

**O‘ZBEKISTON RESPUBLIKASI
MAKTABGACHA VA MAKTAB TA’LIMI VAZIRLIGI
NAMANGAN DAVLAT PEDAGOGIKA INSTITUTI**

“Intellectual fanlar va axborot texnologiyalari” kafedrası

WEB TEXNOLOGIYALARI

fanidan

O‘QUV-USLUBIY MAJMUA

Kurs: Informatika fanini o‘qitish huquqini berish bo‘yicha
kasbiy va pedagogik qayta tayyorlash

Ta’lim dasturi: 10011013

(ishlab chiqarishdan ajralgan va ajralmagan holda)

Umumiy yuklama: 20 soat (ma’ruza – 10, amaliy – 10)

Namangan – 2026

MUNDARIJA

SO‘Z BOSHI	3
I. FANNING O‘QUV DASTURI.....	4
II. ISHCHI O‘QUV DASTURI	7
III. FANNI O‘QITISHDA FOYDALANILADIGAN INTERFAOL TA’LIM METODLARI	10
IV. NAZARIY MATERIALLAR (MA’RUZA MATNLARI)	12
1-MAVZU. WEB TEXNOLOGIYALARI VA INTERNET ASOSLARI. MIJOZ-SERVER ARXITEKTURASI, HTTP/HTTPS PROTOKOLLARI, VEB-BRAUZERLAR ISHLASH PRINSIPL. HTML ASOSLARI VA SEMANTIK HTML5.....	12
2-MAVZU. CSS ASOSLARI: SELEKTORLAR, BOX MODELI, FLEXBOX VA GRID. RESPONSIVE DIZAYN VA BOOTSTRAP FREYMOVORKI	23
3-MAVZU. JAVASCRIPT ASOSLARI VA DOM BILAN ISHLASH. HODISALAR (EVENTS). FETCH API VA JSON	32
4-MAVZU. BACKEND DASTURLASHGA KIRISH: NODE.JS VA EXPRESS.JS. MA’LUMOTLAR BAZASI VA REST API ASOSLARI.....	41
5-MAVZU. VEB-XAVFSIZLIK (XSS, CSRF, SQL INJECTION), VEB-SAYTNI OPTIMALLASHTIRISH, SEO ASOSLARI VA HOSTINGGA JOYLASHTIRISH.....	51
V. AMALIY MASHG‘ULOT MATERIALLARI.....	56
1-AMALIY MASHG‘ULOT. VEB-LOYIHA MUHITINI SOZLASH. SEMANTIK HTML5 YORDAMIDA VEB-SAHIFA YARATISH: FORMALAR, JADVALLAR, MULTIMEDIA ELEMENTLARI.....	58
2-AMALIY MASHG‘ULOT. CSS FLEXBOX, GRID VA MEDIA QUERIES YORDAMIDA RESPONSIVE SAHIFA YARATISH.....	63
3-AMALIY MASHG‘ULOT. JAVASCRIPT VA DOM: INTERAKTIV ELEMENTLAR YARATISH, TASHQI API’DAN MA’LUMOT OLISH	69
4-AMALIY MASHG‘ULOT. NODE.JS VA EXPRESS.JS BILAN ODDIY SERVER VA REST API YARATISH	77
5-AMALIY MASHG‘ULOT. YAKUNIY LOYIHA: RESPONSIVE VEB-SAYTNI GITHUB PAGES / NETLIFY GA JOYLASHTIRISH, TAQDIMOT QILISH VA HIMOYA	85
VI. MUSTAQIL TA’LIM MASHG‘ULOTLARI	90
VII. KEYSLAR BANKI.....	93
VIII. GLOSSARIY	97
IX. TEST SAVOLLARI.....	101
X. YAKUNIY NAZORAT (YOZMA ISH) MATERIALLARI VA BAHOLASH MEZONLARI	113
XI. ADABIYOTLAR RO‘YXATI	120

SO‘Z BOSHI

Mazkur o‘quv-uslubiy majmua Namangan davlat pedagogika institutida tashkil etilgan 10011013 – “Informatika fanini o‘qitish huquqini berish bo‘yicha kasbiy va pedagogik qayta tayyorlash” kursi tinglovchilari uchun “Web texnologiyalari” fanining o‘quv dasturi asosida tayyorlangan. Majmua fanning maqsad va vazifalari, ta’lim natijalari (TN1–TN10), ma’ruza va amaliy mashg‘ulotlar mazmuni hamda mustaqil ta’lim topshiriqlarini to‘liq qamrab oladi.

Fan bo‘yicha jami auditoriya yuklamasi 20 soatni tashkil etadi: ma’ruza mashg‘ulotlari – 10 soat (5 ta mavzu), amaliy mashg‘ulotlar – 10 soat (5 ta mashg‘ulot). Mashg‘ulotlar ketma-ketligi tinglovchini Internet va HTTP protokoli asoslaridan boshlab HTML, CSS, JavaScript, Bootstrap, Node.js/Express.js, ma’lumotlar bazasi, REST API, veb-xavfsizlik, optimallashtirish va SEO orqali responsive veb-saytni yaratish, uni GitHub Pages / Netlify ga joylashtirish va himoya qilishgacha olib boradi.

Majmua quyidagi qismlardan iborat:

- fanning o‘quv dasturi;
- ishchi o‘quv dasturi (mavzular va soatlar taqsimoti, nazorat shakllari);
- fanni o‘qitishda foydalaniladigan interfaol ta’lim metodlari;
- nazariy materiallar (M1–M5 ma’ruza matnlari, reja, tayanch iboralar, nazorat savollari);
- amaliy mashg‘ulot materiallari (A1–A5: maqsad, bajarish tartibi, dastur kodlari, variantli topshiriqlar, baholash mezonlari);
- mustaqil ta’lim topshiriqlari (variantlar, kooperativ topshiriqlar, baholash mezonlari);
- keyslar banki;
- glossariy (o‘zbek, ingliz, rus tillarida atamalar va izohlar);
- test savollari va javoblar kaliti;
- yakuniy nazorat (yozma ish) materiallari va baholash mezonlari;
- foydalanilgan adabiyotlar va axborot manbalari.

Majmuadagi dastur kodlari zamonaviy brauzerlar (Google Chrome, Mozilla Firefox, Microsoft Edge) va Node.js 20 LTS yoki undan keyingi versiyalarida tekshirishga mo‘ljallangan. Kod namunalarini Visual Studio Code muharririda yozish va “Live Server” kengaytmasi orqali ishga tushirish tavsiya etiladi.

Majmuadan informatika o‘qituvchilari, qayta tayyorlash kursi tinglovchilari hamda “Matematika va informatika” ta’lim yo‘nalishi talabalari foydalanishlari mumkin.

Tuzuvchi: D. Sodirova – NamDPI “Intellectual fanlar va axborot texnologiyalari”
kafedra o‘qituvchisi

Taqrizchilar: Sh. Boltabayev – NamDU “Raqamli ta’lim texnologiyalar”
kafedra dotsenti

Sh. Jurayev – NamDU “Raqamli ta’lim texnologiyalar” kafedra dotsenti

Mazkur o‘quv-uslubiy majmua “Intellectual fanlar va axborot texnologiyalari” kafedrasining 2026-yil “___” _____dagi ___-sonli yig‘ilishida muhokama qilingan va Aniq va tabiiy fanlar fakulteti Kengashining 2026-yil “___” _____dagi ___-sonli yig‘ilish bayoni bilan ma’qullangan.

I. FANNING O‘QUV DASTURI

1. Fan ma’lumotlari

Ko‘rsatkich	Qiymat	Ko‘rsatkich	Qiymat
Fan/modul kodi	WEBT	O‘quv yili	2026–2027
Fan/modul turi	Umumkasbiy fanlar / Tanlov	Ta’lim tili	O‘zbek
O‘quv rejadagi tartib raqami	10011013 – 3.08	O‘qitish oyi	8
Jami auditoriya yuklamasi	20 soat	Ma’ruza / Amaliy / Lab.	10 / 10 / –

2. Fanning mazmuni

Fanning maqsadi – tinglovchilarga zamonaviy veb-texnologiyalar, frontend va backend dasturlash asoslari, veb-saytlarni loyihalash va ishlab chiqish vositalarini o‘rgatish, ularning kasbiy faoliyatida axborot-kommunikatsiya texnologiyalaridan samarali foydalanish ko‘nikmalarini shakllantirishdan iborat.

Fanning vazifalari – veb-texnologiyalarning asosiy tushunchalari va mijoz-server arxitekturasini o‘rganish; HTML, CSS va JavaScript texnologiyalarini o‘zlashtirish; responsive (moslashuvchan) veb-dizayn tamoyillarini qo‘llash; backend dasturlash, ma’lumotlar bazalari va API’lar bilan ishlash; veb-xavfsizlik va loyihalarni joylashtirish bo‘yicha amaliy ko‘nikmalarni rivojlantirishdan iborat.

3. Fanni o‘zlashtirish uchun zarur boshlang‘ich bilimlar

- Informatika va raqamli texnologiyalar;
- Zamonaviy dasturlash tillari;
- Kompyuter grafikasi va veb-dizayn.

4. Ta’lim natijalari (TN)

Kod	Ta’lim natijasi
Kasbiy kompetensiyalar	
TN1	veb-texnologiyalarning nazariy asoslari, Internet va mijoz-server arxitekturasini, HTTP protokoli haqida bilimga ega bo‘lishi kerak
TN2	HTML va CSS yordamida veb-sahifalar tuzilishi va dizaynini yaratish tamoyillarini bilishi kerak
TN3	JavaScript dasturlash tili va DOM bilan ishlash imkoniyatlarini bilishi kerak
TN4	responsive (moslashuvchan) veb-dizayn tamoyillari va texnologiyalarini tushunishi kerak
TN5	backend dasturlash, ma’lumotlar bazalari va REST API asoslarini bilishi kerak

Kod	Ta'lim natijasi
TN6	veb-xavfsizlik, autentifikatsiya va veb-ilovalarni joylashtirish jarayonlarini tushunishi kerak
Ko'nikmalar	
TN7	HTML, CSS va JavaScript yordamida to'liq funksional veb-saytlar yaratish ko'nikmasiga ega bo'ladi
TN8	turli qurilmalarga moslashuvchi (responsive) veb-sahifalar yaratish ko'nikmasiga ega bo'ladi
TN9	server tarafi dasturlash, ma'lumotlar bazasi va API bilan ishlash ko'nikmasiga ega bo'ladi
TN10	veb-loyihalarni rejalashtirish, ishlab chiqish, joylashtirish va taqdimot qilish bo'yicha amaliy tajribaga ega bo'ladi

5. Fan mazmuni va mashg'ulotlar shakli

Nº	Mavzu va uning mazmuni	Soat
Ma'ruza (M)		10
M1	Web texnologiyalari va Internet asoslari. Mijoz-server arxitekturasini, HTTP/HTTPS protokollari, veb-brauzerlar ishlash prinsipi. HTML asoslari va semantik HTML5. [1,3,9]	2
M2	CSS asoslari: selektorlar, box modeli, Flexbox va Grid. Responsive dizayn va Bootstrap freymvorki. [3,5,11]	2
M3	JavaScript asoslari va DOM bilan ishlash. Hodisalar (events). Fetch API va JSON. [2,4,6]	2
M4	Backend dasturlashga kirish: Node.js va Express.js. Ma'lumotlar bazasi va REST API asoslari. [7,10]	2
M5	Veb-xavfsizlik (XSS, CSRF, SQL Injection), veb-saytni optimallashtirish, SEO asoslari va hostingga joylashtirish. [5,6]	2
Amaliy mashg'ulot (A)		10
A1	Veb-loyiha muhitini sozlash. Semantik HTML5 yordamida veb-sahifa yaratish: formalar, jadvallar, multimedia elementlari. [3,8,9]	2
A2	CSS Flexbox, Grid va media queries yordamida responsive sahifa yaratish. [3,5]	2
A3	JavaScript va DOM: interaktiv elementlar yaratish, tashqi API'dan ma'lumot olish. [2,4,6]	2
A4	Node.js va Express.js bilan oddiy server va REST API yaratish. [7,10]	2
A5	Yakuniy loyiha: responsive veb-saytni GitHub Pages / Netlify ga joylashtirish, taqdimot qilish va himoya. [3,4,6]	2

6. Mustaqil ta'lim topshiriqlari

Nº	Topshiriq
1	Shaxsiy portfolio veb-saytni HTML va CSS yordamida yaratish va himoya qilish.

№	Topshiriq
2	Responsive landing page (mahsulot yoki xizmat sahifasi) loyihasini yaratish.
3	JavaScript yordamida interaktiv veb-ilova yaratish (kalkulyator, viktorina yoki to-do list).
4	Tashqi API'dan foydalangan holda dinamik veb-ilova ishlab chiqish (ob-havo, valyuta yoki yangiliklar).
5	REST API va ma'lumotlar bazasi bilan ishlovchi oddiy backend loyihasini yaratish (kichik guruhlarda hamkorlikda).
6	Zamonaviy veb-texnologiyalar yoki freymvorklar bo'yicha tahliliy taqdimot tayyorlash (kichik guruhlarda hamkorlikda).

Izoh: mustaqil ish topshiriqlari variativ tavsifga ega; topshiriqlarning 1/3 qismi (5- va 6-topshiriqlar) kichik guruhlarda hamkorlikda ishlashga mo'ljallangan.

7. Ta'lim texnologiyalari va metodlari

- ma'ruzalar;
- amaliy ishlarni bajarish va xulosalash;
- interfaol keys-stadilar;
- blits-so'rov;
- guruhlarda ishlash;
- taqdimotlar qilish;
- jamoa bo'lib ishlash va himoya qilish uchun loyihalar.

8. Kreditlarni olish uchun talablar

Fanga oid nazariy va uslubiy tushunchalarni to'la o'zlashtirish, tahlil natijalarini to'g'ri aks ettira olish, o'rganilayotgan jarayonlar haqida mustaqil mushohada yuritish va nazorat uchun berilgan vazifa va topshiriqlarni bajarish, yakuniy nazorat bo'yicha yozma ishni topshirish.

9. Tinglovchi bilimini baholash mezonlari

Baholash mezonlari ushbu majmuaning X bo'limida batafsil keltirilgan (5 – "a'lo", 4 – "yaxshi", 3 – "qoniqarli", 2 – "qoniqarsiz").

10. Foydalanilgan adabiyotlar

Adabiyotlar ro'yxati majmuaning XI bo'limida keltirilgan. Mavzular yonidagi kvadrat qavs ichidagi raqamlar ([1,3,9]) ushbu ro'yxatdagi tartib raqamlariga mos keladi.

Tuzuvchi: D. Sodirova
Kafedra mudiri: A. Matkarimov

II. ISHCHI O‘QUV DASTURI

1. Fanning o‘quv yuklamasi taqsimoti

Mashg‘ulot turi	Soat	Mashg‘ulotlar soni	Nazorat shakli
Ma’ruza	10	5	Blits-so‘rov, nazorat savollari
Amaliy mashg‘ulot	10	5	Amaliy ishni himoya qilish
Jami auditoriya	20	10	–
Mustaqil ta’lim	–	6 ta topshiriq	Loyiha / taqdimot himoyasi
Yakuniy nazorat	–	–	Yozma ish

2. Ma’ruza mashg‘ulotlarining mavzuli rejası

№	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M1	Web texnologiyalari va Internet asoslari. HTML asoslari va semantik HTML5 1. Internet va WWW. 2. Mijoz-server arxitekturasi. 3. HTTP/HTTPS. 4. Brauzerlar ishlash prinsipi. 5. HTML hujjat strukturasi. 6. Asosiy teglar, ro‘yxat va jadvallar. 7. Semantik teglar. 8. Multimedia va formalar	2	Klaster, blits-so‘rov, insert	TN1, TN2
M2	CSS asoslari: selektorlar, box modeli, Flexbox va Grid. Responsive dizayn va Bootstrap 1. Sintaksis va ulash usullari. 2. Selektorlar va kaskad. 3. Ranglar, fon, shriftlar. 4. Box modeli. 5. Display, Flexbox, Grid, positioning. 6. Responsive dizayn. 7. Bootstrap grid va komponentlar	2	Venn diagrammasi, aqliy hujum	TN2, TN4, TN8
M3	JavaScript asoslari va DOM. Hodisalar. Fetch API va JSON 1. O‘zgaruvchilar va turlar. 2. Operatorlar, shartlar, tsikllar. 3. Funksiyalar, massivlar, obyektlar. 4. DOM. 5. Hodisalar. 6. JSON va Fetch API	2	Muammoli ma’ruza, “Kodni o‘qi”	TN3, TN7
M4	Backend dasturlashga kirish: Node.js va Express.js. Ma’lumotlar bazasi va REST API asoslari 1. Frontend va backend. 2. Node.js va NPM. 3. Express, routing, middleware. 4. MB tushunchasi, SQL va NoSQL. 5. Jadvallar va bog‘lanishlar. 6. SQL so‘rovlari. 7. REST API	2	Keys-stadi, guruhlarda ishlash	TN5, TN9

N ^o	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M5	Veb-xavfsizlik, optimallashtirish, SEO va hostingga joylashtirish 1. Autentifikatsiya, sessiya, cookie. 2. XSS, CSRF, SQL Injection. 3. Optimallashtirish. 4. SEO. 5. Git va hosting	2	Keys-stadi, “Hujum va himoya”	TN6, TN10
Jami		10		

3. Amaliy mashg‘ulotlarning mavzuli rejasi

N ^o	Amaliy mashg‘ulot mavzusi va natijasi	Soat	Shakli	TN
A1	Muhitni sozlash, semantik HTML5 sahifa, formalar, jadval, multimedia. Natija: “Maktab to‘garagi” sahifasi	2	Individual	TN2, TN7
A2	CSS, Flexbox, Grid, media queries. Natija: telefon, planshet va kompyuterga moslashuvchi sahifa	2	Individual	TN2, TN4, TN8
A3	DOM, hodisalar, Fetch API, Bootstrap. Natija: interaktiv elementlar va ob-havo ilovasi	2	Juftlikda	TN3, TN7, TN8
A4	Express server, middleware, SQLite, CRUD REST API, frontend integratsiya. Natija: “Kutubxona” API si	2	Individual	TN5, TN9
A5	Git, GitHub Pages / Netlify, Lighthouse, yakuniy loyiha himoyasi. Natija: Internetda joylashgan sayt	2	Kichik guruh	TN6, TN7, TN10
Jami		10		

4. Mustaqil ta’lim topshiriqlarining taqsimoti

N ^o	Topshiriq	Bog‘liq mavzu	Shakli	Topshirish muddati
1	Shaxsiy portfolio veb-sayti (HTML, CSS)	M1, M2, A1, A2	Individual	A2 mashg‘ulotigacha
2	Responsive landing page	M2, A2	Individual	A3 mashg‘ulotigacha
3	Interaktiv JavaScript ilova	M3, A3	Individual	A4 mashg‘ulotigacha
4	Tashqi API bilan dinamik ilova	M3, A3	Individual	A5 mashg‘ulotigacha
5	REST API va MB bilan backend loyiha	M4, A4	Kichik guruh (3–4 kishi)	A5 mashg‘ulotida
6	Zamonaviy veb-texnologiyalar bo‘yicha tahliliy taqdimot	M1, M5	Kichik guruh (3–4 kishi)	A5 mashg‘ulotida

5. Nazorat turlari va baholash tartibi

Tinglovchilar bilimi joriy va yakuniy nazorat orqali 5 baholik tizimda baholanadi.

Joriy nazorat amaliy mashg'ulotlarda bajarilgan ishlarni himoya qilish, ma'ruzalardagi blits-so'rovlar va mustaqil ta'lim topshiriqlarini himoya qilish orqali amalga oshiriladi. Har bir amaliy ish 5 baholik shkalada baholanadi; joriy baho barcha amaliy ishlar va mustaqil topshiriqlar baholarining o'rtacha arifmetigi sifatida aniqlanadi.

Yakuniy nazorat yozma ish shaklida o'tkaziladi. Yozma ish varianti 3 ta savoldan iborat: ikkita nazariy savol va bitta amaliy (kod yozish yoki tahlil qilish) topshiriq. Yozma ish uchun 80 daqiqa vaqt ajratiladi.

Yakuniy nazoratga barcha amaliy ishlarni topshirgan va joriy nazoratdan kamida "3" baho olgan tinglovchilar qo'yiladi. Fan bo'yicha umumiy baho joriy nazorat (40%) va yakuniy nazorat (60%) natijalarining o'lchangan o'rtachasi sifatida, butun songa yaxlitlangan holda qo'yiladi.

Nazorat turi	Shakli	Ulushi
Joriy nazorat	Amaliy ishlar himoyasi, blits-so'rov, mustaqil ta'lim	40%
Yakuniy nazorat	Yozma ish (3 savol, 80 daqiqa)	60%

6. Fanni o'qitish uchun zarur moddiy-texnik va dasturiy ta'minot

- kompyuter sinfi (har bir tinglovchiga bitta kompyuter), proyektor yoki interaktiv doska;
- Internet tarmog'iga ulanish;
- Google Chrome yoki Mozilla Firefox brauzeri (DevTools vositalari bilan);
- Visual Studio Code muharriri va "Live Server", "Prettier", "SQLite Viewer" kengaytmalari;
- Node.js 20 LTS va NPM, Git versiyalarni boshqarish tizimi;
- GitHub va Netlify bepul hisoblari;
- Postman yoki Thunder Client (API so'rovlarini tekshirish uchun).

III. FANNI O‘QITISHDA FOYDALANILADIGAN INTERFAOL TA’LIM METODLARI

Fanni o‘qitishda tinglovchilarning faolligini oshirish, nazariy bilimlarni amaliyot bilan bog‘lash va kasbiy kompetensiyalarni shakllantirish maqsadida quyidagi interfaol metodlardan foydalaniladi. Har bir metod uchun uning maqsadi, o‘tkazish tartibi va fanga tatbiq etish namunasi keltirilgan.

1. “Blits-so‘rov” metodi

Maqsadi: qisqa vaqt ichida o‘tilgan mavzu bo‘yicha tayanch tushunchalarni takrorlash va tinglovchilarning tayyorgarlik darajasini aniqlash.

O‘tkazish tartibi:

- o‘qituvchi 8–10 ta qisqa savol tayyorlaydi;
- savollar tez sur‘atda navbat bilan beriladi, javob uchun 10–15 soniya ajratiladi;
- noto‘g‘ri javoblar darhol boshqa tinglovchilar tomonidan to‘g‘rilanadi;
- yakunda o‘qituvchi umumiy xulosa qiladi.

Fanga tatbiq etish namunasi (M1 da HTML qismi boshlanishida): “HTML qanday kengaytma bilan saqlanadi?”, “<a> tegining qaysi atributi manzilni bildiradi?”, “Semantik teg deganda nimani tushunasiz?”, “ va farqi?”, “alt atributi nima uchun kerak?”.

2. “Klaster” metodi

Maqsadi: mavzuga oid tushunchalar o‘rtasidagi bog‘lanishlarni grafik shaklda aks ettirish orqali tizimli fikrlashni rivojlantirish.

O‘tkazish tartibi: doska markaziga asosiy tushuncha yoziladi (masalan, “Veb-texnologiyalar”), tinglovchilar unga bog‘liq tushunchalarni tarmoqlar ko‘rinishida qo‘shib boradi: “Frontend” → HTML, CSS, JavaScript; “Backend” → Node.js, Express, ma’lumotlar bazasi; “Protokollar” → HTTP, HTTPS, DNS, TCP/IP. Kichik guruhlar o‘z klasterlarini taqdim etadi va taqqoslaydi.

3. “Insert” metodi

Maqsadi: matn bilan ishlashda faol o‘qish va tanqidiy fikrlashni shakllantirish.

Tinglovchilar ma’ruza matnining bir qismini o‘qiydi va quyidagi belgilar bilan belgilaydi:

Belgi	Ma’nosi
V	Avvaldan bilgan ma’lumotlarim bilan mos keladi
+	Men uchun yangi ma’lumot

Belgi	Ma'nosi
–	Bilganlarimga zid ma'lumot
?	Tushunarsiz, qo'shimcha izoh talab qiladi

Natijalar jadvalga yoziladi va “?” belgisi qo'yilgan joylar birgalikda muhokama qilinadi. M1 mavzusining semantik HTML5 teglari qismida qo'llash tavsiya etiladi.

4. “Venn diagrammasi” metodi

Maqsadi: ikki yoki uch tushunchani taqqoslash, ularning umumiy va farqli jihatlarini aniqlash.

Fanga tatbiq etish namunalari: “Flexbox va CSS Grid”, “SQL va NoSQL ma'lumotlar bazalari”, “Statik va dinamik veb-saytlar”, “**let**, **const** va **var**”. Tinglovchilar kesishgan doiralar ichiga mos xususiyatlarni joylashtiradi.

5. “Aqliy hujum” (Brainstorming) metodi

Maqsadi: muammo bo'yicha qisqa vaqt ichida imkon qadar ko'p g'oya to'plash.

Qoidalar: g'oyalar tanqid qilinmaydi; miqdor sifatdan muhim; boshqalarning g'oyasini rivojlantirish rag'batlantiriladi; barcha g'oyalar yozib boriladi; keyingi bosqichda ular saralanadi.

Namuna: “Maktab sayti foydalanuvchilar uchun qulay bo'lishi uchun qanday talablarga javob berishi kerak?” (M1, M2, M5).

6. “Keys-stadi” metodi

Maqsadi: real yoki real holatga yaqin vaziyatlarni tahlil qilish orqali muammoni aniqlash va yechim ishlab chiqish ko'nikmasini shakllantirish.

Bosqichlari:

- keys matni bilan individual tanishish (5–7 daqiqa);
- muammoni aniqlash va asosiy savollarni shakllantirish;
- kichik guruhlarda yechim variantlarini ishlab chiqish (10–15 daqiqa);
- yechimlarni taqdim etish va himoya qilish;
- o'qituvchi tomonidan umumlashtirish va baholash.

Fan bo'yicha keyslar banki majmuaning VII bo'limida berilgan.

7. “Kodni o'qi va bashorat qil” metodi

Maqsadi: dastur kodini tahlil qilish, natijani oldindan aniqlash orqali algoritmik tafakkurni rivojlantirish.

O'qituvchi ekranga qisqa kod parchasini chiqaradi. Tinglovchilar kodni ishga tushirmasdan natijani daftariga yozadi, so'ngra kod brauzer konsolida bajariladi va natijalar solishtiriladi. Xatoliklar sababi muhokama qilinadi. M3 va A3 mashg'ulotlarida qo'llaniladi.

8. “Hujum va himoya” rolli o'yini

Maqsadi: veb-xavfsizlik zaifliklarini amaliy tushunish.

Guruh ikkiga bo'linadi: “hujumchilar” o'quv maqsadida tayyorlangan, ataylab zaif qilingan lokal sahifadagi zaiflikni (masalan, filtrlanmagan izoh maydoni) topadi va tavsiflaydi, “himoyachilar” esa uni bartaraf etish usulini taklif qiladi va kodga tuzatish kiritadi. Mashg'ulot faqat lokal o'quv muhitida o'tkaziladi. M5 va A4 mashg'ulotlarida qo'llaniladi.

9. Loyiha metodi

Maqsadi: egallangan bilim va ko'nikmalarni yaxlit mahsulot – veb-sayt yaratishga yo'naltirish.

Bosqichlari: g'oyani tanlash → talablarni aniqlash → prototip (wireframe) chizish → ishlab chiqish → test qilish → hostingga joylashtirish → taqdimot va himoya. A5 mashg'ulotida va 5-mustaqil topshiriqda qo'llaniladi.

IV. NAZARIY MATERIALLAR (MA'RUZA MATNLARI)

1-MAVZU. WEB TEXNOLOGIYALARI VA INTERNET ASOSLARI. MIJOZ-SERVER ARXITEKTURASI, HTTP/HTTPS PROTOKOLLARI, VEB-BRAUZERLAR ISHLASH PRINSIPI. HTML ASOSLARI VA SEMANTIK HTML5

Reja:

1. Internet va World Wide Web tushunchalari.
2. Mijoz-server arxitekturasini.
3. HTTP va HTTPS protokollari.
4. Veb-brauzerlarning ishlash prinsipi.
5. HTML tili va HTML hujjat strukturasi.
6. Asosiy teglar: sarlavhalar, matn, havolalar, rasmlar.
7. Ro'yxatlar va jadvallar.
8. Semantik HTML5 teglari.
9. Multimedia elementlari va formalar.

Tayanch iboralar:

Internet, WWW, URL, domen, DNS, IP-manzil, TCP/IP, mijoz, server, HTTP, HTTPS, TLS, so'rov, javob, status kodi, brauzer, render, DevTools, HTML,

teg, element, atribut, `<!DOCTYPE html>`, head, body, havola, rasm, `alt`, ro'yxat, jadval, semantika, header, nav, main, section, article, footer, audio, video, forma.

1. Internet va World Wide Web tushunchalari

Internet – bu yagona TCP/IP protokollar to'plami asosida o'zaro bog'langan millionlab kompyuter tarmoqlarining global tizimidir. Internetning ilk ko'rinishi 1969-yilda AQSh Mudofaa vazirligining ARPANET loyihasi doirasida yaratilgan. 1983-yilda ARPANET to'liq TCP/IP protokoliga o'tkazildi va bu sana zamonaviy Internetning tug'ilgan kuni hisoblanadi.

Ko'pincha "Internet" va "veb" so'zlari sinonim sifatida ishlatiladi, ammo ular farqli tushunchalardir. Internet – bu infratuzilma, ya'ni ma'lumot uzatiladigan "yo'llar" tarmog'i. **World Wide Web (WWW, veb)** esa Internet orqali ishlaydigan xizmatlardan biri bo'lib, gipermatnli hujjatlar (veb-sahifalar) va ular orasidagi havolalar tizimidan iborat. Elektron pochta (SMTP), fayl uzatish (FTP), messenjerlar ham Internet xizmatlari hisoblanadi, lekin ular veb emas.

Veb 1989-yilda Yevropa yadro tadqiqotlari markazi (CERN) xodimi **Tim Berners-Li** tomonidan taklif qilingan. U vebning uchta asosiy tarkibiy qismini yaratgan:

- **HTML** (HyperText Markup Language) – veb-sahifalarni belgilash tili;
- **URL** (Uniform Resource Locator) – resursning yagona manzili;
- **HTTP** (HyperText Transfer Protocol) – gipermatnni uzatish protokoli.

URL tuzilishi

Har bir veb-resurs o'zining noyob manziliga – URL ga ega. Quyidagi URL ni tahlil qilamiz:

```
https://www.namdpi.uz:443/uz/news/index.html?page=2&lang=uz#comments
```

Qism	Qiymat	Vazifasi
Protokol (sxema)	https	Ma'lumot almashish qoidasi
Domen (host)	www.namdpi.uz	Server nomi
Port	443	Serverdagi xizmat raqami (HTTPS uchun standart)
Yo'l (path)	/uz/news/index.html	Serverdagi resurs joylashuvi
So'rov satri (query)	?page=2&lang=uz	Serverga uzatiladigan parametrlar
Fragment	#comments	Sahifa ichidagi qism

IP-manzil va DNS

Internetdagi har bir qurilma **IP-manzil** bilan aniqlanadi. IPv4 manzil 32 bitdan iborat bo'lib, nuqta bilan ajratilgan to'rtta sondan tashkil topadi (masalan,

185.74.5.210). IPv4 manzillar soni (taxminan 4,3 mlrd) yetishmay qolgani sababli 128 bitli **IPv6** standarti joriy etilmoqda (masalan, 2001:0db8:85a3::8a2e:0370:7334).

Insonlar uchun raqamlarni eslab qolish qiyin, shu sababli **domen nomlari** (namdpi.uz, edu.uz) ishlatiladi. Domen nomini IP-manzilga aylantirish vazifasini **DNS (Domain Name System)** bajaradi. DNS – bu Internetning “telefon kitobi”dir. Domenlar ierarxik tuzilishga ega: **.uz** – yuqori darajadagi domen (TLD), **namdpi.uz** – ikkinchi darajali domen, **www.namdpi.uz** – subdomen.

TCP/IP protokollar steki

Internetda ma’lumot uzatish qatlamlarga bo’lingan holda amalga oshiriladi:

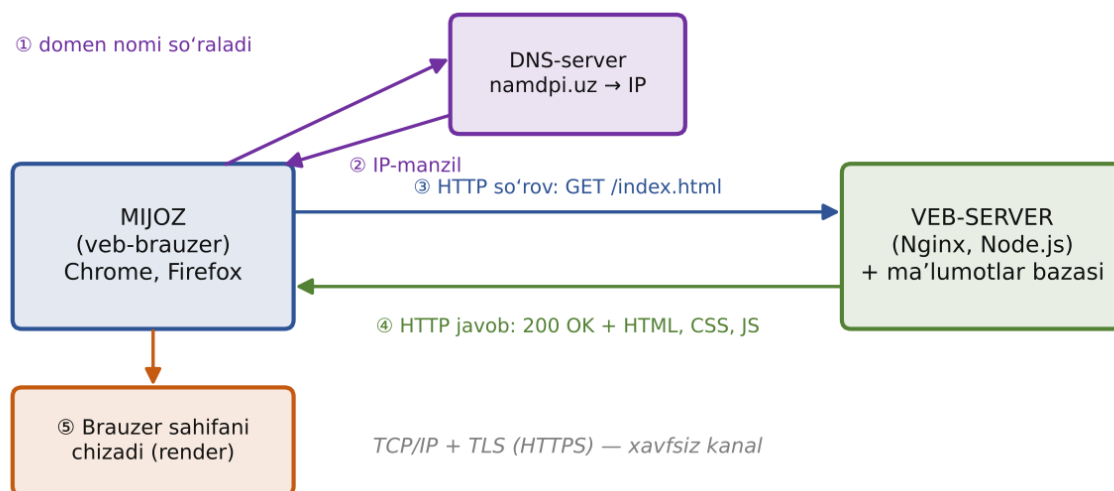
Qatlam	Vazifasi	Protokollar
Amaliy (Application)	Ilovalar o’rtasida ma’lumot almashish	HTTP, HTTPS, DNS, FTP, SMTP
Transport	Ma’lumotni ishonchli yetkazish, paketlarga bo’lish	TCP, UDP
Tarmoq (Internet)	Paketlarni manzilga yo’naltirish (marshrutlash)	IP, ICMP
Kanal (Link)	Fizik muhitda uzatish	Ethernet, Wi-Fi

2. Mijoz-server arxitekturas

Veb **mijoz-server (client-server)** arxitekturasida ishlaydi. Bu modelda vazifalar ikki tomonga taqsimlanadi:

- **Mijoz (client)** – xizmatni so’rovchi dastur. Vebda mijoz vazifasini odatda veb-brauzer bajaradi. Mijoz so’rov yuboradi va olingan javobni foydalanuvchiga ko’rsatadi.
- **Server** – so’rovlarni qabul qilib, qayta ishlovchi va javob qaytaruvchi dastur (yoki kompyuter). Mashhur veb-serverlar: Nginx, Apache HTTP Server, Microsoft IIS, shuningdek Node.js asosidagi ilovalar.

Foydalanuvchi brauzer manzil satriga **https://namdpi.uz** deb yozganda quyidagi jarayon kechadi (1-rasm):



1-rasm. Mijoz-server o'zaro ta'siri va HTTP so'rov-javob sikli

- 1) brauzer DNS-serverdan **nampdi.uz** domeniga mos IP-manzilni so'raydi;
- 2) DNS-server IP-manzilni qaytaradi;
- 3) brauzer ushbu IP-manzildagi serverga TCP ulanish o'rnatadi (HTTPS bo'lsa, qo'shimcha TLS kelishuvi bajariladi) va HTTP so'rov yuboradi;
- 4) server so'rovni qayta ishlaydi va HTML hujjatni javob sifatida qaytaradi;
- 5) brauzer HTML ni tahlil qiladi, unda ko'rsatilgan CSS, JavaScript, rasm fayllarini qo'shimcha so'rovlar bilan yuklaydi va sahifani ekranda chizadi (render qiladi).

Mijoz-server arxitekturasining afzalliklari: ma'lumotlarning markazlashgan holda saqlanishi va boshqarilishi; xavfsizlikni server tomonida ta'minlash imkoniyati; bitta serverga ko'plab mijozlarning ulanishi; mijoz tomonida faqat brauzer bo'lishining yetarliligi.

Statik va dinamik veb-saytlar

Statik sayt – barcha foydalanuvchilarga bir xil, oldindan tayyorlangan HTML fayllarni yuboradi. Ular tez ishlaydi va oddiy hostingda joylashtiriladi (GitHub Pages, Netlify). **Dinamik sayt** – sahifani so'rov paytida server dasturi (Node.js, PHP, Python) yordamida ma'lumotlar bazasidagi ma'lumotlar asosida hosil qiladi (masalan, internet-do'kon, elektron kundalik, hemis.uz).

Frontend, backend va full-stack

Yo'nalish	Vazifasi	Texnologiyalar
Frontend	Foydalanuvchi ko'radigan va o'zaro ta'sir qiladigan qism (interfeys)	HTML, CSS, JavaScript, Bootstrap, React, Vue
Backend	Server mantiqi, ma'lumotlar bazasi, autentifikatsiya, API	Node.js, Express, PHP, Python (Django), SQL
Full-stack	Frontend va backendni birgalikda ishlab chiqish	Yuqoridagilarning barchasi

3. HTTP va HTTPS protokollari

HTTP (HyperText Transfer Protocol) – mijoz va server o‘rtasida gipermatnli ma’lumotlarni uzatish uchun amaliy qatlam protokoli. HTTP **holatsiz (stateless)** protokol hisoblanadi: server har bir so‘rovni oldingi so‘rovlardan mustaqil ravishda qayta ishlaydi. Foydalanuvchini “eslab qolish” uchun cookie, sessiya va tokenlardan foydalaniladi (M5 mavzusiga qarang).

HTTP xabari uch qismdan iborat: boshlang‘ich satr, sarlavhalar (headers) va tana (body). Quyida so‘rov va javob namunasi keltirilgan:

```
GET /uz/news HTTP/1.1
Host: namdpi.uz
User-Agent: Mozilla/5.0 (Windows NT 10.0) Chrome/128.0
Accept: text/html
Accept-Language: uz,ru;q=0.8
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 18452
Cache-Control: max-age=3600
```

```
<!DOCTYPE html>
<html lang="uz"> ... </html>
```

HTTP metodlari

Metod	Vazifasi	Misol
GET	Resursni olish (o‘qish)	Sahifani ochish, ro‘yxatni olish
POST	Yangi ma’lumot yuborish/yaratish	Ro‘yxatdan o‘tish formasi
PUT	Resursni to‘liq yangilash	Profil ma’lumotlarini almashtirish
PATCH	Resursni qisman yangilash	Faqat telefon raqamini o‘zgartirish
DELETE	Resursni o‘chirish	Izohni o‘chirish

HTTP status kodlari

Guruh	Ma’nosi	Eng ko‘p uchraydigan kodlar
1xx	Axborot	100 Continue
2xx	Muvaffaqiyatli	200 OK, 201 Created, 204 No Content
3xx	Yo‘naltirish	301 Moved Permanently, 302 Found, 304 Not Modified
4xx	Mijoz xatosi	400 Bad Request, 401 Unauthorized, 403 Forbidden, 404 Not Found

Guruh	Ma'nosi	Eng ko'p uchraydigan kodlar
5xx	Server xatosi	500 Internal Server Error, 502 Bad Gateway, 503 Service Unavailable

HTTPS

HTTPS (HTTP Secure) – HTTP protokolining **TLS (Transport Layer Security)** shifrlash protokoli ustida ishlaydigan xavfsiz versiyasi. HTTPS uchta vazifani bajaradi:

- **maxfiylik** – uzatilayotgan ma'lumot shifrlanadi, uchinchi shaxs uni o'qiy olmaydi;
- **yaxlitlik** – ma'lumot yo'lda o'zgartirilmaganligi kafolatlanadi;
- **autentifikatsiya** – foydalanuvchi aynan haqiqiy server bilan bog'langanligi SSL/TLS sertifikatini orqali tasdiqlanadi.

Brauzer manzil satridagi qulf belgisi saytning HTTPS orqali ishlayotganini bildiradi. Bugungi kunda Let's Encrypt kabi xizmatlar bepul sertifikat beradi, shuning uchun har qanday sayt HTTPS dan foydalanishi shart deb hisoblanadi. HTTP standart 80-portda, HTTPS esa 443-portda ishlaydi.

HTTP versiyalari

HTTP/1.1 (1997) – bitta ulanishda ketma-ket so'rovlar; HTTP/2 (2015) – bitta ulanishda parallel so'rovlar (multiplexing), sarlavhalarni siqish; HTTP/3 (2022) – UDP asosidagi QUIC protokoli orqali tezroq ulanish.

4. Veb-brauzerlarning ishlash prinsipi

Veb-brauzer – veb-resurslarni so'rash, ularni qayta ishlash va foydalanuvchiga ko'rsatish uchun mo'ljallangan dastur. Hozirgi mashhur brauzerlar: Google Chrome, Microsoft Edge, Mozilla Firefox, Apple Safari, Opera, Yandex Browser.

Brauzerning asosiy komponentlari:

- **foydalanuvchi interfeysi** – manzil satri, tugmalar, xatcho'plar;
- **render dvigateli (rendering engine)** – HTML va CSS ni tahlil qilib, sahifani chizadi: Blink (Chrome, Edge, Opera), Gecko (Firefox), WebKit (Safari);
- **JavaScript dvigateli** – JS kodini bajaradi: V8 (Chrome, Node.js), SpiderMonkey (Firefox), JavaScriptCore (Safari);
- **tarmoq qismi** – HTTP so'rovlarini yuboradi va javoblarni qabul qiladi;
- **ma'lumotlarni saqlash** – cookie, localStorage, IndexedDB, kesh.

Sahifani chizish (render) bosqichlari

- 1) **HTML tahlili** – HTML kod o‘qilib, **DOM** (Document Object Model) daraxti quriladi;
- 2) **CSS tahlili** – stillar o‘qilib, **CSSOM** daraxti quriladi;
- 3) **Render daraxti** – DOM va CSSOM birlashtiriladi, faqat ko‘rinadigan elementlar qoladi;
- 4) **Layout (joylashtirish)** – har bir elementning o‘lchami va ekrandagi o‘rni hisoblanadi;
- 5) **Paint (bo‘yash)** – piksellar ekranga chiziladi;
- 6) **Composite** – qatlamlar birlashtiriladi.

JavaScript kodi DOM ni o‘zgartirganda brauzer layout va paint bosqichlarini qayta bajaradi. Shu sababli DOM ni keragidan ortiq o‘zgartirish sahifa tezligini pasaytiradi.

Brauzerning dasturchi vositalari (DevTools)

Barcha zamonaviy brauzerlarda F12 tugmasi orqali ochiladigan dasturchi vositalari mavjud: **Elements** – HTML va CSS ni ko‘rish va jonli tahrirlash; **Console** – JavaScript xatolarini ko‘rish va kod bajarish; **Network** – barcha HTTP so‘rovlar, status kodlari va yuklanish vaqtini kuzatish; **Application** – cookie va localStorage; **Lighthouse** – sahifa tezligi, SEO va qulaylikni baholash. Ushbu vositalar butun kurs davomida faol ishlatiladi.

5. HTML tili va HTML hujjat strukturasi

HTML (HyperText Markup Language) – veb-sahifalarning tuzilishi va mazmunini tavsiflovchi **belgilash tili**. HTML dasturlash tili emas: unda o‘zgaruvchilar, shartlar va tsikllar yo‘q; u faqat matnning qaysi qismi sarlavha, qaysi qismi paragraf, rasm yoki havola ekanini ko‘rsatadi. Sahifaning tashqi ko‘rinishi CSS, xatti-harakati esa JavaScript yordamida belgilanadi.

HTML hujjat **teglar** (tags) yordamida yoziladi. Teg burchakli qavslar ichidagi kalit so‘zdir. Ko‘pchilik teglar juft bo‘ladi: ochuvchi **<p>** va yopuvchi **</p>**. Ochuvchi teg, mazmun va yopuvchi teg birgalikda **element** deyiladi. Ba’zi elementlar bo‘sh (yopuvchi tegsiz) bo‘ladi: **
, **, **<hr>**, **<input>**, **<meta>**.

Atributlar elementga qo‘shimcha ma’lumot beradi va ochuvchi teg ichida **nom="qiymat"** ko‘rinishida yoziladi:

```
<a href="https://namdpi.uz" target="_blank" title="Institut sayti">NamDPI</a>
```

Global atributlar istalgan elementda ishlatiladi: **id** (noyob identifikator), **class** (CSS sinfi), **style** (ichki stil), **title** (izoh), **lang** (til), **hidden**, **data-*** (maxsus ma’lumotlar).

HTML5 hujjatining asosiy tuzilishi

```
<!DOCTYPE html>
<html lang="uz">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="Informatika to'garagi sahifasi">
  <title>Informatika to'garagi</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Informatika to'garagiga xush kelibsiz!</h1>
  <p>Bu yerda biz veb-sahifa yaratishni o'rganamiz.</p>
  <script src="script.js"></script>
</body>
</html>
```

Element	Vazifasi
<!DOCTYPE html>	Brauzerga hujjat HTML5 standartida yozilganini bildiradi
<html lang="uz">	Ildiz element; lang atributi sahifa tilini ko'rsatadi (qidiruv tizimlari va ekran o'quvchilari uchun muhim)
<head>	Sahifa haqidagi xizmat ma'lumotlari: kodirovka, sarlavha, stillar, meta teglar. Ekranda ko'rinmaydi
<meta charset="UTF-8">	Belgilar kodirovkasi; o'zbek harflari (o', g') to'g'ri chiqishi uchun zarur
<meta name="viewport">	Mobil qurilmalarda sahifa kengligini ekran kengligiga moslaydi
<title>	Brauzer yorlig'ida va qidiruv natijalarida ko'rinadigan nom
<body>	Sahifaning ekranda ko'rinadigan barcha mazmuni

HTML fayllar **.html** kengaytmasi bilan saqlanadi. Saytning bosh sahifasi odatda **index.html** deb nomlanadi – server papka so'ralganda aynan shu faylni qaytaradi.

6. Asosiy teglar: sarlavhalar, matn, havolalar, rasmlar

Sarlavhalar va matn

HTML da oltita darajali sarlavha mavjud: **<h1>** (eng muhim) dan **<h6>** gacha. Sahifada bitta **<h1>** bo'lishi va darajalarni tashlab ketmaslik (h1 dan keyin h3 emas, h2) tavsiya etiladi – bu SEO va qulaylik uchun muhim.

```
<h1>Web texnologiyalari</h1>
<h2>HTML asoslari</h2>
<p>HTML - <strong>belgilash tili</strong>, u <em>dasturlash tili emas</em>.</p>
<p>Formula:  $H_{2+0} \cdot x^{2+} + y^{2+}$ </p>
<p>Birinchi qator<br>Ikkinchi qator</p>
<hr>
<blockquote cite="https://w3.org">Veb - hamma uchun.</blockquote>
<pre><code>console.log("Salom");</code></pre>
```

**** – mazmunan muhim matn (qalin), **** – urg‘u (kursiv), **<mark>** – ajratilgan matn, **<sub>/><sup>** – pastki/yuqori indeks, **<code>** – kod, **<pre>** – formatlangan matn (bo‘shliqlar saqlanadi), **<blockquote>** – iqtibos.

Havolalar

Havola **<a>** (anchor) tegi bilan yaratiladi; asosiy atributi **href**:

```
<a href="https://developer.mozilla.org">MDN (absolyut yo‘l)</a>
<a href="pages/about.html">Biz haqimizda (nisbiy yo‘l)</a>
<a href="../index.html">Bosh sahifaga (bir daraja yuqori)</a>
<a href="#aloqa">Sahifadagi "Aloqa" bo‘limiga</a>
<a href="mailto:info@namdpi.uz">Xat yozish</a>
<a href="tel:+998692270000">Qo‘ng‘iroq qilish</a>
<a href="files/dastur.pdf" download>Dasturni yuklab olish</a>
<a href="https://namdpi.uz" target="_blank" rel="noopener">Yangi oynada</a>
```

Absolyut yo‘l to‘liq URL ni ko‘rsatadi, **nisbiy yo‘l** joriy faylga nisbatan joylashuvni bildiradi. **target="_blank"** havolani yangi oynada ochadi; xavfsizlik uchun unga **rel="noopener"** qo‘shiladi.

Rasmlar

```


<figure>
  
  <figcaption>1-rasm. O‘quvchilar natijalari</figcaption>
</figure>
```

alt atributi majburiy: rasm yuklanmasa, uning o‘rnida matn chiqadi; ko‘zi ojiz foydalanuvchilarning ekran o‘quvchi dasturlari aynan shu matnni o‘qiydi; qidiruv tizimlari rasm mazmunini shu orqali biladi. Veb uchun asosiy rasm formatlari: JPEG (fotosuratlar), PNG (shaffoflik), SVG (vektor, logotiplar), WebP va AVIF (zamonaviy, kichik hajmli).

7. Ro'yxatlar va jadvallar

Ro'yxatlar

```
<ul>                                <!-- tartiblanmagan (markerli) -->
  <li>HTML</li>
  <li>CSS</li>
  <li>JavaScript
    <ol type="a">                    <!-- ichma-ich tartiblangan ro'yxat -->
      <li>O'zgaruvchilar</li>
      <li>Funksiyalar</li>
    </ol>
  </li>
</ul>

<ol start="5" reversed> ... </ol>  <!-- 5 dan boshlab teskari tartibda -->

<dl>                                <!-- ta'riflar ro'yxati -->
  <dt>HTML</dt>  <dd>Belgilash tili</dd>
  <dt>CSS</dt>   <dd>Stillar jadvali tili</dd>
</dl>
```

Jadvallar

Jadvallar faqat jadval ko'rinishidagi ma'lumotlar uchun ishlatiladi (sahifa maketini jadval bilan qurish eskirgan usul).

```
<table>
  <caption>9-"A" sinf informatika baholari</caption>
  <thead>
    <tr><th>№</th><th>F.I.Sh.</th><th>1-chorak</th><th>2-chorak</th></tr>
  </thead>
  <tbody>
    <tr><td>1</td><td>Aliyev Jasur</td><td>5</td><td>5</td></tr>
    <tr><td>2</td><td>Karimova Malika</td><td>4</td><td>5</td></tr>
  </tbody>
  <tfoot>
    <tr><td colspan="2">O'rtacha</td><td>4,5</td><td>5,0</td></tr>
  </tfoot>
</table>
```

<tr> – qator, **<th>** – sarlavha katakchasi, **<td>** – oddiy katakcha. **colspan** katakchani gorizontaal, **rowspan** vertikal birlashtiradi. **<thead>**, **<tbody>**, **<tfoot>** jadval qismlarini mantiqiy ajratadi.

8. Semantik HTML5 teglari

Semantika – elementning ma'nosi. HTML5 gacha sahifa qismlari **<div id="header">**, **<div class="menu">** kabi nomsiz bloklar bilan qurilgan. HTML5 sahifa qismlarining ma'nosini bildiruvchi maxsus teglarni kiritdi:

Teg	Vazifasi
<header>	Sahifa yoki bo'limning sarlavha qismi (logotip, nom, qidiruv)
<nav>	Asosiy navigatsiya havolalari bloki (menyu)

Teg	Vazifasi
<code><main></code>	Sahifaning asosiy, noyob mazmuni (sahifada bitta bo‘ladi)
<code><section></code>	Mavzu bo‘yicha guruhlangan bo‘lim (odatda sarlavhaga ega)
<code><article></code>	Mustaqil, alohida tarqatilishi mumkin bo‘lgan mazmun: yangilik, maqola, izoh
<code><aside></code>	Asosiy mazmunga bilvosita aloqador qism: yon panel, reklama, foydali havolalar
<code><footer></code>	Pastki qism: mualliflik huquqi, aloqa ma’lumotlari
<code><figure></code> , <code><figcaption></code>	Rasm, diagramma yoki kod va uning izohi
<code><time></code>	Sana va vaqt: <code><time datetime="2026-09-01">1-sentabr</time></code>

```

<body>
  <header>
    <h1>Maktab yangiliklari</h1>
    <nav>
      <a href="index.html">Bosh sahifa</a>
      <a href="news.html">Yangiliklar</a>
      <a href="contact.html">Aloqa</a>
    </nav>
  </header>
  <main>
    <article>
      <h2>Informatika olimpiadasi g‘oliblari</h2>
      <p><time datetime="2026-10-10">10-oktabr</time> kuni...</p>
    </article>
    <section>
      <h2>E’lonlar</h2> ...
    </section>
  </main>
  <aside>Foydali havolalar</aside>
  <footer>&copy; 2026 Maktab №5</footer>
</body>

```

Semantik teglarning afzalliklari: kodni o‘qish va qo‘llab-quvvatlash osonlashadi; qidiruv tizimlari mazmunni yaxshiroq tushunadi (SEO); ekran o‘quvchi dasturlar sahifa bo‘ylab qulay harakatlanadi (accessibility); CSS stillarini yozish soddalashadi.

`<div>` (blokli) va `<span>` (qatordagi) – semantik ma’noga ega bo‘lmagan universal konteynerlar; ular faqat mos semantik teg bo‘lmagan hollarda, asosan stil berish uchun ishlatiladi.

Blokli va qatordagi elementlar

Blokli (block) elementlar yangi qatordan boshlanadi va butun kenglikni egallaydi: `<div>`, `<p>`, `<h1>`–`<h6>`, `<ul>`, `<table>`, `<section>`. **Qatordagi (inline)** elementlar matn oqimida joylashadi va faqat o‘z mazmuni kengligini egallaydi: `<span>`, `<a>`, `<strong>`, `<em>`, `<img>`.

9. Multimedia elementlari va formalar

Audio va video

HTML5 gacha audio va video uchun Flash kabi qo‘shimcha pluginlar talab qilingan. HTML5 da ular brauzerning o‘zida qo‘llab-quvvatlanadi:

```
<audio controls>
  <source src="media/madhiya.mp3" type="audio/mpeg">
  <source src="media/madhiya.ogg" type="audio/ogg">
  Brauzeringiz audio elementini qo‘llab-quvvatlamaydi.
</audio>

<video width="640" height="360" controls poster="media/muqova.jpg" muted>
  <source src="media/dars.mp4" type="video/mp4">
  <source src="media/dars.webm" type="video/webm">
  <track src="media/subtitr_uz.vtt" kind="subtitles" srclang="uz"
label="O‘zbekcha">
</video>

<iframe width="560" height="315" src="https://www.youtube.com/embed/VIDEO_ID"
title="Video dars" allowfullscreen></iframe>
```

Asosiy atributlar: **controls** – boshqaruv paneli, **autoplay** – avtomatik ijro (ko‘pchilik brauzerlarda faqat **muted** bilan ishlaydi), **loop** – takrorlash, **muted** – ovozsiz, **poster** – video yuklanguncha ko‘rsatiladigan rasm, **<track>** – subtitrlar.

Formalar

Formalar foydalanuvchidan ma’lumot qabul qilish uchun ishlatiladi:

```
<form action="/register" method="POST">
  <fieldset>
    <legend>To‘garakka ro‘yxatdan o‘tish</legend>

    <label for="ism">Ism:</label>
    <input type="text" id="ism" name="ism" required minlength="2">

    <label for="email">E-pochta:</label>
    <input type="email" id="email" name="email" placeholder="ism@mail.uz">

    <label for="yosh">Yosh:</label>
    <input type="number" id="yosh" name="yosh" min="7" max="18">

    <p>Sinf:
      <label><input type="radio" name="smena" value="1"> 1-smena</label>
      <label><input type="radio" name="smena" value="2"> 2-smena</label>
    </p>

    <label for="yonalish">Yo‘nalish:</label>
    <select id="yonalish" name="yonalish">
      <option value="web">Veb-dasturlash</option>
      <option value="python">Python</option>
    </select>

    <label for="izoh">Izoh:</label>
```

```
<textarea id="izoh" name="izoh" rows="4"></textarea>

<label><input type="checkbox" name="rozilik" required> Shartlarga
roziman</label>
<button type="submit">Yuborish</button>
</fieldset>
</form>
```

HTML5 da yangi **input** turlari qo‘shilgan: **email**, **tel**, **url**, **number**, **range**, **date**, **time**, **color**, **search**, **file**. Mobil qurilmalarda ular mos klaviaturani chiqaradi. Validatsiya atributlari (**required**, **min**, **max**, **minlength**, **maxlength**, **pattern**) ma’lumotni yuborishdan oldin brauzerning o‘zida tekshiradi. **<label for="...">** maydon bilan bog‘lanadi va qulaylikni oshiradi.

Muhim: brauzerdagi validatsiya faqat foydalanuvchiga qulaylik uchun. Ma’lumot server tomonida ham albatta tekshirilishi kerak, chunki brauzer tekshiruvini osongina chetlab o‘tish mumkin (M5 mavzusiga qarang).

Mavzu bo‘yicha xulosa

Internet – global tarmoq infratuzilmasi, veb esa uning ustida ishlovchi gipermatnli xizmatdir. Veb mijoz-server arxitekturasida, HTTP/HTTPS protokollari orqali ishlaydi; brauzer HTML va CSS dan DOM va CSSOM daraxtlarini qurib, sahifani chizadi. HTML veb-sahifaning skeletini tashkil etadi: to‘g‘ri tuzilgan HTML5 hujjati **<!DOCTYPE html>**, **<head>** va **<body>** qismlariga ega, semantik teglar sahifa tuzilishini mazmunli qiladi, multimedia elementlari esa pluginlarsiz audio va video joylashtirish imkonini beradi. Yozilgan kodni W3C validatori (validator.w3.org) orqali tekshirish tavsiya etiladi.

Nazorat savollari

1. Internet va World Wide Web o‘rtasidagi farq nimada?
2. URL qanday qismlardan iborat? DNS qanday vazifani bajaradi?
3. Mijoz-server arxitekturasining ishlash tartibini tushuntiring.
4. GET va POST metodlarining farqi nimada? 200, 404 va 500 status kodlari nimani bildiradi?
5. HTTPS qanday xavfsizlik kafolatlarini beradi?
6. Brauzerda sahifani chizish bosqichlarini sanab bering.
7. HTML5 hujjatining minimal tuzilishini yozing. **<meta name="viewport">** nima uchun kerak?
8. Nisbiy va absolyut yo‘lning farqini misolda tushuntiring. **alt** atributining ahamiyati nimada?
9. Semantik teglarni sanang. **<section>** va **<article>** ning farqi nimada?
10. HTML5 formalarida qanday maydon turlari va validatsiya atributlari bor?

Adabiyotlar: [1], [3], [9]

2-MAVZU. CSS ASOSLARI: SELEKTORLAR, BOX MODEL, FLEXBOX VA GRID. RESPONSIVE DIZAYN VA BOOTSTRAP FREYMVORKI

Reja:

1. CSS sintaksisi va vazifasi.
2. Selektor turlari.
3. Ranglar, fon va shriftlar.
4. Box modeli.
5. Display xossalari, Flexbox, Grid va pozitsiyalash.
6. Responsive dizayn tamoyillari. Bootstrap asoslari.
7. Bootstrap grid tizimi va komponentlari.

Tayanch iboralar:

CSS, qoida, selektor, xossa, kaskad, spetsifiklik, meros, psevdosinf, RGB, HEX, HSL, box modeli, margin, padding, border, **box-sizing**, display, position, Flexbox, Grid, **fr**, responsive dizayn, mobile-first, media query, breakpoint, freymvork, Bootstrap, CDN, container, row, col, utility sinflar, komponent.

1. CSS sintaksisi va vazifasi

CSS (Cascading Style Sheets – kaskadli stillar jadvali) – HTML hujjatining tashqi ko‘rinishini (ranglar, shriftlar, o‘lchamlar, joylashuv, animatsiya) tavsiflovchi til. CSS mazmun (HTML) va ko‘rinishni (dizayn) ajratish imkonini beradi: bitta CSS fayl o‘nlab sahifalarning dizaynini boshqaradi, dizaynni o‘zgartirish uchun HTML ni tahrirlash talab etilmaydi.

CSS **qoidalar** to‘plamidan iborat. Har bir qoida **selektor** va **e‘lonlar blokidan** tashkil topadi:

```
selektor {
  xossa: qiymat;
  xossa: qiymat;
}

h1 {
  color: #2F5496;           /* matn rangi */
  font-size: 32px;         /* shrift o'lchami */
  text-align: center;      /* tekislash */
}
```

CSS ni HTML ga ulash usullari

Usul	Yozilishi	Qo‘llanilishi
Ichki (inline)	<code><p style="color: red;">Matn</p></code>	Bir elementga tezkor stil; tavsiya etilmaydi
Ichiga o‘rnatilgan (internal)	<code><head> ichida <style> p { color: red; } </style></code>	Bitta sahifa uchun

Usul	Yozilishi	Qo'llanilishi
Tashqi (external)	<code><link rel="stylesheet" href="css/style.css"></code>	Butun sayt uchun; eng to'g'ri usul

Tashqi CSS fayl brauzer tomonidan keshlanadi, shuning uchun sahifalar tezroq yuklanadi.

2. Selektor turlari

Selektor	Misol	Nimani tanlaydi
Universal	<code>* { }</code>	Barcha elementlarni
Teg (tip)	<code>p { }</code>	Barcha <code><p></code> elementlarni
Sinf (class)	<code>.karta { }</code>	<code>class="karta"</code> bo'lgan elementlarni
Identifikator (id)	<code>#asosiy { }</code>	<code>id="asosiy"</code> bo'lgan yagona elementni
Guruh	<code>h1, h2, h3 { }</code>	Sanab o'tilgan barcha elementlarni
Avlod (descendant)	<code>nav a { }</code>	<code>nav</code> ichidagi barcha <code>a</code> larni
Bevosita farzand	<code>ul > li { }</code>	<code>ul</code> ning bevosita farzandi bo'lgan <code>li</code> larni
Qo'shni	<code>h2 + p { }</code>	<code>h2</code> dan keyin kelgan birinchi <code>p</code> ni
Atribut	<code>input[type="email"] { }</code>	Atribut qiymati mos elementlarni
Psevdo-sinf	<code>a:hover, li:nth-child(2n), input:focus</code>	Holat yoki tartib bo'yicha
Psevdo-element	<code>p::first-line, .belgi::before</code>	Elementning qismini yoki qo'shimcha mazmunni

```
.menu a:hover { color: orange; } /* sichqoncha ustida */
tr:nth-child(even) { background: #f2f2f2; } /* juft qatorlar */
input:invalid { border-color: red; } /* noto'g'ri kiritilgan maydon */
.muhim::before { content: "⚠"; } /* oldiga belgi qo'shish */
```

Kaskad, spetsifiklik va meros

Bitta elementga bir nechta qoida mos kelsa, qaysi biri qo'llanishini **kaskad** qoidalari belgilaydi:

- **muhimlik** – `!important` belgisi bo'lgan e'lon ustun (faqat zarurat bo'lganda ishlatiladi);
- **spetsifiklik (aniqlik darajasi)** – inline stil (1,0,0,0) > id (0,1,0,0) > sinf, atribut, psevdo-sinf (0,0,1,0) > teg, psevdo-element (0,0,0,1);
- **tartib** – spetsifikligi teng bo'lsa, keyin yozilgan qoida ustun.

Masalan, `#menu .link` selektorining spetsifikligi (0,1,1,0), `nav a` niki esa (0,0,0,2) – birinchisi ustun.

Meros (inheritance) – ba’zi xossalar (`color`, `font-family`, `font-size`, `line-height`) ota elementdan farzand elementlarga avtomatik o’tadi. `margin`, `padding`, `border`, `background` esa meros bo‘lib o‘tmaydi.

3. Ranglar, fon va shriftlar

Rang berish usullari

```
color: red; /* nomi bilan (140 dan ortiq nom) */
color: #2F5496; /* HEX: #RRGGBB */
color: rgb(47, 84, 150); /* RGB: 0-255 */
color: rgba(47, 84, 150, 0.5); /* RGBA: shaffoflik 0-1 */
color: hsl(218, 52%, 39%); /* HSL: ton, to‘yinganlik, yorqinlik */
```

Fon xossalari

```
.banner {
  background-color: #EAF1FB;
  background-image: url("../images/fon.jpg");
  background-size: cover; /* butun maydonni qoplash */
  background-position: center;
  background-repeat: no-repeat;
}
.tugma {
  background: linear-gradient(90deg, #2F5496, #548235); /* gradient */
}
```

Shrift va matn xossalari

```
body {
  font-family: "Roboto", Arial, sans-serif; /* zaxira shriftlar bilan */
  font-size: 16px;
  line-height: 1.5; /* qatorlar oralig‘i */
  color: #333;
}
h1 { font-weight: 700; text-transform: uppercase; letter-spacing: 1px; }
a { text-decoration: none; }
```

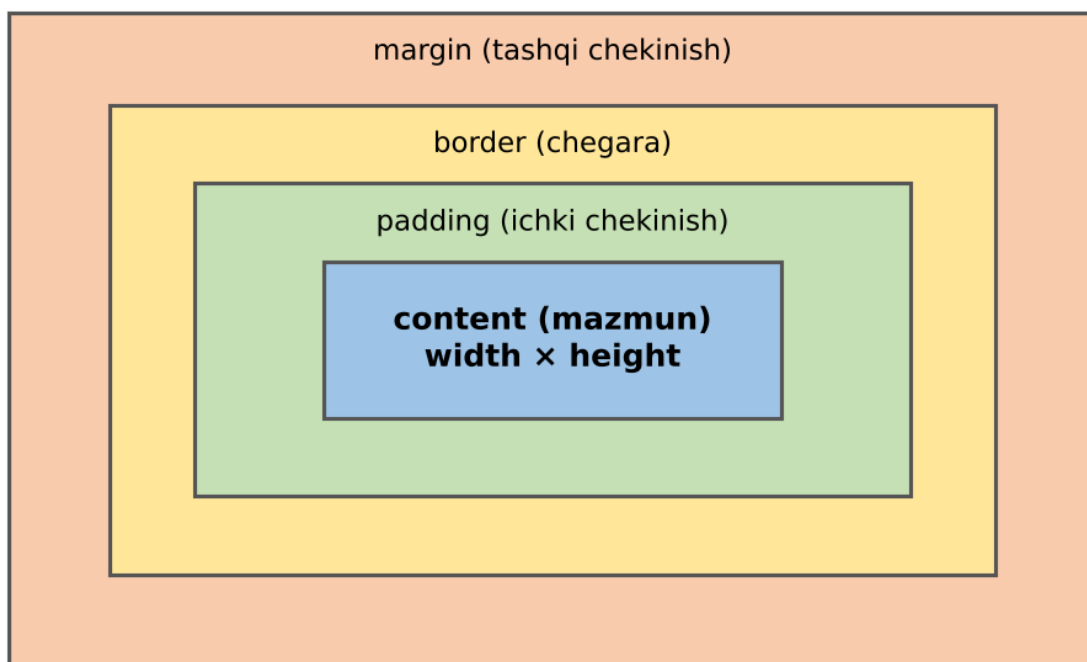
O‘lchov birliklari: `px` – piksel (absolyut); `%` – ota elementga nisbatan; `em` – ota elementning shrift o‘lchamiga nisbatan; `rem` – ildiz (`html`) elementning shrift o‘lchamiga nisbatan; `vw/vh` – brauzer oynasi kengligi/balandligining 1% i. Moslashuvchan dizaynda `rem`, `%`, `vw` birliklaridan foydalanish tavsiya etiladi.

Google Fonts xizmatidan shrift ulash:

```
<link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap"
rel="stylesheet">
```

4. Box modeli

CSS da har bir element to‘g‘ri to‘rtburchak – **quti (box)** sifatida qoraladi. Box modeli to‘rt qatlamdan iborat (2-rasm):



2-rasm. CSS box modeli

- **content** – mazmun (matn, rasm), o‘lchami **width** va **height** bilan beriladi;
- **padding** – mazmun va chegara orasidagi ichki chekinish (fon rangi bilan bo‘yaladi);
- **border** – chegara chizig‘i;
- **margin** – element va qo‘shni elementlar orasidagi tashqi chekinish (shaffof).

```
.karta {  
  width: 300px;  
  padding: 20px;           /* to‘rt tomondan */  
  border: 2px solid #2F5496;  
  margin: 10px auto;       /* yuqori-past 10px, chap-o‘ng avtomatik  
(markazlash) */  
  border-radius: 12px;     /* burchaklarni yumaloqlash */  
  box-shadow: 0 4px 12px rgba(0,0,0,0.15);  
}
```

Qisqa yozuv tartibi soat mili bo‘yicha: **margin: 10px 20px 30px 40px;** → yuqori 10, o‘ng 20, past 30, chap 40.

Elementning umumiy kengligi. Standart holatda (**box-sizing: content-box**) **width** faqat mazmun kengligini bildiradi. Yuqoridagi **.karta** ning ekrandagi kengligi: $300 + 2 \cdot 20 + 2 \cdot 2 = 344 \text{ px}$. Hisob-kitobni soddalashtirish uchun zamonaviy loyihalarda quyidagi qoida qo‘llanadi:

```
*, ::before, ::after { box-sizing: border-box; }
```

Bunda **width** padding va border ni ham o‘z ichiga oladi: **.karta** aynan 300 px egallaydi.

Margin birlashishi (margin collapse): ikki blokli element vertikal joylashganda ularning margin’lari qo‘shilmaydi, kattasi olinadi (masalan, 20px va 30px → 30px).

5. Display xossalari, Flexbox, Grid va pozitsiyalash

display xossasi

Qiymat	Xususiyati
block	Yangi qatordan boshlanadi, butun kenglikni egallaydi; width/height ishlaydi
inline	Matn oqimida joylashadi; width/height va vertikal margin ishlamaydi
inline-block	Qatorda joylashadi, lekin width/height qabul qiladi (tugmalar uchun qulay)
none	Element ko‘rinmaydi va joy egallamaydi (visibility: hidden dan farqli)
flex	Bir o‘lchovli moslashuvchan konteyner (qator yoki ustun)
grid	Ikki o‘lchovli to‘r (qator va ustunlar)

Flexbox va Grid – qisqacha

```
.menu {
  display: flex;
  justify-content: space-between; /* asosiy o‘q bo‘yicha taqsimlash */
  align-items: center;           /* ko‘ndalang o‘q bo‘yicha tekislash */
  gap: 16px;                     /* elementlar oralig‘i */
  flex-wrap: wrap;               /* sig‘masa keyingi qatorga */
}
.galereya {
  display: grid;
  grid-template-columns: repeat(3, 1fr); /* 3 ta teng ustun */
  gap: 20px;
}
```

Flexbox elementlarni bitta yo‘nalishda (navigatsiya paneli, tugmalar qatori, kartani markazlash) joylashtirish uchun, **Grid** esa butun sahifa maketi va galereyalar uchun qulay. Ular quyida va A2 amaliy mashg‘ulotida batafsil o‘rganiladi.

position xossasi

Qiymat	Xususiyati	Qo‘llanilishi
static	Standart; oddiy hujjat oqimida	—
relative	O‘z joyiga nisbatan siljiydi, joyi saqlanadi	absolute farzandlar uchun tayanch

Qiymat	Xususiyati	Qo'llanilishi
absolute	Eng yaqin pozitsiyalangan otaga nisbatan; oqimdan chiqadi	Belgi (badge), ochiluvchi menyu
fixed	Brauzer oynasiga nisbatan; aylantirganda joyida qoladi	“Yuqoriga” tugmasi, chat belgisi
sticky	Aylantirilganda belgilangan joyga “yopishib” qoladi	Yopishqoq menyu, jadval sarlavhasi

```
.karta { position: relative; }
.karta .belgi {
  position: absolute;
  top: 10px; right: 10px;      /* kartaning yuqori o'ng burchagidan */
  background: crimson; color: white; padding: 2px 8px; border-radius: 8px;
}
header { position: sticky; top: 0; z-index: 100; }
```

z-index – bir-birining ustiga tushgan pozitsiyalangan elementlarning qatlam tartibi (kattaroq qiymat ustda).

6. Responsive dizayn tamoyillari. Bootstrap asoslari

Responsive (moslashuvchan) veb-dizayn – sahifa maketining qurilma ekrani o'lchamiga (smartfon, planshet, noutbuk, katta monitor) avtomatik moslashishi. Bu atamani 2010-yilda Ethan Marcotte taklif qilgan. Responsive dizaynning uchta asosiy tarkibiy qismi:

- **moslashuvchan to'r (fluid grid)** – o'lchamlar piksel o'rniga foiz va nisbiy birliklarda beriladi;
- **moslashuvchan rasmlar** – **max-width: 100%; height: auto;**;
- **media so'rovlar (media queries)** – ekran kengligiga qarab turli stillar qo'llash.

```
/* mobile-first: avval kichik ekran uchun yoziladi */
.kartalar { display: grid; grid-template-columns: 1fr; gap: 16px; }

@media (min-width: 768px) {          /* planshet va kattaroq */
  .kartalar { grid-template-columns: repeat(2, 1fr); }
}
@media (min-width: 1200px) {        /* katta ekranlar */
  .kartalar { grid-template-columns: repeat(4, 1fr); }
}
```

Mobile-first yondashuvda stillar avval eng kichik ekran uchun yoziladi, so'ngra **min-width** media so'rovlari bilan kattaroq ekranlar uchun kengaytiriladi.

CSS freymvorklari

Freymvork – tez-tez uchraydigan vazifalarni hal qilish uchun tayyor kod, qoidalar va komponentlar to'plami. CSS freymvorklari dizaynni noldan yozmasdan,

tayyor sinflar yordamida tez va bir xil uslubda sahifa yaratish imkonini beradi. Mashhur CSS freymvorklari: **Bootstrap**, Tailwind CSS, Bulma, Foundation, Materialize.

Bootstrap – 2011-yilda Twitter kompaniyasi dasturchilari Mark Otto va Jacob Thornton tomonidan yaratilgan, dunyodagi eng mashhur ochiq kodli frontend freymvork. Joriy asosiy versiyasi – Bootstrap 5 (jQuery ga bog‘liq emas). Uning tarkibida: 12 ustunli moslashuvchan grid tizimi; tayyor komponentlar (tugmalar, navigatsiya, kartalar, modal oynalar, karusel, formalar); yordamchi (utility) sinflar; JavaScript plaginlari.

Bootstrap ni ulash (CDN orqali)

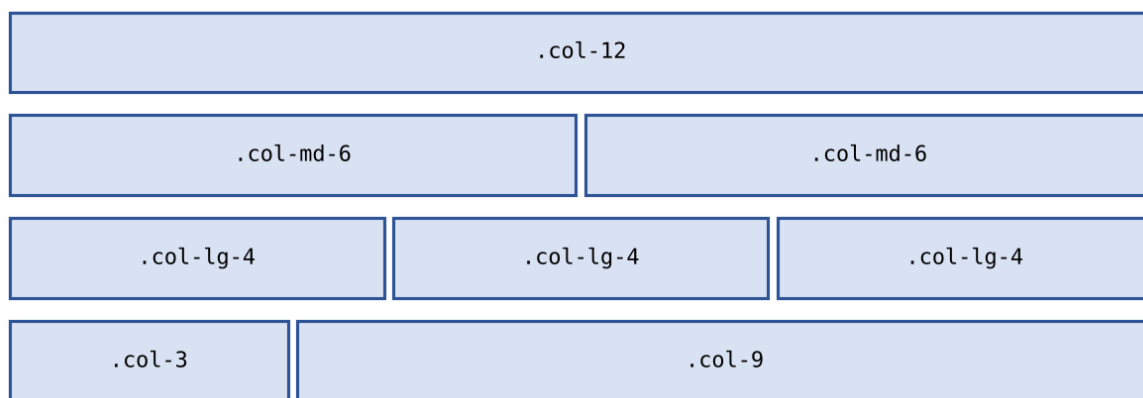
```
<!DOCTYPE html>
<html lang="uz">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Bootstrap sahifa</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
  <h1 class="text-center text-primary mt-4">Salom, Bootstrap!</h1>

  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js">
  </script>
</body>
</html>
```

CDN (Content Delivery Network) – fayllarni foydalanuvchiga geografik jihatdan eng yaqin serverdan yetkazib beruvchi tarmoq.

7. Bootstrap grid tizimi va komponentlari

Bootstrap grid tizimi Flexbox asosida qurilgan va uch darajali tuzilishga ega: **container** (konteyner) → **row** (qator) → **col** (ustun). Har bir qator **12 ta** ustunga bo‘linadi (3-rasm).



3-rasm. Bootstrap 12 ustunli grid tizimi

Prefiks	Ekran kengligi	Qurilma	Misol
col-	< 576 px	Smartfon (barcha)	col-12
col-sm-	≥ 576 px	Katta smartfon	col-sm-6
col-md-	≥ 768 px	Planshet	col-md-4
col-lg-	≥ 992 px	Noutbuk	col-lg-3
col-xl-	≥ 1200 px	Kompyuter	col-xl-2
col-xxl-	≥ 1400 px	Katta monitor	col-xxl-2

```
<div class="container">
  <div class="row g-4">
    <!-- telefonda 1 ta, planshetda 2 ta, noutbukda 3 ta ustun -->
    <div class="col-12 col-md-6 col-lg-4">1-blok</div>
    <div class="col-12 col-md-6 col-lg-4">2-blok</div>
    <div class="col-12 col-md-12 col-lg-4">3-blok</div>
  </div>
</div>
```

Yordamchi (utility) sinflar

Bootstrap stillarni bevosita HTML da sinflar orqali berish imkonini beradi: **m-3**, **mt-4**, **px-2** (margin/padding: 0–5), **text-center**, **text-primary**, **bg-light**, **fw-bold**, **d-flex**, **justify-content-between**, **align-items-center**, **d-none** **d-md-block** (telefonda yashirish), **rounded**, **shadow**, **w-100**, **img-fluid**.

Asosiy komponentlar

```
<!-- Navigatsiya paneli (telefonda "gamburger" menyuga aylanadi) -->
<nav class="navbar navbar-expand-lg navbar-dark bg-primary">
  <div class="container">
    <a class="navbar-brand" href="#">Maktab №5</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
      data-bs-target="#menyu"><span class="navbar-toggler-
icon"></span></button>
    <div class="collapse navbar-collapse" id="menyu">
      <ul class="navbar-nav ms-auto">
        <li class="nav-item"><a class="nav-link active" href="#">Bosh
sahifa</a></li>
```

```

        <li class="nav-item"><a class="nav-link"
href="#togaraklar">To'garaklar</a></li>
    </ul>
</div>
</div>
</nav>

<!-- Karta -->
<div class="card shadow-sm" style="max-width: 20rem;">
    
    <div class="card-body">
        <h5 class="card-title">Veb-dasturlash</h5>
        <p class="card-text">HTML, CSS va JavaScript ni o'rganamiz.</p>
        <button class="btn btn-success" data-bs-toggle="modal" data-bs-
target="#yozilish">
            Yozilish</button>
    </div>
</div>

<!-- Modal oyna -->
<div class="modal fade" id="yozilish" tabindex="-1">
    <div class="modal-dialog"><div class="modal-content">
        <div class="modal-header"><h5 class="modal-title">Ro'yxatdan o'tish</h5>
        <button type="button" class="btn-close" data-bs-
dismiss="modal"></button></div>
        <div class="modal-body">Ismingizni kiriting...</div>
    </div></div>
</div>

```

Tugma uslublari: **btn-primary**, **btn-secondary**, **btn-success**, **btn-danger**, **btn-warning**, **btn-outline-primary**; o'lchamlari: **btn-sm**, **btn-lg**. Boshqa komponentlar: **alert**, **badge**, **carousel**, **accordion**, **table** **table-striped**, **form-control**, **pagination**, **spinner**.

Bootstrap ning afzalliklari: tez prototip yaratish, kafolatlangan moslashuvchanlik, brauzerlararo moslik, keng hujjatlar (getbootstrap.com). Kamchiliklari: saytlar bir-biriga o'xshab qolishi, ishlatilmagan kod hisobiga fayl hajmining ortishi.

Mavzu bo'yicha xulosa

CSS veb-sahifaga dizayn beradi va HTML dan alohida faylda saqlanishi kerak. Selektorlar stil qo'llanadigan elementlarni tanlaydi, kaskad va spetsifiklik ziddiyatlarni hal qiladi. Box modeli elementning o'lchamini, **display**, Flexbox, Grid va **position** esa uning joylashuvini belgilaydi. Media so'rovlar va mobile-first yondashuv sahifani istalgan qurilmaga moslashtiradi, Bootstrap esa tayyor grid tizimi va komponentlar yordamida moslashuvchan sahifalarni tez yaratish imkonini beradi.

Nazorat savollari

1. CSS ning vazifasi nima va u HTML ga qanday usullarda ulanadi?
2. Sinf va identifikator selektorlarining farqi nimada? Spetsifiklik qanday hisoblanadi?

3. Box modeli qatlamlarini tushuntiring.
4. `width: 200px; padding: 15px; border: 5px solid;` bo'lgan elementning `content-box` va `border-box` holatlaridagi kengligi qancha?
5. `absolute`, `fixed` va `sticky` pozitsiyalashning farqlarini ayting.
6. Flexbox va Grid ning farqi nimada? Qaysi holatda qaysi birini tanlash kerak?
7. Responsive dizaynning uchta asosiy tarkibiy qismini ayting. Mobile-first nima?
8. Bootstrap grid tizimi qanday tuzilgan? `col-12` `col-md-6` `col-lg-4` qanday natija beradi?
9. Bootstrap ning qaysi komponentlarini bilasiz?
10. Bootstrap ning afzallik va kamchiliklari nimada?

Adabiyotlar: [3], [5], [11]

3-MAVZU. JAVASCRIPT ASOSLARI VA DOM BILAN ISHLASH. HODISALAR (EVENTS). FETCH API VA JSON

Reja:

1. JavaScript tili. O‘zgaruvchilar va ma’lumot turlari.
2. Operatorlar, shartli operatorlar va tsikllar.
3. Funksiyalar, massivlar va obyektlar.
4. Document Object Model (DOM).
5. Hodisalar (events) bilan ishlash.
6. JSON formati va Fetch API.

Tayanch iboralar:

JavaScript, ECMAScript, **let**, **const**, ma’lumot turi, operator, qat’iy tenglik, **if**, **for**, funksiya, strelkali funksiya, massiv, obyekt, DOM, **querySelector**, **textContent**, **innerHTML**, **classList**, hodisa, **addEventListener**, **preventDefault**, JSON, **JSON.parse**, **JSON.stringify**, Fetch API, Promise, **async/await**, **try...catch**, CORS.

1. JavaScript tili. O‘zgaruvchilar va ma’lumot turlari

JavaScript (JS) – veb-sahifalarga interaktivlik beruvchi, brauzerda bajariladigan yuqori darajali, dinamik turlanuvchi dasturlash tili. JavaScript 1995-yilda Brendan Eich tomonidan 10 kunda yaratilgan. Tilning standarti **ECMAScript** deb ataladi; 2015-yildagi ES6 (ES2015) versiyasidan boshlab har yili yangi versiya chiqariladi. Bugungi kunda JavaScript nafaqat brauzerda, balki serverda (Node.js), mobil va desktop ilovalarda ham ishlatiladi.

Veb-sahifadagi uch texnologiyaning vazifasi: **HTML** – mazmun va tuzilish (“skelet”), **CSS** – ko‘rinish (“kiyim”), **JavaScript** – xatti-harakat (“muskullar”).

JavaScript ni sahifaga ulash

```
<!-- tashqi fayl (tavsiya etiladi) -->
<script src="js/app.js" defer></script>

<!-- sahifa ichida -->
<script>
  console.log("Salom, dunyo!");
</script>
```

defer atributi skriptni HTML to‘liq tahlil qilingandan keyin bajarishni bildiradi – shunda skript DOM elementlariga xatosiz murojaat qila oladi. Natijalarni tekshirish uchun brauzer konsoli (F12 → Console) va **console.log()** ishlatiladi.

O'zgaruvchilar

```
let yosh = 25;           // qiymati o'zgarishi mumkin
const PI = 3.14159;      // o'zgarmas (qayta qiymat berib bo'lmaydi)
var eski = "eskirgan usul"; // ES6 gacha; zamonaviy kodda ishlatilmaydi

yosh = 26;               // to'g'ri
// PI = 3;               // xato: TypeError
```

Kalit so'z	Ko'rinish sohasi	Qayta qiymat berish	Tavsiya
var	Funksiya	Mumkin	Ishlatmaslik
let	Blok { }	Mumkin	O'zgaradigan qiymatlar uchun
const	Blok { }	Mumkin emas	Standart tanlov

Ma'lumot turlari

Tur	Tavsif	Misol
string	Satr	"Salom", 'NamDPI', shablon satr
number	Butun va haqiqiy sonlar	42, 3.14, NaN, Infinity
bigint	Juda katta butun sonlar	9007199254740993n
boolean	Mantiqiy qiymat	true, false
undefined	Qiymat berilmagan	let x;
null	Ataylab "bo'sh" qiymat	let y = null;
symbol	Noyob identifikator	Symbol("id")
object	Murakkab tur: obyekt, massiv, funksiya	{ism: "Ali"}, [1, 2, 3]

O'zgaruvchi turini **typeof** operatori aniqlaydi: **typeof 42** → "number".
Shablon satrlar (backtick belgisi bilan) ichiga **\${...}** orqali ifoda qo'yish mumkin.

2. Operatorlar, shartli operatorlar va tsikllar

Arifmetik: + - * / % ** (% – qoldiq, ** – daraja). **Taqqoslash:** > < >= <=, == (turini o'zgartirib taqqoslaydi), === (qat'iy: qiymat va tur), !=, !==. **Mantiqiy:** && (va), || (yoki), ! (inkor).

```
console.log(5 == "5"); // true - tur avtomatik o'zgartiriladi
console.log(5 === "5"); // false - turlar har xil
console.log("5" + 3); // "53" - satrga ulanadi
console.log("5" - 3); // 2 - songa aylantiriladi
```

Qoida: har doim qat'iy taqqoslash **===** va **!==** dan foydalaning.

Shartli operatorlar

```
const ball = 78;
let baho;
if (ball >= 86) {
    baho = 5;
} else if (ball >= 71) {
    baho = 4;
} else if (ball >= 55) {
    baho = 3;
} else {
    baho = 2;
}

const holat = ball >= 55 ? "o'tdi" : "o'tmadi";    // ternar operator

switch (new Date().getDay()) {
    case 0: case 6: console.log("Dam olish kuni"); break;
    default: console.log("Ish kuni");
}
```

Tsikllar

```
for (let i = 1; i <= 5; i++) {
    console.log(`${i} x 7 = ${i * 7}`);
}

let n = 10;
while (n > 0) { n -= 3; }

const fanlar = ["HTML", "CSS", "JS"];
for (const fan of fanlar) {           // massiv elementlari bo'yicha
    console.log(fan);
}
```

break tsiklni to'xtatadi, **continue** joriy qadamni o'tkazib yuboradi.

3. Funksiyalar, massivlar va obyektlar

Funksiya – nom berilgan, qayta ishlatiladigan kod bloki. JavaScriptda funksiya bir necha usulda e'lon qilinadi:

```
// 1. Funksiya e'loni
function yuza(a, b) {
    return a * b;
}

// 2. Funksiya ifodasi
const perimetr = function (a, b) {
    return 2 * (a + b);
};

// 3. Strelkali funksiya (ES6)
const kvadrat = x => x * x;
const salom = (ism = "mehmon") => `Salom, ${ism}!`; // standart parametr

console.log(yuza(4, 5)); // 20
console.log(kvadrat(9)); // 81
console.log(salom()); // Salom, mehmon!
```

Massivlar

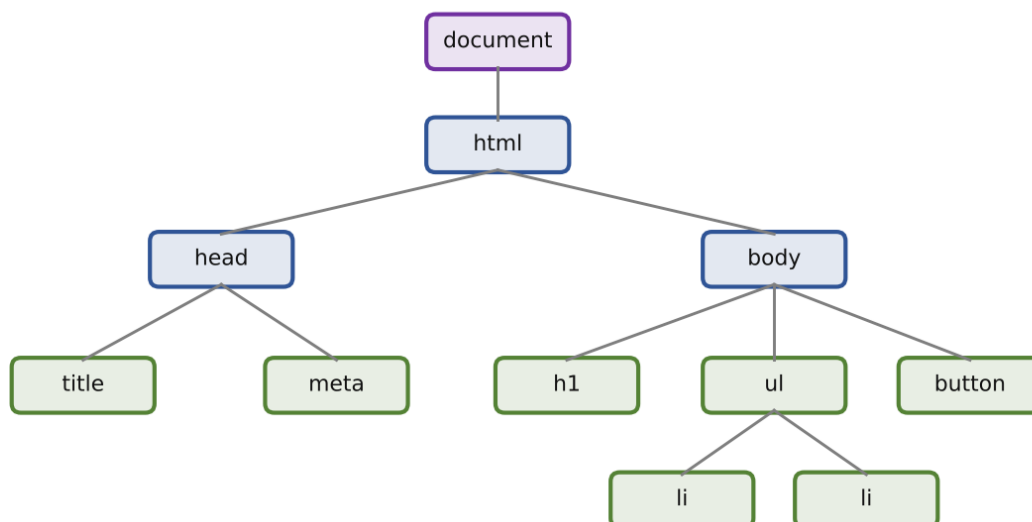
```
const baholar = [5, 4, 3, 5, 4];
baholar.push(5); // oxiriga qo'shish
console.log(baholar.length); // 6
const ikkilangan = baholar.map(b => b * 2); // har birini o'zgartirish
const alochilar = baholar.filter(b => b === 5); // saralash
const yigindi = baholar.reduce((s, b) => s + b, 0); // yig'ish
console.log((yigindi / baholar.length).toFixed(2)); // o'rtacha: 4.33
```

Obyektlar

```
const talaba = {
    ism: "Malika",
    yosh: 20,
    fanlar: ["HTML", "CSS"],
    tanishtir() {
        return `Men ${this.izm}, ${this.yosh} yoshdaman`;
    }
};
console.log(talaba.izm); // nuqta orqali murojaat
console.log(talaba["yosh"]); // kvadrat qavs orqali
talaba.guruh = "IQT-1"; // yangi xossa qo'shish
console.log(talaba.tanishtir());
```

4. Document Object Model (DOM)

DOM (Document Object Model) – brauzer HTML hujjatdan hosil qiladigan, uning har bir elementini **obyekt** sifatida ifodalovchi daraxtsimon model va ushbu modelni boshqarish uchun dasturiy interfeys (API). DOM orqali JavaScript sahifadagi istalgan elementni topishi, mazmunini, atributini, stilini o'zgartirishi, yangi element qo'shishi yoki o'chirishi mumkin.



4-rasm. HTML hujjatning DOM daraxti

DOM daraxtining ildizi **document** obyekt. Har bir element – **tugun (node)**. Tugunlar o‘rtasida ota-ona (parent), farzand (child) va aka-uka (sibling) munosabatlari mavjud.

Elementlarni tanlash

```

document.getElementById("sarlavha"); // id bo'yicha (bitta)
document.querySelector(".karta"); // CSS selektor bo'yicha birinchisi
document.querySelectorAll("ul li"); // barcha moslar (NodeList)
document.getElementsByClassName("karta"); // sinf bo'yicha (HTMLCollection)
  
```

Zamonaviy kodda universal **querySelector** va **querySelectorAll** metodlari ishlatiladi, chunki ular CSS selektorlarining barcha imkoniyatlarini qo‘llab-quvvatlaydi.

Elementlarni o‘zgartirish

```

const h1 = document.querySelector("h1");
h1.textContent = "Yangi sarlavha"; // faqat matn (xavfsiz)
h1.style.color = "crimson"; // inline stil
h1.classList.add("faol"); // sinf qo'shish
h1.classList.toggle("yashirin"); // bor bo'lsa olib tashlash, yo'q bo'lsa qo'shish

const rasm = document.querySelector("img");
rasm.setAttribute("src", "images/yangi.png");
rasm.alt = "Yangi rasm";

// yangi element yaratish va qo'shish
const li = document.createElement("li");
li.textContent = "JavaScript";
document.querySelector("#fanlar").append(li);

// elementni o'chirish
document.querySelector(".eski-elon")?.remove();
  
```

`textContent` va `innerHTML` farqi: `innerHTML` satrni HTML sifatida talqin qiladi. Agar unga foydalanuvchi kiritgan ma'lumot berilsa, XSS hujumiga yo'l ochiladi (M5). Foydalanuvchi ma'lumotlari uchun doimo `textContent` dan foydalaning.

5. Hodisalar (events) bilan ishlash

Hodisa (event) – sahifada sodir bo'ladigan voqea: tugma bosilishi, klaviatura tugmasining bosilishi, forma yuborilishi, sahifa yuklanishi va h.k. JavaScript hodisalarga **hodisa ishlovchilari** (event listener) orqali javob beradi.

Hodisa	Qachon sodir bo'ladi
<code>click</code> , <code>dblclick</code>	Sichqoncha bilan bosilganda
<code>mouseover</code> , <code>mouseout</code>	Kursor element ustiga kelganda/ketganda
<code>keydown</code> , <code>keyup</code>	Klaviatura tugmasi bosilganda/qo'yib yuborilganda
<code>input</code> , <code>change</code>	Maydon qiymati o'zgarganda
<code>submit</code>	Forma yuborilganda
<code>DOMContentLoaded</code> , <code>load</code>	DOM tayyor bo'lganda / barcha resurslar yuklanganda
<code>scroll</code> , <code>resize</code>	Sahifa aylantirilganda / oyna o'lchami o'zgarganda

Misol: hisoblagich va tun rejimi

```
<h2 id="son">0</h2>
<button id="qosh">+1</button>
<button id="tozalash">Tozalash</button>
<button id="rejim">Tun rejimi</button>

<script>
  let hisob = 0;
  const sonEl = document.querySelector("#son");

  document.querySelector("#qosh").addEventListener("click", () => {
    hisob++;
    sonEl.textContent = hisob;
  });

  document.querySelector("#tozalash").addEventListener("click", () => {
    hisob = 0;
    sonEl.textContent = hisob;
  });

  document.querySelector("#rejim").addEventListener("click", () => {
    document.body.classList.toggle("tun");
  });
</script>
```

Hodisa obyekti va formani qayta ishlash

Ishlovchi funksiyaga hodisa haqidagi ma'lumotlarni saqlovchi **hodisa obyekti** (`event` yoki `e`) uzatiladi: `e.target` – hodisa sodir bo'lgan element, `e.key` –

bosilgan tugma, `e.preventDefault()` – brauzerning standart harakatini (masalan, formani yuborib sahifani yangilashni) bekor qilish.

```
<form id="forma">
  <input id="ism" placeholder="Ismingiz">
  <button>Yuborish</button>
</form>
<p id="xabar"></p>

<script>
  document.querySelector("#forma").addEventListener("submit", (e) => {
    e.preventDefault();
    const ism = document.querySelector("#ism").value.trim();
    const xabar = document.querySelector("#xabar");
    if (ism.length < 2) {
      xabar.textContent = "Ism kamida 2 ta harfdan iborat bo'lsin";
      xabar.style.color = "red";
    } else {
      xabar.textContent = `Rahmat, ${ism}! Ma'lumot qabul qilindi.`;
      xabar.style.color = "green";
    }
  });
</script>
```

Hodisalarni delegatsiya qilish: ko'plab bir xil elementlarga alohida ishlovchi ilish o'rniga, ularning umumiy ota elementiga bitta ishlovchi ilinadi va `e.target` orqali aniq element aniqlanadi. Bu dinamik qo'shilgan elementlar uchun ham ishlaydi.

```
document.querySelector("#royxat").addEventListener("click", (e) => {
  if (e.target.matches("button.ochir")) {
    e.target.closest("li").remove();
  }
});
```

6. JSON formati va Fetch API

JSON (JavaScript Object Notation) – ma'lumot almashishning matnli, yengil formati. U JavaScript obyekt sintaksisiga asoslangan, lekin barcha dasturlash tillarida qo'llab-quvvatlanadi. JSON da kalitlar va satrlar faqat qo'sh tirnoqda yoziladi; funksiya va izohlar ruxsat etilmaydi.

```
{
  "id": 1,
  "title": "O'tkan kunlar",
  "year": 1926,
  "available": true,
  "author": { "id": 1, "name": "Abdulla Qodiriy" },
  "tags": ["roman", "tarixiy"]
}
```

JavaScriptda: `JSON.stringify(obj)` – obyektни JSON satrga, `JSON.parse(satr)` – JSON satrni obyektga aylantiradi.

Fetch API va asinxron so‘rovlar

Fetch API – brauzerda HTTP so‘rovlarini yuborish uchun zamonaviy JavaScript interfeysi. `fetch(url)` funksiyasi tarmoqqa so‘rov yuboradi va **Promise** (kelajakdagi natija va‘dasi) qaytaradi. Tarmoq so‘rovi vaqt talab qilgani sababli u **asinxron** bajariladi: javob kelguncha sahifa “qotib qolmaydi”.

```
// tashqi API dan ma'lumot olish (GET)
async function foydalanuvchilar() {
  try {
    const javob = await fetch("https://jsonplaceholder.typicode.com/users");
    if (!javob.ok) throw new Error(`HTTP xato: ${javob.status}`);
    const royxat = await javob.json(); // JSON -> JavaScript massivi
    royxat.forEach(u => console.log(u.name, u.email));
  } catch (xato) {
    console.error("So'rov bajarilmadi:", xato.message);
  }
}
foydalanuvchilar();
```

`async` kalit so‘zi funksiyani asinxron qiladi, `await` esa Promise natijasini kutadi. `javob.ok` – status kodi 200–299 oralig‘ida ekanini bildiradi; tarmoq yoki server xatolari `try...catch` bilan ushlanadi.

Ma'lumotni serverga yuborish (POST) va formani integratsiya qilish

```
<form id="kitobForma">
  <input name="title" placeholder="Kitob nomi" required>
  <input name="year" type="number" placeholder="Yili">
  <button>Qo'shish</button>
</form>
<ul id="royxat"></ul>

<script>
const royxat = document.querySelector("#royxat");

async function yuklash() {
  const javob = await fetch("/api/books"); // GET so'rov
  const kitoblar = await javob.json();
  royxat.innerHTML = "";
  kitoblar.forEach(k => {
    const li = document.createElement("li");
    li.textContent = `${k.title} (${k.year ?? "-"})`;
    royxat.append(li);
  });
}

document.querySelector("#kitobForma").addEventListener("submit", async (e) => {
  e.preventDefault();
  const malumot = Object.fromEntries(new FormData(e.target));
  const javob = await fetch("/api/books", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
  });
```

```
        body: JSON.stringify(malumot)
    });
    if (javob.ok) { e.target.reset(); yuklash(); }
    else alert((await javob.json()).error);
});

yuklash();
</script>
```

`async/await` yordamida asinxron so'rovlar ketma-ket kod ko'rinishida yoziladi.

CORS (Cross-Origin Resource Sharing). Brauzer xavfsizlik maqsadida bir domendagi sahifaning boshqa domendagi API ga so'rov yuborishini cheklaydi. Agar frontend va backend turli manzillarda bo'lsa, serverda ruxsat berish kerak: `npm install cors` va `app.use(cors({ origin: "https://mening-saytim.uz" }))`.

API larni sinash uchun **Postman** yoki VS Code ning **Thunder Client** kengaytmasidan foydalaniladi.

CORS (Cross-Origin Resource Sharing). Brauzer xavfsizlik maqsadida bir domendagi sahifaning boshqa domendagi API ga so'rov yuborishini cheklaydi. Ochiq API lar (Open-Meteo, JSONPlaceholder) CORS ga ruxsat bergani uchun ularga to'g'ridan-to'g'ri murojaat qilish mumkin. O'z serveringiz boshqa manzilda bo'lsa, serverda ruxsat beriladi (M4).

Mavzu bo'yicha xulosa

JavaScript veb-sahifaga "jon" beradi. O'zgaruvchilar `let/const` bilan e'lon qilinadi, taqqoslashda `===` ishlatiladi. Funksiyalar, massivlar va obyektlar ma'lumotlarni tartibli qayta ishlash imkonini beradi. DOM orqali JavaScript sahifa elementlarini topadi va o'zgartiradi, hodisalar orqali foydalanuvchi harakatlariga javob beradi. Fetch API va JSON yordamida sahifa tashqi API va serverlardan ma'lumot olib, sahifani qayta yuklamasdan yangilaydi.

Nazorat savollari

1. `let`, `const` va `var` ning farqi nimada?
2. `==` va `===` operatorlarining farqini misolda ko'rsating.
3. Funksiya e'lon qilishning qanday usullari bor?
4. `map`, `filter`, `reduce` metodlari nima qiladi?
5. DOM nima? `querySelector` va `querySelectorAll` farqi nimada?
6. `textContent` va `innerHTML` dan qaysi biri xavfsizroq va nima uchun?
7. `addEventListener` qanday ishlaydi? `e.preventDefault()` qachon ishlatiladi?
8. JSON formati qanday qoidalarga ega? `JSON.parse` va `JSON.stringify` nima qiladi?

9. `fetch()` nima qaytaradi? `async/await` nima uchun kerak?

10. `fetch()` yordamida POST so'rov qanday yuboriladi?

Adabiyotlar: [2], [4], [6]

4-MAVZU. BACKEND DASTURLASHGA KIRISH: NODE.JS VA EXPRESS.JS. MA'LUMOTLAR BAZASI VA REST API ASOSLARI

Reja:

1. Frontend va backend. Server tushunchasi.
2. Node.js platformasi va NPM.
3. Express.js bilan server yaratish.
4. Ma'lumotlar bazasi tushunchasi.
5. Jadvallar, kalitlar va bog'lanishlar.
6. SQL so'rovlari.
7. REST API dizayni va Express da amalga oshirish.

Tayanch iboralar:

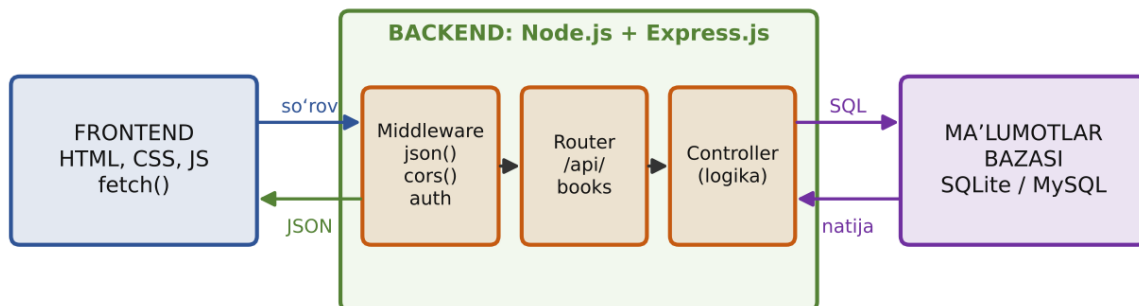
backend, server, Node.js, V8, asinxronlik, event loop, NPM, `package.json`, modul, Express.js, marshrut, `req`, `res`, middleware, `next()`, statik fayllar, ma'lumotlar bazasi, MBBT, jadval, birlamchi kalit, tashqi kalit, SQL, NoSQL, SQLite, CRUD, SELECT, INSERT, UPDATE, DELETE, JOIN, API, REST, resurs, endpoint, parametrlangan so'rov.

1. Frontend va backend. Server tushunchasi

Statik sahifalarda barcha foydalanuvchilar bir xil mazmunni ko'radi. Ammo ro'yxatdan o'tish, ma'lumot saqlash, qidiruv, to'lov, shaxsiy kabinet kabi imkoniyatlar uchun **server tomonida** ishlovchi dastur – **backend** kerak.

Mezon	Frontend	Backend
Qayerda bajariladi	Foydalanuvchi brauzerida	Serverda
Vazifasi	Interfeys, vizual ko'rinish, foydalanuvchi bilan o'zaro ta'sir	Biznes mantiq, ma'lumotlar bazasi, autentifikatsiya, API
Tillar	HTML, CSS, JavaScript	JavaScript (Node.js), PHP, Python, Java, C#, Go
Kodni ko'rish	Foydalanuvchi kodni ko'ra oladi	Kod foydalanuvchidan yashirin
Xavfsizlik	Ishonchsiz muhit	Maxfiy ma'lumotlar faqat shu yerda saqlanadi

Server so'zi ikki ma'noda ishlatiladi: 1) doimiy ishlab turuvchi kuchli kompyuter (apparat); 2) so'rovlarni ma'lum portda kutib, ularga javob qaytaruvchi dastur. Veb-dasturlashda ko'pincha ikkinchi ma'no nazarda tutiladi.



5-rasm. Veb-ilovanning frontend – backend – ma'lumotlar bazasi arxitekturası

2. Node.js platformasi va NPM

Node.js – Google Chrome brauzerining V8 JavaScript dvigateli asosida qurilgan, JavaScript kodini brauzerdan tashqarida (serverda, kompyuterda) bajaruvchi **muhit (runtime)**. 2009-yilda Ryan Dahl tomonidan yaratilgan. Node.js tufayli frontend va backendni bitta tilda – JavaScriptda yozish mumkin bo'ldi.

Node.js ning asosiy xususiyatlari:

- **asinxron, bloklanmaydigan kiritish-chiqarish** – fayl o'qish yoki bazaga so'rov kutilayotganda boshqa so'rovlar qayta ishlanadi;
- **hodisalar tsikli (event loop)** – bitta oqimda minglab ulanishlarni samarali boshqaradi;
- **ulkan ekotizim** – NPM omborida 2 milliondan ortiq paket;
- **kross-platformalilik** – Windows, Linux, macOS.

Node.js ni nodejs.org saytidan LTS (uzoq muddat qo'llab-quvvatlanadigan) versiyasini yuklab o'rnatish tavsiya etiladi. O'rnatilganini tekshirish:

```
node -v      # v20.x.x
npm -v      # 10.x.x
```

NPM (Node Package Manager)

NPM – Node.js paketlar menejeri: tayyor kutubxonalarni o'rnatish, yangilash va loyiha bog'liqliklarini boshqarish vositasi.

```
mkdir kutubxona-api && cd kutubxona-api
npm init -y                # package.json faylini yaratish
npm install express        # paketni o'rnatish (node_modules papkasiga)
npm install -D nodemon     # faqat ishlab chiqish uchun paket
npm run dev                # package.json dagi skriptni ishga tushirish
```

package.json – loyiha haqidagi ma'lumotlar (nomi, versiyasi, skriptlar, bog'liqliklar) saqlanadigan fayl:

```
{
  "name": "kutubxona-api",
  "version": "1.0.0",
  "type": "module",
  "scripts": {
    "start": "node server.js",
    "dev": "nodemon server.js"
  },
  "dependencies": { "express": "^4.21.0" },
  "devDependencies": { "nodemon": "^3.1.0" }
}
```

node_modules papkasi Git omboriga yuklanmaydi (**.gitignore** ga qo‘shiladi): uni istalgan vaqtda **npm install** buyrug‘i bilan **package.json** asosida qayta tiklash mumkin.

Node.js modullari

```
// matematika.js
export function qoshish(a, b) { return a + b; }

// app.js
import { qoshish } from "./matematika.js";
import fs from "fs"; // o‘rnatilgan modul (fayl tizimi)

console.log(qoshish(2, 3));
fs.writeFileSync("salom.txt", "Salom, Node.js!");
```

3. Express.js bilan server yaratish

Express.js – Node.js uchun eng mashhur, minimalistik veb-freymvork. U marshrutlash, middleware, so‘rov va javoblar bilan ishlashni sezilarli darajada soddalashtiradi.

Eng oddiy server

```
// server.js
import express from "express";

const app = express();
const PORT = 3000;

app.get("/", (req, res) => {
  res.send("<h1>Salom! Bu Express serveri</h1>");
});

app.listen(PORT, () => {
  console.log(`Server ishga tushdi: http://localhost:${PORT}`);
});
```

Serverni `node server.js` (yoki `npm run dev`) buyrug'i bilan ishga tushirib, brauzerda `http://localhost:3000` manzilini ochamiz. `localhost` – joriy kompyuterning o'zi (IP: 127.0.0.1), `3000` – port raqami.

Routing (marshrutlash)

Marshrut (route) – HTTP metodi va URL yo'lining kombinatsiyasi hamda unga mos ishlovchi funksiya. `req` (request) – kelgan so'rov haqidagi ma'lumotlar, `res` (response) – javob yuborish uchun obyekt.

```
app.get("/about", (req, res) => res.send("Biz haqimizda"));

// URL parametri: /talaba/15
app.get("/talaba/:id", (req, res) => {
  res.send(`Talaba ID: ${req.params.id}`);
});

// so'rov satri: /qidiruv?soz=html
app.get("/qidiruv", (req, res) => {
  res.json({ qidirilgan: req.query.soz });
});

app.post("/xabar", (req, res) => {
  res.status(201).json({ holat: "qabul qilindi", malumot: req.body });
});

// mos marshrut topilmasa - 404
app.use((req, res) => res.status(404).send("Sahifa topilmadi"));
```

Javob metodlari: `res.send()` – matn/HTML, `res.json()` – JSON, `res.status(kod)` – status kodi, `res.sendFile()` – fayl, `res.redirect()` – yo'naltirish.

Loyiha kattalashganda marshrutlar `express.Router()` yordamida alohida fayllarga ajratiladi:

```
// routes/books.js
import { Router } from "express";
const router = Router();
router.get("/", (req, res) => res.json([]));
export default router;

// server.js
import booksRouter from "./routes/books.js";
app.use("/api/books", booksRouter);
```

Middleware

Middleware (oraliq dastur) – so'rov kelgani va javob yuborilishi orasida ketma-ket bajariladigan funksiyalar zanjiri. Har bir middleware `req`, `res` va `next` parametrlarini oladi; `next()` chaqirilsa, boshqaruv keyingi middleware ga o'tadi.

```
// o'zimiz yozgan middleware: har bir so'rovni qayd qilish (logger)
app.use((req, res, next) => {
  console.log(`${new Date().toISOString()} ${req.method} ${req.url}`);
  next();
});

app.use(express.json()); // JSON tanani o'qish (req.body)
app.use(express.urlencoded({ extended: true })); // HTML forma ma'lumotlarini o'qish
```

Middleware vazifalari: so'rovlarni qayd qilish, JSON ni tahlil qilish, autentifikatsiyani tekshirish, xatolarni ushlash, CORS sozlash, so'rovlar sonini cheklash.

Statik fayllar

```
app.use(express.static("public"));
```

Shu bitta qator **public** papkasidagi **index.html**, **style.css**, **app.js**, rasmlarni to'g'ridan-to'g'ri brauzerga yetkazib beradi. Odatiy loyiha tuzilishi:

```
kutubxona-api/
├── package.json
├── server.js
├── routes/
│   └── books.js
└── public/
    ├── index.html
    ├── css/style.css
    └── js/app.js
```

4. Ma'lumotlar bazasi tushunchasi

Ma'lumotlar bazasi (MB) – ma'lum bir sohaga oid, tartiblangan holda saqlanadigan va kompyuter yordamida qayta ishlanadigan o'zaro bog'liq ma'lumotlar to'plami. MB bilan ishlashni ta'minlovchi dasturiy tizim **ma'lumotlar bazasini boshqarish tizimi (MBBT)** deyiladi.

Veb-ilovalarda ma'lumotlarni oddiy fayllarda emas, MB da saqlash zarur, chunki MBBT bir vaqtda ko'plab foydalanuvchilarning murojaatini, ma'lumotlarning yaxlitligini, tez qidiruvni, zaxira nusxa olishni va kirish huquqlarini boshqarishni ta'minlaydi.

Mezon	Relyatsion (SQL) MB	NoSQL MB
Tuzilishi	Qat'iy sxemali jadvallar (qator va ustunlar)	Hujjatlar (JSON), kalit-qiymat, graf, ustunli
So'rov tili	SQL (standart)	Har bir tizimning o'z API si
Bog'lanishlar	Kalitlar va JOIN orqali	Odatda ma'lumot hujjat ichiga joylanadi

Mezon	Relyatsion (SQL) MB	NoSQL MB
Masshtablash	Asosan vertikal (kuchliroq server)	Gorizontal (ko‘p serverlar)
Qo‘llanilishi	Moliya, ta‘lim tizimlari, buyurtmalar	Katta hajmli, tez o‘zgaruvchan ma‘lumotlar, chat, IoT
Misollar	MySQL, PostgreSQL, SQLite, MS SQL Server, Oracle	MongoDB, Redis, Firebase Firestore, Cassandra

Ushbu kursda o‘rnatishni talab qilmaydigan, butun bazani bitta faylda saqlaydigan **SQLite** dan foydalanamiz. Undagi SQL bilimlari MySQL va PostgreSQL da ham to‘liq qo‘llanadi.

5. Jadvallar, kalitlar va bog‘lanishlar

Relyatsion MB da ma‘lumotlar **jadvallar** ko‘rinishida saqlanadi. Jadvalning har bir qatori – **yozuv** (record), har bir ustuni – **maydon** (field) deyiladi. Har bir maydon ma‘lum turga ega: **INTEGER**, **REAL**, **TEXT**, **DATE**, **BOOLEAN**.

- **Birlamchi kalit (PRIMARY KEY)** – jadvaldagi har bir yozuvni noyob aniqlovchi maydon (masalan, **id**).
- **Tashqi kalit (FOREIGN KEY)** – boshqa jadvalning birlamchi kalitiga havola qiluvchi maydon; jadvallarni bog‘laydi.

Bog‘lanish turlari:

- **birga-bir (1:1)** – bir talabaga bitta talabalik guvohnomasi;
- **birga-ko‘p (1:N)** – bir muallifning ko‘p kitobi bor (eng ko‘p uchraydi);
- **ko‘pga-ko‘p (M:N)** – bir talaba ko‘p fanni o‘qiydi, bir fanni ko‘p talaba o‘qiydi; oraliq jadval orqali amalga oshiriladi.

Misol sifatida “Maktab kutubxonasi” bazasini loyihalaymiz:

```
CREATE TABLE authors (
  id      INTEGER PRIMARY KEY AUTOINCREMENT,
  name    TEXT NOT NULL,
  country TEXT
);

CREATE TABLE books (
  id      INTEGER PRIMARY KEY AUTOINCREMENT,
  title   TEXT NOT NULL,
  year    INTEGER CHECK (year > 1000),
  available INTEGER DEFAULT 1,
  author_id INTEGER,
  FOREIGN KEY (author_id) REFERENCES authors(id)
);
```

6. SQL so'rovlari

SQL (Structured Query Language) – relyatsion MB lar bilan ishlash uchun standart so'rovlar tili. Ma'lumotlar ustida bajariladigan to'rtta asosiy amal **CRUD** deb ataladi: Create (yaratish), Read (o'qish), Update (yangilash), Delete (o'chirish).

CRUD	SQL buyrug'i	Vazifasi
Create	INSERT	Yangi yozuv qo'shish
Read	SELECT	Yozuvlarni tanlab olish
Update	UPDATE	Mavjud yozuvni o'zgartirish
Delete	DELETE	Yozuvni o'chirish

INSERT – ma'lumot qo'shish

```
INSERT INTO authors (name, country) VALUES
('Abdulla Qodiriy', 'O'zbekiston'),
('Cho'lpon', 'O'zbekiston'),
('Jack London', 'AQSh');

INSERT INTO books (title, year, author_id) VALUES
('O'tkan kunlar', 1926, 1),
('Mehrobdan chayon', 1929, 1),
('Kecha va kunduz', 1936, 2),
('Martin Eden', 1909, 3);
```

SELECT – ma'lumotlarni tanlash

```
SELECT * FROM books; -- barcha ustun va qatorlar
SELECT title, year FROM books WHERE year < 1930; -- shart bilan
SELECT * FROM books WHERE title LIKE '%kun%'; -- matn ichida qidirish
SELECT * FROM books WHERE year BETWEEN 1920 AND 1930
ORDER BY year DESC LIMIT 5; -- saralash va cheklash
SELECT COUNT(*) AS soni, MIN(year), MAX(year) FROM books; -- agregat funksiyalar
SELECT author_id, COUNT(*) FROM books GROUP BY author_id; -- guruhlash
```

WHERE shartida taqqoslash (**=**, **<>**, **>**, **<**), mantiqiy (**AND**, **OR**, **NOT**) operatorlari, **IN**, **BETWEEN**, **LIKE**, **IS NULL** ishlatiladi.

UPDATE va DELETE

```
UPDATE books SET available = 0 WHERE id = 2; -- kitob berildi
DELETE FROM books WHERE id = 4;
```

Diqqat: **UPDATE** va **DELETE** buyruqlarida **WHERE** shartini unutish jadvaldagi **barcha** yozuvlarning o'zgarishiga yoki o'chishiga olib keladi.

JOIN – jadvallarni bog‘lash

```
SELECT b.title, b.year, a.name AS muallif
FROM books AS b
INNER JOIN authors AS a ON b.author_id = a.id
WHERE a.country = 'O‘zbekiston'
ORDER BY b.year;
```

JOIN turi	Natija
INNER JOIN	Faqat ikkala jadvalda mosligi bor yozuvlar
LEFT JOIN	Chap jadvalning barcha yozuvlari, o‘ngdan mosi bo‘lmasa NULL
RIGHT JOIN	O‘ng jadvalning barcha yozuvlari
FULL OUTER JOIN	Ikkala jadvalning barcha yozuvlari

Node.js dan SQLite bilan ishlash

```
// db.js (npm install better-sqlite3)
import Database from "better-sqlite3";
const db = new Database("library.db");

const kitoblar = db.prepare(
  `SELECT b.id, b.title, a.name AS author
   FROM books b JOIN authors a ON a.id = b.author_id`
).all();

const qosh = db.prepare("INSERT INTO books (title, year, author_id) VALUES (?, ?, ?)");
const natija = qosh.run("Sarob", 1937, 4);
console.log(natija.lastInsertRowid);
export default db;
```

So‘roq belgilari (?) – **parametrlangan so‘rov**. Qiymatlar SQL matniga qo‘lda ulanmaydi, balki alohida uzatiladi. Bu SQL Injection hujumidan himoyaning asosiy usuli (M5).

7. REST API dizayni va Express da amalga oshirish

API (Application Programming Interface) – dasturlarning bir-biri bilan muloqot qilish qoidalari va interfeysi. Veb-API orqali frontend (yoki mobil ilova) serverdan HTTP so‘rovlari yordamida ma’lumot oladi va yuboradi.

REST (Representational State Transfer) – 2000-yilda Roy Fielding tomonidan taklif qilingan veb-xizmatlar arxitektura uslubi. REST tamoyillari:

- **mijoz-server** – interfeys va ma’lumotlarni saqlash ajratilgan;
- **holatsizlik (stateless)** – har bir so‘rov o‘zi uchun zarur barcha ma’lumotni (jumladan, tokenni) o‘z ichiga oladi;
- **keshlanuvchanlik** – javoblar kesh qilinishi mumkinligini bildiradi;

- **yagona interfeys** – resurslar URL orqali aniqlanadi, ular ustidagi amallar HTTP metodlari bilan bajariladi;
- **qatlamli tizim** – mijoz oraliq serverlar (proksi, balanslovchi) borligini bilmasligi mumkin.

Resurslar va endpoint'lar

REST da asosiy tushuncha – **resurs** (kitob, talaba, buyurtma). Resurs URL da **ko'plikdagi ot** bilan ifodalanadi, amal esa HTTP metodi bilan beriladi (URL da fe'l ishlatilmaydi: `/getBooks` emas, `GET /api/books`).

Metod	Endpoint	Amal	Javob
GET	<code>/api/books</code>	Barcha kitoblar ro'yxati	200 + massiv
GET	<code>/api/books/5</code>	5-kitob ma'lumoti	200 yoki 404
POST	<code>/api/books</code>	Yangi kitob qo'shish	201 + yangi obyekt
PUT	<code>/api/books/5</code>	5-kitobni to'liq yangilash	200
PATCH	<code>/api/books/5</code>	5-kitobni qisman yangilash	200
DELETE	<code>/api/books/5</code>	5-kitobni o'chirish	204
GET	<code>/api/authors/1/books</code>	1-muallifning kitoblari	200
GET	<code>/api/books?year=1926&sort=title</code>	Filtrlash va saralash	200

Express da REST API

```
import express from "express";
import db from "./db.js";
const app = express();
app.use(express.json());
app.use(express.static("public"));

app.get("/api/books", (req, res) => {
  const rows = db.prepare(`SELECT b.*, a.name AS author FROM books b
    LEFT JOIN authors a ON a.id = b.author_id`).all();
  res.json(rows);
});

app.get("/api/books/:id", (req, res) => {
  const book = db.prepare("SELECT * FROM books WHERE id = ?").get(req.params.id);
  if (!book) return res.status(404).json({ error: "Kitob topilmadi" });
  res.json(book);
});

app.post("/api/books", (req, res) => {
  const { title, year, author_id } = req.body;
  if (!title) return res.status(400).json({ error: "Kitob nomi majburiy" });
  const info = db.prepare("INSERT INTO books (title, year, author_id) VALUES (?, ?, ?)")
    .run(title, year, author_id);
});
```

```

    res.status(201).json({ id: info.lastInsertRowid, title, year, author_id });
  });

app.delete("/api/books/:id", (req, res) => {
  db.prepare("DELETE FROM books WHERE id = ?").run(req.params.id);
  res.status(204).end();
});

app.listen(3000);

```

Frontend bunday API ga `fetch()` orqali murojaat qiladi (M3). Agar frontend va backend turli manzillarda bo'lsa, serverda CORS ga ruxsat beriladi: `npm install cors` va `app.use(cors({ origin: "https://mening-saytim.uz" }))`. API larni sinash uchun **Postman** yoki VS Code ning **Thunder Client** kengaytmasidan foydalaniladi.

JSON – REST API larda ma'lumot almashishning asosiy formati (M3). Express da `res.json(obyekt)` obyektini JSON ga aylantirib yuboradi, `express.json()` middleware i esa kelgan JSON tanani `req.body` ga o'giradi.

Mavzu bo'yicha xulosa

Dinamik veb-ilova uchun backend zarur. Node.js JavaScriptni serverda bajaradi, NPM paketlarni boshqaradi, Express.js esa marshrutlash, middleware va statik fayllar yordamida server yaratishni soddalashtiradi. Ilova ma'lumotlari ma'lumotlar bazasida saqlanadi: relyatsion MB jadvallar va kalitlar orqali tashkil etiladi, SQL so'rovlari CRUD amallarini bajaradi. REST API resurslarni URL va HTTP metodlari orqali ochib beradi, ma'lumotlar JSON formatida almashinadi.

Nazorat savollari

1. Frontend va backendning asosiy farqlarini ayting.
2. Node.js nima va uning asosiy xususiyatlari qanday?
3. `package.json` va `node_modules` ning vazifasi nima?
4. Express.js da marshrut qanday yaratiladi? `req.params` va `req.query` farqi nimada?
5. Middleware nima? `next()` funksiyasi nima uchun kerak?
6. SQL va NoSQL bazalarining asosiy farqlari nimada?
7. Birlamchi va tashqi kalitlarning vazifasi nima? Bog'lanish turlariga misol keltiring.
8. CRUD amallari va ularga mos SQL buyruqlarini ayting. `INNER JOIN` va `LEFT JOIN` farqi nimada?
9. REST arxitekturasining asosiy tamoyillarini sanang.
10. "Talabalar" resursi uchun REST endpoint'lar ro'yxatini tuzing.

Adabiyotlar: [7], [10]

5-MAVZU. VEB-XAVFSIZLIK (XSS, CSRF, SQL INJECTION), VEB-SAYTNI OPTIMALLASHTIRISH, SEO ASOSLARI VA HOSTINGGA JOYLASHTIRISH

Reja:

1. Foydalanuvchi autentifikatsiyasi, sessiyalar va cookie'lar.
2. Asosiy veb-hujumlar: XSS, CSRF, SQL Injection. Xavfsiz kod yozish prinsiplari.
3. Veb-ilovalarni optimallashtirish.
4. SEO asoslari, meta teglar va Google Analytics.
5. Veb-saytni hostingga joylashtirish.

Tayanch iboralar:

autentifikatsiya, avtorizatsiya, parol xeshi, bcrypt, sessiya, cookie, **HttpOnly**, **Secure**, **SameSite**, XSS, CSRF, SQL Injection, OWASP Top 10, CSP, HTTPS, minifikatsiya, kesh, lazy loading, WebP, Core Web Vitals, Lighthouse, SEO, meta teg, sitemap, robots.txt, Open Graph, Google Analytics, hosting, domen, Git, repozitoriy, commit, GitHub Pages, Netlify.

1. Foydalanuvchi autentifikatsiyasi, sessiyalar va cookie'lar

Foydalanuvchi bilan ishlovchi har qanday tizimda uch tushuncha farqlanadi:

- **identifikatsiya** – foydalanuvchining kimligini aytishi (login, e-pochta);
- **autentifikatsiya** – haqiqatan ham o'sha shaxs ekanini tekshirish (parol, SMS-kod, barmoq izi);
- **avtorizatsiya** – tasdiqlangan foydalanuvchiga qaysi amallarga ruxsat borligini aniqlash (masalan, o'qituvchi baho qo'ya oladi, o'quvchi esa faqat ko'radi).

Parollarni saqlash

Parollar bazada **hech qachon ochiq holda saqlanmaydi**. Ular maxsus sekin xeshlash algoritmlari (bcrypt, Argon2) yordamida qaytarib bo'lmaydigan **xeshga** aylantiriladi. Kirishda foydalanuvchi kiritgan parol xeshlanib, bazadagi xesh bilan solishtiriladi.

```
import bcrypt from "bcrypt";
const xesh = await bcrypt.hash("Parol123!", 10); // ro'yxatdan o'tishda
// $2b$10$Vd3k... - bazaga shu saqlanadi
const togri = await bcrypt.compare("Parol123!", xesh); // kirishda: true/false
```

Cookie va sessiyalar

HTTP holatsiz protokol bo'lgani uchun server foydalanuvchini keyingi so'rovlarda "tanishi" uchun maxsus mexanizm kerak. **Cookie** – server tomonidan

Set-Cookie sarlavhasi orqali brauzerga yuboriladigan va brauzer har bir keyingi so‘rovda serverga avtomatik qaytaradigan kichik matnli ma’lumot.

Sessiyaga asoslangan autentifikatsiya: foydalanuvchi muvaffaqiyatli kimgach, server sessiya yaratadi va uning tasodifiy identifikatorini (session ID) cookie sifatida yuboradi. Sessiya ma’lumotlari serverda saqlanadi.

```
Set-Cookie: sid=a8f3k29xQ; HttpOnly; Secure; SameSite=Strict; Max-Age=3600
```

Atribut	Vazifasi
HttpOnly	Cookie ga JavaScript orqali murojaat qilib bo‘lmaydi (XSS orqali o‘g‘irlashdan himoya)
Secure	Cookie faqat HTTPS orqali yuboriladi
SameSite=Strict/Lax	Boshqa saytdan kelgan so‘rovlarda cookie yuborilmaydi (CSRF dan himoya)
Max-Age, Expires	Cookie amal qilish muddati

2. Asosiy veb-hujumlar: XSS, CSRF, SQL Injection. Xavfsiz kod yozish prinsiplari

OWASP (Open Worldwide Application Security Project) tashkiloti veb-ilovalardagi eng xavfli zaifliklarning “OWASP Top 10” ro‘yxatini muntazam e’lon qiladi. Unga kirish nazoratining buzilishi, kriptografik xatolar, in’ektsiyalar (SQL Injection, XSS) va boshqalar kiradi. Quyida eng ko‘p uchraydigan uchta hujum ko‘rib chiqiladi. Ushbu ma’lumotlar faqat himoyani to‘g‘ri tashkil etish maqsadida beriladi.

XSS (Cross-Site Scripting)

Mohiyati: tajovuzkor sayt sahifasiga zararli JavaScript kodini joylashtiradi; boshqa foydalanuvchilar sahifani ochganda u ularning brauzerida bajariladi va, masalan, cookie yoki tokeni o‘g‘irlaydi. Sabab – foydalanuvchi kiritgan ma’lumotni tekshirmasdan HTML sifatida chiqarish.

```
// ZAIF KOD: foydalanuvchi izohi HTML sifatida qo‘shiladi
izohlarDiv.innerHTML += `<p>${izoh}</p>`;

// XAVFSIZ KOD: matn sifatida qo‘shiladi
const p = document.createElement("p");
p.textContent = izoh;
izohlarDiv.append(p);
```

Himoya: chiqarishda ma’lumotni ekranlash (**textContent**, shablon dvigatellarining avtomatik ekranlashi); kirishda ma’lumotni tekshirish; **Content Security Policy (CSP)** sarlavhasi orqali faqat ishonchli manbalardan skript yuklashga ruxsat berish; cookie larga **HttpOnly** atributi.

CSRF (Cross-Site Request Forgery)

Mohiyati: foydalanuvchi o‘z hisobiga kirgan holda boshqa (zararli) saytni ochadi; ushbu sayt foydalanuvchi nomidan uning brauzeri orqali birinchi saytga so‘rov yuboradi (masalan, parolni o‘zgartirish). Brauzer cookie ni avtomatik qo‘shgani uchun server so‘rovni haqiqiy deb qabul qiladi.

Himoya: cookie larga **SameSite=Strict** yoki **Lax** atributi; holatni o‘zgartiruvchi har bir formada maxsus, tasodifiy **CSRF-token**; muhim amallarni faqat POST/PUT/DELETE orqali bajarish (GET orqali emas); tokenni cookie da emas, **Authorization** sarlavhasida yuborish.

SQL Injection

Mohiyati: foydalanuvchi kiritgan ma’lumot SQL so‘rov matniga to‘g‘ridan-to‘g‘ri ulanganda, maxsus tuzilgan kiritma so‘rov mantiqini o‘zgartirib yuboradi va ruxsatsiz ma’lumot olish yoki o‘chirishga olib keladi.

```
// ZAIF KOD: satrlarni ulash
db.prepare(`SELECT * FROM users WHERE login = '${login}'`).get();

// XAVFSIZ KOD: parametrlangan so‘rov
db.prepare("SELECT * FROM users WHERE login = ?").get(login);
```

Himoya: faqat parametrlangan so‘rovlar yoki ORM dan foydalanish; ma’lumotlar bazasi foydalanuvchisiga minimal huquq berish; kiritilgan ma’lumot turini va uzunligini tekshirish.

Xavfsiz kod yozish prinsiplari

- **hech qachon foydalanuvchi kiritgan ma’lumotga ishonmaslik** – serverda har doim tekshirish;
- **minimal imtiyozlar** – har bir foydalanuvchi va dastur faqat zarur huquqlarga ega bo‘lsin;
- maxfiy kalit va parollarni kodda emas, **muhit o‘zgaruvchilarida (.env)** saqlash va **.env** ni Git ga yuklamaslik;
- butun sayt uchun **HTTPS**;
- paketlarni yangilab borish (**npm audit**), xavfsizlik sarlavhalarini o‘rnatish (**helmet** paketi);
- xatolik xabarlarida ichki ma’lumotlarni (SQL so‘rov, fayl yo‘li) ko‘rsatmaslik;
- kirish urinishlari sonini cheklash (**express-rate-limit**).

3. Veb-ilovalarni optimallashtirish

Sahifa yuklanish tezligi foydalanuvchi tajribasi va qidiruv tizimidagi o‘rinni bevosita belgilaydi: tadqiqotlarga ko‘ra, mobil foydalanuvchilarning katta qismi 3

soniyadan ko‘p yuklanadigan saytni tark etadi. Google sahifa sifatini **Core Web Vitals** ko‘rsatkichlari bilan baholaydi:

Ko‘rsatkich	Mazmuni	Yaxshi qiymat
LCP (Largest Contentful Paint)	Eng katta element ko‘ringuncha vaqt	$\leq 2,5$ s
INP (Interaction to Next Paint)	Foydalanuvchi harakatiga javob tezligi	≤ 200 ms
CLS (Cumulative Layout Shift)	Yuklanishda elementlarning “sakrashi”	$\leq 0,1$

Optimallashtirish usullari

- **rasmlarni optimallashtirish** – zamonaviy formatlar (WebP, AVIF), o‘lchamni ekranga moslash, siqish (Squoosh, TinyPNG), **srcset** orqali turli ekranlar uchun turli o‘lcham;
- **lazy loading** – ekranda ko‘rinmayotgan rasm va iframelarni foydalanuvchi ularga yaqinlashgandagina yuklash: `` (width/height berilishi CLS ni oldini oladi);
- **minifikatsiya** – CSS va JS fayllardan bo‘shliq va izohlarni olib tashlash; fayllarni birlashtirish;
- **siqish** – serverda Gzip yoki Brotli;
- **keshlash** – **Cache-Control** sarlavhasi, CDN;
- skriptlarni **defer/async** bilan yuklash, keraksiz kutubxonalardan voz kechish;
- shriftlarni **font-display: swap** bilan yuklash.

```

```

Sahifa sifatini tekshirish vositalari: Chrome DevTools → **Lighthouse** (Performance, Accessibility, Best Practices, SEO bo‘yicha 0–100 ball), PageSpeed Insights (pagespeed.web.dev).

4. SEO asoslari, meta teglar va Google Analytics

SEO (Search Engine Optimization) – saytning qidiruv tizimlarida (Google, Yandex, Bing) yuqori o‘rinlarda ko‘rinishi uchun amalga oshiriladigan chora-tadbirlar majmui.

Texnik va ichki SEO:

- har bir sahifa uchun noyob, 50–60 belgili `<title>` va 150–160 belgili `meta description`;
- bitta `<h1>` va mantiqiy sarlavhalar ierarxiyasi, semantik teglar;
- rasmlarda mazmunli `alt` matni;
- tushunarli URL lar (`/togaraklar/web-dasturlash, ?id=123` emas);
- moslashuvchan dizayn va yuqori tezlik, HTTPS;
- `sitemap.xml` (sahifalar xaritasi) va `robots.txt` (indekslash qoidalari) fayllari;
- sifatli, noyob va muntazam yangilanadigan mazmun, ichki havolalar.

```
<head>
  <title>Web-dasturlash to‘garagi | Maktab №5, Namangan</title>
  <meta name="description" content="5-maktabdagi bepul veb-dasturlash to‘garagi:
    HTML, CSS, JavaScript. Ro‘yxatdan o‘ting!">
  <link rel="canonical" href="https://maktab5.uz/togaraklar/web">
  <!-- Open Graph: ijtimoiy tarmoqlarda ulashganda ko‘rinish -->
  <meta property="og:title" content="Web-dasturlash to‘garagi">
  <meta property="og:description" content="HTML, CSS, JavaScript ni o‘rganing">
  <meta property="og:image" content="https://maktab5.uz/images/og-web.jpg">
  <meta property="og:type" content="website">
</head>
```

```
# robots.txt
User-agent: *
Disallow: /admin/
Sitemap: https://maktab5.uz/sitemap.xml
```

Google Analytics – sayt tashrifchilari statistikasini (tashriflar soni, manbalari, qurilmalari, eng ko‘p ko‘rilgan sahifalar, sahifada o‘tkazilgan vaqt) yig‘uvchi bepul xizmat. Joriy versiyasi – GA4. Ulanish uchun analytics.google.com da “oqim” (stream) yaratiladi va berilgan kod sahifaning `<head>` qismiga joylanadi. Sayt Google qidiruvida qanday ko‘rinishini kuzatish uchun **Google Search Console** ishlatiladi. Shaxsiy ma’lumotlar to‘planganda foydalanuvchini maxfiylik siyosati orqali xabardor qilish lozim.

5. Veb-saytni hostingga joylashtirish

Hosting – veb-sayt fayllarini Internetga doimiy ulangan serverda joylashtirib, uni istalgan foydalanuvchi uchun ochiq qilish xizmati. Saytni Internetda e’lon qilish uchun uchta narsa kerak: sayt fayllari, ular joylashadigan server (hosting) va manzil (domen nomi yoki hosting bergan subdomen).

Hosting turi	Xususiyati	Misollar
Statik hosting	Faqat HTML, CSS, JS, rasm fayllari; server dasturi ishlamaydi. Ko‘pincha bepul, CDN va HTTPS bilan	GitHub Pages, Netlify, Vercel, Cloudflare Pages

Hosting turi	Xususiyati	Misollar
Platforma xizmati (PaaS)	Node.js, Python kabi server ilovalarini ishga tushiradi, MB ulash mumkin	Render, Railway, Heroku
Virtual server (VPS)	To'liq boshqariladigan server; sozlash bilim talab etadi	DigitalOcean, mahalliy provayderlar (uz-hosting)
Umumiy (shared) hosting	Bitta serverni ko'p sayt bo'lishadi; odatda PHP + MySQL	Mahalliy hosting provayderlari

Git va GitHub

Git – fayllardagi o'zgarishlar tarixini saqlovchi versiyalarni boshqarish tizimi. U kodning har bir holatini (**commit**) saqlaydi, xato bo'lsa oldingi holatga qaytish, bir nechta dasturchining bir loyihada parallel ishlashini ta'minlaydi. **GitHub** – Git omborlarini (**repository**) Internetda saqlash va jamoaviy ishlash platformasi.

```
git init # papkada ombor yaratish
git add . # o'zgarishlarni tayyorlash
git commit -m "Saytning birinchi versiyasi" # tarixga yozish
git branch -M main
git remote add origin https://github.com/LOGIN/maktab-sayti.git
git push -u origin main # GitHub ga yuborish
```

Keyingi o'zgarishlardan so'ng faqat `git add .`, `git commit -m "izoh"` va `git push` buyruqlari takrorlanadi. Maxfiy ma'lumotlar (`.env`) va `node_modules` papkasi `.gitignore` fayli orqali omborga yuklanmaydi.

GitHub Pages va Netlify

GitHub Pages – GitHub omboridagi statik saytni bepul e'lon qiladi: ombor → Settings → Pages → Branch: `main` → Save. Sayt <https://LOGIN.github.io/ombor-nomi/> manzilida ochiladi.

Netlify – GitHub omboriga ulanib, har bir `git push` dan keyin saytni avtomatik yangilaydi (uzluksiz joylashtirish – *continuous deployment*). Sayt `nom.netlify.app` manzilini oladi, o'z domeningizni ulash va bepul HTTPS sertifikatini avtomatik beriladi. Papkani brauzerga sudrab tashlash (drag and drop) orqali ham joylashtirish mumkin.

Domen va HTTPS

O'z domen nomini (masalan, `maktab5.uz`) ro'yxatdan o'tkazish uchun domen registratoriga murojaat qilinadi; `.uz` zonasidagi domenlar akkreditatsiyadan o'tgan registratorlar orqali beriladi. So'ngra DNS sozlamalarida domen hosting serveriga yo'naltiriladi (A yoki CNAME yozuvi). Zamonaviy hostinglar Let's Encrypt sertifikatini yordamida HTTPS ni avtomatik yoqadi.

Joylashtirishdan oldingi tekshiruv ro'yxati

- barcha havolalar va rasm yo'llari nisbiy va kichik-katta harflar mos (serverlar **Rasm.JPG** va **rasm.jpg** ni farqlaydi);
- bosh sahifa **index.html** deb nomlangan;
- sayt telefonda to'g'ri ko'rinadi, konsolda xatolar yo'q;
- **title**, **description** va boshqa SEO meta teglar to'ldirilgan;
- rasmlar optimallashtirilgan, Lighthouse ko'rsatkichlari qoniqarli;
- maxfiy kalit va parollar omborga tushmagan.

Mavzu bo'yicha xulosa

Xavfsiz veb-ilova parollarni xeshlab saqlaydi, kiritilgan ma'lumotlarni serverda tekshiradi, parametrlangan so'rovlar va ekranlash orqali SQL Injection va XSS dan, SameSite cookie va CSRF-tokenlar orqali CSRF dan himoyalaniadi. Optimallashtirish va SEO saytni tez va qidiruv tizimlarida ko'rinadigan qiladi. Tayyor sayt Git yordamida GitHub ga yuklanadi va GitHub Pages yoki Netlify kabi hostinglarda bepul, HTTPS bilan e'lon qilinadi.

Nazorat savollari

1. Identifikatsiya, autentifikatsiya va avtorizatsiyaning farqi nimada?
2. Nima uchun parollar xesh ko'rinishida saqlanadi?
3. **HttpOnly**, **Secure**, **SameSite** cookie atributlari nima uchun kerak?
4. XSS hujumining mohiyati va undan himoyalaniish usullari qanday?
5. CSRF hujumi qanday amalga oshadi va qanday oldi olinadi?
6. Parametrlangan so'rov SQL Injection dan qanday himoya qiladi?
7. Core Web Vitals ko'rsatkichlarini ayting. Lazy loading nima?
8. SEO uchun qanday meta teglar va fayllar muhim?
9. Git da **add**, **commit**, **push** buyruqlari nima qiladi?
10. Statik saytni GitHub Pages va Netlify ga joylashtirish bosqichlarini ayting.

Adabiyotlar: [5], [6]

V. AMALIY MASHG'ULOT MATERIALLARI

Amaliy mashg'ulotlar kompyuter sinfida o'tkaziladi. Har bir mashg'ulot quyidagi tuzilishga ega: mavzu, maqsad, kutilayotgan natijalar, kerakli jihozlar, qisqacha nazariy ma'lumot, ishni bajarish tartibi (namunaviy kodlar bilan), mustaqil bajarish uchun variantli topshiriqlar, baholash mezonlari va nazorat savollari.

Amaliy ishni topshirish tartibi. Tinglovchi namunaviy topshiriqni bajaradi, so'ngra o'z variantidagi topshiriqni mustaqil bajaradi va natijani o'qituvchiga ko'rsatib, nazorat savollariga javob beradi (himoya). Ish fayllari **familiya_A1** ko'rinishidagi papkada saqlanadi. Variant raqami tinglovchining jurnaldagi tartib

raqami bo'yicha aniqlanadi (raqam 10 dan katta bo'lsa, 10 ga bo'lgandagi qoldiq olinadi; qoldiq 0 bo'lsa – 10-variant).

Amaliy ishlar uchun umumiy baholash mezon:

Baho	Mezon
5	Namunaviy va variant topshiriqlari to'liq, xatosiz bajarilgan; kod toza, izohlangan, validatsiyadan o'tadi; nazorat savollariga to'liq javob berilgan; qo'shimcha ijodiy element mavjud
4	Topshiriqlar to'liq bajarilgan, lekin 1–2 ta kichik kamchilik bor (dizayn, kod uslubi); nazorat savollariga asosan to'g'ri javob berilgan
3	Namunaviy topshiriq bajarilgan, variant topshirig'i qisman bajarilgan; savollarga javoblar to'liq emas
2	Topshiriq bajarilmagan yoki ko'chirilgan; savollarga javob berilmagan

1-AMALIY MASHG‘ULOT. VEB-LOYIHA MUHITINI SOZLASH. SEMANTIK HTML5 YORDAMIDA VEB-SAHIFA YARATISH: FORMALAR, JADVALLAR, MULTIMEDIA ELEMENTLARI

Maqsad: veb-ishlab chiqish muhitini sozlash; semantik HTML5 teglari, formalar, jadvallar va multimedia elementlaridan foydalanib ko‘p sahifali sayt tuzilishini yaratish.

Kutilayotgan natijalar (TN2, TN7): tinglovchi VS Code muharririda loyiha papkasini tashkil eta oladi; HTML5 hujjatining to‘g‘ri tuzilishini yozadi; semantik teglar, jadval, forma, audio va video elementlarini joylashtiradi; kodni W3C validatorida tekshiradi.

Kerakli jihozlar: Visual Studio Code, “Live Server” kengaytmasi, Google Chrome, namunaviy rasm, audio (mp3) va video (mp4) fayllar.

Qisqacha nazariy ma’lumot

HTML5 hujjati `<!DOCTYPE html>` deklaratsiyasi, `<head>` va `<body>` qismlaridan iborat. Sahifa qismlari semantik teglar (`header`, `nav`, `main`, `section`, `article`, `aside`, `footer`) bilan belgilanadi. Formalar `<form>`, `<input>`, `<select>`, `<textarea>`, `<button>`; jadvallar `<table>`, `<tr>`, `<th>`, `<td>`; multimedia `<audio>`, `<video>` teglari bilan yaratiladi (M1 mavzusiga qarang).

Ishni bajarish tartibi

1-qadam. Muhitni sozlash

- 1) code.visualstudio.com saytidan VS Code ni yuklab o‘rnating.
- 2) Extensions (Ctrl+Shift+X) bo‘limidan “Live Server” (Ritwick Dey) va “Prettier – Code formatter” kengaytmalarini o‘rnating.
- 3) Ish stolida `familiya_A1` papkasini yarating va uni VS Code da oching (File → Open Folder).
- 4) Quyidagi tuzilishni yarating:

```
familiya_A1/  
├── index.html          (bosh sahifa)  
├── royxat.html         (ro‘yxatdan o‘tish formasi)  
├── images/            logo.png, dars.jpg  
└── media/              intro.mp4, musiqa.mp3
```

Maslahat: VS Code da bo‘sh `index.html` faylida ! belgisini yozib Tab tugmasini bossangiz, HTML5 shabloni avtomatik hosil bo‘ladi (Emmet). `ul>li*3` yozib Tab bossangiz, uchta `li` li ro‘yxat yaratiladi.

2-qadam. Bosh sahifa (index.html)

```
<!DOCTYPE html>  
<html lang="uz">  
<head>
```

```

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="Maktab informatika to'garagi sayti">
<title>Informatika to'garagi | Bosh sahifa</title>
</head>
<body>
  <header>
    
    <h1>"Yosh dasturchi" informatika to'garagi</h1>
    <nav>
      <a href="index.html">Bosh sahifa</a> |
      <a href="#jadval">Dars jadvali</a> |
      <a href="#media">Video dars</a> |
      <a href="royxat.html">Ro'yxatdan o'tish</a>
    </nav>
  </header>

  <main>
    <section id="haqida">
      <h2>To'garak haqida</h2>
      <p>To'garakda o'quvchilar <strong>HTML</strong>, <strong>CSS</strong>
va
      <strong>JavaScript</strong> asosida veb-sayt yaratishni
o'rganadilar.</p>
      <figure>
        
        <figcaption>1-rasm. To'garak mashg'ulotidan lavha</figcaption>
      </figure>
      <h3>O'rganiladigan mavzular</h3>
      <ol>
        <li>HTML va semantik teglar</li>
        <li>CSS va dizayn</li>
        <li>JavaScript va interaktivlik</li>
      </ol>
    </section>

    <section id="jadval">
      <h2>Dars jadvali</h2>
      <table border="1">
        <caption>2026-2027 o'quv yili, 1-yarim yillik</caption>
        <thead>
          <tr><th>Kun</th><th>Vaqt</th><th>Mavzu</th><th>Xona</th></tr>
        </thead>
        <tbody>
          <tr><td>Dushanba</td><td>14:00-15:30</td><td>HTML</td>
            <td rowspan="2">204-xona</td></tr>
          <tr><td>Chorshanba</td><td>14:00-15:30</td><td>CSS</td></tr>
          <tr><td>Juma</td><td colspan="2">Loyiha ustida erkin
ishlash</td>
            <td>Kompyuter sinfi</td></tr>
        </tbody>
      </table>
    </section>

    <section id="media">
      <h2>Video dars</h2>
      <video width="480" controls poster="images/dars.jpg">
        <source src="media/intro.mp4" type="video/mp4">
        Brauzeringiz video elementini qo'llab-quvvatlamaydi.
      </video>
    </section>
  </main>
</body>
</html>

```

```

        </video>
        <h3>Fon musiqasi</h3>
        <audio controls>
            <source src="media/musiq.mp3" type="audio/mpeg">
        </audio>
    </section>

    <article>
        <h2>Yangilik: maktab olimpiadasi</h2>
        <p><time datetime="2026-11-15">15-noyabr</time> kuni informatika
            olimpiadasining maktab bosqichi bo'lib o'tadi.</p>
    </article>
</main>

<aside>
    <h3>Foydali havolalar</h3>
    <ul>
        <li><a href="https://developer.mozilla.org" target="_blank"
rel="noopener">MDN</a></li>
        <li><a href="https://www.w3schools.com" target="_blank"
rel="noopener">W3Schools</a></li>
    </ul>
</aside>

<footer>
    <p>&copy; 2026 "Yosh dasturchi" to'garagi. Aloqa:
        <a href="mailto:togarak@maktab.uz">togarak@maktab.uz</a></p>
</footer>
</body>
</html>

```

3-qadam. Ro'yxatdan o'tish formasi (royxat.html)

```

<!DOCTYPE html>
<html lang="uz">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ro'yxatdan o'tish</title>
</head>
<body>
    <h1>To'garakka ro'yxatdan o'tish</h1>
    <form action="#" method="post">
        <fieldset>
            <legend>Shaxsiy ma'lumotlar</legend>
            <p><label for="fio">F.I.Sh.:</label><br>
                <input type="text" id="fio" name="fio" required minlength="5"></p>
            <p><label for="sinf">Sinf:</label><br>
                <input type="number" id="sinf" name="sinf" min="5" max="11"
required></p>
            <p><label for="tel">Telefon:</label><br>
                <input type="tel" id="tel" name="tel" placeholder="+998901234567"
pattern="\+998[0-9]{9}"></p>
            <p><label for="email">E-pochta:</label><br>
                <input type="email" id="email" name="email"></p>
            <p><label for="sana">Tug'ilgan sana:</label><br>
                <input type="date" id="sana" name="sana"></p>
        </fieldset>
    </form>

```

```

        <legend>Qiziqishlar</legend>
        <label><input type="checkbox" name="qiz" value="web"> Veb-
dizayn</label>
        <label><input type="checkbox" name="qiz" value="dastur">
Dasturlash</label>
        <label><input type="checkbox" name="qiz" value="robot">
Robototexnika</label>
        <p>Smena:
        <label><input type="radio" name="smena" value="1" checked> 1-
smena</label>
        <label><input type="radio" name="smena" value="2"> 2-
smena</label></p>
        <p><label for="daraja">Bilim darajasi:</label>
        <select id="daraja" name="daraja">

<option>Boshlang'ich</option><option>O'rta</option><option>Yuqori</option>
        </select></p>
        <p><label for="izoh">O'zingiz haqingizda:</label><br>
        <textarea id="izoh" name="izoh" rows="4" cols="40"></textarea></p>
        </fieldset>
        <button type="submit">Yuborish</button>
        <button type="reset">Tozalash</button>
    </form>
    <p><a href="index.html">← Bosh sahifaga qaytish</a></p>
</body>
</html>

```

4-qadam. Tekshirish

- 1) `index.html` faylida o'ng tugma → "Open with Live Server". Sahifa `http://127.0.0.1:5500` manzilida ochiladi.
- 2) Barcha havolalar, jadval, video va audio ishlashini tekshiring.
- 3) Formada maydonlarni bo'sh qoldirib "Yuborish" tugmasini bosing va brauzer validatsiyasini kuzating.
- 4) F12 → Elements bo'limida DOM daraxtini ko'zdan kechiring.
- 5) `validator.w3.org/#validate_by_input` sahifasida kodni tekshiring va xatolarni tuzating.

Mustaqil bajarish uchun topshiriqlar

Har bir variant bo'yicha semantik teglardan foydalangan holda kamida ikki sahifali sayt yarating. Saytda: navigatsiya, kamida bitta jadval (`colspan` yoki `rowspan` bilan), kamida 5 maydonli forma (kamida 3 xil `input` turi va validatsiya), rasm (`figure` bilan) va video yoki audio bo'lishi shart.

Variant	Sayt mavzusi	Jadval va forma mazmuni
1	Maktab kutubxonasi	Kitoblar ro'yxati; kitob buyurtma qilish formasi
2	Sport to'garagi	Mashg'ulotlar jadvali; a'zolikka ariza
3	Oshxona menyusi	Haftalik taomnoma; taklif va shikoyatlar formasi

Variant	Sayt mavzusi	Jadval va forma mazmuni
4	Sinf rahbari sahifasi	O'quvchilar davomati; ota-onalar bilan aloqa formasi
5	Ekskursiya agentligi	Sayohatlar narxlari; sayohatga yozilish formasi
6	Ingliz tili kursi	Guruhlar jadvali; darajani aniqlash testi formasi
7	Kompyuter do'koni	Mahsulot tavsiflari; buyurtma formasi
8	Olimpiada natijalari	Ishtirokchilar reytingi; ishtirok uchun ariza
9	Shahar muzeyi	Ko'rgazmalar jadvali; chipta band qilish formasi
10	Maktab gazetasi	Sonlar arxivi; maqola yuborish formasi

Nazorat savollari

1. Live Server kengaytmasi qanday vazifani bajaradi?
2. Nima uchun bosh sahifa `index.html` deb nomlanadi?
3. Sahifangizda qaysi semantik teglardan foydalandingiz va nima uchun?
4. `<figure>` va `<figcaption>` nima uchun kerak?
5. `pattern` atributi qanday ishlaydi?
6. `<label for>` atributi nima beradi?
7. Video bir nechta `<source>` bilan nima uchun beriladi?

2-AMALIY MASHG'ULOT. CSS FLEXBOX, GRID VA MEDIA QUERIES YORDAMIDA RESPONSIVE SAHIFA YARATISH

Maqsad: tashqi CSS fayl orqali sahifaga dizayn berish; Flexbox va Grid yordamida navigatsiya paneli, kartalar va murakkab sahifa maketini yaratish; media so'rovlar yordamida sahifani telefon, planshet va kompyuter ekranlariga moslashtirish.

Kutilayotgan natijalar (TN2, TN4, TN8): tinglovchi CSS faylni HTML ga ulaydi; selektorlar va box modeli xossalarini qo'llaydi; Flexbox bilan navigatsiya paneli va Grid bilan kartalar maketini yaratadi; mobile-first uslubida media so'rovlar yozadi va sahifani turli qurilmalarda tekshiradi.

Kerakli jihozlar: VS Code, Live Server, Chrome DevTools (Device Toolbar – Ctrl+Shift+M), A1 mashg'ulotida yaratilgan sayt.

Qisqacha nazariy ma'lumot

Flexbox – bir o'lchovli joylashtirish modeli. Konteyner (`display: flex`) xossalari: `flex-direction` (row/column), `justify-content` (asosiy o'q: flex-start, center, space-between, space-around), `align-items` (ko'ndalang o'q), `flex-wrap`, `gap`. Element xossalari: `flex: 1` (bo'sh joyni egallash), `order`, `align-self`.

CSS Grid – ikki o'lchovli joylashtirish modeli. Konteyner xossalari: `grid-template-columns`, `grid-template-rows`, `gap`, `grid-template-areas`. `fr` birligi – bo'sh joyning ulushi; `repeat(auto-fit, minmax(250px, 1fr))` – sig'ishiga qarab ustunlar sonini avtomatik tanlash.

Media so'rov – ekran kengligiga qarab CSS qoidalarini qo'llash sharti: `@media (min-width: 768px) { ... }`. Asosiy chegaralar (breakpoints): 576, 768, 992, 1200 px. **Mobile-first** yondashuvida avval telefon uchun stillar yoziladi, so'ngra `min-width` bilan kattaroq ekranlar uchun kengaytiriladi. **Tranzitsiya** (`transition`) – xossa qiymatining silliq o'zgarishi.

Ishni bajarish tartibi

1-qadam. Loyiha tuzilishi

`familiya_A2` papkasida `index.html` va `css/style.css` fayllarini yarating.

```
<!DOCTYPE html>
<html lang="uz">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>To'garaklar</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <header class="sarlavha">
    <div class="logo">Yosh<span>Dasturchi</span></div>
    <nav>
```

```

        <ul class="menu">
            <li><a href="#" class="faol">Bosh sahifa</a></li>
            <li><a href="#togaraklar">To'garaklar</a></li>
            <li><a href="#ustozlar">Ustozlar</a></li>
            <li><a href="#aloqa">Aloqa</a></li>
        </ul>
    </nav>
</header>

<section class="banner">
    <h1>Kelajak kasbini bugun o'rganing!</h1>
    <p>Veb-dizayn, dasturlash va robototexnika bo'yicha bepul to'garaklar</p>
    <a href="#togaraklar" class="tugma">Batafsil</a>
</section>

<main class="konteyner">
    <h2 id="togaraklar">Bizning to'garaklar</h2>
    <div class="kartalar">
        <article class="karta">
            <span class="belgi">Yangi</span>
            <h3>Veb-dizayn</h3>
            <p>HTML va CSS yordamida chiroyli saytlar yaratish.</p>
            <a href="#" class="tugma">Yozilish</a>
        </article>
        <article class="karta">
            <h3>Python</h3>
            <p>Dasturlash asoslari va algoritmlar.</p>
            <a href="#" class="tugma">Yozilish</a>
        </article>
        <article class="karta">
            <h3>Robototexnika</h3>
            <p>Arduino platasi bilan loyihalar.</p>
            <a href="#" class="tugma">Yozilish</a>
        </article>
        <article class="karta">
            <h3>Grafik dizayn</h3>
            <p>Figma va Photoshop asoslari.</p>
            <a href="#" class="tugma">Yozilish</a>
        </article>
    </div>
</main>

<div class="maket">
    <header class="m-header">Header</header>
    <aside class="m-aside">Yon panel</aside>
    <main class="m-main">Asosiy mazmun</main>
    <footer class="m-footer">Footer</footer>
</div>

<footer class="pastki">&copy; 2026 YoshDasturchi</footer>
</body>
</html>

```

2-qadam. CSS stillari (css/style.css)

```

/* 1. Umumiy sozlamalar */
*, ::before, ::after { box-sizing: border-box; margin: 0; padding: 0; }
:root {
    --asosiy: #2F5496;
    /* CSS o'zgaruvchilari */
}

```

```

    --qoshimcha: #F4B183;
    --fon: #F7F9FC;
}
body { font-family: "Segoe UI", Arial, sans-serif; line-height: 1.6;
    color: #333; background: var(--fon); }

/* 2. Navigatsiya paneli - Flexbox */
.sarlavha {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 16px 40px;
    background: var(--asosiy);
}
.logo { color: white; font-size: 24px; font-weight: 700; }
.logo span { color: var(--qoshimcha); }
.menu { display: flex; gap: 24px; list-style: none; }
.menu a { color: white; text-decoration: none; padding: 6px 10px; border-radius: 6px;
    transition: background 0.3s; }
.menu a:hover, .menu a.faol { background: rgba(255, 255, 255, 0.2); }

/* 3. Banner - fon va markazlash */
.banner {
    min-height: 300px;
    display: flex; flex-direction: column; justify-content: center; align-items: center;
    text-align: center; color: white; padding: 40px 20px;
    background: linear-gradient(135deg, #2F5496, #548235);
}
.banner h1 { font-size: 40px; margin-bottom: 12px; }

/* 4. Tugma - box modeli */
.tugma {
    display: inline-block;
    padding: 10px 24px;
    margin-top: 16px;
    border: 2px solid var(--qoshimcha);
    border-radius: 24px;
    background: var(--qoshimcha);
    color: #222; text-decoration: none; font-weight: 600;
}
.tugma:hover { background: transparent; color: var(--qoshimcha); }

/* 5. Kartalar - Grid */
.konteyner { max-width: 1200px; margin: 40px auto; padding: 0 20px; }
.konteyner h2 { text-align: center; margin-bottom: 24px; color: var(--asosiy); }
.kartalar {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 24px;
}
.karta {
    position: relative;
    background: white;
    padding: 24px;
    border-radius: 12px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);
}
.karta h3 { color: var(--asosiy); margin-bottom: 8px; }

```

```

.karta .tugma { background: var(--asosiy); border-color: var(--asosiy); color:
white; }
.belgi {
    position: absolute; top: 12px; right: 12px;
    background: crimson; color: white; font-size: 12px;
    padding: 2px 10px; border-radius: 10px;
}

/* 6. Murakkab sahifa maketi - grid-template-areas */
.maket {
    max-width: 1200px; margin: 40px auto; height: 400px;
    display: grid;
    grid-template-columns: 250px 1fr;
    grid-template-rows: 80px 1fr 60px;
    grid-template-areas:
        "header header"
        "aside main"
        "footer footer";
    gap: 10px;
}
.maket > * { display: grid; place-items: center; border-radius: 8px; color: white;
}
.m-header { grid-area: header; background: #2F5496; }
.m-aside { grid-area: aside; background: #7030A0; }
.m-main { grid-area: main; background: #548235; }
.m-footer { grid-area: footer; background: #C55A11; }

.pastki { text-align: center; padding: 20px; background: #222; color: #ccc; }

```

3-qadam. Box modelini tadqiq qilish

- 1) F12 → Elements bo‘limida **.karta** elementini tanlang. Pastdagi “Computed” yorlig‘ida box modeli diagrammasini ko‘ring.
- 2) **box-sizing: border-box** qatorini vaqtincha o‘chirib, kartaning o‘lchami qanday o‘zgarishini kuzating va izohlang.
- 3) DevTools da **.kartalar** elementi yonidagi “grid” belgisini bosib, grid chiziqlarini ko‘ring.
- 4) Brauzer oynasini toraytirib, kartalar soni avtomatik kamayishini kuzating.

4-qadam. Flexbox o‘yini (qo‘shimcha)

flexboxfroggy.com va cssgridgarden.com o‘yinlarining kamida 10 ta bosqichini bajaring.

5-qadam. Media so‘rovlar bilan moslashuvchanlik (mobile-first)

css/style.css faylining oxiriga quyidagi qoidalarni qo‘shing. Avval telefon uchun asosiy stillar yoziladi, so‘ngra planshet va kompyuter uchun kengaytiriladi.

```

/* 7. Telefon (asosiy holat): menyu ustun ko‘rinishida, kartalar bitta ustunda */
.sarlavha { flex-direction: column; gap: 12px; padding: 16px; }
.menu { flex-direction: column; align-items: center; gap: 8px; }
.banner h1 { font-size: 28px; }
.kartalar { grid-template-columns: 1fr; }
.maket {

```

```

height: auto;
grid-template-columns: 1fr;
grid-template-rows: auto;
grid-template-areas: "header" "main" "aside" "footer";
}
.maket > * { min-height: 80px; }

/* 8. Planshet: >= 768px */
@media (min-width: 768px) {
  .sarlavha { flex-direction: row; padding: 16px 40px; }
  .menu { flex-direction: row; gap: 24px; }
  .banner h1 { font-size: 36px; }
  .kartalar { grid-template-columns: repeat(2, 1fr); }
  .maket {
    height: 400px;
    grid-template-columns: 200px 1fr;
    grid-template-rows: 80px 1fr 60px;
    grid-template-areas: "header header" "aside main" "footer footer";
  }
}

/* 9. Kompyuter: >= 1200px */
@media (min-width: 1200px) {
  .banner h1 { font-size: 48px; }
  .kartalar { grid-template-columns: repeat(4, 1fr); }
  .maket { grid-template-columns: 250px 1fr; }
}

/* 10. Hover effekti va silliq o'tish */
.karta { transition: transform 0.3s ease, box-shadow 0.3s ease; }
.karta:hover { transform: translateY(-8px); box-shadow: 0 12px 24px rgba(0, 0, 0, 0.15); }

/* 11. Rasmlar konteynerdan chiqib ketmasligi uchun */
img { max-width: 100%; height: auto; }

```

Diqqat: 2-qadamdagi **.kartalar** uchun yozilgan **repeat(auto-fit, minmax(250px, 1fr))** qoidasi ham moslashuvchan ishlaydi. Bu qadamda ustunlar sonini media so'rovlar orqali aniq boshqarish usuli ko'rsatilgan; ikkala usulni solishtiring.

6-qadam. Turli qurilmalarda tekshirish

- 1) DevTools → Device Toolbar (Ctrl+Shift+M) da iPhone SE (375 px), iPad (768 px) va 1440 px kengliklarda sahifani ko'ring.
- 2) Har bir kenglikda menyu, kartalar soni va maket qanday o'zgarishini jadvalga yozing.
- 3) Sahifada gorizontaal aylantirish paydo bo'lmasligini tekshiring.
- 4) Haqiqiy telefonda tekshirish uchun kompyuter va telefon bitta Wi-Fi tarmog'ida bo'lsa, telefonda Live Server ko'rsatgan manzildagi **127.0.0.1** o'rniga kompyuterning lokal IP-manzilini yozing.

Element	375 px	768 px	1440 px
Navigatsiya menyusi			

Element	375 px	768 px	1440 px
Kartalar ustunlari soni			
Maket (header/aside/main)			

Mustaqil bajarish uchun topshiriqlar

A1 mashg'ulotidagi variantingiz saytiga tashqi CSS fayl orqali dizayn bering. Shartlar: CSS o'zgaruvchilari; mobile-first uslubida kamida 2 ta media so'rov (sahifa 375, 768 va 1440 px kengliklarda to'g'ri ko'rinishi kerak); Flexbox bilan navigatsiya paneli; Grid bilan kamida 6 ta kartadan iborat blok; kamida bitta **position: absolute** elementi; psevdosinflar (**:hover**, **:nth-child**); jadvalga stil (chegara, juft qatorlar rangi).

Variant	Qo'shimcha talab
1	Kitob kartalari: muqova, nom, muallif, "Mavjud/Band" belgisi
2	Grid-template-areas bilan 3 ustunli maket (chap menyu, mazmun, e'lonlar)
3	Taom kartalari, narx belgisi position: absolute bilan
4	O'quvchilar ro'yxati jadvali: sticky sarlavha va hover effekti
5	Sayohat kartalari: fon rasmi va ustiga tushgan matn
6	Narxlar jadvali (3 ta tarif kartasi, o'rtadagisi ajratilgan)
7	Mahsulot kartalari: rasm, narx, chegirma belgisi, "Savatga" tugmasi
8	Reyting jadvali: birinchi uch o'rin rang bilan ajratilgan
9	Rasmlar galereyasi: grid, turli o'lchamdagi katakchalar (span 2)
10	Gazeta maketi: sarlavha, 3 ustunli maqolalar, yon panel

Nazorat savollari

1. CSS o'zgaruvchilari qanday e'lon qilinadi va ishlatiladi?
2. **justify-content** va **align-items** ning farqi nimada?
3. **repeat(auto-fit, minmax(250px, 1fr))** ifodasini tushuntiring.
4. **grid-template-areas** qanday ishlaydi?
5. **.belgi** elementi nima uchun kartaga nisbatan joylashadi?
6. Flexbox va Grid ni qachon tanlash kerak?
7. Mobile-first yondashuvida nima uchun **min-width** ishlatiladi?
8. Sahifangiz telefonda qanday o'zgaradi va buni qaysi qoidalar ta'minlaydi?

3-AMALIY MASHG‘ULOT. JAVASCRIPT VA DOM: INTERAKTIV ELEMENTLAR YARATISH, TASHQI API‘DAN MA‘LUMOT OLISH

Maqsad: JavaScript yordamida DOM elementlarini tanlash, o‘zgartirish va hodisalarni qayta ishlash orqali interaktiv elementlar yaratish; Fetch API, `async/await` va JSON yordamida tashqi API dan ma‘lumot olib, uni sahifada ko‘rsatish.

Kutilayotgan natijalar (TN3, TN7, TN8): tinglovchi “gamburger” menyuni rejimi, akkordeon va “yuqoriga” tugmasini yaratadi; `localStorage` dan foydalanadi; tashqi API ga asinxron so‘rov yuboradi, JSON javobni qayta ishlaydi, xatoliklarni `try...catch` bilan ushlaydi va natijani Bootstrap kartalarida ko‘rsatadi.

Kerakli jihozlar: VS Code, Live Server, Chrome DevTools, Internet, A2 mashg‘ulotidagi sahifa, Open-Meteo bepul API si (kalit talab qilmaydi): open-meteo.com.

Ish shakli: juftlikda (biri kod yozadi, ikkinchisi tekshiradi, har 15 daqiqada rollar almashadi).

Qisqacha nazariy ma‘lumot

DOM elementlari `document.querySelector()` bilan tanlanadi, `textContent`, `classList`, `style` orqali o‘zgartiriladi, foydalanuvchi harakatlariga `addEventListener()` orqali javob beriladi (M3). Tashqi API ga so‘rov `fetch()` bilan yuboriladi; javob `await javob.json()` orqali JavaScript obyektiga aylantiriladi.

```
// Destrukturizatsiya
const shahar = { nom: "Namangan", lat: 41.0, lon: 71.67 };
const { nom, lat, lon } = shahar;
const [birinchi, ...qolgani] = [1, 2, 3, 4];           // rest operator
const yangi = { ...shahar, aholi: 700000 };           // spread operator

// Promise va async/await
async function olish(url) {
  try {
    const javob = await fetch(url);
    if (!javob.ok) throw new Error(`HTTP xato: ${javob.status}`);
    return await javob.json();
  } catch (xato) {
    console.error("So‘rovda xatolik:", xato.message);
  }
}
```

Ishni bajarish tartibi

I qism. DOM bilan interaktiv elementlar

1-qadam. HTML (index.html)

```
<header class="sarlavha">
  <div class="logo">Yosh<span>Dasturchi</span></div>
  <button class="gamburger" id="gamburger" aria-label="Menyu">&#9776;</button>
```

```

    <nav>
      <ul class="menu" id="menu">
        <li><a href="#">Bosh sahifa</a></li>
        <li><a href="#savollar">Savollar</a></li>
        <li><a href="#aloqa">Aloqa</a></li>
      </ul>
    </nav>
    <button id="rejim" class="rejim-tugma">🔍</button>
  </header>

<section class="banner"><h1 class="paydo">Responsive sahifa</h1></section>

<main class="konteyner">
  <div class="kartalar"> ... (A2 dagi kartalar) ... </div>

  <h2 id="savollar">Ko'p beriladigan savollar</h2>
  <div class="akkordeon">
    <div class="savol">
      <button class="savol-sarlavha">To'garak pullikmi?</button>
      <div class="javob"><p>Yo'q, to'garak mutlaqo bepul.</p></div>
    </div>
    <div class="savol">
      <button class="savol-sarlavha">Qaysi sinflar qatnashadi?</button>
      <div class="javob"><p>5-11-sinf o'quvchilari.</p></div>
    </div>
  </div>
</main>
<button id="yuqoriga" class="yuqoriga">↑</button>
<script src="js/app.js" defer></script>

```

2-qadam. Interaktiv elementlar uchun CSS

```

/* Mobile-first: kichik ekran uchun asosiy stillar */
.sarlavha { display: flex; flex-wrap: wrap; justify-content: space-between;
            align-items: center; padding: 12px 20px; background: var(--asosiy); }
.gamburger { display: block; font-size: 28px; color: white;
            background: none; border: none; cursor: pointer; }
nav { width: 100%; }
.menu { display: none; flex-direction: column; gap: 8px; list-style: none; padding:
12px 0; }
.menu.ochiq { display: flex; } /* JS shu sinfni qo'shadi */
.kartalar { display: grid; grid-template-columns: 1fr; gap: 16px; }
.banner h1 { font-size: 28px; }

/* Planshet: >= 768px */
@media (min-width: 768px) {
  .gamburger { display: none; }
  nav { width: auto; }
  .menu { display: flex; flex-direction: row; padding: 0; }
  .kartalar { grid-template-columns: repeat(2, 1fr); }
  .banner h1 { font-size: 36px; }
}

/* Kompyuter: >= 1200px */
@media (min-width: 1200px) {
  .kartalar { grid-template-columns: repeat(4, 1fr); }
  .banner h1 { font-size: 48px; }
}

```

```

/* Hover va tranzitsiya */
.karta { transition: transform 0.3s ease, box-shadow 0.3s ease; }
.karta:hover { transform: translateY(-8px) scale(1.02);
               box-shadow: 0 12px 24px rgba(0, 0, 0, 0.15); }

/* Kalit kadrli animatsiya */
@keyframes paydoBolish {
  from { opacity: 0; transform: translateY(30px); }
  to   { opacity: 1; transform: translateY(0); }
}
.paydo { animation: paydoBolish 1s ease-out; }

/* Akkordeon */
.savol-sarlavha { width: 100%; text-align: left; padding: 14px; font-size: 16px;
                  border: none; background: #e9eef7; cursor: pointer; margin-top:
8px; }
.javob { max-height: 0; overflow: hidden; transition: max-height 0.4s ease;
         background: white; padding: 0 14px; }
.savol.fao1 .javob { max-height: 200px; }

/* Yuqoriga tugmasi */
.yuqoriga { position: fixed; bottom: 24px; right: 24px; width: 48px; height: 48px;
            border-radius: 50%; border: none; background: var(--asosiy); color:
white;
            font-size: 22px; cursor: pointer; opacity: 0; pointer-events: none;
            transition: opacity 0.3s; }
.yuqoriga.korinadi { opacity: 1; pointer-events: auto; }

/* Tun rejimi */
body.tun { --fon: #121212; color: #e0e0e0; }
body.tun .karta, body.tun .javob { background: #1e1e1e; color: #e0e0e0; }
body.tun .savol-sarlavha { background: #2a2a2a; color: #e0e0e0; }

```

3-qadam. JavaScript va DOM (js/app.js)

```

// 1. Gamburger menyu
const gamburger = document.querySelector("#gamburger");
const menu = document.querySelector("#menu");
gamburger.addEventListener("click", () => {
  menu.classList.toggle("ochiq");
});

// 2. Tun rejimi (tanlov localStorage da saqlanadi)
const rejimTugma = document.querySelector("#rejim");
if (localStorage.getItem("rejim") === "tun") {
  document.body.classList.add("tun");
  rejimTugma.textContent = "☒";
}
rejimTugma.addEventListener("click", () => {
  const tunmi = document.body.classList.toggle("tun");
  rejimTugma.textContent = tunmi ? "☒" : "☑";
  localStorage.setItem("rejim", tunmi ? "tun" : "kun");
});

// 3. Akkordeon – hodisalar delegatsiyasi
document.querySelector(".akkordeon").addEventListener("click", (e) => {
  if (!e.target.matches(".savol-sarlavha")) return;
  const savol = e.target.parentElement;

```

```

document.querySelectorAll(".savol").forEach(s => {
    if (s !== savol) s.classList.remove("faol"); // boshqalarini yopish
});
savol.classList.toggle("faol");
});

// 4. "Yuqoriga" tugmasi
const yuqoriga = document.querySelector("#yuqoriga");
window.addEventListener("scroll", () => {
    yuqoriga.classList.toggle("korinadi", window.scrollY > 300);
});
yuqoriga.addEventListener("click", () => {
    window.scrollTo({ top: 0, behavior: "smooth" });
});

// 5. Kartalar soni va joriy yil
const kartalarSon = document.querySelectorAll(".karta").length;
console.log(`Sahifada ${kartalarSon} ta karta bor`);
const pastki = document.querySelector(".pastki");
if (pastki) pastki.textContent = `@ ${new Date().getFullYear()} YoshDasturchi`;

```

4-qadam. Tekshirish

- 1) DevTools → Device Toolbar (Ctrl+Shift+M) da iPhone SE (375 px), iPad (768 px) va 1440 px kengliklarda sahifani ko'ring.
- 2) 375 px kenglikda hamburger menyu ishlashini tekshiring.
- 3) Tun rejimini yoqib, sahifani yangilang – tanlov saqlanib qolishini tekshiring (Application → Local Storage).
- 4) Console da xatolar yo'qligiga ishonch hosil qiling.

II qism. Tashqi API dan ma'lumot olish: ob-havo ilovasi

Ushbu qismda Bootstrap 5 va Open-Meteo API si asosida tanlangan shaharning joriy ob-havosi va 7 kunlik prognozini ko'rsatuvchi ilova yaratiladi. Fayllar o'sha loyihaning **weather.html** va **js/weather.js** fayllariga yoziladi.

5-qadam. Ob-havo ilovasi interfeysi (weather.html)

```

<!DOCTYPE html>
<html lang="uz">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Ob-havo ilovasi</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body class="bg-light">
    <nav class="navbar navbar-dark bg-primary mb-4">
        <div class="container"><span class="navbar-brand"> Ob-havo</span></div>
    </nav>

    <div class="container">
        <div class="row justify-content-center mb-4">

```

```

        <div class="col-12 col-md-8 col-lg-6">
            <div class="input-group">
                <select id="shahar" class="form-select">
                    <option value="41.00,71.67">Namangan</option>
                    <option value="41.31,69.28">Toshkent</option>
                    <option value="39.65,66.96">Samarqand</option>
                    <option value="40.78,72.34">Andijon</option>
                    <option value="40.38,71.78">Farg'ona</option>
                    <option value="39.77,64.42">Buxoro</option>
                </select>
                <button id="korsat" class="btn btn-primary">Ko'rsatish</button>
            </div>
        </div>
    </div>

    <div id="yuklanmoqda" class="text-center d-none">
        <div class="spinner-border text-primary"></div>
    </div>
    <div id="xato" class="alert alert-danger d-none"></div>

    <div id="joriy" class="row justify-content-center mb-4"></div>
    <h4 class="text-center mb-3">7 kunlik prognoz</h4>
    <div id="prognoz" class="row g-3"></div>
</div>

<script src="js/weather.js" defer></script>
</body>
</html>

```

6-qadam. JavaScript (js/weather.js)

```

const kodlar = { // WMO ob-havo kodlari (qisqartirilgan)
    0: ["Ochiq osmon", "☀️"], 1: ["Asosan ochiq", "☁️"], 2: ["Qisman bulutli", "☁️"],
    3: ["Bulutli", "☁️"], 45: ["Tuman", "🌫️"], 61: ["Yengil yomg'ir", "🌧️"],
    63: ["Yomg'ir", "🌧️"], 71: ["Yengil qor", "❄️"], 73: ["Qor", "❄️"],
    95: ["Momaqalldiroq", "❄️"]
};

const tavsif = kod => kodlar[kod] ?? ["Noma'lum", "❓"];
const $ = sel => document.querySelector(sel);

async function obHavoOl(lat, lon) {
    const url = `https://api.open-
meteo.com/v1/forecast?latitude=${lat}&longitude=${lon}`
    +
    "&current=temperature_2m,relative_humidity_2m,wind_speed_10m,weather_code"
    +
    "&daily=weather_code,temperature_2m_max,temperature_2m_min&timezone=auto";
    const javob = await fetch(url);
    if (!javob.ok) throw new Error(`Server javobi: ${javob.status}`);
    return javob.json();
}

function joriyniChiz(nom, { current }) {
    const [matn, belgi] = tavsif(current.weather_code);
    $("#joriy").innerHTML = "";
    const ustun = document.createElement("div");
    ustun.className = "col-12 col-md-8 col-lg-6";
    ustun.innerHTML = `

```

```

        <div class="card text-center shadow-sm">
          <div class="card-body">
            <h2 class="card-title"></h2>
            <div class="display-1">${belgi}</div>
            <p class="display-5 fw-
bold">${Math.round(current.temperature_2m)}°C</p>
            <p class="text-muted">${matn}</p>
            <span class="badge bg-info me-2">Namlik:
${current.relative_humidity_2m}%</span>
            <span class="badge bg-secondary">Shamol: ${current.wind_speed_10m}
km/soat</span>
          </div>
        </div>`;
        ustun.querySelector(".card-title").textContent = nom; // matn - xavfsiz
        usulda
        $("#joriy").append(ustun);
      }

function prognozniChiz({ daily }) {
  const kunlar = ["Yak", "Dush", "Sesh", "Chor", "Pay", "Jum", "Shan"];
  $("#prognoz").innerHTML = daily.time.map((sana, i) => {
    const [matn, belgi] = tavsif(daily.weather_code[i]);
    const kun = kunlar[new Date(sana).getDay()];
    return `
      <div class="col-6 col-md-4 col-lg">
        <div class="card h-100 text-center">
          <div class="card-body p-2">
            <h6>${kun}<br><small class="text-
muted">${sana.slice(5)}</small></h6>
            <div class="fs-2">${belgi}</div>
            <small>${matn}</small>
            <p class="mb-0"><b>${Math.round(daily.temperature_2m_max[i])}°</b>
/
              ${Math.round(daily.temperature_2m_min[i])}°</p>
            </div>
          </div>
        </div>`;
    </div>`);
  }).join("");
}

async function korsat() {
  const tanlov = $("#shahar");
  const [lat, lon] = tanlov.value.split(",");
  const nom = tanlov.options[tanlov.selectedIndex].text;
  $("#xato").classList.add("d-none");
  $("#yuklanmoqda").classList.remove("d-none");
  try {
    const malumot = await obHavoOl(lat, lon);
    joriyniChiz(nom, malumot);
    prognozniChiz(malumot);
  } catch (xato) {
    $("#xato").textContent = "Ma'lumot olinmadi: " + xato.message;
    $("#xato").classList.remove("d-none");
  } finally {
    $("#yuklanmoqda").classList.add("d-none");
  }
}

$("#korsat").addEventListener("click", korsat);
korsat(); // sahifa ochilganda Namangan uchun

```

7-qadam. Tekshirish

- 1) Live Server da sahifani oching, turli shaharlarni tanlang.
- 2) F12 → Network bo‘limida **forecast?...** so‘rovini toping, Response yorlig‘ida JSON tuzilishini o‘rganing.
- 3) Internetni vaqtincha o‘chirib “Ko‘rsatish” ni bosing – xato xabari chiqishini tekshiring.
- 4) Sahifani turli ekran kengliklarida ko‘ring: prognoz kartalari telefonda 2 ta, planshetda 3 ta, kompyuterda 7 ta ustunda joylashishi kerak.

Mustaqil bajarish uchun topshiriqlar

1-topshiriq. O‘z variantingiz saytiga quyidagi jadvaldagi interaktiv elementni JavaScript yordamida qo‘shing.

Variant	Interaktiv element
1	Kitoblar ro‘yxatida nom bo‘yicha jonli qidiruv (input hodisasi)
2	Mashg‘ulot kunini tanlaganda shu kun jadvalini ko‘rsatuvchi yorliqlar (tabs)
3	Taomlar narxini hisoblovchi oddiy savat (miqdor + / –, umumiy summa)
4	O‘quvchilar ro‘yxatini familiya bo‘yicha saralash tugmasi
5	Rasmlar slayderi (oldinga/orqaga tugmalari, avtomatik almashish)
6	Modal oyna: “Darajani bilish” tugmasi bosilganda ochiladi, Esc bilan yopiladi
7	Mahsulotlarni toifa bo‘yicha filtrlash tugmalari
8	Ballar kiritilganda reytingni avtomatik qayta tartiblash
9	Rasm bosilganda kattalashtirib ko‘rsatuvchi lightbox
10	Maqola matnidagi so‘zlar sonini va o‘qish vaqtini hisoblash

2-topshiriq. Quyidagi bepul API lardan biri asosida Bootstrap da responsive ilova yarating. Talablar: **async/await**, **try...catch**, yuklanish indikatori (spinner), xato xabari (alert), natijalar Bootstrap kartalarida.

Variant	Ilova	API manzili
1	Valyuta kurslari (O‘zbekiston Markaziy banki)	cbu.uz/uz/arkhiv-kursov-valyut/json/
2	Tasodifiy foydali maslahat	api.adviceslip.com/advice
3	Mamlakatlar haqida ma’lumot (bayroq, poytaxt, aholi)	restcountries.com/v3.1/name/{nom}
4	Foydalanuvchilar ro‘yxati va qidiruv	jsonplaceholder.typicode.com/users
5	Postlar va izohlar	jsonplaceholder.typicode.com/posts

Variant	Ilova	API manzili
6	Tasodifiy foydalanuvchi profili kartasi	randomuser.me/api/
7	GitHub foydalanuvchi profili	api.github.com/users/{login}
8	Kitob qidiruvi	openlibrary.org/search.json?q={soz}
9	Universitetlar ro'yxati (davlat bo'yicha)	universities.hipolabs.com/search?country=Uzbekistan
10	Bir nechta shahar ob-havosini taqqoslash jadvali	api.open-meteo.com/v1/forecast

Nazorat savollari

1. `classList.toggle()` metodi qanday ishlaydi?
2. Akkordeonda hodisalar delegatsiyasining afzalligi nimada?
3. `localStorage` nima uchun ishlatildi?
4. `defer` atributi bo'lmasa skript qanday xatoga olib kelishi mumkin?
5. `fetch()` nima qaytaradi? `await` nima uchun kerak?
6. Nima uchun `javob.ok` alohida tekshiriladi?
7. `try...catch...finally` bloklarining vazifalari qanday?
8. Shahar nomi nima uchun `textContent` orqali qo'yildi?

4-AMALIY MASHG‘ULOT. NODE.JS VA EXPRESS.JS BILAN ODDIY SERVER VA REST API YARATISH

Maqsad: Node.js loyihasini yaratish; Express.js serverida marshrutlar va middleware sozlash; SQLite ma’lumotlar bazasi asosida to‘liq CRUD REST API yaratish va uni `fetch()` orqali frontendga ulash.

Kutilayotgan natijalar (TN5, TN9): tinglovchi NPM orqali loyiha yaratadi, Express server yozadi, o‘z middleware ini yaratadi, MB jadvallarini bog‘lanishlar bilan loyihalaydi, REST tamoyillariga mos endpoint’lar yaratadi, kiruvchi ma’lumotlarni tekshiradi va API ni frontend formasi bilan bog‘laydi.

Kerakli jihozlar: Node.js 20 LTS, VS Code, “SQLite Viewer” va “Thunder Client” kengaytmalari, terminal (VS Code: Ctrl+`).

Qisqacha nazariy ma’lumot

Express ilovasi so‘rovni middleware zanjiri orqali o‘tkazadi va mos marshrut ishlovchisiga yetkazadi. Relyatsion MB da jadvallar birlamchi va tashqi kalitlar orqali bog‘lanadi; ma’lumotlar SQL so‘rovlari bilan boshqariladi. REST API da resurs URL bilan, amal esa HTTP metodi bilan ifodalanadi: GET – o‘qish, POST – yaratish, PUT – yangilash, DELETE – o‘chirish (M4). `better-sqlite3` paketi SQLite bazasi bilan sinxron, sodda ishlash imkonini beradi.

Ishni bajarish tartibi

1-qadam. Loyihani yaratish

```
mkdir kutubxona && cd kutubxona
npm init -y
npm install express better-sqlite3
npm install -D nodemon
```

`package.json` fayliga `"type": "module"` qatorini va skriptlarni qo‘shing:

```
"type": "module",
"scripts": { "start": "node server.js", "dev": "nodemon server.js" }
```

2-qadam. Ma’lumotlar bazasini loyihalash (db.js)

“Maktab kutubxonasi” bazasi uch jadvaldan iborat: `authors` (mualliflar), `books` (kitoblar, 1:N bog‘lanish), `loans` (kitob berish qaydlari).

```
// db.js
import Database from "better-sqlite3";
const db = new Database("kutubxona.db");
db.pragma("foreign_keys = ON");

db.exec(`
CREATE TABLE IF NOT EXISTS authors (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL,
```

```

    country TEXT
  );
  CREATE TABLE IF NOT EXISTS books (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    title TEXT NOT NULL,
    year INTEGER,
    genre TEXT,
    author_id INTEGER REFERENCES authors(id)
  );
  CREATE TABLE IF NOT EXISTS loans (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    book_id INTEGER NOT NULL REFERENCES books(id),
    reader TEXT NOT NULL,
    taken_at TEXT DEFAULT (date('now')),
    returned INTEGER DEFAULT 0
  );
`);

// boshlang'ich ma'lumotlar (faqat jadval bo'sh bo'lsa)
const soni = db.prepare("SELECT COUNT(*) AS n FROM authors").get().n;
if (soni === 0) {
  const muallif = db.prepare("INSERT INTO authors (name, country) VALUES (?, ?)");
  const kitob = db.prepare(
    "INSERT INTO books (title, year, genre, author_id) VALUES (?, ?, ?, ?)");
  const toldir = db.transaction(() => {
    muallif.run("Abdulla Qodiriy", "O'zbekiston");
    muallif.run("Cho'lpon", "O'zbekiston");
    muallif.run("O'tkir Hoshimov", "O'zbekiston");
    muallif.run("Antoine de Saint-Exupéry", "Fransiya");
    kitob.run("O'tkan kunlar", 1926, "roman", 1);
    kitob.run("Mehrobdan chayon", 1929, "roman", 1);
    kitob.run("Kecha va kunduz", 1936, "roman", 2);
    kitob.run("Dunyoning ishlari", 1982, "qissa", 3);
    kitob.run("Ikki eshik orasi", 1986, "roman", 3);
    kitob.run("Kichkina shahzoda", 1943, "ertak", 4);
  });
  toldir();
}
export default db;

```

db.transaction() – bir nechta amalni bitta tranzaksiyada bajaradi: yoki hammasi bajariladi, yoki hech biri.

3-qadam. Express server, middleware va marshrutlar (server.js)

```

import express from "express";
import db from "../db.js";

const app = express();
const PORT = 3000;

// 1. O'z middleware imiz: so'rovlar jurnali va bajarilish vaqti
app.use((req, res, next) => {
  const boshlanish = Date.now();
  res.on("finish", () => {
    console.log(`${req.method} ${req.url} → ${res.statusCode} (${Date.now() - boshlanish} ms)`);
  });
  next();
});

```

```

    });
    next();
  });
  // 2. O'rnatilgan middleware lar
  app.use(express.json());
  app.use(express.static("public"));

  // 3. Marshrutlar
  app.get("/api/salom", (req, res) => {
    res.json({ xabar: "Kutubxona serveri ishlayapti", vaqt: new Date() });
  });

  // barcha kitoblar muallif nomi bilan (LEFT JOIN - muallifsiz kitoblar ham
  chiqadi), janr filtri
  app.get("/api/books", (req, res) => {
    const { genre } = req.query;
    let sql = `SELECT b.id, b.title, b.year, b.genre, a.name AS author
      FROM books b LEFT JOIN authors a ON a.id = b.author_id`;
    const params = [];
    if (genre) { sql += " WHERE b.genre = ?"; params.push(genre); }
    sql += " ORDER BY b.year";
    res.json(db.prepare(sql).all(...params));
  });

  // muallif va uning kitoblari soni (LEFT JOIN + GROUP BY)
  app.get("/api/authors", (req, res) => {
    const rows = db.prepare(`
      SELECT a.id, a.name, a.country, COUNT(b.id) AS kitoblar_soni
      FROM authors a LEFT JOIN books b ON b.author_id = a.id
      GROUP BY a.id ORDER BY kitoblar_soni DESC`).all();
    res.json(rows);
  });

  // yangi muallif qo'shish (INSERT)
  app.post("/api/authors", (req, res) => {
    const { name, country } = req.body;
    if (!name || name.trim().length < 2)
      return res.status(400).json({ error: "Muallif nomi kamida 2 belgi bo'lsin" });
    const info = db.prepare("INSERT INTO authors (name, country) VALUES (?, ?)")
      .run(name.trim(), country ?? null);
    res.status(201).json({ id: info.lastInsertRowid, name, country });
  });

  // kitob berish va qo'lda bo'lgan kitoblar
  app.post("/api/loans", (req, res) => {
    const { book_id, reader } = req.body;
    const info = db.prepare("INSERT INTO loans (book_id, reader) VALUES (?, ?)")
      .run(book_id, reader);
    res.status(201).json({ id: info.lastInsertRowid });
  });
  app.get("/api/loans/active", (req, res) => {
    res.json(db.prepare(`
      SELECT l.id, l.reader, l.taken_at, b.title
      FROM loans l JOIN books b ON b.id = l.book_id
      WHERE l.returned = 0`).all());
  });

  // 4. 404 va xatoliklarni ushlovchi middleware

```

```

app.use((req, res) => res.status(404).json({ error: "Marshrut topilmadi" }));
app.use((err, req, res, next) => {
  console.error(err);
  res.status(500).json({ error: "Serverda ichki xatolik" });
});

app.listen(PORT, () => console.log(`Server: http://localhost:${PORT}`));

```

4-qadam. Ishga tushirish va tekshirish

- 1) Terminalda `npm run dev` buyrug'ini bering. `nodemon` fayl o'zgarganda serverni avtomatik qayta ishga tushiradi.
- 2) Brauzerda quyidagi manzillarni oching:
`http://localhost:3000/api/salom, /api/books, /api/books?genre=roman, /api/authors, /api/yoq` (404).
- 3) POST so'rovlarni VS Code "Thunder Client" kengaytmasi yoki Postman orqali yuboring: `POST /api/authors`, tana: `{ "name": "Erkin Vohidov", "country": "O'zbekiston" }`.
- 4) Terminaldagi jurnal yozuvlarini kuzating.
- 5) `kutubxona.db` faylini SQLite Viewer da ochib, jadvallarni ko'ring.

5-qadam. Kitoblar uchun to'liq CRUD REST API

`server.js` faylida 404 va xatoliklarni ushlovchi middleware'lardan **oldin** quyidagi marshrutlarni qo'shing (`GET /api/books` 3-qadamda yaratilgan):

```

// kiruvchi ma'lumotni tekshirish
function tekshir({ title, year }) {
  if (!title || title.trim().length < 2 || title.length > 200)
    return "Kitob nomi 2-200 belgidan iborat bo'lsin";
  if (year && (!Number.isInteger(Number(year)) || year < 1000 || year > 2100))
    return "Yil noto'g'ri kiritilgan";
  return null;
}

// GET /api/books/:id - bitta kitob
app.get("/api/books/:id", (req, res) => {
  const kitob = db.prepare("SELECT * FROM books WHERE id = ?").get(req.params.id);
  if (!kitob) return res.status(404).json({ error: "Kitob topilmadi" });
  res.json(kitob);
});

// POST /api/books - yangi kitob
app.post("/api/books", (req, res) => {
  const xato = tekshir(req.body);
  if (xato) return res.status(400).json({ error: xato });
  const { title, year, genre, author_id } = req.body;
  const info = db.prepare(
    "INSERT INTO books (title, year, genre, author_id) VALUES (?, ?, ?, ?)"
  ).run(title.trim(), year || null, genre || null, author_id || null);
  res.status(201).json({ id: info.lastInsertRowid, title, year, genre, author_id });
});

```

```

});

// PUT /api/books/:id - kitobni yangilash
app.put("/api/books/:id", (req, res) => {
  const xato = tekshir(req.body);
  if (xato) return res.status(400).json({ error: xato });
  const { title, year, genre } = req.body;
  const info = db.prepare("UPDATE books SET title = ?, year = ?, genre = ? WHERE id = ?")
    .run(title.trim(), year || null, genre || null, req.params.id);
  if (info.changes === 0) return res.status(404).json({ error: "Kitob topilmadi" });
  res.json({ ok: true });
});

// DELETE /api/books/:id - kitobni o'chirish
app.delete("/api/books/:id", (req, res) => {
  const info = db.prepare("DELETE FROM books WHERE id = ?").run(req.params.id);
  if (info.changes === 0) return res.status(404).json({ error: "Kitob topilmadi" });
  res.status(204).end();
});

```

API ni Thunder Client orqali tekshiring:

Metod	URL	Tana (JSON)	Kutilgan javob
GET	/api/books	—	200 + massiv
GET	/api/books/1	—	200 + obyekt
GET	/api/books/999	—	404
POST	/api/books	{"title": "Sarob", "year": 1937, "author_id": 1}	201
POST	/api/books	{"title": "x"}	400
PUT	/api/books/7	{"title": "Sarob", "year": 1937}	200
DELETE	/api/books/7	—	204

6-qadam. API ni frontend bilan bog'lash

public/index.html faylini yarating:

```

<!DOCTYPE html>
<html lang="uz">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Maktab kutubxonasi</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body class="bg-light">
<div class="container py-4">
  <h1 class="h3 mb-4">📖 Maktab kutubxonasi</h1>
  <form id="kitobForma" class="row g-2 mb-3">
    <div class="col-md-5"><input name="title" class="form-control"
placeholder="Kitob nomi" required></div>
    <div class="col-md-2"><input name="year" type="number" class="form-control"
placeholder="Yil"></div>
    <div class="col-md-3"><input name="genre" class="form-control"
placeholder="Janr"></div>
    <div class="col-md-2"><button class="btn btn-success w-
100">Qo'shish</button></div>
  </form>
  <div id="xabar" class="text-danger mb-2"></div>
  <ul id="royxat" class="list-group"></ul>
</div>
<script src="app.js" defer></script>
</body>
</html>

```

public/app.js faylini yarating:

```

const $ = s => document.querySelector(s);

async function api(url, options = {}) {
  const javob = await fetch(url, {
    ...options,
    headers: { "Content-Type": "application/json" }
  });
  if (javob.status === 204) return null;
  const malumot = await javob.json();
  if (!javob.ok) throw new Error(malumot.error);
  return malumot;
}

async function chiz() {
  const kitoblar = await api("/api/books");
  $("#royxat").replaceChildren(...kitoblar.map(k => {
    const li = document.createElement("li");
    li.className = "list-group-item d-flex justify-content-between align-items-center";
    li.textContent = `${k.title}${k.year ? " (" + k.year + ")" : ""}` +
      `${k.author ? " - " + k.author : ""}`; // XSS dan himoya
    const btn = document.createElement("button");
    btn.className = "btn btn-sm btn-outline-danger";
    btn.textContent = "O'chirish";
    btn.addEventListener("click", async () => {
      await api(`/api/books/${k.id}`, { method: "DELETE" });
      chiz();
    });
  }));
}

```

```

    });
    li.append(btn);
    return li;
  }));
}

$("#kitobForma").addEventListener("submit", async (e) => {
  e.preventDefault();
  $("#xabar").textContent = "";
  try {
    const malumot = Object.fromEntries(new FormData(e.target));
    await api("/api/books", { method: "POST", body: JSON.stringify(malumot) });
    e.target.reset();
    chiz();
  } catch (xato) {
    $("#xabar").textContent = xato.message;
  }
});

chiz();

```

Brauzerda <http://localhost:3000> manzilini oching: kitob qo'shing, o'chiring, bitta harfli nom kiritib server tekshiruvi ishlashini ko'ring. Kitob nomi sifatida **Qalin** kiriting – u HTML sifatida emas, oddiy matn sifatida ko'rinishi kerak. DevTools → Network bo'limida har bir so'rov metodi, status kodi va JSON javobini kuzating.

7-qadam. SQL mashqlari (qo'shimcha)

SQLite Viewer yoki alohida skript orqali quyidagi so'rovlarni yozing va natijasini tekshiring:

- 1) 1930-yildan oldin yozilgan kitoblar.
- 2) Har bir janrdagi kitoblar soni (**GROUP BY genre**).
- 3) O'zbek mualliflarining kitoblari, yili bo'yicha kamayish tartibida.
- 4) Hech qanday kitobi bo'lmagan mualliflar (**LEFT JOIN ... WHERE b.id IS NULL**).
- 5) Eng ko'p kitobga ega muallif (**ORDER BY ... LIMIT 1**).
- 6) 5-kitobning yilini 1987 ga o'zgartiring (**UPDATE**) va tekshiring.

Mustaqil bajarish uchun topshiriqlar

Variantingiz bo'yicha kamida 3 jadvalli (1:N bog'lanishli) MB loyihalang va bitta asosiy resurs uchun to'liq CRUD REST API (GET ro'yxat, GET bitta, POST, PUT, DELETE) hamda kamida bitta JOIN li endpoint yarating. Talablar: to'g'ri status kodlari, parametrlangan SQL so'rovlari, serverda kiruvchi ma'lumot tekshiruvi, Bootstrap frontend (**public** papkada) va API hujjati (metod, URL, tana, javob kodlari) jadval ko'rinishida.

Variant	Ma'lumotlar bazasi mavzusi va jadvallari
1	Maktab: sinflar, o'quvchilar, baholar
2	Sport klubi: murabbiylar, guruhlar, a'zolar
3	Oshxona: toifalar, taomlar, buyurtmalar
4	Olimpiada: fanlar, ishtirokchilar, natijalar
5	Internet-do'kon: toifalar, mahsulotlar, buyurtmalar
6	Kurs markazi: o'qituvchilar, kurslar, tinglovchilar
7	Shifoxona navbati: shifokorlar, bemorlar, qabullar
8	Kinoteatr: filmlar, seanslar, chiptalar
9	Avtopark: haydovchilar, avtomobillar, reyslar
10	Blog: mualliflar, maqolalar, izohlar

Nazorat savollari

1. `npm init -y` va `npm install` buyruqlari nima qiladi? `nodemon` nima uchun kerak?
2. Yozgan middleware ingiz qanday ishlaydi? `next()` chaqirilmasa nima bo'ladi?
3. Xatoliklarni ushlovchi middleware boshqalardan nimasi bilan farq qiladi?
4. `/api/authors` marshrutida nima uchun `LEFT JOIN` ishlatildi?
5. API ingizdagi endpoint'lar REST tamoyillariga qanday mos keladi?
6. 201, 204, 400 va 404 status kodlarini qaysi holatlarda qaytardingiz?
7. Kiruvchi ma'lumotni nima uchun serverda ham tekshirish kerak?
8. Frontendda kitob nomlari nima uchun `textContent` orqali chiqarildi?

5-AMALIY MASHG'ULOT. YAKUNIY LOYIHA: RESPONSIVE VEB-SAYTNI GITHUB PAGES / NETLIFY GA JOYLASHTIRISH, TAQDIMOT QILISH VA HIMOYA

Maqsad: kurs davomida egallangan bilim va ko'nikmalarni yakuniy loyihada umumlashtirish; Git va GitHub orqali jamoaviy ishlash; saytni optimallashtirib, GitHub Pages yoki Netlify ga joylashtirish; loyihani taqdim etish va himoya qilish.

Kutilayotgan natijalar (TN6, TN7, TN8, TN10): tinglovchi veb-loyihani rejalashtiradi, Git ombori yaratadi, tarmoqlar (branch) va pull request orqali jamoada ishlaydi, saytni Internetda e'lon qiladi, Lighthouse ko'rsatkichlarini yaxshilaydi va loyihani himoya qiladi.

Kerakli jihozlar: avvalgi mashg'ulotlardagi loyihalar, Git (git-scm.com), GitHub va Netlify hisoblari, Chrome Lighthouse, proyektor.

Ish shakli: kichik guruhlarda (3–4 kishi). Loyiha mashg'ulotdan oldin mustaqil ta'lim doirasida tayyorlanadi, mashg'ulotda joylashtirish yakunlanadi va himoya o'tkaziladi.

Qisqacha nazariy ma'lumot

Git – fayllardagi o'zgarishlar tarixini saqlovchi taqsimlangan versiyalarni boshqarish tizimi (2005, Linus Torvalds). Asosiy tushunchalar: **repository (ombor)**, **commit** (o'zgarishlar surati), **branch** (tarmoq – parallel ishlash yo'nalishi), **merge** (birlashtirish), **remote** (masofaviy ombor, masalan GitHub), **pull request** (o'zgarishni ko'rib chiqish va birlashtirish so'rovi).

Buyruq	Vazifasi
<code>git init</code>	Joriy papkada yangi ombor yaratish
<code>git status</code>	O'zgargan fayllar holati
<code>git add .</code>	O'zgarishlarni commit uchun tayyorlash
<code>git commit -m "izoh"</code>	O'zgarishlarni tarixga yozish
<code>git log --oneline</code>	Commitlar tarixi
<code>git branch nom, git switch nom</code>	Tarmoq yaratish va unga o'tish
<code>git merge nom</code>	Tarmoqni joriy tarmoqqa birlashtirish
<code>git remote add origin URL</code>	Masofaviy omborni ulash
<code>git push, git pull</code>	O'zgarishlarni yuborish va olish
<code>git clone URL</code>	Masofaviy ombordan nusxa olish

Ishni bajarish tartibi

1-bosqich. Rejalashtirish (mashg'ulotdan oldin)

1) Guruh (3–4 kishi) mavzu tanlaydi va loyiha pasportini to'ldiradi.

2) Sahifalar va funksiyalar ro'yxati, maket eskizi (qog'ozda yoki Figma/Excalidraw da) tayyorlanadi.

3) Vazifalar a'zolar o'rtasida taqsimlanadi va GitHub omborida "Issues" sifatida yoziladi.

Loyiha pasporti bandi	Mazmuni
Loyiha nomi	
Maqsadli auditoriya	
Hal qilinadigan muammo	
Sahifalar ro'yxati	
Asosiy funksiyalar	
Texnologiyalar	
Foydalanilgan AI vositalari va maqsadi	
A'zolar va vazifalar	
Sayt havolasi va GitHub ombori	

2-bosqich. Ishlab chiqish

Loyiha quyidagi **majburiy talablarga** javob berishi kerak:

- kamida 4 ta sahifa yoki bo'lim, semantik HTML5;
- to'liq moslashuvchan dizayn (Bootstrap yoki o'z CSS ingiz, Flexbox/Grid, media so'rovlar);
- JavaScript interaktivligi (kamida 2 ta funksiya: filtr, qidiruv, forma tekshiruvi, modal va h.k.);
- tashqi API dan ma'lumot olish yoki forma ma'lumotlarini qayta ishlash;
- forma va kiruvchi ma'lumotlar tekshiruvi, XSS dan himoya (**textContent**);
- SEO meta teglari, optimallashtirilgan rasmlar, lazy loading;
- Git tarixida barcha a'zolarining committlari;
- GitHub Pages yoki Netlify da ishlab turgan versiya.

Qo'shimcha (yuqori baho uchun): o'z Express + SQLite REST API backendi (A4), tun rejimi, ko'p tillilik, Lighthouse Accessibility ≥ 95 .

AI vositalaridan foydalanish (ixtiyoriy). Bugungi kunda dasturchilar sun'iy intellekt yordamchilaridan kod yozish, xatolarni topish, kodni tushuntirish, test yozish va hujjatlashtirishda keng foydalanadi. Biroq AI tomonidan yaratilgan kodda mantiqiy xatolar, eskirgan usullar va xavfsizlik zaifliklari bo'lishi mumkin. Shu sababli **AI yordamchi, javobgarlik esa dasturchida** tamoyiliga amal qilinadi.

Samarali so'rov (prompt) tuzish qoidalari:

- **kontekst bering** – loyiha nimaligi, qaysi texnologiyalar ishlatilayotgani;

- **vazifani aniq va kichik qismlarga bo‘ling** – “butun saytni yoz” emas, “navigatsiya panelini yoz”;
- **cheklovlarni ko‘rsating** – “faqat HTML, CSS va oddiy JavaScript”, “Bootstrap 5”, “innerHTML ishlatma”;
- **natija formatini so‘rang** – “har bir qatorga izoh yoz”, “fayllar bo‘yicha alohida ber”;
- **tushuntirishni so‘rang** – “nima uchun aynan shu usulni tanlading?”;
- **iteratsiya qiling** – natijani sinab ko‘ring, xatoni AI ga qaytarib, tuzatishni so‘rang.

Prompt namunasi

Men maktab informatika to‘garagi uchun responsive sayt yaratyapman. Texnologiyalar: HTML5, CSS3, Bootstrap 5.3 (CDN), oddiy JavaScript (ES6), freymvorksiz. Vazifa: "To‘garaklar" bo‘limi uchun kartalar ro‘yxatini yarat. Ma’lumotlar JS massivida (nom, tavsif, o‘qituvchi, kunlar). Yuqorida toifa bo‘yicha filtr tugmalari bo‘lsin. Talablar: telefonda 1, planshetda 2, kompyuterda 3 ustun; DOM ga matnni faqat textContent orqali qo‘y; har bir funksiya ustiga o‘zbek tilida qisqa izoh yoz. Oxirida kod qanday ishlashini 5 ta bandda tushuntir.

AI kodini tekshirish ro‘yxati: kod ishlaydimi va konsolda xato yo‘qmi; har bir qatorni tushuntira olamanmi; **innerHTML** ga foydalanuvchi ma’lumoti tushmayaptimi; maxfiy kalitlar kodda emasmi; SQL so‘rovlari parametrlanganmi; sahifa telefonda to‘g‘ri ko‘rinadimi; Lighthouse ko‘rsatkichlari qoniqarlirmi; litsenziya va mualliflik huquqi buzilmaganmi.

Akademik halollik: himoyada loyihadagi har qanday kod qatorini tushuntira olish talab etiladi. AI dan foydalanilgan bo‘lsa, bu README faylida va taqdimotda ko‘rsatiladi (qaysi vosita, qaysi qismlar uchun).

3-bosqich. Git ni sozlash va ombor yaratish (guruh sardori)

```
git config --global user.name "Ism Familiya"
git config --global user.email "pochta@example.com"

cd maktab-sayti
git init
echo "node_modules/" > .gitignore
echo ".env" >> .gitignore
git add .
git commit -m "Saytning dastlabki versiyasi"
```

GitHub da **maktab-sayti** nomli ombor yarating va unga ulaning:

```
git branch -M main
git remote add origin https://github.com/LOGIN/maktab-sayti.git
git push -u origin main
```

Ombor sozlamalari → Collaborators bo‘limida guruh a‘zolarini taklif qiling.

4-bosqich. Jamoaviy ishlash (har bir a‘zo)

```
git clone https://github.com/LOGIN/maktab-sayti.git
cd maktab-sayti
git switch -c seo-qoshish          # o‘z vazifasi uchun yangi tarmoq
# ... fayllarni tahrirlash ...
git add .
git commit -m "SEO meta teglari qo‘shildi"
git push -u origin seo-qoshish
```

GitHub da “Compare & pull request” tugmasini bosib, pull request oching. Guruh sardori o‘zgarishlarni ko‘rib chiqadi (“Files changed”), izoh qoldiradi va “Merge” qiladi. Boshqa a‘zolar `git switch main` va `git pull` bilan yangilangan versiyani oladi.

Vazifalar taqsimoti namunasi: 1-a‘zo – sahifalar tuzilishi va SEO meta teglari; 2-a‘zo – dizayn va moslashuvchanlik; 3-a‘zo – JavaScript funksiyalari va rasmlarni optimallashtirish.

Ziddiyat (conflict): ikki kishi bitta faylning bitta qatorini o‘zgartirsa, Git `<<<<<<`, `=====`, `>>>>>>` belgilari bilan ziddiyatni ko‘rsatadi. Kerakli variant qoldirilib, belgilar o‘chiriladi, so‘ng `git add` va `git commit` bajariladi.

5-bosqich. Optimallashtirish va SEO

```
<head>
<title>“Yosh dasturchi” to‘garagi | 5-maktab, Namangan</title>
<meta name="description" content="5-maktabdagi bepul informatika to‘garaklari:
    veb-dizayn, Python, robototexnika. Jadval va ro‘yxatdan o‘tish.">
<meta property="og:title" content="“Yosh dasturchi” to‘garagi">
<meta property="og:image" content="https://LOGIN.github.io/maktab-
sayti/images/og.jpg">
<link rel="icon" href="icons/icon-192.png">
<meta name="theme-color" content="#2F5496">
</head>
...

```

Rasmlarni squoosh.app orqali WebP formatiga o‘tkazing va kerakli o‘lchamga kichraytiring. `robots.txt` va oddiy `sitemap.xml` fayllarini yarating.

6-bosqich. Hostingga joylashtirish

GitHub Pages: ombor → Settings → Pages → Source: “Deploy from a branch”, Branch: `main`, papka / (`root`) → Save. Bir-ikki daqiqadan so‘ng sayt <https://LOGIN.github.io/maktab-sayti/> manzilida ochiladi.

Netlify (muqobil): app.netlify.com → “Add new site” → “Import an existing project” → GitHub → omborni tanlang → Deploy. Har bir `git push` dan keyin sayt

avtomatik yangilanadi. Site settings → “Change site name” orqali manzilni o‘zgartirish mumkin.

7-bosqich. Lighthouse tahlili

- 1) Joylashtirilgan saytni Chrome da inkognito rejimida oching, F12 → Lighthouse → Mobile → “Analyze page load”.
- 2) Performance, Accessibility, Best Practices, SEO ballarini jadvalga yozing.
- 3) Tavsiyalarni (“Opportunities”, “Diagnostics”) o‘qib, kamida 3 ta kamchilikni tuzating.
- 4) Qayta tahlil qilib, “oldin/keyin” natijalarini solishtiring.
- 5) Saytni telefonda ochib, barcha sahifa va funksiyalar ishlashini tekshiring.

Ko‘rsatkich	Oldin	Keyin	Qilingan o‘zgarish
Performance			
Accessibility			
Best Practices			
SEO			

8-bosqich. Taqdimot va himoya (mashg‘ulotda)

Har bir guruhga 7 daqiqa taqdimot va 3 daqiqa savol-javob beriladi. Taqdimot rejasi:

- 1) Muammo va loyiha g‘oyasi (1 daqiqa).
- 2) Saytning jonli namoyishi – telefonda va kompyuterda (3 daqiqa).
- 3) Texnik yechimlar: arxitektura, eng qiziqarli kod qismi (1,5 daqiqa).
- 4) Git bilan jamoaviy ish va joylashtirish jarayoni; AI vositalaridan foydalanilgan bo‘lsa – tajriba (1 daqiqa).
- 5) Lighthouse natijalari va keyingi rivojlantirish rejalari (0,5 daqiqa).

Yakuniy loyihani baholash mezonlari

Mezon	Ulushi	Izoh
Funksionallik va majburiy talablarning bajarilishi	30%	Barcha talablar ishlaydi
Dizayn, moslashuvchanlik, foydalanish qulayligi	15%	Telefon, planshet, kompyuter
Kod sifati va xavfsizlik	15%	Tuzilish, izohlar, XSS/SQLi himoyasi
Joylashtirish, Git bilan jamoaviy ish	10%	Ishchi havola, a‘zolar commitlari
Optimallashtirish va SEO (Lighthouse)	10%	Ko‘rsatkichlar ≥ 85
Kodni tushuntira olish (AI dan foydalanilgan bo‘lsa – mas’uliyatli qo‘llash)	10%	Savollarga javob

Mezon	Ulushi	Izoh
Taqdimot va himoya	10%	Vaqt, ravonlik, jamoaviy ishtirok

Ulushlar bo'yicha jami natija 5 baholik shkalaga o'tkaziladi: 86–100% – “5”, 71–85% – “4”, 55–70% – “3”, 55% dan kam – “2”.

Yakuniy loyiha mavzulari

№	Mavzu
1	Maktab sayti: yangiliklar, o'qituvchilar, dars jadvali, qabul arizasi
2	Elektron kutubxona: kitoblar katalogi, qidiruv, kitob band qilish
3	Onlayn viktorina platformasi: savollar banki, taymer, natijalar reytingi
4	O'qituvchining shaxsiy portfoliosi va elektron dars materiallari
5	To'garak va kurslar markazi: yo'nalishlar, ro'yxatdan o'tish, sharhlar
6	Mahalla/maktab e'lonlar taxtasi (CRUD, autentifikatsiya)
7	Shaxsiy moliya hisobchisi: kirim-chiqim, diagrammalar (Chart.js)
8	Ob-havo va valyuta ma'lumotlari paneli (bir nechta API)
9	Maktab olimpiadasi natijalari tizimi
10	Turistik yo'riqnoma: Namangan viloyati diqqatga sazovor joylari, xarita

Nazorat savollari

1. `git add` va `git commit` farqi nimada? `.gitignore` nima uchun kerak?
2. Nima uchun har bir vazifa alohida tarmoqda bajariladi? Pull request nima?
3. GitHub Pages va Netlify ning farqlari nimada?
4. Lighthouse dagi qaysi tavsiyalarni bajardingiz?
5. Loyihangizning tuzilishi va arxitekturasini tushuntiring.
6. Saytingiz XSS hujumidan qanday himoyalangan?
7. Saytingiz telefonda qanday ko'rinishga moslashadi?
8. Loyihani yana qanday rivojlantirish mumkin?

VI. MUSTAQIL TA'LIM MASHG'ULOTLARI

Mustaqil ta'limni tashkil etish tartibi

Mustaqil ta'lim auditoriya mashg'ulotlarida olingan bilim va ko'nikmalarni mustahkamlash, chuqurlashtirish hamda ularni real loyihalarda qo'llash maqsadida tashkil etiladi. Fan bo'yicha 6 ta mustaqil topshiriq belgilangan: 1–4-topshiriqlar individual, 5–6-topshiriqlar (jami topshiriqlarning 1/3 qismi) kichik guruhlarda (3–4 kishi) hamkorlikda bajariladi. Har bir topshiriq **variantiv** tavsifga ega: tinglovchi taklif etilgan variantlardan birini tanlaydi yoki o'qituvchi bilan kelishgan holda o'z mavzusini taklif qiladi.

Topshirish shakli: loyiha fayllari GitHub omboriga joylanadi va havolasi o‘qituvchiga yuboriladi; 1–5-topshiriqlar uchun Internetdagi ishchi havola (GitHub Pages / Netlify yoki backend uchun lokal namoyish) ham taqdim etiladi. Topshiriq qisqa (3–5 daqiqa) og‘zaki himoya bilan qabul qilinadi. Topshirish muddatlari ishchi o‘quv dasturida ko‘rsatilgan.

Mustaqil ish uchun umumiy baholash mezon (5 baholik shkala):

Baho	Mezon
5	Barcha talablar to‘liq bajarilgan, loyiha ishlaydi va Internetda joylashgan; kod toza va izohlangan; dizayn moslashuvchan; himoyada barcha savollarga asosli javob berilgan; ijodiy yondashuv mavjud
4	Talablar asosan bajarilgan, 1–2 ta kichik kamchilik bor; himoyada savollarga asosan to‘g‘ri javob berilgan
3	Talablarning yarmidan ko‘pi bajarilgan; jiddiy kamchiliklar mavjud; himoyada javoblar to‘liq emas
2	Topshiriq bajarilmagan, ko‘chirilgan yoki tinglovchi o‘z kodini tushuntira olmaydi

1-topshiriq. Shaxsiy portfolio veb-saytini HTML va CSS yordamida yaratish va himoya qilish

Maqsad: HTML5 semantik tuzilishi va CSS dizayn ko‘nikmalarini mustahkamlash (TN2, TN7).

Talablar:

- kamida 3 ta sahifa yoki bo‘lim: “Men haqimda”, “Ishlarim/Loyihalarim”, “Aloqa”;
- semantik teglar (**header**, **nav**, **main**, **section**, **footer**);
- tashqi CSS fayl, CSS o‘zgaruvchilari, Flexbox va Grid;
- kamida 4 ta loyiha/ish kartasi, rasm (**alt** bilan);
- aloqa formasi (kamida 3 maydon, HTML5 validatsiya);
- W3C validatorida xatosiz.

Variantlar: 1) informatika o‘qituvchisi portfoliosi; 2) talaba-dasturchi portfoliosi; 3) grafik dizayner portfoliosi; 4) fotograf portfoliosi; 5) o‘qituvchining metodik ishlanmalari ko‘rgazmasi; 6) o‘z variantingiz (o‘qituvchi bilan kelishilgan holda).

Himoya savollari: sahifa tuzilishini va qaysi semantik teglar nima uchun ishlatilganini tushuntiring; kartalar maketi qanday qurilgan; spetsifiklik bo‘yicha ziddiyatga uchradingizmi va uni qanday hal qildingiz?

2-topshiriq. Responsive landing page (mahsulot yoki xizmat sahifasi) loyihasini yaratish

Maqsad: moslashuvchan dizayn tamoyillari va Bootstrap freymvorkini amalda qo'llash (TN4, TN8).

Talablar:

- bir sahifali sayt: banner (hero), afzalliklar, narxlar/tariflar, mijozlar fikri uchun joy, FAQ (akkordeon), aloqa formasi, footer;
- mobile-first yondashuv, kamida 3 ta ekran o'lchamida to'g'ri ko'rinish (375, 768, 1440 px);
- Bootstrap grid va kamida 5 xil komponent (navbar, card, accordion, modal, button, badge va h.k.);
- CSS tranzitsiya yoki animatsiya;
- optimallashtirilgan rasmlar (WebP, `loading="lazy"`).

Variantlar: 1) maktab o'quv markazi; 2) mobil ilova taqdimoti; 3) kitob do'koni; 4) IT-kurs; 5) mahalliy hunarmandchilik mahsulotlari; 6) sport zali; 7) ekoturizm xizmati; 8) o'z variantingiz.

Mijozlar fikri bo'limi uchun haqiqiy shaxslar nomidan to'qilgan sharhlar yozilmaydi: "Namuna sharh" deb belgilangan matn yoki bo'sh shablon ishlatiladi.

Himoya savollari: sahifa telefonda qanday o'zgaradi va buni qaysi sinflar/media so'rovlar ta'minlaydi; Bootstrap ning qaysi komponentlarini ishlatdingiz; sahifa yuklanish tezligini qanday oshirdingiz?

3-topshiriq. JavaScript yordamida interaktiv veb-ilova yaratish (kalkulyator, viktorina yoki to-do list)

Maqsad: JavaScript, DOM va hodisalar bilan ishlash ko'nikmasini mustahkamlash (TN3, TN7).

Umumiy talablar: kod alohida `.js` faylda, `let/const`, strekali funksiyalar, massiv metodlari (`map`, `filter`, `reduce`), hodisalar delegatsiyasi, ma'lumotlarni `localStorage` da saqlash, foydalanuvchi kiritgan matnni faqat `textContent` orqali chiqarish, moslashuvchan interfeys.

Variant	Funksional talablar
Kalkulyator	To'rt amal, qavslar yoki foiz, klaviaturadan kiritish (<code>keydown</code>), hisoblashlar tarixi (oxirgi 10 ta), nolga bo'lishni tekshirish; <code>eval()</code> ishlatilmaydi
Viktorina	Kamida 10 ta savol JS massivida (obyektlar), har savolga taymer, natija va to'g'ri javoblarni ko'rsatish, eng yaxshi natijani saqlash, savollarni aralashtirish
To-do list	Vazifa qo'shish, tahrirlash, bajarilgan deb belgilash, o'chirish; filtr (barchasi / bajarilgan / bajarilmagan); muddat va muhimlik darajasi; sahifa yangilanganda saqlanib qolish

Variant	Funksional talablar
Baholar jurnali	O'quvchi va baho qo'shish, o'rtacha ball, eng yuqori/past natija, saralash, CSV ko'rinishida eksport
Lug'at-trenajyor	Inglizcha–o'zbekcha so'zlar, kartochka rejimi, test rejimi, o'rganilgan so'zlar statistikasi

Himoya savollari: ma'lumotlar qanday tuzilmada saqlanadi; qaysi hodisalardan foydalanildi; `localStorage` bilan ishlash kodini tushuntiring; nima uchun `innerHTML` ishlatilmadi?

4-topshiriq. Tashqi API'dan foydalangan holda dinamik veb-ilova ishlab chiqish (ob-havo, valyuta yoki yangiliklar)

Maqsad: Fetch API, `async/await`, JSON bilan ishlash ko'nikmasini rivojlantirish (TN3, TN5).

Talablar:

- kamida bitta ochiq API dan ma'lumot olish (kalit talab qilinsa, u ommaviy repozitoriyga yuklanmaydi);
- yuklanish holati (spinner), xato holati, bo'sh natija holati;
- qidiruv yoki filtr imkoniyati;
- natijalarni kartalar yoki jadvalda chiqarish, kamida bitta diagramma (Chart.js) – yuqori baho uchun;
- oxirgi qidiruvlarni `localStorage` da saqlash.

Variantlar: 1) O'zbekiston viloyat markazlari ob-havosi (Open-Meteo); 2) valyuta konvertori (cbu.uz API); 3) mamlakatlar ensiklopediyasi (REST Countries); 4) kitob qidiruvi (Open Library); 5) GitHub foydalanuvchi va omborlari statistikasi; 6) Vikipediya maqolalari qidiruvi (Wikipedia API); 7) o'z variantingiz.

Himoya savollari: API so'rovi URL ini va parametrlarini tushuntiring; javob JSON tuzilishi qanday; xatoliklar qanday ushlanadi; CORS muammosiga duch keldingizmi?

5-topshiriq. REST API va ma'lumotlar bazasi bilan ishlovchi oddiy backend loyihasini yaratish (kichik guruhlarda hamkorlikda)

Maqsad: Node.js, Express, SQL va REST API asosida to'liq stekli ilova yaratish; jamoada ishlash ko'nikmasini rivojlantirish (TN5, TN6, TN9).

Guruh tarkibi va rollar (3–4 kishi): backend dasturchi (marshrutlar, MB), frontend dasturchi (interfeys, `fetch`), test va xavfsizlik bo'yicha mas'ul (API ni Thunder Client da sinash, validatsiya, XSS/SQLi tekshiruv), loyiha menejeri va hujjatlashtiruvchi (Git, README, API hujjati). Kichik guruhda rollar birlashtirilishi mumkin.

Talablar:

- kamida 3 ta bog‘langan jadval (SQLite), boshlang‘ich ma’lumotlar bilan;
- kamida bitta resurs uchun to‘liq CRUD (GET, GET/:id, POST, PUT, DELETE), kamida bitta JOIN li endpoint;
- to‘g‘ri HTTP status kodlari va JSON xato xabarlarini;
- parametrlangan SQL so‘rovlari, serverda ma’lumot tekshiruvini;
- frontendda foydalanuvchi ma’lumotlarini faqat **textContent** orqali chiqarish (XSS dan himoya);
- Bootstrap frontend (**public** papkada);
- GitHub ombori: har bir a’zoning commitlari, README va API hujjati.

Variantlar: 1) maktab elektron kundaligi (sinf, o‘quvchi, baho); 2) kutubxona (muallif, kitob, ijara); 3) to‘garaklar (to‘garak, o‘qituvchi, a’zo); 4) oshxona buyurtmalari; 5) olimpiada natijalari; 6) e’lonlar taxtasi; 7) o‘z variantingiz.

Guruh ishini baholash: guruh bahosi (loyiha natijasi) – 70%, individual hissa (Git tarixidagi commitlar, himoyadagi javoblar, o‘zaro baholash varag‘i) – 30%.

O‘zaro baholash varag‘i (har bir a’zo to‘ldiradi)	A’zo 1	A’zo 2	A’zo 3	A’zo 4
Vazifalarni o‘z vaqtida bajardi (1–5)				
Jamoaga yordam berdi, fikr almashdi (1–5)				
Ish sifati (1–5)				

6-topshiriq. Zamonaviy veb-texnologiyalar yoki freymvorklar bo‘yicha tahliliy taqdimot tayyorlash (kichik guruhlarda hamkorlikda)

Maqsad: veb-texnologiyalarning zamonaviy tendensiyalarini tahlil qilish, manbalar bilan ishlash va taqdimot ko‘nikmasini rivojlantirish (TN1, TN10).

Talablar:

- 12–15 slayddan iborat taqdimot (10 daqiqa) va 2–3 sahifali tahliliy ma’lumotnoma;
- texnologiyaning vazifasi, tarixi, afzallik va kamchiliklari, muqobillari bilan taqqoslash jadvali;
- amaliy namoyish (kichik kod namunasi yoki demo);
- ta’lim jarayonida (maktabda) qo‘llash imkoniyatlari bo‘yicha takliflar;
- kamida 5 ta ishonchli manba (rasmiy hujjatlar, ilmiy maqolalar), manbalarga havolalar;
- AI vositalaridan foydalanilgan bo‘lsa, bu ko‘rsatiladi va ma’lumotlar rasmiy manbalar bilan tekshiriladi.

Mavzular:

1. React, Vue va Angular freymvorklarining qiyosiy tahlili.
2. Tailwind CSS va Bootstrap: yondashuvlar farqi.
3. TypeScript: JavaScriptga tur tizimi.
4. Next.js va server tomonida render qilish (SSR).
5. WebAssembly va brauzerdagi yuqori unumdorlik.
6. Progressive Web Apps: imkoniyatlar va cheklovlar.
7. Headless CMS va JAMstack arxitekturasini.
8. Veb-qulaylik (accessibility) va WCAG standarti.
9. Sun'iy intellekt yordamchilarining veb-dasturlashdagi o'rni.
10. Serverless texnologiyalar va bulutli hosting (Netlify, Vercel, Firebase).
11. GraphQL va REST API ni taqqoslash.
12. Veb-xavfsizlik: OWASP Top 10 ning so'nggi versiyasi.

Baholash mezonlari: mazmunning to'liqligi va ilmiyligi – 30%; tahlil va taqqoslash chuqurligi – 25%; amaliy namoyish – 20%; taqdimot sifati va vaqtga rioya – 15%; savollarga javob – 10%.

VII. KEYSLAR BANKI

Keyslar ma'ruza va amaliy mashg'ulotlarda "Keys-stadi" metodi asosida (III bo'lim) kichik guruhlarda yechiladi. Har bir keys uchun vaziyat, tinglovchilar uchun topshiriqlar va o'qituvchi uchun yechim yo'nalishi keltirilgan.

1-keys. "Sayt ochilmayapti" (M1)

Vaziyat. Maktab sayti maktab5.uz kecha ishlagan edi. Bugun ota-onalar "Saytga kira olmayapmiz" deb murojaat qilmoqda. Ba'zilar brauzer "Bu sayt topilmadi (DNS_PROBE_FINISHED_NXDOMAIN)" xabarini, boshqalarida "Sizning ulanishingiz xavfsiz emas (NET::ERR_CERT_DATE_INVALID)" xabarini, uchinchi guruhda esa "502 Bad Gateway" sahifasini ko'rsatmoqda.

Topshiriqlar: 1) har bir xabar mijoz-server o'zaro ta'sirining qaysi bosqichidagi muammoni bildiradi? 2) har bir holat uchun ehtimoliy sabab va yechimni ko'rsating; 3) muammoni aniqlashda brauzerning qaysi vositalaridan foydalanish mumkin?

Yechim yo'nalishi: DNS xatosi – domen muddati tugagan yoki DNS yozuvlari noto'g'ri (domen registratorida tekshirish, to'lovni yangilash). Sertifikat xatosi – SSL/TLS sertifikati muddati o'tgan (Let's Encrypt sertifikatini yangilash, avtomatik yangilashni sozlash). 502 – oraliq server (Nginx) asosiy ilovadan javob ololmayapti: Node.js ilovasi to'xtagan (server jurnallarini ko'rish, ilovani qayta

ishga tushirish, jarayon menejeri – PM2). Tekshirish: DevTools → Network, status kodlari, `nslookup` buyrug'i.

2-keys. “Ko‘zi ojiz o‘quvchi va maktab sayti” (M1)

Vaziyat. Maktabda ko‘rish qobiliyati past o‘quvchi ta’lim oladi. U ekran o‘quvchi dastur (NVDA) yordamida maktab saytidan dars jadvalini topa olmasligini aytdi. Sayt kodi tekshirilganda quyidagilar aniqlandi: barcha bo‘limlar `<div>` bilan qurilgan, rasmlarda `alt` yo‘q, dars jadvali rasm (screenshot) ko‘rinishida joylangan, menyu havolalari `<div onclick="...">` bilan yasalgan, sahifada `lang` atributi ko‘rsatilmagan.

Topshiriqlar: 1) aniqlangan kamchiliklarning har biri foydalanuvchiga qanday to‘sqinlik qiladi? 2) sahifani semantik HTML5 asosida qayta tuzish rejasini tuzing; 3) dars jadvalini qulay ko‘rinishda qanday joylash kerak?

Yechim yo‘nalishi: semantik teglar (`header`, `nav`, `main`, `footer`) ekran o‘quvchiga bo‘limlar bo‘ylab tez harakatlanish imkonini beradi; `alt` matnlari; jadval rasm emas, `<table>` bilan `<caption>` va `<th scope="col">` orqali; havolalar `<a href>` bilan (klaviaturadan Tab orqali tanlanadi); `<html lang="uz">`; Lighthouse → Accessibility orqali tekshirish; WCAG tamoyillari.

3-keys. “Telefonda sayt buzilib ketadi” (M2)

Vaziyat. O‘qituvchi kompyuterda chiroyli ko‘rinadigan sayt yaratdi. O‘quvchilar telefonda ochganda: matn juda mayda, sahifani gorizontal surish kerak, uchta ustun bir-birining ustiga chiqib ketgan, rasmlar ekrandan tashqariga chiqqan. Kodda `.ustun { width: 400px; float: left; }, img { width: 1200px; }` qoidalari bor, `<head>` da viewport meta tegi yo‘q.

Topshiriqlar: 1) muammolarning sabablarini aniqlang; 2) kodni responsive dizayn tamoyillariga mos ravishda qayta yozing (CSS Grid yoki Bootstrap varianti); 3) natijani qanday tekshirasiz?

Yechim yo‘nalishi: `<meta name="viewport" content="width=device-width, initial-scale=1">` qo‘shish; qat’iy piksel kengliklar o‘rniga `max-width: 100%, %, fr; float` o‘rniga Grid/Flexbox; mobile-first media so‘rovlar yoki Bootstrap `col-12 col-md-6 col-lg-4`; DevTools Device Toolbar bilan sinash.

4-keys. “Tugma bosilsa hech narsa bo‘lmaydi” (M3)

Vaziyat. Tinglovchi quyidagi kodni yozdi, ammo tugma bosilganda natija chiqmayapti. Konsolda: `Uncaught TypeError: Cannot read properties of null (reading 'addEventListener')`.

```
<head>
  <script src="app.js"></script>
</head>
<body>
  <input id="son" type="number"> <button id="hisobla">Kvadrat</button>
  <p id="natija"></p>
</body>
```

```
// app.js
const tugma = document.querySelector("#hisoblash");
tugma.addEventListener("click", () => {
  const x = document.querySelector("#son").value;
  document.querySelector("#natija").textContent = x + x;
});
```

Topshiriqlar: 1) xatoni keltirib chiqargan barcha sabablarni toping (kamida 3 ta); 2) to‘g‘ri kodni yozing; 3) bunday xatolarning oldini olish uchun qanday odatlarni tavsiya qilasiz?

Yechim yo‘nalishi: skript `<head>` da `defer` siz ulangan – DOM hali tayyor emas; selektordagi id noto‘g‘ri (`#hisoblash` emas `#hisobla`); `value` satr qaytaradi – `x + x` satrlarni ulaydi, `Number(x) ** 2` kerak (vazifa kvadrat). Tavsiyalar: `defer`, konsolni kuzatish, `console.log` bilan oraliq tekshiruv, kiritmani songa aylantirish va `isNaN` tekshiruvi.

5-keys. “Bootstrapmi yoki o‘z CSS imizmi?” (M2)

Vaziyat. Ikki o‘qituvchi maktab sayti uchun yondashuv bo‘yicha bahslashmoqda. Birinchisi: “Bootstrap bilan bir kunda tayyor sayt qilamiz”. Ikkinchisi: “Bootstrap bilan barcha saytlar bir xil ko‘rinadi, fayl hajmi katta, o‘quvchilar CSS ni tushunmay qoladi; o‘zimiz Flexbox va Grid bilan yozamiz”.

Topshiriqlar: 1) Venn diagrammasi yordamida ikki yondashuvni taqqoslang; 2) maktab sayti va o‘quvchilarga CSS ni o‘rgatish – ikki xil maqsad uchun qaysi yondashuvni tavsiya qilasiz? 3) murosali yechim taklif qiling.

Yechim yo‘nalishi: Bootstrap – tez prototip, sinovdan o‘tgan komponentlar, jamoaviy ishda yagona uslub; kamchiliklari – o‘xshashlik, ortiqcha kod. O‘z CSS – to‘liq nazorat, kichik hajm, o‘quv maqsadida tushunish chuqurroq. Murosa: o‘quv jarayonida avval sof CSS, so‘ng Bootstrap; saytda Bootstrap + o‘z mavzu (CSS o‘zgaruvchilari orqali ranglarni moslash), faqat kerakli komponentlarni ulash.

6-keys. “Excel dan ma’lumotlar bazasiga” (M4)

Vaziyat. Maktab kutubxonachisi barcha ma’lumotlarni bitta Excel jadvalida yuritadi: “Kitob nomi | Muallif | Muallif mamlakati | O‘quvchi F.I.Sh. | Sinf | Olingan sana | Qaytarilganmi”. Bitta kitob bir necha marta berilgani uchun nomi va muallifi

ko‘p marta takrorlanadi, muallif nomi turli qatorlarda turlicha yozilgan (“A. Qodiriy”, “Abdulla Qodiriy”), qaysi kitoblar hozir qo‘lda ekanini aniqlash qiyin.

Topshiriqlar: 1) mavjud yondashuvning kamchiliklarini aniqlang; 2) relyatsion MB sxemasini loyihalang (jadvallar, maydonlar, kalitlar, bog‘lanish turlari); 3) “hozir qo‘lda bo‘lgan kitoblar va ularni olgan o‘quvchilar” so‘rovini SQL da yozing; 4) ushbu ma’lumotlar uchun REST endpoint’lar ro‘yxatini tuzing.

Yechim yo‘nalishi: takrorlanish (ortiqchalik), nomuvofiqlik, yangilash anomaliyalari. Jadvallar: `authors(id, name, country)`, `books(id, title, author_id)`, `students(id, full_name, class)`, `loans(id, book_id, student_id, taken_at, returned_at)`. So‘rov: `SELECT b.title, s.full_name, l.taken_at FROM loans l JOIN books b ON b.id = l.book_id JOIN students s ON s.id = l.student_id WHERE l.returned_at IS NULL;` Endpoint’lar: `GET/POST /api/books`, `GET /api/loans?active=true`, `POST /api/loans`, `PATCH /api/loans/:id` (qaytarish).

7-keys. “Izohlar bo‘limidagi g‘alati oyna” (M5)

Vaziyat. Maktab saytidagi yangiliklarga izoh qoldirish imkoniyati bor. Bir kuni sahifani ochgan har bir foydalanuvchida kutilmagan oyna chiqa boshladi va administrator tizimdan avtomatik chiqib ketganini sezdi. Tekshiruvda izohlardan birida HTML teglar va skript borligi, izohlar sahifaga `innerHTML` orqali qo‘shilishi, administrator sessiya cookie sida `HttpOnly` atributi yo‘qligi aniqlandi.

Topshiriqlar: 1) hujum turini aniqlang va uning mexanizmini tushuntiring; 2) qanday zarar yetkazilishi mumkin edi? 3) kamida 4 ta himoya chorasini taklif qiling va kodda ko‘rsating; 4) hodisadan keyin qanday tashkiliy choralar ko‘rilishi kerak?

Yechim yo‘nalishi: saqlanadigan (stored) XSS. Zarar: sessiya cookie sini o‘g‘irlash va administrator nomidan harakat qilish, sahifani o‘zgartirish. Himoya: chiqarishda `textContent`/ekranlash; serverda kiritmani tekshirish va tozalash; cookie ga `HttpOnly`, `Secure`, `SameSite`; CSP sarlavhasi (`helmet`). Tashkiliy: zararli izohni o‘chirish, barcha sessiyalarni bekor qilish, parollarni almashtirish, jurnallarni tahlil qilish, kodni ko‘rib chiqish amaliyotini joriy etish.

8-keys. “Parollar bazasi oshkor bo‘ldi” (M5)

Vaziyat. Kichik o‘quv markazi saytining zaxira nusxasi tasodifan ochiq papkada qolib ketdi. Bazadagi `users` jadvalida parollar ochiq holda saqlangan. Ko‘plab foydalanuvchilar xuddi shu parolni elektron pochta va ijtimoiy tarmoqlarida ham ishlatadi. Bundan tashqari, kodda `const SECRET = "12345"` qatori topildi va u GitHub dagi ochiq omborga yuklangan.

Topshiriqlar: 1) qanday xatolarga yo‘l qo‘yilgan? 2) foydalanuvchilar uchun qanday xavflar paydo bo‘ldi? 3) to‘g‘ri arxitekturaning taklif qiling: parollar, maxfiy kalitlar va zaxira nusxalar qanday saqlanishi kerak? 4) foydalanuvchilarga qanday tavsiyalar berasiz?

Yechim yo‘nalishi: parollar faqat bcrypt/Argon2 xeshi ko‘rinishida; maxfiy kalit uzun tasodifiy satr, `.env` da, `.gitignore` ga qo‘shilgan, oshkor bo‘lgan kalitni darhol almashtirish (eski kalit bilan berilgan barcha sessiya va tokenlar bekor bo‘ladi); zaxira nusxalar shifrlangan va ochiq papkadan tashqarida; minimal huquqlar. Foydalanuvchilarga: parolni almashtirish, har bir xizmat uchun alohida parol, parol menejeri va ikki bosqichli autentifikatsiya.

9-keys. “Sayt sekin yuklanadi va Google da ko‘rinmaydi” (M5)

Vaziyat. Maktab saytining bosh sahifasi telefonda 9 soniyada yuklanadi. Lighthouse natijalari: Performance – 28, SEO – 61. Tahlil: bosh sahifada 12 ta 4–6 MB hajmli JPEG fotosurat; 5 ta JavaScript kutubxonasi `<head>` da `defer` siz ulangan, ulardan 3 tasi ishlatilmaydi; barcha sahifalarning `<title>` i “Bosh sahifa”; `meta description` yo‘q; URL lar `index.php?page=17` ko‘rinishida.

Topshiriqlar: 1) Core Web Vitals ko‘rsatkichlariga qaysi muammolar ta’sir qiladi? 2) muhimlik tartibida optimallashtirish rejasini tuzing; 3) SEO bo‘yicha kamida 5 ta tuzatishni taklif qiling.

Yechim yo‘nalishi: rasmlarni WebP ga o‘tkazish va kichraytirish, `srcset`, `loading="lazy"`, `width/height` (LCP, CLS); keraksiz skriptlarni olib tashlash, qolganlariga `defer` (LCP, INP); Gzip/Brotli va keshlash. SEO: noyob `title` va `description`, tushunarli URL, bitta `h1`, `alt` matnlar, `sitemap.xml`, `robots.txt`, Google Search Console ga ulash.

10-keys. “Sayt kompyuterda ishlaydi, hostingda – yo‘q” (M5, A5)

Vaziyat. Tinglovchilar guruhi yakuniy loyihani GitHub Pages ga joylashtirdi. Kompyuterda Live Server orqali hammasi ishlagan edi, ammo Internetdagi manzilda: logotip va ba’zi rasmlar ko‘rinmaydi; “Biz haqimizda” havolasi 404 xatosini beradi; stillar umuman yuklanmagan; ob-havo bloki ishlamaydi. Tekshiruvda: HTML da ``, papkada esa `images/logo.png`; havola `href="/about.html"`, sayt esa `https://login.github.io/maktab-sayti/` manzilida; CSS `C:\Users\Ali\Desktop\sayt\style.css` yo‘li bilan ulangan; API so‘rovi `http://` orqali yuborilgan; omborga `.env` fayli ham yuklangan.

Topshiriqlar: 1) har bir muammoning sababini aniqlang; 2) to‘g‘ri yo‘llar va sozlamalarni yozing; 3) joylashtirishdan oldin bajariladigan tekshiruv ro‘yxatini

tuzing; 4) `.env` fayli omborga tushib qolgani qanday xavf tug‘diradi va nima qilish kerak?

Yechim yo‘nalishi: Linux serverlar fayl nomidagi katta-kichik harflarni farqlaydi – nomlarni bir xil (kichik harflarda) yozish; GitHub Pages da sayt pastki papkada joylashgani uchun `/about.html` domen ildiziga ishora qiladi – nisbiy yo‘l `about.html` yoki `./about.html` ishlatish; kompyuterdagi absolyut fayl yo‘li (`C:\...`) Internetda mavjud emas – `css/style.css`; HTTPS sahifadan `http://` so‘rovlar bloklanadi (mixed content) – `https://` ishlatish; `.env` ni `.gitignore` ga qo‘shish, ombordan o‘chirish va undagi barcha kalit/parollarni darhol almashtirish (Git tarixida saqlanib qoladi). Tekshiruv: M5 dagi “Joylashtirishdan oldingi tekshiruv ro‘yxati”, DevTools → Console va Network.

11-keys. “AI yozgan kod” (M5, A5)

Vaziyat. Tinglovchi AI yordamchisidan “Express da login sahifasi yozib ber” deb so‘radi va olingan kodni o‘zgartirmasdan loyihasiga qo‘ydi. Kod ishlaydi, lekin unda: SQL so‘rov satrlarni ulash orqali tuzilgan; parollar bazada ochiq matn ko‘rinishida solishtiriladi; xato bo‘lganda foydalanuvchiga to‘liq SQL xato matni chiqariladi; Express ning juda eski versiyasi ko‘rsatilgan.

Topshiriqlar: 1) kodning xavfli joylarini aniqlang va tuzating; 2) AI ga qanday so‘rov (prompt) berilganda xavfsizroq natija olish mumkin edi? 3) AI kodini loyihaga qo‘shishdan oldin tekshirish tartibini ishlab chiqing.

Yechim yo‘nalishi: parametrlangan so‘rovlar, bcrypt, umumiy xato xabarlari, `npm install express@latest` va `npm audit`. Prompt: texnologiyalar versiyasi, xavfsizlik talablari (bcrypt, parametrlangan so‘rov, helmet), izohlar va tushuntirish so‘rash. Tekshirish ro‘yxati: A5 mashg‘ulotidagi “AI kodini tekshirish ro‘yxati”.

VIII. GLOSSARIY

№	O‘zbekcha	Inglizcha	Ruscha	Izoh
1	Animatsiya	Animation	Анимация	CSS da <code>@keyframes</code> yordamida kadrlar asosida xossalarning vaqt bo‘yicha o‘zgarishi
2	API	Application Programming Interface	Программный интерфейс приложения	Dasturlarning o‘zaro muloqot qilish qoidalari va interfeysi; veb-API HTTP orqali ishlaydi
3	Asinxronlik	Asynchrony	Асинхронность	Uzoq davom etadigan amal (tarmoq so‘rovi) natijasini kutmasdan boshqa kodni bajarish imkoniyati

№	O'zbekcha	Inglizcha	Ruscha	Izoh
4	Atribut	Attribute	Атрибут	HTML elementining qo'shimcha xususiyati: href , src , alt , class
5	Autentifikatsiya	Authentication	Аутентификация	Foydalanuvchi kimligini tasdiqlash jarayoni (parol, kod, biometriya)
6	Avtorizatsiya	Authorization	Авторизация	Tasdiqlangan foydalanuvchining ma'lum amallarni bajarishga ruxsatini aniqlash
7	Backend	Backend	Бэкенд	Veb-ilovanning serverda bajariladigan qismi: mantiq, MB, API
8	Bootstrap	Bootstrap	Bootstrap	12 ustunli grid va tayyor komponentlarga ega mashhur CSS/JS freymvork
9	Box modeli	Box model	Блочная модель	Elementni content, padding, border va margin qatlamlaridan iborat quti sifatida ifodalash
10	Brauzer	Web browser	Веб-браузер	Veb-resurslarni so'rab, ularni foydalanuvchiga ko'rsatuvchi dastur
11	CDN	Content Delivery Network	Сеть доставки контента	Fayllarni foydalanuvchiga eng yaqin serverdan yetkazuvchi tarmoq
12	Commit	Commit	Коммит	Git omboridagi o'zgarishlarning izohli surati; loyiha tarixining bir nuqtasi
13	Cookie	Cookie	Куки	Server brauzerda saqlaydigan va har so'rovda qaytariladigan kichik matnli ma'lumot
14	CORS	Cross-Origin Resource Sharing	Совместное использование ресурсов между источниками	Boshqa domendagi resursga brauzer so'rovlariga ruxsat berish mexanizmi
15	CRUD	Create, Read, Update, Delete	CRUD	Ma'lumotlar ustidagi to'rt asosiy amal: yaratish, o'qish, yangilash, o'chirish
16	CSP	Content Security Policy	Политика безопасности контента	Sahifada qaysi manbalardan skript va resurslar yuklanishi mumkinligini belgilovchi sarlavha
17	CSRF	Cross-Site Request Forgery	Межсайтовая подделка запроса	Foydalanuvchi nomidan uning brauzeri orqali soxta so'rov yuborish hujumi

№	O'zbekcha	Inglizcha	Ruscha	Izoh
18	CSS	Cascading Style Sheets	Каскадные таблицы стилей	Veb-sahifa ko'rinishini tavsiflovchi til
19	DNS	Domain Name System	Система доменных имён	Domen nomlarini IP-manzillarga aylantiruvchi tizim
20	DOM	Document Object Model	Объектная модель документа	HTML hujjatning brauzerdagi daraxtsimon obyekt modeli va uni boshqarish interfeysi
21	Domen	Domain name	Доменное имя	Saytning inson uchun tushunarli nomi: namdpi.uz
22	Element	Element	Элемент	Ochuvchi teg, mazmun va yopuvchi tegdan iborat HTML tuzilma birligi
23	Endpoint	Endpoint	Конечная точка	API dagi HTTP metodi va URL kombinatsiyasi: GET /api/books
24	Express.js	Express.js	Express.js	Node.js uchun minimalistik veb-freymvork
25	Fetch API	Fetch API	Fetch API	Brauzerda HTTP so'rovlarini yuborish uchun zamonaviy JavaScript interfeysi
26	Flexbox	Flexible Box Layout	Флексбокс	Elementlarni bir o'lchovda joylashtiruvchi CSS modeli
27	Freymvork	Framework	Фреймворк	Ilova yaratish uchun tayyor tuzilma, qoidalar va komponentlar to'plami
28	Frontend	Frontend	Фронтенд	Veb-ilovaning foydalanuvchi brauzerida ishlaydigan qismi
29	Funksiya	Function	Функция	Nom berilgan, qayta ishlatiladigan kod bloki
30	Git	Git	Git	Taqsimlangan versiyalarni boshqarish tizimi
31	GitHub Pages	GitHub Pages	GitHub Pages	GitHub omboridagi statik saytni bepul e'lon qiluvchi hosting xizmati
32	Grid (CSS)	CSS Grid Layout	CSS Grid	Ikki o'lchovli (qator va ustun) joylashtirish modeli
33	Hodisa	Event	Событие	Sahifada sodir bo'ladigan voqea: bosish, kiritish, yuklanish
34	Hosting	Hosting	Хостинг	Saytni Internetda doimiy ishlab turadigan serverga joylashtirish xizmati

№	O'zbekcha	Inglizcha	Ruscha	Izoh
35	HTML	HyperText Markup Language	Язык гипертекстовой разметки	Veb-sahifa tuzilishi va mazmunini belgilash tili
36	HTTP	HyperText Transfer Protocol	Протокол передачи гипертекста	Mijoz va server o'rtasida ma'lumot uzatish protokoli
37	HTTPS	HTTP Secure	Защищённый HTTP	TLS bilan shifrlangan xavfsiz HTTP
38	IP-manzil	IP address	IP-адрес	Tarmoqdagi qurilmaning raqamli identifikatori
39	JavaScript	JavaScript	JavaScript	Brauzer va serverda bajariladigan dinamik dasturlash tili
40	JOIN	JOIN	Соединение (JOIN)	Bir nechta jadvallarni umumiy maydon bo'yicha bog'lovchi SQL amali
41	JSON	JavaScript Object Notation	JSON	Ma'lumot almashishning yengil matnli formati
42	Kesh	Cache	Кэш	Qayta foydalanish uchun vaqtincha saqlangan ma'lumotlar nusxasi
43	Lazy loading	Lazy loading	Ленивая загрузка	Resurslarni faqat kerak bo'lganda (ekranga yaqinlashganda) yuklash
44	Ma'lumotlar bazasi	Database	База данных	Tartiblangan holda saqlanadigan o'zaro bog'liq ma'lumotlar to'plami
45	Media so'rov	Media query	Медиазапрос	Ekran xususiyatiga qarab CSS qo'llash sharti: <code>@media (min-width: 768px)</code>
46	Meta teg	Meta tag	Мета-тег	Sahifa haqidagi xizmat ma'lumotlarini beruvchi <code><meta></code> elementi
47	Middleware	Middleware	Промежуточное ПО	So'rov va javob orasida bajariladigan funksiyalar zanjiri
48	Mijoz	Client	Клиент	Serverdan xizmat so'rovchi dastur (brauzer)
49	Minifikatsiya	Minification	Минификация	Kod hajmini kamaytirish uchun bo'shliq va izohlarni olib tashlash
50	Netlify	Netlify	Netlify	Git omboriga ulanib, saytni avtomatik joylashtiruvchi bulutli hosting platformasi

№	O'zbekcha	Inglizcha	Ruscha	Izoh
51	Node.js	Node.js	Node.js	V8 dvigateli asosidagi serverda JavaScript bajarish muhiti
52	NoSQL	NoSQL	NoSQL	Jadvalsiz (hujjatli, kalit-qiyamat va b.) ma'lumotlar bazalari
53	NPM	Node Package Manager	Менеджер пакетов Node	Node.js paketlarini o'rnatish va boshqarish vositasi
54	Optimallashtirish	Optimization	Оптимизация	Sayt tezligi va samaradorligini oshirish choralari
55	Parametrlangan so'rov	Parameterized query	Параметризованный запрос	Qiymatlar SQL matnidan alohida uzatiladigan so'rov; SQL Injection dan himoya
56	Port	Port	Порт	Kompyuterdagi tarmoq xizmatining raqami: HTTP – 80, HTTPS – 443
57	Pozitsiyalash	Positioning	Позиционирование	Elementni position xossasi yordamida joylashtirish
58	Relyatsion MB	Relational database	Реляционная база данных	Ma'lumotlar bog'langan jadvallarda saqlanadigan MB
59	Render	Rendering	Отрисовка	Brauzerning HTML va CSS asosida sahifani ekranda chizishi
60	Repozitoriy (ombor)	Repository	Репозиторий	Loyiha fayllari va ularning o'zgarishlar tarixi saqlanadigan Git ombori
61	Responsive dizayn	Responsive web design	Адаптивный дизайн	Sahifaning turli ekran o'lchamlariga moslashishi
62	REST	Representational State Transfer	REST	Resurslar va HTTP metodlariga asoslangan veb-xizmatlar arxitektura uslubi
63	Routing (marshrutlash)	Routing	Маршрутизация	URL va HTTP metodiga qarab so'rovni tegishli ishlovchiga yo'naltirish
64	Selektor	Selector	Селектор	CSS qoidasi qo'llanadigan elementlarni tanlash ifodasi
65	Semantik teg	Semantic element	Семантический тег	Mazmun ma'nosini bildiruvchi HTML5 tegi: header , article
66	SEO	Search Engine Optimization	Поисковая оптимизация	Saytning qidiruv tizimlaridagi o'rnini yaxshilash
67	Server	Server	Сервер	So'rovlarni qabul qilib, javob qaytaruvchi dastur yoki kompyuter

№	O'zbekcha	Inglizcha	Ruscha	Izoh
68	Sessiya	Session	Сессия	Foydalanuvchi holatini serverda saqlash mexanizmi
69	Spetsifiklik	Specificity	Специфичность	Bir nechta CSS qoidasi ziddiyatida qaysi biri ustunligini belgilovchi ko'rsatkich
70	SQL	Structured Query Language	Язык структурированных запросов	Relyatsion MB lar bilan ishlash uchun so'rovlar tili
71	SQL Injection	SQL Injection	SQL-инъекция	Kiritma orqali SQL so'rov mantiqini o'zgartirish hujumi
72	Status kodi	Status code	Код состояния	So'rov natijasini bildiruvchi uch xonali raqam: 200, 404, 500
73	Teg	Tag	Тег	Burchakli qavsdagi HTML belgilash kalit so'zi
74	TLS	Transport Layer Security	Протокол защиты транспортного уровня	Tarmoqdagi ma'lumotlarni shifrlash protokoli
75	Tranzitsiya	Transition	Переход	CSS xossasi qiymatining silliq o'zgarishi
76	URL	Uniform Resource Locator	Унифицированный указатель ресурса	Internetdagi resursning yagona manzili
77	Validatsiya	Validation	Валидация	Kiritilgan ma'lumotlarning to'g'riligini tekshirish
78	Veb-server	Web server	Веб-сервер	HTTP so'rovlariga javob beruvchi dastur: Nginx, Apache
79	Viewport	Viewport	Область просмотра	Brauzer oynasidagi sahifaning ko'rinadigan qismi
80	WWW	World Wide Web	Всемирная паутина	Internet orqali ishlovchi gipermatnli hujjatlar tizimi
81	XSS	Cross-Site Scripting	Межсайтовый скриптинг	Sahifaga zararli skript joylashtirish hujumi

IX. TEST SAVOLLARI

Test savollari tinglovchilarning nazariy bilimlarini joriy nazorat va o'z-o'zini tekshirish maqsadida baholash uchun mo'ljallangan. Har bir savolga to'rtta javob varianti berilgan, ulardan faqat bittasi to'g'ri. Javoblar kaliti bo'lim oxirida keltirilgan. Testdan joriy nazoratda foydalanilganda 25 ta savoldan iborat variant tuziladi (har bir mavzudan 5 tadan); to'g'ri javoblar ulushi 86% va undan yuqori – “5”, 71–85% – “4”, 55–70% – “3”, 55% dan kam – “2”.

M1. Web texnologiyalari, Internet asoslari va HTML5

1. World Wide Web (vab) qaysi tashkilotda va kim tomonidan taklif qilingan?
 - A) Microsoft, Brendan Eich
 - B) Google, Ryan Dahl
 - C) MIT, Bill Geyts
 - D) CERN, Tim Berners-Li
2. Domen nomini IP-manzilga aylantiruvchi tizim qanday ataladi?
 - A) DNS
 - B) TLS
 - C) FTP
 - D) HTTP
3. HTTPS protokoli standart holatda qaysi portda ishlaydi?
 - A) 443
 - B) 80
 - C) 21
 - D) 3000
4. HTTP protokolining "holatsiz" (stateless) bo'lishi nimani anglatadi?
 - A) Ulanish hech qachon uzilmaydi
 - B) Ma'lumot shifrlanmaydi
 - C) Server har bir so'rovni oldingi so'rovlardan mustaqil qayta ishlaydi
 - D) Server so'rovlarga javob bermaydi
5. 404 status kodi nimani bildiradi?
 - A) So'rov muvaffaqiyatli bajarildi
 - B) Resurs boshqa manzilga ko'chirildi
 - C) Server ichki xatosi
 - D) So'ralgan resurs topilmadi
6. Qaysi HTTP metodi odatda yangi resurs yaratish uchun ishlatiladi?
 - A) DELETE
 - B) GET
 - C) POST
 - D) HEAD
7. <https://namdpi.uz/news?page=2> URL idagi [?page=2](#) qismi nima deb ataladi?
 - A) Fragment
 - B) Domen
 - C) So'rov satri (query string)
 - D) Protokol

8. Mijoz-server arxitekturasida vebdagi mijoz vazifasini odatda nima bajaradi?
- A) Veb-brauzer
 - B) Marshrutizator
 - C) Ma'lumotlar bazasi
 - D) DNS-server
9. Brauzer HTML koddan qanday daraxtsimon model quradi?
- A) CSSOM
 - B) DOM
 - C) JSON
 - D) URL
10. Chrome brauzerining render dvigateli qaysi?
- A) SpiderMonkey
 - B) Gecko
 - C) Blink
 - D) WebKit
11. Qaysi status kodlar guruhi server xatolarini bildiradi?
- A) 4xx
 - B) 5xx
 - C) 2xx
 - D) 3xx
12. HTTPS qanday kafolatlarni ta'minlaydi?
- A) Faqat keshlashni
 - B) Faqat sahifa tezligini
 - C) Maxfiylik, yaxlitlik va server autentifikatsiyasi
 - D) Faqat SEO ni
13. Web 2.0 bosqichining asosiy xususiyati nima?
- A) Foydalanuvchilar o'zlari kontent yaratadi
 - B) Faqat matnli brauzerlar
 - C) Faqat statik sahifalar mavjud
 - D) Blokcheyn asosidagi markazlashmagan veb
14. Brauzerda sahifa chizilishining to'g'ri ketma-ketligi qaysi?
- A) CSSOM → paint → DOM → layout
 - B) Layout → DOM → paint → CSSOM
 - C) Paint → layout → DOM → CSSOM
 - D) DOM va CSSOM → render daraxti → layout → paint
15. HTML5 hujjati qaysi deklaratsiya bilan boshlanadi?
- A) <!HTML>

- B) <?xml version="1.0"?>
C) <html5>
D) <!DOCTYPE html>
- 16.** O'zbek harflari to'g'ri ko'rinishi uchun qaysi meta teg zarur?
A) <meta charset="UTF-8">
B) <meta name="viewport">
C) <meta name="author">
D) <meta http-equiv="refresh">
- 17.** Rasm yuklanmaganda ko'rindigan va ekran o'quvchi dasturlar o'qiydigan matn qaysi atributda beriladi?
A) alt
B) src
C) title
D) href
- 18.** Havola yaratish uchun qaysi teg ishlatiladi?
A) <link>
B) <url>
C) <href>
D) <a>
- 19.** Tartiblangan (raqamlangan) ro'yxat qaysi teg bilan yaratiladi?
A)
B) <dl>
C)
D)
- 20.** Jadval katakchasini gorizontal birlashtirish uchun qaysi atribut ishlatiladi?
A) cellpadding
B) colspan
C) rowspan
D) merge
- 21.** Sahifaning asosiy, noyob mazmuni uchun mo'ljallangan semantik teg qaysi?
A) <section>
B) <aside>
C) <main>
D) <div>
- 22.** Mustaqil, alohida tarqatilishi mumkin bo'lgan mazmun (yangilik, maqola) uchun qaysi teg mos?
A) <article>

- B) <nav>
 - C) <footer>
 - D)
23. Quyidagilardan qaysi biri qatordagi (inline) element?
- A)
 - B) <div>
 - C) <section>
 - D) <p>
24. Video elementida boshqaruv panelini ko'rsatish uchun qaysi atribut kerak?
- A) controls
 - B) autoplay
 - C) loop
 - D) poster
25. E-pochta manzilini tekshiruvchi maydon qaysi **input** turi bilan yaratiladi?
- A) type="text"
 - B) type="email"
 - C) type="mail"
 - D) type="address"
26. Forma maydonini to'ldirish majburiy ekanini qaysi atribut bildiradi?
- A) disabled
 - B) required
 - C) placeholder
 - D) readonly
27. **** havolasidagi yo'l qanday turga kiradi?
- A) Fragment
 - B) Absolyut yo'l
 - C) Protokol
 - D) Nisbiy yo'l
28. Mobil qurilmalarda sahifa kengligini ekranga moslovchi teg qaysi?
- A) <link rel="icon">
 - B) <meta name="viewport" content="width=device-width, initial-scale=1">
 - C) <meta charset="UTF-8">
 - D) <base href="/">

M2. CSS, responsive dizayn va Bootstrap

29. Butun sayt uchun CSS ni ulashning eng to'g'ri usuli qaysi?
- A) Tashqi fayl (<link rel="stylesheet">)
 - B) style atributi

- C) tegi
D) Har bir sahifada <style>
30. **.karta** selektori nimani tanlaydi?
A) class="karta" bo'lgan elementlarni
B) id="karta" bo'lgan elementni
C) <karta> teglarini
D) Barcha elementlarni
31. Qaysi selektor spetsifikligi eng yuqori?
A) .menu
B) #menu
C) nav
D) *
32. **ul > li** selektori nimani tanlaydi?
A) ul ning bevosita farzandi bo'lgan li larni
B) Faqat birinchi li ni
C) ul ichidagi barcha li larni
D) ul dan keyingi li ni
33. Box modelida mazmun va chegara orasidagi ichki chekinish qanday ataladi?
A) outline
B) border
C) margin
D) padding
34. **width: 200px; padding: 10px; border: 5px solid;** bo'lgan elementning **content-box** holatidagi umumiy kengligi qancha?
A) 215px
B) 200px
C) 230px
D) 220px
35. **box-sizing: border-box** qanday ta'sir qiladi?
A) Element ko'rinmay qoladi
B) margin ni nolga tenglaydi
C) width ga padding va border ham kiradi
D) Chegarani olib tashlaydi
36. Qaysi xossa meros bo'lib o'tadi?
A) border
B) color
C) margin

- D) padding
37. `display: none` va `visibility: hidden` farqi nimada?
- A) `display: none` da element joy egallamaydi
 - B) `display: none` faqat matnga ta'sir qiladi
 - C) `visibility: hidden` da element joy egallamaydi
 - D) Farqi yo'q
38. Aylantirganda ham oynaning belgilangan joyida qoladigan element uchun qaysi qiymat ishlatiladi?
- A) `position: relative`
 - B) `position: fixed`
 - C) `position: inherit`
 - D) `position: static`
39. `rgba(0, 0, 0, 0.5)` dagi `0.5` nimani bildiradi?
- A) Ko'k rang ulushi
 - B) Yorqinlik
 - C) Rang tonini
 - D) Shaffoflik darajasi
40. Qaysi o'lchov birligi ildiz (html) element shrift o'lchamiga nisbatan hisoblanadi?
- A) em
 - B) vh
 - C) px
 - D) rem
41. Sichqoncha element ustiga kelgandagi holat uchun qaysi psevd-sinf ishlatiladi?
- A) `:visited`
 - B) `:active`
 - C) `:hover`
 - D) `:focus`
42. Flexbox konteynerida elementlarni asosiy o'q bo'yicha taqsimlovchi xossa qaysi?
- A) `justify-content`
 - B) `align-items`
 - C) `grid-gap`
 - D) `flex-wrap`
43. `grid-template-columns: repeat(3, 1fr)` qanday natija beradi?
- A) Bitta ustun 3 marta takrorlanadi
 - B) 3px kenglikdagi ustunlar
 - C) Uchta teng kenglikdagi ustun

- D) Uchta qator
44. Bootstrap grid tizimida qator nechta ustunga bo‘linadi?
- A) 10
 - B) 12
 - C) 16
 - D) 24
45. Bootstrap da `col-md-6` sinfi qanday ekran kengligidan boshlab ta’sir qiladi?
- A) $\geq 992\text{px}$
 - B) $\geq 768\text{px}$
 - C) Barcha ekranlarda
 - D) $\geq 576\text{px}$
46. Bootstrap grid tuzilishining to‘g‘ri ketma-ketligi qaysi?
- A) grid $\rightarrow$ col $\rightarrow$ row
 - B) row $\rightarrow$ container $\rightarrow$ col
 - C) container $\rightarrow$ row $\rightarrow$ col
 - D) col $\rightarrow$ row $\rightarrow$ container
47. Mobile-first yondashuvida media so‘rovlarda asosan qaysi shart ishlatiladi?
- A) orientation
 - B) resolution
 - C) min-width
 - D) max-width

M3. JavaScript, DOM, Fetch API va JSON

48. Qiymatini qayta o‘zgartirib bo‘lmaydigan o‘zgaruvchi qaysi kalit so‘z bilan e’lon qilinadi?
- A) var
 - B) const
 - C) let
 - D) static
49. `5 === "5"` ifodasining natijasi nima?
- A) undefined
 - B) false
 - C) TypeError
 - D) true
50. `"5" + 3` ifodasining natijasi nima?
- A) 2
 - B) NaN
 - C) 8

D) "53"

51. `typeof null` qanday natija qaytaradi?

A) "number"

B) "undefined"

C) "null"

D) "object"

52. Massivning har bir elementidan yangi massiv hosil qiluvchi metod qaysi?

A) `push()`

B) `forEach()`

C) `indexOf()`

D) `map()`

53. Strelkali funksiyaning to'g'ri yozilishi qaysi?

A) `const f = x => x * 2;`

B) `const f = x -> x * 2;`

C) `const f = (x) { return x * 2 }`

D) `function => f(x) { }`

54. CSS selektori bo'yicha birinchi mos elementni qaytaruvchi metod qaysi?

A) `document.querySelectorAll()`

B) `document.querySelector()`

C) `document.getElementsByTagName()`

D) `document.write()`

55. Foydalanuvchi kiritgan matnni sahifaga xavfsiz chiqarish uchun qaysi xossa ishlatiladi?

A) `outerHTML`

B) `textContent`

C) `innerHTML`

D) `insertAdjacentHTML`

56. Elementga sinf qo'shish yoki mavjud bo'lsa olib tashlash uchun qaysi metod ishlatiladi?

A) `className.replace()`

B) `classList.toggle()`

C) `setAttribute()`

D) `classList.add()`

57. Hodisa ishlovchisini qo'shish uchun qaysi metod ishlatiladi?

A) `addEventListener()`

B) `attachEvent()`

C) `onEvent()`

- D) listen()
58. Forma yuborilganda sahifa yangilanishini to'xtatish uchun nima chaqiriladi?
- A) e.stopPropagation()
 - B) e.preventDefault()
 - C) return true
 - D) form.reset()
59. Hodisa sodir bo'lgan elementga hodisa obyektining qaysi xossasi orqali murojaat qilinadi?
- A) e.timeStamp
 - B) e.key
 - C) e.type
 - D) e.target
60. Skript HTML to'liq tahlil qilingandan keyin bajarilishi uchun `<script>` ga qaysi atribut qo'shiladi?
- A) async
 - B) type="module"
 - C) nomodule
 - D) defer
61. Yangi HTML element yaratuvchi metod qaysi?
- A) document.createElement()
 - B) document.appendChild()
 - C) document.addElement()
 - D) document.createElement()
62. JSON formatida kalitlar qanday yoziladi?
- A) Bitta tirnoq ichida
 - B) Qo'sh tirnoq ichida
 - C) Tirnoqsiz
 - D) Kvadrat qavs ichida
63. JavaScript obyektini JSON satrga aylantiruvchi metod qaysi?
- A) JSON.parse()
 - B) JSON.stringify()
 - C) Object.toJSON()
 - D) JSON.convert()
64. `fetch()` funksiyasi nima qaytaradi?
- A) Massiv
 - B) Promise
 - C) JSON obyekt

D) Satr

M4. Node.js, Express.js, ma'lumotlar bazasi va REST API

65. Node.js qaysi JavaScript dvigateli asosida qurilgan?

A) JavaScriptCore

B) Chakra

C) V8

D) SpiderMonkey

66. Loyiha bog'liqliklari va skriptlari qaysi faylda saqlanadi?

A) package.json

B) index.html

C) node_modules

D) .gitignore

67. Express ilovasida GET marshrut qanday yaratiladi?

A) `express.get("/yol")`

B) `app.route.get("/yol")`

C) `app.get("/yol", (req, res) => { })`

D) `app.listen("/yol")`

68. `/talaba/:id` marshrutida `id` qiymatini olish uchun nima ishlatiladi?

A) `res.params.id`

B) `req.query.id`

C) `req.params.id`

D) `req.body.id`

69. `/qidiruv?soz=html` so'rovida `soz` qiymati qayerdan olinadi?

A) `req.query.soz`

B) `req.body.soz`

C) `req.headers.soz`

D) `req.params.soz`

70. Middleware boshqaruvni keyingi funksiyaga uzatishi uchun nima chaqiradi?

A) `next()`

B) `return false`

C) `res.send()`

D) `app.use()`

71. So'rov tanasidagi JSON ni `req.body` ga aylantiruvchi middleware qaysi?

A) `express.Router()`

B) `express.static()`

C) `express.json()`

D) `express.urlencoded()`

72. **public** papkadagi fayllarni brauzerga berish uchun qaysi qator yoziladi?
- A) app.static("public")
 - B) app.use(express.static("public"))
 - C) app.get("public")
 - D) express.public()
73. Fayl o'zgarganda Node.js serverini avtomatik qayta ishga tushiruvchi vosita qaysi?
- A) npm init
 - B) Live Server
 - C) nodemon
 - D) git
74. Frontend qaysi muhitda bajariladi?
- A) Faqat veb-serverda
 - B) DNS-serverda
 - C) Foydalanuvchi brauzerida
 - D) Ma'lumotlar bazasida
75. Jadvaldagi har bir yozuvni noyob aniqlovchi maydon nima deb ataladi?
- A) Birlamchi kalit (PRIMARY KEY)
 - B) Tashqi kalit (FOREIGN KEY)
 - C) Indeks
 - D) Ustun nomi
76. Boshqa jadvalning birlamchi kalitiga havola qiluvchi maydon qanday ataladi?
- A) Birlamchi kalit
 - B) Tashqi kalit
 - C) Unikal kalit
 - D) Kompozit kalit
77. Jadvaldan ma'lumot tanlab olish uchun qaysi SQL buyrug'i ishlatiladi?
- A) UPDATE
 - B) CREATE
 - C) INSERT
 - D) SELECT
78. Mavjud yozuvni o'zgartirish uchun qaysi SQL buyrug'i ishlatiladi?
- A) ALTER
 - B) UPDATE
 - C) MODIFY
 - D) INSERT

- 79. DELETE FROM books;** buyrug'i WHERE siz bajarilsa nima bo'ladi?
- A) Jadval tuzilishi o'chadi, ma'lumotlar qoladi
 - B) Hech narsa o'chmaydi
 - C) Faqat birinchi yozuv o'chadi
 - D) Jadvaldagi barcha yozuvlar o'chadi
- 80.** Faqat ikkala jadvalda mosligi bor yozuvlarni qaytaruvchi JOIN turi qaysi?
- A) INNER JOIN
 - B) CROSS JOIN
 - C) FULL OUTER JOIN
 - D) LEFT JOIN
- 81.** Quyidagilardan qaysi biri NoSQL ma'lumotlar bazasi?
- A) MySQL
 - B) MongoDB
 - C) SQLite
 - D) PostgreSQL
- 82.** REST tamoyillariga ko'ra kitoblar ro'yxatini olish uchun qaysi endpoint to'g'ri?
- A) POST /api/books/list
 - B) GET /api/book/all/get
 - C) GET /api/getBooks
 - D) GET /api/books
- 83.** REST API da 5-kitobni o'chirish uchun qaysi so'rov ishlatiladi?
- A) GET /api/books/delete/5
 - B) POST /api/deleteBook?id=5
 - C) DELETE /api/books/5
 - D) PUT /api/books/5/remove
- 84.** Yangi resurs muvaffaqiyatli yaratilganda qaysi status kodi qaytariladi?
- A) 204 No Content
 - B) 302 Found
 - C) 200 OK
 - D) 201 Created
- 85.** Boshqa domendagi API ga brauzer so'rovlariga ruxsat berish mexanizmi qanday ataladi?
- A) CDN
 - B) CSRF
 - C) CSP
 - D) CORS

86. Relyatsion MB da “ko‘pga-ko‘p” bog‘lanish qanday amalga oshiriladi?

- A) Bitta maydonga bir nechta qiymat yozish orqali
- B) Oraliq (bog‘lovchi) jadval orqali
- C) Faqat NoSQL bazada amalga oshirish mumkin
- D) Jadvallarni birlashtirib bitta jadval qilish orqali

M5. Veb-xavfsizlik, optimallashtirish, SEO va hosting

87. Foydalanuvchining kimligini tasdiqlash jarayoni qanday ataladi?

- A) Shifrlash
- B) Autentifikatsiya
- C) Avtorizatsiya
- D) Identifikatsiya

88. Parollarni bazada saqlashning to‘g‘ri usuli qaysi?

- A) Teskari tartibda yozish
- B) bcrypt kabi algoritm bilan xeshlash
- C) Ochiq matnda saqlash
- D) Base64 da kodlash

89. Cookie ga JavaScript orqali murojaat qilishni taqiqlovchi atribut qaysi?

- A) Max-Age
- B) SameSite
- C) Secure
- D) HttpOnly

90. Sahifaga zararli skript joylashtirish hujumi qanday ataladi?

- A) CSRF
- B) DDoS
- C) XSS
- D) SQL Injection

91. SQL Injection dan himoyalaniishning asosiy usuli qaysi?

- A) So‘rovlarni katta harflarda yozish
- B) Faqat GET metodidan foydalanish
- C) CSS ni tashqi faylga chiqarish
- D) Parametrlangan so‘rovlar

92. Foydalanuvchi nomidan uning brauzeri orqali soxta so‘rov yuborish hujumi qaysi?

- A) XSS
- B) Brute force
- C) CSRF
- D) Phishing

- 93.** Ekranda ko‘rinmayotgan rasmlarni keyinroq yuklash uchun qaysi atribut ishlatiladi?
- A) loading="lazy"
 - B) preload="none"
 - C) defer
 - D) async
- 94.** Core Web Vitals ko‘rsatkichlaridan qaysi biri yuklanishdagi elementlar “sakrashini” o‘lchaydi?
- A) TTFB
 - B) INP
 - C) LCP
 - D) CLS
- 95.** Qidiruv natijalarida ko‘rinadigan sahifa tavsifi qaysi teg orqali beriladi?
- A) <meta name="description">
 - B) <title description>
 - C) <description>
 - D) <meta charset>
- 96.** Qidiruv robotlariga indekslash qoidalarini beruvchi fayl qaysi?
- A) sitemap.json
 - B) robots.txt
 - C) index.txt
 - D) seo.xml
- 97.** Git da o‘zgarishlarni omborga (tarixga) yozuvchi buyruq qaysi?
- A) git commit
 - B) git push
 - C) git init
 - D) git clone
- 98.** Mahalliy ombordagi commitlarni GitHub ga yuboruvchi buyruq qaysi?
- A) git push
 - B) git add
 - C) git pull
 - D) git status
- 99.** Quyidagilardan qaysi biri statik saytni bepul joylashtirish xizmati?
- A) Postman
 - B) MySQL
 - C) Express.js
 - D) GitHub Pages

100. GitHub Pages dagi sayt Live Server da ishlagan holda rasmlar ko‘rinmasa, eng ehtimoliy sabab qaysi?

- A) CSS fayl juda katta
- B) Sayt HTTPS da ishlamoqda
- C) Brauzerda JavaScript o‘chirilgan
- D) Fayl nomlaridagi katta-kichik harflar mos emas

Test javoblari kaliti

Nº	Javob	Nº	Javob	Nº	Javob	Nº	Javob	Nº	Javob
1	D	21	C	41	C	61	D	81	B
2	A	22	A	42	A	62	B	82	D
3	A	23	A	43	C	63	B	83	C
4	C	24	A	44	B	64	B	84	D
5	D	25	B	45	B	65	C	85	D
6	C	26	B	46	C	66	A	86	B
7	C	27	D	47	C	67	C	87	B
8	A	28	B	48	B	68	C	88	B
9	B	29	A	49	B	69	A	89	D
10	C	30	A	50	D	70	A	90	C
11	B	31	B	51	D	71	C	91	D
12	C	32	A	52	D	72	B	92	C
13	A	33	D	53	A	73	C	93	A
14	D	34	C	54	B	74	C	94	D
15	D	35	C	55	B	75	A	95	A
16	A	36	B	56	B	76	B	96	B
17	A	37	A	57	A	77	D	97	A
18	D	38	B	58	B	78	B	98	A
19	C	39	D	59	D	79	D	99	D
20	B	40	D	60	D	80	A	100	D

X. YAKUNIY NAZORAT (YOZMA ISH) MATERIALLARI VA BAHOLASH MEZONLARI

Yakuniy nazorat yozma ish shaklida o‘tkaziladi. Har bir variant 3 ta topshiriqdan iborat: 1–2-savollar – nazariy (ta’rif, tamoyil, taqqoslash, misollar bilan yoritish), 3-topshiriq – amaliy (kod yozish, kodni tahlil qilish yoki hisoblash). Yozma ish uchun 80 daqiqa vaqt beriladi. Yozma ishda qo‘shimcha adabiyot, telefon va Internetdan foydalanish taqiqlanadi.

1. Yakuniy nazorat uchun nazariy savollar

1. Internet va World Wide Web tushunchalari, ularning farqi va vebning asosiy tarkibiy qismlari.
2. URL tuzilishi. Domen nomlari va DNS tizimining ishlash prinsipi.
3. TCP/IP protokollar steki qatlamlari va ularning vazifalari.
4. Mijoz-server arxitekturasini: brauzerda manzil kiritilgandan sahifa ko'ringuncha bo'lgan jarayon.
5. HTTP protokoli: so'rov va javob tuzilishi, HTTP metodlari.
6. HTTP status kodlari guruhlari va ularga misollar.
7. HTTPS va TLS: xavfsiz ulanish qanday ta'minlanadi.
8. Veb-brauzerning tarkibiy qismlari va sahifani chizish (render) bosqichlari.
9. Veb-texnologiyalarning rivojlanish tarixi. Web 1.0, Web 2.0, Web 3.0.
10. Veb-texnologiyalarning zamonaviy tendensiyalari va ularning ta'limdagi ahamiyati.
11. HTML5 hujjatining tuzilishi va head qismidagi asosiy elementlar.
12. HTML da matn, havola va rasm teglari. Nisbiy va absolyut yo'llar.
13. HTML da ro'yxatlar va jadvallar. colspan va rowspan atributlari.
14. Semantik HTML5 teglari va ularning afzalliklari.
15. HTML5 multimedia elementlari: audio va video.
16. HTML formalar: maydon turlari va brauzer validatsiyasi.
17. CSS ning vazifasi va uni HTML ga ulash usullari.
18. CSS selektor turlari va psevdosinflar.
19. Kaskad, spetsifiklik va meros tushunchalari.
20. CSS da ranglar, fon va shrift xossalari. O'lchov birliklari.
21. CSS box modeli va box-sizing xossasi.
22. display xossasi qiymatlari. Blokli va qatordagi elementlar.
23. CSS da pozitsiyalash: static, relative, absolute, fixed, sticky.
24. Flexbox va CSS Grid: asosiy xossalar va qo'llanilish sohalari.
25. JavaScript tili: o'zgaruvchilar (let, const, var) va ma'lumot turlari.
26. JavaScript operatorlari, shartli operatorlar va tsikllar.
27. JavaScript da funksiyalar: e'lon qilish usullari, strekali funksiyalar.
28. JavaScript da massivlar va obyektlar. map, filter, reduce metodlari.
29. DOM tushunchasi. Elementlarni tanlash va o'zgartirish usullari.
30. JavaScript da hodisalar bilan ishlash. Hodisa obyekti va delegatsiya.
31. Responsive veb-dizayn tamoyillari. Media so'rovlar va mobile-first yondashuv.
32. Bootstrap freymvorki: ulash, grid tizimi va breakpointlar.

33. Bootstrap komponentlari va yordamchi sinflar.
34. Frontend va backend: vazifalari, texnologiyalari va farqlari.
35. Node.js platformasi: xususiyatlari va asinxron ishlash modeli.
36. NPM va package.json. Node.js modullari.
37. Express.js da server yaratish va marshrutlash.
38. Express.js da middleware tushunchasi. Statik fayllar bilan ishlash.
39. Ma'lumotlar bazasi va MBBT. Relyatsion va NoSQL bazalarini taqqoslash.
40. Relyatsion MB da jadvallar, kalitlar va bog'lanish turlari.
41. SQL so'rovlari: SELECT, INSERT, UPDATE, DELETE va WHERE sharti.
42. SQL da JOIN turlari va agregat funksiyalar.
43. REST arxitekturasi tamoyillari va endpoint'larni loyihalash qoidalari.
44. JSON formati. Fetch API va async/await yordamida API ga murojaat.
45. Autentifikatsiya va avtorizatsiya. Parollarni xavfsiz saqlash.
46. Cookie va sessiyalar. Cookie xavfsizlik atributlari.
47. Git versiyalarni boshqarish tizimi: asosiy tushunchalar va buyruqlar.
48. XSS va CSRF hujumlari va ulardan himoyalaniish usullari.
49. SQL Injection hujumi va xavfsiz kod yozish prinsiplari.
50. Veb-ilovalarni optimallashtirish usullari va Core Web Vitals.
51. SEO asoslari: meta teglar, sitemap, robots.txt. Google Analytics.
52. Veb-saytni hostingga joylashtirish: hosting turlari, GitHub Pages, Netlify, domen va HTTPS.

2. Yakuniy nazorat uchun amaliy topshiriqlar

1. Semantik teglardan foydalanib header, nav, main (ikkita section bilan) va footer qismlaridan iborat HTML5 sahifa kodini yozing.
2. "Dars jadvali" jadvalini yozing: 4 ustun, 4 qator, bitta katakcha colspan bilan, jadval sarlavhasi (caption) bilan.
3. Ism (majburiy), e-pochta, yosh (7–18 oralig'ida), jins (radio), yo'nalish (select) maydonlari va yuborish tugmasidan iborat HTML forma yozing.
4. Uchta kartani gorizontaal joylashtiruvchi, kartalar orasida 20px oraliq bo'lgan va kichik ekranda bitta ustunga o'tuvchi CSS kodini (Flexbox yoki Grid) yozing.
5. width: 300px; padding: 20px; border: 4px solid; margin: 10px bo'lgan elementning content-box va border-box holatlaridagi ekrandagi kengligini hisoblang va izohlang.

6. Sichqoncha ustiga kelganda 0,3 soniyada biroz kattalashadigan va soya paydo bo'ladigan tugma uchun CSS kodini yozing.
7. Navigatsiya paneli uchun CSS yozing: logotip chapda, menyu o'ngda, panel aylantirganda yuqorida yopishib qolsin.
8. Berilgan baholar massivi [5, 4, 3, 5, 2, 4] uchun o'rtacha bahoni, "5" lar sonini va 3 dan past baholarni topuvchi JavaScript kodini yozing.
9. Tugma bosilganda input ga kiritilgan sonning kvadratini paragrafga chiqaruvchi HTML va JavaScript kodini yozing (kiritma tekshiruvi bilan).
10. To-do ro'yxati uchun kod yozing: input va tugma orqali vazifa qo'shish, har bir vazifa yonidagi tugma bilan o'chirish (delegatsiya orqali).
11. Tugma bosilganda sahifaga "tun" sinfini qo'shuvchi/olib tashlovchi va tanlovni localStorage da saqlovchi JavaScript kodini yozing.
12. Bootstrap yordamida telefonda 1, planshetda 2, kompyuterda 4 ustunli kartalar blokini yozing.
13. Fetch API va async/await yordamida <https://jsonplaceholder.typicode.com/users> manzilidan foydalanuvchilarni olib, ismlarini ro'yxatga chiqaruvchi kodni xatolarni ushlab bilan yozing.
14. Express.js da 3000-portda ishlovchi, "/" va "/about" marshrutlariga javob beruvchi va mos marshrut topilmasa 404 qaytaruvchi server kodini yozing.
15. Express.js da har bir so'rov metodini va URL ini konsolga yozuvchi middleware yozing va uni ulang.
16. "Talabalar" (id, fio, guruh_id) va "Guruhlar" (id, nomi) jadvallarini yaratuvchi SQL kodini kalitlar bilan yozing.
17. "Talabalar" va "Guruhlar" jadvalaridan har bir talabaning F.I.Sh. va guruh nomini chiqaruvchi, F.I.Sh. bo'yicha saralangan SQL so'rovni yozing.
18. Har bir guruhdagi talabalar sonini chiqaruvchi SQL so'rovni yozing (talabasi yo'q guruhlar ham ko'rinsin).
19. "Mahsulotlar" resursi uchun REST endpoint'lar jadvalini tuzing (metod, URL, amal, muvaffaqiyatli javob kodi).
20. Express.js da GET /api/books (ro'yxat) va POST /api/books (qo'shish, nom majburiy) marshrutlarini massiv asosida yozing.
21. Quyidagi kodning zaifligini aniqlang va tuzating: `list.innerHTML += "<li>" + comment + "</li>";`
22. Quyidagi kodning zaifligini aniqlang va tuzating: `db.prepare("SELECT * FROM users WHERE login = " + login + "").get();`
23. Saytni GitHub ga yuklash va GitHub Pages da e'lon qilish uchun Git buyruqlari ketma-ketligini yozing va har birini izohlang.

24. Maktab sayti bosh sahifasi uchun SEO ga mos head qismini yozing: title, description, viewport, Open Graph teglari.

25. Express.js da GET /api/books/:id marshrutini yozing: kitob topilmasa 404 holat kodi va JSON xato xabari qaytarsin.

3. Yozma ish variantlari

Variantlar shunday tuzilganki, har bir variantdagi ikki nazariy savol fanning turli bo'limlariga (frontend va backend/xavfsizlik) tegishli bo'ladi. Raqamlar yuqoridagi ro'yxatlardagi tartib raqamlariga mos keladi.

Variant	Nazariy savollar	Amaliy topshiriq
1	15. HTML5 multimedia elementlari: audio va video. 46. Cookie va sessiyalar. Cookie xavfsizlik atributlari.	15. Express.js da har bir so'rov metodini va URL ini konsolga yozuvchi middleware yozing va uni ulang.
2	19. Kaskad, spetsifiklik va meros tushunchalari. 35. Node.js platformasi: xususiyatlari va asinxron ishlash modeli.	20. Express.js da GET /api/books (ro'yxat) va POST /api/books (qo'shish, nom majburiy) marshrutlarini massiv asosida yozing.
3	21. CSS box modeli va box-sizing xossasi. 38. Express.js da middleware tushunchasi. Statik fayllar bilan ishlash.	11. Tugma bosilganda sahifaga "tun" sinfini qo'shuvchi/olib tashlovchi va tanlovni localStorage da saqllovchi JavaScript kodini yozing.
4	5. HTTP protokoli: so'rov va javob tuzilishi, HTTP metodlari. 48. XSS va CSRF hujumlari va ulardan himoyalash usullari.	10. To-do ro'yxati uchun kod yozing: input va tugma orqali vazifa qo'shish, har bir vazifa yonidagi tugma bilan o'chirish (delegatsiya orqali).
5	30. JavaScript da hodisalar bilan ishlash. Hodisa obyekti va delegatsiya. 45. Autentifikatsiya va avtorizatsiya. Parollarni xavfsiz saqlash.	17. "Talabalar" va "Guruhlar" jadvallaridan har bir talabaning F.I.Sh. va guruh nomini chiqaruvchi, F.I.Sh. bo'yicha saralangan SQL so'rovni yozing.
6	29. DOM tushunchasi. Elementlarni tanlash va o'zgartirish usullari. 37. Express.js da server yaratish va marshrutlash.	14. Express.js da 3000-portda ishlovchi, "/" va "/about" marshrutlariga javob beruvchi va mos marshrut topilmasa 404 qaytaruvchi server kodini yozing.
7	2. URL tuzilishi. Domen nomlari va DNS tizimining ishlash prinsipi. 34. Frontend va backend: vazifalari, texnologiyalari va farqlari.	24. Maktab sayti bosh sahifasi uchun SEO ga mos head qismini yozing: title, description, viewport, Open Graph teglari.
8	32. Bootstrap freymvorki: ulash, grid tizimi va breakpointlar. 39. Ma'lumotlar bazasi va MBBT. Relyatsion va NoSQL bazalarini taqqoslash.	3. Ism (majburiy), e-pochta, yosh (7–18 oralig'ida), jins (radio), yo'nalish (select) maydonlari va yuborish tugmasidan iborat HTML forma yozing.

Variant	Nazariy savollar	Amaliy topshiriq
9	3. TCP/IP protokollar steki qatlamlari va ularning vazifalari. 40. Relyatsion MB da jadvallar, kalitlar va bog‘lanish turlari.	18. Har bir guruhdagi talabalar sonini chiqaruvchi SQL so‘rovni yozing (talabasi yo‘q guruhlar ham ko‘rinsin).
10	7. HTTPS va TLS: xavfsiz ulanish qanday ta‘minlanadi. 49. SQL Injection hujumi va xavfsiz kod yozish prinsiplari.	23. Saytni GitHub ga yuklash va GitHub Pages da e‘lon qilish uchun Git buyruqlari ketma-ketligini yozing va har birini izohlang.
11	11. HTML5 hujjatining tuzilishi va head qismidagi asosiy elementlar. 51. SEO asoslari: meta teglar, sitemap, robots.txt. Google Analytics.	25. Express.js da GET /api/books/:id marshrutini yozing: kitob topilmasa 404 holat kodi va JSON xato xabari qaytarsin.
12	28. JavaScript da massivlar va obyektlar. map, filter, reduce metodlari. 44. JSON formati. Fetch API va async/await yordamida API ga murojaat.	5. width: 300px; padding: 20px; border: 4px solid; margin: 10px bo‘lgan elementning content-box va border-box holatlaridagi ekrandagi kengligini hisoblang va izohlang.
13	24. Flexbox va CSS Grid: asosiy xossalar va qo‘llanilish sohalari. 41. SQL so‘rovlari: SELECT, INSERT, UPDATE, DELETE va WHERE sharti.	21. Quyidagi kodning zaifligini aniqlang va tuzating: <code>list.innerHTML += "" + comment + "";</code>
14	16. HTML formalar: maydon turlari va brauzer validatsiyasi. 50. Veb-ilovalarni optimallashtirish usullari va Core Web Vitals.	6. Sichqoncha ustiga kelganda 0,3 soniyada biroz kattalashadigan va soya paydo bo‘ladigan tugma uchun CSS kodini yozing.
15	17. CSS ning vazifasi va uni HTML ga ulash usullari. 42. SQL da JOIN turlari va agregat funksiyalar.	9. Tugma bosilganda input ga kiritilgan sonning kvadratini paragrafga chiqaruvchi HTML va JavaScript kodini yozing (kiritma tekshiruvi bilan).
16	33. Bootstrap komponentlari va yordamchi sinflar. 52. Veb-saytni hostingga joylashtirish: hosting turlari, GitHub Pages, Netlify, domen va HTTPS.	22. Quyidagi kodning zaifligini aniqlang va tuzating: <code>db.prepare("SELECT * FROM users WHERE login = " + login + "").get();</code>
17	14. Semantik HTML5 teglari va ularning afzalliklari. 47. Git versiyalarni boshqarish tizimi: asosiy tushunchalar va buyruqlar.	7. Navigatsiya paneli uchun CSS yozing: logotip chapda, menyu o‘ngda, panel aylantirganda yuqorida yopishib qolsin.
18	13. HTML da ro‘yxatlar va jadvallar. colspan va rowspan atributlari. 43. REST arxitekturasi tamoyillari va endpoint‘larni loyihalash qoidalari.	4. Uchta kartani gorizontol joylashtiruvchi, kartalar orasida 20px oraliq bo‘lgan va kichik ekranda bitta ustunga o‘tuvchi CSS kodini (Flexbox yoki Grid) yozing.

Variant	Nazariy savollar	Amaliy topshiriq
19	8. Veb-brauzerning tarkibiy qismlari va sahifani chizish (render) bosqichlari. 36. NPM va package.json. Node.js modullari.	8. Berilgan baholar massivi [5, 4, 3, 5, 2, 4] uchun o'rtacha bahoni, "5" lar sonini va 3 dan past baholarni topuvchi JavaScript kodini yozing.
20	1. Internet va World Wide Web tushunchalari, ularning farqi va vebning asosiy tarkibiy qismlari. 46. Cookie va sessiyalar. Cookie xavfsizlik atributlari.	13. Fetch API va async/await yordamida https://jsonplaceholder.typicode.com/users manzilidan foydalanuvchilarni olib, ismlarini ro'yxatga chiqaruvchi kodni xatolarni ushlash bilan yozing.
21	4. Mijoz-server arxitekturasini: brauzerda manzil kiritilgandan sahifa ko'ringuncha bo'lgan jarayon. 35. Node.js platformasi: xususiyatlari va asinxron ishlash modeli.	1. Semantik teglardan foydalanib header, nav, main (ikkita section bilan) va footer qismlaridan iborat HTML5 sahifa kodini yozing.
22	9. Veb-texnologiyalarning rivojlanish tarixi. Web 1.0, Web 2.0, Web 3.0. 38. Express.js da middleware tushunchasi. Statik fayllar bilan ishlash.	19. "Mahsulotlar" resursi uchun REST endpoint'lar jadvalini tuzing (metod, URL, amal, muvaffaqiyatli javob kodi).
23	22. display xossasi qiymatlari. Blokli va qatordagi elementlar. 48. XSS va CSRF hujumlari va ulardan himoyalash usullari.	12. Bootstrap yordamida telefonda 1, planshetda 2, kompyuterda 4 ustunli kartalar blokini yozing.
24	18. CSS selektor turlari va psevdosinflar. 45. Autentifikatsiya va avtorizatsiya. Parollarni xavfsiz saqlash.	16. "Talabalar" (id, fio, guruh_id) va "Guruhlar" (id, nomi) jadvallarini yaratuvchi SQL kodini kalitlar bilan yozing.
25	26. JavaScript operatorlari, shartli operatorlar va tsikllar. 37. Express.js da server yaratish va marshrutlash.	2. "Dars jadvali" jadvalini yozing: 4 ustun, 4 qator, bitta katakcha colspan bilan, jadval sarlavhasi (caption) bilan.
26	23. CSS da pozitsiyalash: static, relative, absolute, fixed, sticky. 34. Frontend va backend: vazifalari, texnologiyalari va farqlari.	15. Express.js da har bir so'rov metodini va URL ini konsolga yozuvchi middleware yozing va uni ulang.
27	25. JavaScript tili: o'zgaruvchilar (let, const, var) va ma'lumot turlari. 39. Ma'lumotlar bazasi va MBBT. Relyatsion va NoSQL bazalarini taqqoslash.	20. Express.js da GET /api/books (ro'yxat) va POST /api/books (qo'shish, nom majburiy) marshrutlarini massiv asosida yozing.
28	6. HTTP status kodlari guruhlari va ularga misollar. 40. Relyatsion MB da jadvallar, kalitlar va bog'lanish turlari.	11. Tugma bosilganda sahifaga "tun" sinfini qo'shuvchi/olib tashlovchi va tanlovni localStorage da saqlovchi JavaScript kodini yozing.

Variant	Nazariy savollar	Amaliy topshiriq
29	10. Veb-texnologiyalarning zamonaviy tendensiyalari va ularning ta'limdagi ahamiyati. 49. SQL Injection hujumi va xavfsiz kod yozish prinsiplari.	10. To-do ro'yxati uchun kod yozing: input va tugma orqali vazifa qo'shish, har bir vazifa yonidagi tugma bilan o'chirish (delegatsiya orqali).
30	12. HTML da matn, havola va rasm teglari. Nisbiy va absolyut yo'llar. 51. SEO asoslari: meta teglar, sitemap, robots.txt. Google Analytics.	17. "Talabalar" va "Guruhlar" jadvallaridan har bir talabaning F.I.Sh. va guruh nomini chiqaruvchi, F.I.Sh. bo'yicha saralangan SQL so'rovni yozing.

4. Yozma ish variantining namunaviy ko'rinishi

NAMANGAN DAVLAT PEDAGOGIKA INSTITUTI

10011013 – Informatika fanini o'qitish huquqini berish bo'yicha kasbiy va pedagogik qayta tayyorlash kursi

“Web texnologiyalari” fanidan yakuniy nazorat (yozma ish)

1-VARIANT

1. HTML5 multimedia elementlari: audio va video.
2. Cookie va sessiyalar. Cookie xavfsizlik atributlari.
3. Express.js da har bir so'rov metodini va URL ini konsolga yozuvchi middleware yozing va uni ulang.

Tinglovchi F.I.Sh.: _____ Guruh: _____ Sana: _____

Kafedra mudiri: _____ A. Matkarimov Tuzuvchi: _____
D. Sodirova

5. Yozma ishni baholash mezonlari

Har bir topshiriq 5 baholik shkala bo'yicha alohida baholanadi. Yozma ishning yakuniy bahosi uch topshiriq baholarining o'rta arifmetigi sifatida aniqlanib, butun songa yaxlitlanadi (0,5 va undan yuqori kasr yuqoriga yaxlitlanadi). Biror topshiriq “2” ga baholansa, umumiy baho “4” dan yuqori bo'lmaydi.

Baho	Nazariy savol	Amaliy topshiriq
5	Savol to'liq, ilmiy va mantiqiy ketma-ketlikda yoritilgan; tushunchalar to'g'ri ta'riflangan; misollar, sxema yoki kod parchalari keltirilgan; mustaqil fikr va xulosa mavjud	Kod to'liq, sintaktik va mantiqan to'g'ri; zamonaviy yondashuvlar va xavfsizlik qoidalariga amal qilingan; kod izohlangan yoki yechim tushuntirilgan
4	Savol asosan to'g'ri yoritilgan, ilmiy chalkashlik yo'q, lekin misollar yoki ayrim tafsilotlar yetarli emas	Kod asosan to'g'ri, 1–2 ta kichik sintaktik xato yoki kamchilik bor, umumiy yechim ishlaydi

Baho	Nazariy savol	Amaliy topshiriq
3	Savolga umumiy tushuncha darajasida javob berilgan, bayon tor doirada, ayrim chalkashliklar bor	Yechim g'oyasi to'g'ri, lekin kod to'liq emas yoki jiddiy xatolar mavjud
2	Javob berilmagan yoki savol mazmuniga mos emas, jiddiy xatolar bor, ko'chirish holati aniqlangan	Topshiriq bajarilmagan yoki yechim mutlaqo noto'g'ri

6. Fan bo'yicha tinglovchi bilimni baholash mezonlari (o'quv dasturiga muvofiq)

a) "5" (a'lo) baho olish uchun tinglovchining bilim darajasi quyidagilarga javob berishi lozim:

- fanning mohiyati va mazmunini to'liq yoritib olsa;
- fandagi mavzularni bayon qilishda ilmiylik va mantiqiylik saqlanib, ilmiy xatolik va chalkashliklarga yo'l qo'ymas;
- fan bo'yicha mavzu materiallarining nazariy yoki amaliy ahamiyati haqida aniq tasavvurga ega bo'lsa;
- fan doirasida mustaqil erkin fikrlash qobiliyatini namoyon eta olsa;
- berilgan savollarga aniq va lo'nda javob bera olsa;
- konspektga puxta tayyorlangan bo'lsa;
- mustaqil topshiriqlarni to'liq va aniq bajargan bo'lsa;
- barcha amaliy ko'nikma va malakalarni o'zlashtirgan bo'lsa;
- nazariy bilimlarni turli vaziyatda qo'llay olsa;
- tizimli yondashish, uzviylikka amal qilsa.

b) "4" (yaxshi) baho olish uchun:

- fanning mohiyati va mazmunini tushungan, fandagi mavzularni bayon qilishda ilmiy va mantiqiy chalkashliklarga yo'l qo'ymas;
- fanning mazmuni va amaliy ahamiyatini tushungan bo'lsa;
- fan bo'yicha berilgan vazifa va topshiriqlarni o'quv dasturi doirasida bajarsa;
- fan bo'yicha berilgan savollarga to'g'ri javob bera olsa;
- fan bo'yicha konspektini puxta shakllantirgan bo'lsa;
- fan bo'yicha mustaqil topshiriqlarni to'liq bajargan bo'lsa;
- barcha amaliy ko'nikma va malakalarni o'zlashtirishga harakat qilgan bo'lsa.

d) "3" (qoniqarli) baho olish uchun:

- fan haqida umumiy tushunchaga ega bo'lsa;
- fandagi mavzularni tor doirada yoritib, bayon qilishda ayrim chalkashliklarga yo'l qo'yilsa;

- fan bo'yicha mustaqil topshiriqlarni qisman bajargan bo'lsa;
- bayon qilish ravon bo'lmasa;
- fan bo'yicha savollarga mujmal va chalkash javoblar olinsa;
- fan bo'yicha matn puxta shakllantirilmagan bo'lsa.

e) quyidagi hollarda tinglovchining bilim darajasi “2” (qoniqarsiz) baho bilan baholanishi mumkin:

- fan bo'yicha mashg'ulotlarga tayyorgarlik ko'rilmagan bo'lsa;
- fan bo'yicha mashg'ulotlarga doir hech qanday tasavvurga ega bo'lmasa;
- fan bo'yicha mustaqil topshiriqlarni to'liq bajarmagan bo'lsa;
- fan bo'yicha matnlarni boshqalardan ko'chirib olganligi sezilib tursa (plagiat aniqlansa);
- fan bo'yicha matnda jiddiy xato va chalkashliklarga yo'l qo'yilgan bo'lsa;
- fanga doir berilgan savollarga javob olinmasa;
- fanni bilmasa.

XI. ADABIYOTLAR RO'YXATI

Asosiy adabiyotlar

1. Aripov M., Haydarov A. Informatika va axborot texnologiyalari. Darslik. – T., 2020. – 368 b.
2. Duckett J. JavaScript and jQuery: Interactive Front-End Web Development. – Wiley, 2014. – 640 p.
3. Duckett J. HTML and CSS: Design and Build Websites. – Wiley, 2011. – 490 p.
4. Robbins J.N. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics. 5th ed. – O'Reilly Media, 2018. – 808 p.

Qo'shimcha adabiyotlar

5. Pilgrim M. Dive Into HTML5. – O'Reilly Media, 2011. – 222 p.
6. Haverbeke M. Eloquent JavaScript. Third Edition. – No Starch Press, 2018. – 472 p.
7. Nixon R. Learning PHP, MySQL & JavaScript. – O'Reilly Media, 2021. – 832 p.

Axborot manbalari

8. <https://www.w3schools.com/> – veb-texnologiyalar bo'yicha o'quv resursi.
9. <https://developer.mozilla.org/> – MDN Web Docs rasmiy hujjatlari.
10. <https://nodejs.org/en/docs/> – Node.js rasmiy hujjatlari.
11. <https://getbootstrap.com/> – Bootstrap freymvorki rasmiy sayti.

Xalqaro universitetlarning tegishli o‘quv manbalari

12. <https://cs50.harvard.edu/web/> – Harvard University. CS50’s Web Programming with Python and JavaScript.

13. <https://web.stanford.edu/class/cs142/> – Stanford University. CS142: Web Applications.

Amaliy mashg‘ulotlarda foydalaniladigan qo‘shimcha elektron resurslar

- <https://expressjs.com/> – Express.js rasmiy hujjatlari.
- <https://sqlite.org/docs.html> – SQLite hujjatlari.
- <https://owasp.org/www-project-top-ten/> – OWASP Top 10 veb-xavfsizlik zaifliklari.
- <https://web.dev/> – Google: veb-sahifalar tezligi, Core Web Vitals va SEO bo‘yicha qo‘llanmalar.
- <https://validator.w3.org/> – W3C HTML validatori.
- <https://git-scm.com/book/> – “Pro Git” kitobi (bepul elektron versiya).
- <https://open-meteo.com/> – ochiq ob-havo API si.
- <https://flexboxfroggy.com/>, <https://cssgridgarden.com/> – Flexbox va Grid bo‘yicha o‘yinli mashqlar.