|  |
| --- |
| **Full stack development** |
| AKBAR MAMANAZAROV |

**Unit 7:** Full stack development

**O'qituvchi:** Muxammadsobir Turdaliyev

**Guruh raqmi:** 24-212

**Talaba ID raqami:** 240258

**Taqdim etilgan sana:** 27.05.2025

**BTEC O'QUVCHILAR TOPSHIRIQLARINI BAHOLASH VA DEKLRITSIYASI**

Baholash uchun ishlarnni taqdim etganda, har bir o'quvchi ish o'ziniki ekanligini tasdiqlovchi deklaratsiyani imzolashi kerak.

|  |  |
| --- | --- |
| **O'quvchi (talaba) identifikatori:** | 240258 |
| **Baholovchi nomi:** | Muxammadsobir Turdaliyev |
| **BTEC dasturi nomi:** | Pearson BTEC Higher Nationals in Information Technologies |
| **Birlik yoki komponent raqami va nomi:** | Full stack development |
| **Topshiriq nomi:** | Full stack development |
| **Topshiriq topshirilgan sana:** | 27.05.2025 |

Iltimos, har bir topshiriq uchun berilgan ishlarni sanab o'ting. Ishlarni topish mumkin bo'lgan sahifa raqamlarini ko'rsating yoki ishlarning mohiyatini tavsiflang (masalan, diagramma, rasm).

|  |  |  |
| --- | --- | --- |
| **Topshiriq vazifasi ma'lumoti** | **Ishlar taqdim etildi** | **Sahifa** |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| **O'quvchi deklaratsiyasi**  Ushbu topshiriq uchun taqdim etilgan ish meniki ekanligini tasdiqlayman. Ishda foydalanilgan manbalarga aniq havola qildim. Men noto'g'ri deklaratsiya noto'g'ri ishlashning bir shakli ekanligini tushunaman.  O'quvchi imzosi: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Sana: 27.05.2025 |

**KIRISH**

Bugungi raqamli asrda kompaniyalar uchun tezkorlik, qulaylik va ishonch asosiy tamoyillarga aylanib bormoqda. Ayniqsa, IT xizmatlari bilan shug‘ullanadigan kichik korxonalar uchun mijozlar bilan to‘g‘ridan-to‘g‘ri, oddiy va samarali aloqa o‘rnatish nihoyatda muhim. Dern-Support kompaniyasi aynan shunday yo‘nalishda faoliyat yuritadi — ya’ni kompyuter tizimlarini ta’mirlash, texnik xizmat ko‘rsatish va foydalanuvchilarga texnik yordam berish bilan shug‘ullanadi. Ammo kompaniya hozirgi kunda qog‘ozli jarayonlar va telefon orqali xizmat ko‘rsatishga tayanmoqda. Bu esa ishning sifatiga, samaradorligiga va mijozlarning qoniqish darajasiga bevosita ta’sir qilmoqda.

Shu sababli, ushbu loyiha orqali kompaniyaning kundalik ish faoliyatini avtomatlashtirish, mijozlar bilan bog‘lanish tizimini soddalashtirish va xizmat sifatini oshirish uchun zamonaviy to‘liq stek asosida ishlaydigan veb-ilova ishlab chiqish maqsad qilindi. Loyihada frontend va backend qismlari birgalikda ishlab chiqilib, real muammolarni hal qiluvchi, funksional va foydalanuvchiga qulay interfeysga ega tizim yaratilmoqda.

Ushbu tizim mijozlarga o‘zlari uchun shaxsiy hisob ochish, xizmat buyurtma qilish, muammo haqida xabar yuborish va xizmat narxini bilish imkonini beradi. Boshqa tarafdan, admin foydalanuvchilar esa xabarlarni ko‘rib chiqish, ularni holatiga qarab saralash, kerak bo‘lsa o‘chirish yoki o‘qilgan deb belgilash, shuningdek foydalanuvchilar ro‘yxatini yuritish kabi amallarni bajarishi mumkin. Bularning barchasi to‘liq stek yondashuvi orqali amalga oshiriladi. Backendda Node.js va Express.js ishlatilgan bo‘lib, ma’lumotlar JSON fayllarda saqlanmoqda. Frontend esa HTML, CSS (Tailwind) va JavaScript orqali qurilgan.

Loyiha doirasida avvalo kompaniyaning ehtiyojlari chuqur tahlil qilindi. Shundan so‘ng, foydalanuvchilarning roli asosida tizim strukturasini yaratish yo‘lga qo‘yildi. Har bir foydalanuvchi — u mijoz bo‘ladimi yoki administrator — o‘z ehtiyojiga mos sahifaga ega bo‘ladi. Kirish va ro‘yxatdan o‘tish sahifalari, foydalanuvchilarning shaxsiy kabineti, admin panel, xabarlar bo‘limi va xizmatlar haqidagi sahifalar — bularning barchasi o‘zaro uzviy bog‘langan holda ishlaydi. Bunda foydalanuvchi tajribasi birinchi o‘ringa qo‘yilgan.

Tizimning asosiy ustunligi — uning soddaligi va tezkor ishlashidir. Murakkab CMS yoki katta hajmli ma’lumotlar bazasi o‘rniga, oddiy JSON fayllardan foydalanilgan. Bu kichik korxonalar uchun ayni muddao: ham tejamkor, ham texnik xizmat ko‘rsatish oson. Dastlabki versiya ishlab chiqilib, foydalanuvchilardan fikr-mulohaza olindi va ularning asosida bir qancha yaxshilanishlar amalga oshirildi. Masalan, foydalanuvchi nomi takrorlanmasligi uchun autentifikatsiya bo‘limi kuchaytirildi, xabar yuborishdagi validatsiya tekshiruvlari aniqlandi, admin panelga foydalanuvchi rollari asosida kirish imkoniyati yaratildi.

Bu loyiha orqali nafaqat kompaniya ichki ishlarini raqamlashtirish, balki biznesning keyingi rivojlanishiga mustahkam poydevor qo‘yish ham ko‘zda tutilgan. Masalan, hozircha lokal fayllarda saqlanayotgan ma’lumotlar kelajakda MongoDB yoki MySQL bazasiga o‘tkazilishi, foydalanuvchilar soni oshganda esa tizimda autentifikatsiya tizimi JWT yoki OAuth kabi zamonaviy texnologiyalar bilan mustahkamlanishi rejalashtirilmoqda.

Shuningdek, frontend dizayn ham mijozlarning fikriga moslab tanlandi — sodda, zamonaviy va moslashuvchan. Tailwind CSS yordamida barcha sahifalar moslamalarga mos holatda ishlab chiqilgan: kompyuter, planshet va telefonlarda to‘liq mos ko‘rinadi. Natijada, foydalanuvchi qayerda bo‘lishidan qat’iy nazar, kompaniya bilan bemalol bog‘lanish imkoniga ega bo‘ladi.

Bu loyiha orqali kompaniyaning butun xizmat jarayoni raqamli shaklga o‘tkazilmoqda. Bu esa nafaqat mijozlar uchun qulaylik, balki kompaniya ichki ishlari uchun aniqlik, statistik tahlil va ish unumdorligini oshirish imkonini beradi. Yaratilgan to‘liq stek veb-ilova real ehtiyojlarga javob beradigan, foydalanuvchi ehtiyojini tushunadigan va foydalanishda soddaligi bilan ajralib turadigan zamonaviy yechimdir.

**1-TOPSHIRIQ**

Bu loyiha ustida ishlash davomida men to‘liq stek (full stack) yondashuvining real hayotdagi muammolarni qanday hal qilishga yordam berishini o‘z ko‘zim bilan ko‘rdim. Ayniqsa, Dern-Support kompaniyasi uchun ishlab chiqilgan platforma misolida, to‘liq stek yondashuvi nafaqat texnik, balki biznes va foydalanuvchi tajribasi nuqtayi nazaridan ham qanday foyda keltirishini chuqur angladim.

**Texnik yondashuv**

Dastlab kompaniya ehtiyojlarini o‘rganganimda, eng katta muammo bu xizmat so‘rovlarini faqat telefon orqali qabul qilinishi, ma’lumotlar esa biron joyda markazlashmagan holda saqlanishi ekanligini tushundim. Shunday ekan, men ishlab chiqqan yechim bu muammoni to‘liq stek yondashuvi yordamida hal qila oldi.

Frontendni HTML, CSS (Tailwind) va JavaScript yordamida yaratdim. U orqali mijozlar tizimga kiradi, ro‘yxatdan o‘tadi va xabar qoldiradi. Backend qismi esa Node.js va Express.js orqali yozildi. Bu server ma’lumotlarni JSON fayllarda saqlaydi va kerak bo‘lsa ularni qayta ishlaydi. Men uchun bu yondashuv ancha qulay bo‘ldi, chunki barcha tizim bitta texnologik to‘plamda boshqarildi.

**Ish jarayoniga ta’siri**

Loyihani yakka holda bajarayotganim sababli, to‘liq stekni tanlaganim juda foydali bo‘ldi. Men frontend va backend kodlarini birgalikda boshqarib, turli komponentlar o‘rtasida tez muvofiqlashtirishga erishdim. Bu nafaqat ishlab chiqish jarayonini tezlashtirdi, balki tizimni sinab ko‘rishda, xatolarni topishda va ularni tuzatishda ham yengillik berdi. Ayniqsa, foydalanuvchi roli asosida sahifalarga kirishni boshqarish (masalan, admin va oddiy foydalanuvchi farqini ajratish) uchun backend va frontendni bog‘lashda tajribam ancha oshdi.

**Foydalanuvchi tajribasiga ta’siri**

Men yaratayotgan tizimda foydalanuvchining qulayligini har doim birinchi o‘ringa qo‘ydim. Chunki mijozlar uchun qulay interfeys ularning tizimdan foydalanish istagini oshiradi. Kirish/ro‘yxatdan o‘tish sahifalarini iloji boricha soddalashtirdim, xabar yuborish formasini esa faqat zarur maydonlar bilan to‘ldirdim. Natijada, foydalanuvchilar o‘z so‘rovlarini hech qanday ortiqcha murakkabliksiz yubora olishdi.

Adminlar uchun esa maxsus “dashboard” sahifasini yaratdim. U orqali ular kelgan xabarlarni ko‘ra oladi, ularni saralab, holatini o‘zgartiradi yoki kerak bo‘lsa o‘chiradi. Men bu sahifani ko‘proq real ehtiyojlarga moslab, minimal, lekin samarali ko‘rinishda tuzdim.

Bu loyiha orqali men shuni tushundimki, to‘liq stek faqat kod yozish bilan bog‘liq emas. U biznes jarayonlarini raqamlashtirish, statistik tahlillar qilish va qaror qabul qilishda katta foyda beradi. Masalan, har bir yuborilgan xabar sanasi, mavzusi, holati va mijoz ma’lumotlari bilan birga saqlanmoqda. Bu ma’lumotlar asosida keyinchalik kompaniya xizmatlar sifati, eng ko‘p uchraydigan muammolar yoki xizmatga ketadigan o‘rtacha vaqt haqida to‘liq tahlil yurita oladi.

Men kod yozayotganda buni hisobga olib, har bir foydalanuvchi harakatini (xabar yuborish, holatni o‘zgartirish) maxsus “timestamp” bilan birga yozilishini ta’minladim. Bu esa statistik tahlil uchun juda muhim bo‘ldi.

**Kengaytirish va moslashuv**

Men bu loyihani yaratishda kelajakni ham hisobga oldim. Hozircha JSON fayllardan foydalanilgan bo‘lsa-da, kerak bo‘lsa MongoDB yoki boshqa ma’lumotlar bazasiga oson o‘tkazish mumkin bo‘lgan strukturani tuzdim. Bundan tashqari, foydalanuvchi autentifikatsiyasi ham shunday yozilganki, uni kelajakda JWT yoki boshqa xavfsizlik protokollari bilan kengaytirish oson.

Frontend esa Tailwind yordamida to‘liq moslashuvchan qilingan: telefon, planshet yoki kompyuter — qaysi qurilmada ochilmasin, sahifa o‘zini avtomatik moslashtiradi. Men bu ishga alohida e’tibor berdim, chunki foydalanuvchilar har doim bir xil sifatli tajribani his qilishi kerak deb hisoblayman.

**FULLSTACK YONDASHUV ASOSIDA VEB-LOYIHA ISHLAB CHIQISH**

Men Dern-Support uchun veb-platforma ishlab chiqish jarayonida fullstack yondashuvni tanladim, chunki bu usul aynan mening vaziyatimga eng to‘g‘ri keladigan echim edi. Kompaniyaning ehtiyoji oddiy: mijozlar bilan tez, qulay va avtomatlashtirilgan aloqa o‘rnatish, xizmat so‘rovlarini qabul qilish, ularni ko‘rib chiqish va adminlar uchun nazorat tizimini yaratish kerak edi. Bu ehtiyojni hal qilishda fullstack uslub menga juda ko‘p yordam berdi.

Avvalo, foydali tomonlarini aytadigan bo‘lsam — bu usul menga mustaqil harakat qilish imkonini berdi. Men frontend (interfeys) qismini Tailwind CSS va JavaScript yordamida soddalashtirilgan, foydalanuvchiga qulay ko‘rinishda qurdim. Aynan shu paytda foydalanuvchi nima ko‘radi, qanday forma to‘ldiradi, qayerga bosadi — bularni aniq his qildim. So‘ng backend (server logikasi) qismini Node.js va Express yordamida yozdim. Bu yerda xabarlarni saqlash, foydalanuvchini autentifikatsiyadan o‘tkazish, va ma’lumotlar oqimini boshqarish kabi ishlarni yo‘lga qo‘ydim.

Shunday qilib, men bir paytning o‘zida ikkita yo‘nalishni bitta tizimda bog‘lay oldim. Har qanday o‘zgarishni darhol test qilib, xatoni o‘zim tuzata oldim. Bu meni nafaqat vaqt, balki ortiqcha muvofiqlashtirishdan ham qutqardi. Ayniqsa, test qilish va debugging jarayoni ancha tezlashdi.

Loyihani yakka o‘zim qilayotganim uchun bu yondashuv menga pul va vaqt tejashda ham foydali bo‘ldi. Hech kimdan kutmasdan, barcha kod ustidan o‘zim nazorat qildim. Bunda menga eng ko‘p yordam bergan narsa — frontend va backendda bir xil til, ya’ni JavaScript ishlatilgani bo‘ldi. Bu nafaqat kodlar o‘rtasida uyg‘unlik berdi, balki mantiqan ham boshqaruvni osonlashtirdi.

Endi kamchilik tomonlariga to‘xtaladigan bo‘lsam — to‘g‘risi, hamma narsani o‘zim qilishim ba’zida qiyin bo‘ldi. Ayniqsa, vaqtni to‘g‘ri taqsimlashda qiynalgan paytlarim bo‘ldi. Masalan, frontenddagi dizayn bilan ko‘p shug‘ullanib qolganimda, backenddagi xavfsizlikni biroz kechroq ko‘rib chiqganman. Yoki aksincha. Yana bir qiyin joyi — har bir narsani o‘zing chuqur tushunishga majbur bo‘lasan. Faqat ishlashi yetarli emas, har bir funksiya nima qiladi, nima uchun shunday ishlaydi, va bu foydalanuvchi uchun qanday natija beradi — bularning hammasini hisobga olmasang, tizimda muammolar chiqishi mumkin.

Yana shuni ham aytishim kerakki, fullstack yondashuv o‘z ichiga juda katta mas’uliyatni oladi. Har bir kichik o‘zgarish butun tizimga ta’sir qilishi mumkin. Shu bois, men kod yozayotganimda ham, foydalanuvchi qanday ko‘radi, admin qanday boshqaradi, serverda nima bo‘ladi — bularning har birini oldindan tasavvur qilib harakat qildim.

Kengroq mulohaza qilib aytadigan bo‘lsam, fullstack yondashuv kichik loyihalar uchun ayni muddao. Ayniqsa, bitta odam yoki kichik jamoa ishlayotgan bo‘lsa, bu yondashuv juda foydali. Men yaratgan platforma real ehtiyojlarga mos keldi: foydalanuvchi hisob yaratadi, so‘rov yuboradi, adminlar esa ularni boshqaradi. Kelajakda bu tizimni kengaytirish, masalan, MongoDB bazasiga o‘tish, yoki autentifikatsiyani kuchaytirish ham juda oson bo‘ladi. Chunki asos barqaror va modulli tuzilgan.

Xulosa qilib aytsam, men fullstack yondashuv orqali nafaqat texnik jihatdan, balki tashkiliy va amaliy jihatdan ham foydali tajriba oldim. Har bir bosqichda tizimning qanday ishlashini o‘zim nazorat qildim, har bir xatolikni o‘z vaqtida tuzatdim, va eng muhimi — foydalanuvchi uchun haqiqatan qulay tizim yarata oldim.

**FULLSTACK YONDASHUVNING ZAMONAVIY IT SANOATIDAGI AHAMIYATI VA AMALIY QO‘LLANISHI**

Men fullstack yondashuvda ishlashga kirishganimda, bu faqat kichik loyihalar uchun emas, balki butun dunyo bo‘yicha IT kompaniyalarida eng ko‘p ishlatiladigan yondashuvlardan biri ekanini tushundim. Ayniqsa, bugungi sanoat muhitida tezlik, samaradorlik va moslashuvchanlik asosiy mezonlarga aylangan bir paytda, fullstack dasturchilar ko‘plab loyihalarda asosiy rollarni bajarishyapti.

Masalan, **startap kompaniyalar** dastlabki versiyalarni ishlab chiqishda byudjetni tejash uchun fullstack dasturchini yollashadi. Chunki u bitta o‘zi butun tizimni – frontenddan tortib backendgacha – ishlab chiqib bera oladi. Bu ularga ortiqcha jamoa tuzmasdan tezda tayyor mahsulot chiqarish imkonini beradi. Men bu tajribani o‘zim ham boshdan kechirdim: Dern-Support kompaniyasi uchun ishlab chiqqan platformamda butun loyihani yakka o‘zim bajarishga to‘g‘ri keldi, va aynan fullstack yondashuv buning uddasidan chiqishimga yordam berdi.

Katta IT kompaniyalar ham, masalan, **Netflix**, **Airbnb**, **Uber** kabi texnologik gigantlar, fullstack yondashuvdan keng foydalanishadi. Ular o‘z ilovalari uchun **MERN stack** yoki **MEAN stack** kabi to‘plamlarni tanlashadi, bu degani — bir xil texnologiyalar asosida barcha qatlamlar boshqariladi: MongoDB (bazaga), Express (server), React yoki Angular (interfeys), va Node.js (backend). Bu arxitektura ularga kodni qayta ishlashda moslashuvchanlik, test qilishda aniqlik va bir xil texnologik muhitni yaratadi.

Masalan, Netflix platformasi orqali foydalanuvchilar film tomosha qilganda ularning tanlovi frontendda React orqali ko‘rsatiladi, bu tanlov backend orqali tahlil qilinadi va keyingi tavsiyalar avtomatik tarzda chiqariladi. Bu bir qarashda oddiy ko‘ringan jarayon aslida backend va frontendning uzviy ishlashi orqali amalga oshadi.

Men o‘zimning loyiha jarayonimda bu yondashuvning qulayligini yaqqol his qildim. Foydalanuvchi interfeysi (kirish, ro‘yxatdan o‘tish, xabar yuborish) bilan backenddagi funksiyalar (xabarlarni saqlash, foydalanuvchi ma’lumotlarini qayta ishlash) birgalikda ishlashi orqali muammosiz tizim yaratdim. Bu nafaqat ishlab chiqish bosqichida, balki keyinchalik test, takomillashtirish va kengaytirish paytida ham katta yordam berdi.

Shuni aytishim mumkinki bugungi sanoatda fullstack dasturlash universal yondashuv sifatida qabul qilinadi. U dasturchiga kodning har ikki tomonini boshqarish, loyihani to‘liq nazorat qilish va foydalanuvchining ehtiyojiga mos, ishlaydigan tizimlar yaratish imkonini beradi. Aynan shu sababli men ham bu yondashuvni tanladim va real ehtiyojlar asosida amalda sinab ko‘rdim.

**TECHREPAIRSERVICE LOYIHASIDA FULLSTACK YONDASHUVNING AMALIY QO‘LLANILISHI: TAJRIBA, AFZALLIKLAR VA SINOVLAR**

Men o‘z loyihamni – **TechRepairService** deb nomlagan veb-platformani ishlab chiqishda **fullstack yondashuvni** tanladim. Bu qarorim tasodifiy emas edi, balki real ehtiyojlar asosida kelib chiqqan edi. Loyihaning maqsadi kompyuter ta’mirlash xizmatlarini raqamlashtirish, mijozlar bilan bog‘lanishni osonlashtirish va administratorlarga xizmat so‘rovlarini boshqarish imkonini berish edi.

**Afzalliklar:**

1. **Bitta tizimda hamma narsa:**  
   Fullstack yondashuv menga bitta tizim ichida foydalanuvchi interfeysi (frontend) va server logikasini (backend) birgalikda ishlab chiqish imkonini berdi. Masalan, foydalanuvchi xabar yuborganida bu xabarni qanday qabul qilish, qayerga yozish va qachon adminga ko‘rsatish kerakligini to‘liq o‘zim belgilay oldim.
2. **Vaqt va resurs jihatdan tejamli:**  
   Men frontend va backendda alohida jamoa yollamasdan, barchasini o‘zim bajardim. Bu menga ishlab chiqish vaqtini qisqartirishga, sinovlarni tezroq o‘tkazishga va xatolarni tez aniqlashga yordam berdi.
3. **Texnologik muvofiqlik:**  
   Frontend va backendda JavaScriptdan foydalanganim sababli, kodlar o‘zaro yaxshi bog‘landi. Express.js va Tailwind CSS kombinatsiyasi esa menga kerakli funksiyalarni soddalik bilan yozish imkonini berdi.
4. **Moslashuvchanlik va kengayish imkoniyati:**  
   Loyihani ishlab chiqqan kodlarim shunday tuzilgan ediki, agar kelajakda MongoDB yoki PostgreSQL bazasiga o‘tish zarurati tug‘ilsa, backend strukturasini o‘zgartirishga hojat qolmaydi. Bu menga professional darajada o‘ylab kod yozishni o‘rgatdi.

**Kamchiliklar:**

1. **Ko‘p vazifa va yuklama:**  
   Men ham frontend, ham backendga javobgar bo‘lganim uchun, ba’zida qaysi biriga ko‘proq e’tibor qaratishni tanlay olmay qiynaldim. Dizayn ustida ishlayotganimda serverdagi autentifikatsiya yoki validatsiya masalalarini kechiktirib yuborgan holatlarim ham bo‘ldi.
2. **Tajriba talab qiladi:**  
   Har bir texnologiyani alohida o‘rganish va uni to‘g‘ri ishlatish ko‘p vaqt va bilim talab qildi. Dastlabki sinovlarda xatoliklar bo‘lgan, ba’zilarini tuzatishim uchun qo‘shimcha o‘rganishga to‘g‘ri kelgan.
3. **Xavfsizlikni mustahkamlash qiyinlashadi:**  
   Foydalanuvchi ma’lumotlarini to‘g‘ri saqlash, kirishni cheklash, admin panelga ruxsatsiz kirishni oldini olish – bu kabi xavfsizlik masalalariga alohida e’tibor berishga to‘g‘ri keldi. Bu qismlar anchagina mas’uliyat talab qildi.

Fullstack yondashuv menga faqat kod yozish emas, balki butun tizimni boshqarish, foydalanuvchi ehtiyojini chuqur tushunish, ish jarayonini boshidan oxirigacha o‘zim nazorat qilish imkonini berdi. Shu sababli, bu yondashuv faqat texnik vosita emas, balki strategik qaror ham bo‘ldi.

Men shuni angladimki, fullstack yondashuv faqat kichik loyihalar uchun emas — u katta kompaniyalar, real biznes ehtiyojlari, mijoz tajribasi va xizmat sifati uchun ham juda foydali. Shuningdek, bu yondashuv dasturchini ko‘proq mustaqil, mas’uliyatli va tizimni boshidan o‘zlashtirishga undaydi.

**2-TOPSHIRIQ**

**Mijoz ehtiyojlariga asoslangan texnik YECHIM**

Loyihani boshlashimdan oldin, mijozning asosiy talablarini diqqat bilan o‘rganib chiqdim. TechRepairService nomli IT texnik xizmat ko‘rsatish kompaniyasining ehtiyojlari aniq edi: foydalanuvchilarga xizmatga yozilish, xabar yuborish, narx so‘rash, va shuningdek kompaniya uchun kundalik ishlarni boshqarish, xodimlar va foydalanuvchilar o‘rtasidagi muloqotni avtomatlashtirish kerak edi. Shuni hisobga olgan holda, men qulay, xavfsiz va foydalanuvchi uchun sodda interfeysga ega bo‘lgan to‘liq stek (fullstack) yechim taklif qildim.

**Biznes konteksti va muammo mazmuni:**

TechRepairService doimiy ravishda mijozlar soni oshib borayotgan kichik biznes bo‘lib, asosiy muammo – xizmatga yozilish, buyurtma berish va xizmat holatini kuzatishning qo‘lda amalga oshirilayotganligidir. Bu jarayon vaqtni oladi, chalkashliklarga sabab bo‘ladi va xizmat sifatini pasaytiradi. Shu sababli, mijozlar bilan raqamli shaklda aloqaga chiqadigan web-platforma ishlab chiqish zarur edi.

**Taklif etilayotgan yechim tavsifi:**

Men ishlab chiqqan yechim – bu foydalanuvchilar uchun xizmatlarga yozilish, aloqa formasi orqali muammo yuborish, foydalanuvchi hisobi yaratish va admin panel orqali xabarlarni boshqarish imkonini beruvchi fullstack veb-ilova. Frontend HTML, CSS, JS orqali yozilgan; backend esa Node.js va Express.js orqali boshqariladi. Ma’lumotlar user.json fayliga yoziladi. Har bir modul mustaqil ishlaydi, ammo umumiy tizimga uyg‘unlashtirilgan.

**Funktsional talablar:**

* Mijoz hisob yaratishi va xizmatga yozilishi mumkin
* Xabar yuborish formasi ishlaydi va backendga ulanadi
* Admin panel mavjud, u orqali foydalanuvchi xabarlarini ko‘rish va o‘chirish mumkin
* Har bir foydalanuvchi uchun to‘liq ma’lumot saqlanadi (fullName, email, phone, message)
* Ma’lumotlar JSON formatida saqlanadi va yangilanadi

**Funktsional bo‘lmagan talablar:**

* Sahifa mobil qurilmalarda ham moslashgan (responsive)
* Tizim xavfsizligi oddiy tekshiruv bilan ta’minlangan
* Brauzerlararo moslik tekshirildi
* Dizayn professional va foydalanuvchi uchun tushunarli

**KPI (asosiy samaradorlik ko‘rsatkichlari):**

* Har oy xizmatga yozilayotgan foydalanuvchilar soni
* Foydalanuvchi formani to‘g‘ri to‘ldirishi va yuborishi (validatsiya muvaffaqiyati)
* Admin tomonidan o‘qilgan va o‘chirilgan xabarlar soni
* Xizmat ko‘rsatish vaqtida kechikish holatlarining kamayishi

**Tavakkalchiliklar va oqibatlar:**

* JSON fayli asosidagi bazaga haddan ortiq yuklama bo‘lsa, tormozlik ehtimoli mavjud
* Agar foydalanuvchi formani noto‘g‘ri to‘ldirsa va validatsiya ishlamasa, bu tizimga noto‘g‘ri ma’lumot kirishiga olib keladi
* Xavfsizlik kuchli parolsiz tizimlarda buzilish xavfini oshiradi

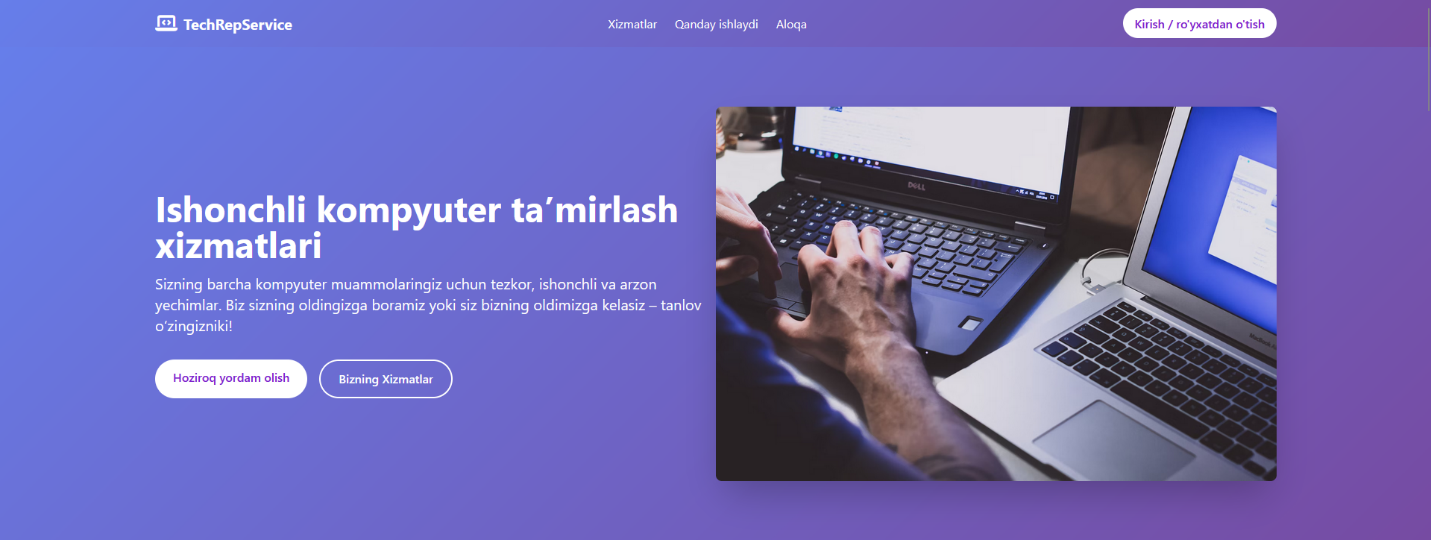
Shu asosda men mijozga yuqoridagi yechimni taklif qildim. Taklifdan oldin albatta foydalanuvchi tajribasi, texnik talablar, xavfsizlik va ishlash tezligi kabi jihatlarni hisobga oldim. Har bir funksiyani ishlab chiqishdan oldin uni qog‘ozda chizdim, algoritm va oqimlarni oldindan rejalashtirdim. Yechimim mijozga nafaqat funksional, balki estetik va foydalanuvchiga qulay tizimni taklif qila oldi.

Loyihamni aynan mijozga taklif qilganimda ular rozi bo‘lishdi, sababi taklif aniq, moslashtirilgan va real muammolarni yechishga qaratilgan edi. Bu bosqichda ishlab chiqqan hujjatlarim yechim sifatini ko‘rsatib bera oldi.

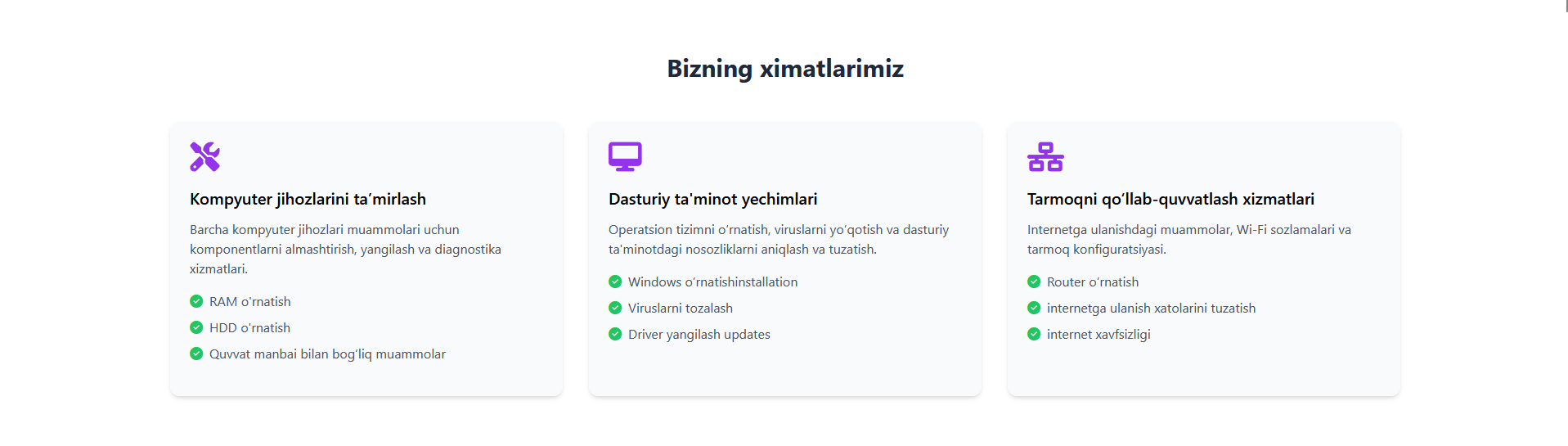
**FOYDALANUVCHIDAN DIZAYNERGACHA: REAL MULOHAZALAR ASOSIDA ISHLAB CHIQILGAN VEB-ILOVA TAJRIBASI**

Loyihamni ishlab chiqish davomida men dizaynlarimni doimo yopiq holda saqlamaganman. Aksincha, har bir bosqichda uni tanishlarim, sinfdoshlarim va ustozim bilan baham ko‘rib, fikr olganman. Aynan shunday fikrlar orqali men web-ilovamning ko‘plab jihatlarini yaxshilashga muvaffaq bo‘ldim. Web dizayn yaratish uchun men [Figma.com](https://www.figma.com) saytida foydalandim quyidagi link orqali men o’z web dizaynimni ko’rish mumkin: <https://www.figma.com/design/L8ywqSbxPV4e8fawtpaq5C/code.to.design-playground-%E2%80%94-by-%E2%80%B9div%E2%80%BARIOTS--Community-?node-id=0-1&t=Lx0snT825p1ThKDq-1>

Dastlabki dizaynimda asosiy sahifa (hero section) soddaroq edi. Men uni ko‘rsatganimda, tanishim vizual jihatdan fon rasmi yetarlicha professional ko‘rinmasligini aytdi. Shundan so‘ng men ushbu fon rasmni o‘zgartirib, yuqori sifatli va IT sohasini ifodalovchi surat bilan almashtirdim. Bu sahifa birinchi ko‘rishda ishonch uyg‘otishi kerak edi – va aynan shunday bo‘ldi.



Keyin xizmatlar bo‘limini foydalanuvchiga aniq tushunarli bo‘lishi uchun uchta asosiy kategoriya – “Kompyuter ta’miri”, “Dasturiy yechimlar”, va “Tarmoq xizmatlari” tarzida ajratdim. Fikr bildirganlar bu tuzilmani qulay deb baholashdi. Boshida har bir xizmat haqida matn haddan ortiq qisqa edi, shuning uchun ularni texnik jihatdan boyitib, xizmatlar haqida batafsilroq yozdim.



Men yaratgan “TechRepService” loyihamda mijozlarga xizmat ko‘rsatish jarayoni iloji boricha sodda, tushunarli va qulay shaklda tashkil etilgan. Aynan shu sababli, men “Servisimiz qanday ishlaydi?” degan bo‘limni ishlab chiqdim va u orqali butun xizmat ko‘rsatish tartibini uchta oddiy bosqichga ajratdim. Bu bo‘lim foydalanuvchining savollariga avvaldan javob beradi, uni chalkashtirmaydi va qanday harakat qilish kerakligini vizual tarzda tushuntiradi.

|

Avvalambor foydalanuvchining ro‘yxatdan o‘tish va login sahifalarini ko‘rib chiqqanman. Dastlabki versiyada bu sahifalar juda soddaligi sababli, foydalanuvchi rolini tanlash imkoniyati yo‘q edi. Men uni qayta ishlab chiqib, foydalanuvchiga “User” yoki “Admin” rolini tanlash imkonini berdim. Shuningdek, parolni tasdiqlash maydoni va email tekshiruv funksiyasini qo‘shdim.

24/7 call center qismi **TechRepService** loyihamda foydalanuvchi bilan aloqa qilish va muammoni tezkor hal etishning eng muhim komponenti bo‘lgan **24/7 Call Center Support** bo‘limidir. Men bu bo‘limni ishlab chiqishda foydalanuvchining ishonchini qozonish, doimiy yordam borligini his qilish va muammosini istalgan vaqtda hal etish imkonini yaratishni maqsad qilganman.



Chap tomonda joylashgan rasmda biz jamoa muhokamasi, loyihalash va muammolarni hal qilish jarayonini aks ettirdik. Bu foydalanuvchiga shuni ko‘rsatadiki, bizning call center orqasida faqat avtomatik tizim emas, balki haqiqiy odamlar — mutaxassislar ishlayapti. Bu esa ishonchni yanada oshiradi.

Bo‘lim sarlavhasida aniq va kuchli signal berilgan: **“24/7 Call Center Support”**. Bu orqali men xizmatlarimiz doimiy ishlashini, ya’ni foydalanuvchi kechasi-yu kunduz bog‘lanishi mumkinligini ta’kidladim.

Rostini aytsam, bu bo‘limda eng katta e’tiborni men “Qanday ishlaydi?” degan 5 bosqichli ko‘rsatmaga qaratdim:

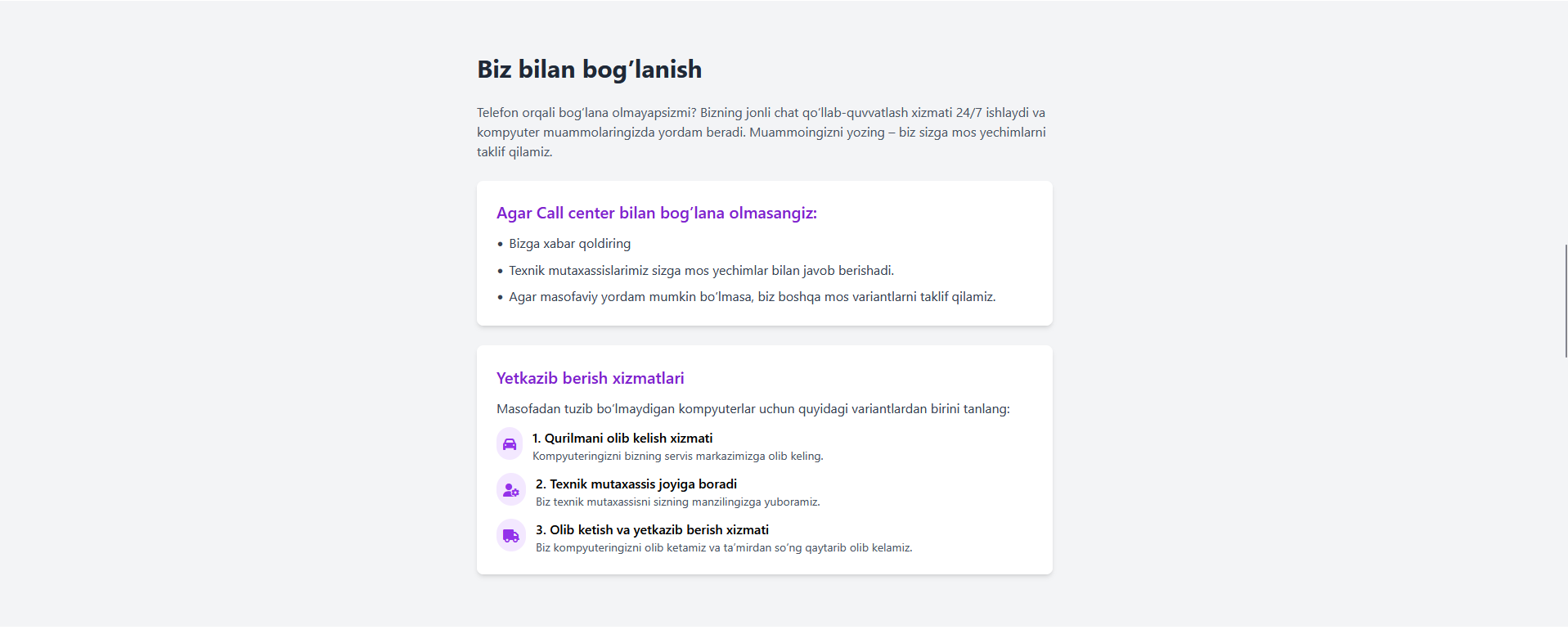
1. **Bog‘lanish** – foydalanuvchi telefon yoki jonli chat orqali bizga murojaat qiladi.
2. **Tushuntirish** – texnik mutaxassis foydalanuvchining muammosini diqqat bilan tinglaydi.
3. **Masofadan hal qilish** – muammo oddiy bo‘lsa, darhol bosqichma-bosqich yo‘l-yo‘riq beriladi.
4. **Joyiga borish** – agar muammo murakkab bo‘lsa, biz texnik mutaxassisni mijoz manziliga yuboramiz.
5. **To‘lov** – muammo hal qilingach, foydalanuvchi xizmat uchun to‘lovni amalga oshiradi.

Ushbu yondashuv foydalanuvchining vaqtini tejaydi va unga qulaylik yaratadi. Ayniqsa, to‘lovning faqat xizmat ko‘rsatilgach olinishi — bu juda kuchli ishonch omili bo‘lib xizmat qiladi.

Quyidagi telefon raqami +998 99 652 6025 esa har doim ochiq turadigan xizmat kanalimiz hisoblanadi. Bu raqam foydalanuvchiga “Siz yolg‘iz emassiz” degan ishorani beradi.

Bu sahifa foydalanuvchi bilan birinchi real aloqani ta’minlaydi. Men buni yaratishda dizayn, axborot va ishonchni uyg‘unlashtirdim. Foydalanuvchining har bir bosqichda o‘zini qulay va tushungan holatda his qilishi mening asosiy maqsadim edi — va bu sahifa aynan shu vazifani bajaradi.

**“Biz bilan bog‘lanish”** bo‘limi — mening loyihamda foydalanuvchi bilan aloqa qilishdagi eng muhim kanallardan biridir. Men bu qismni ishlab chiqayotganda asosiy e’tiborni oddiylik, qulaylik va har xil foydalanuvchilar uchun moslashuvchanlikka qaratdim.



Bo‘lim sarlavhasi — **"Biz bilan bog‘lanish"** — foydalanuvchiga shuni bildiradiki, bu yer orqali ular o‘z muammosini yetkazishlari va shunga yarasha yechim olishlari mumkin. Matnda yozilgan jumlalar orqali men foydalanuvchiga telefon, chat yoki boshqa aloqa vositalari orqali istalgan vaqtda murojaat qilish imkoniyati borligini aniq bayon qildim.

Sahifaning ikki qismdan iborat bo‘lishi — bu bo‘limni nafaqat funksional, balki foydalanuvchi tajribasi (UX) nuqtai nazaridan ham qulay qilgan.

Birinchi qismda, **Agar Call Center bilan bog‘lana olmasangiz**, degan sarlavha ostida uchta muqobil yechim taqdim etilgan:

* Xabar qoldirish,
* Mutaxassis javobi,
* Masofaviy yordam yoki boshqa variantlar.

Bu yondashuv shuni anglatadiki, foydalanuvchi aloqa liniyasi band bo‘lsa ham, o‘z muammosini hech bo‘lmaganda yozma tarzda qoldirib ketishi mumkin. Bu esa mijoz tajribasini optimallashtiradi.

Ikkinchi qism esa **Yetkazib berish xizmatlari** haqida. Aynan mana shu bo‘limni ishlab chiqishda men foydalanuvchilarning texnik qurilmani markazga olib bora olmasligi yoki joyida tuzatish talab qilinadigan holatlariga yechim topishga harakat qildim. Bu yerda uchta variant berilgan:

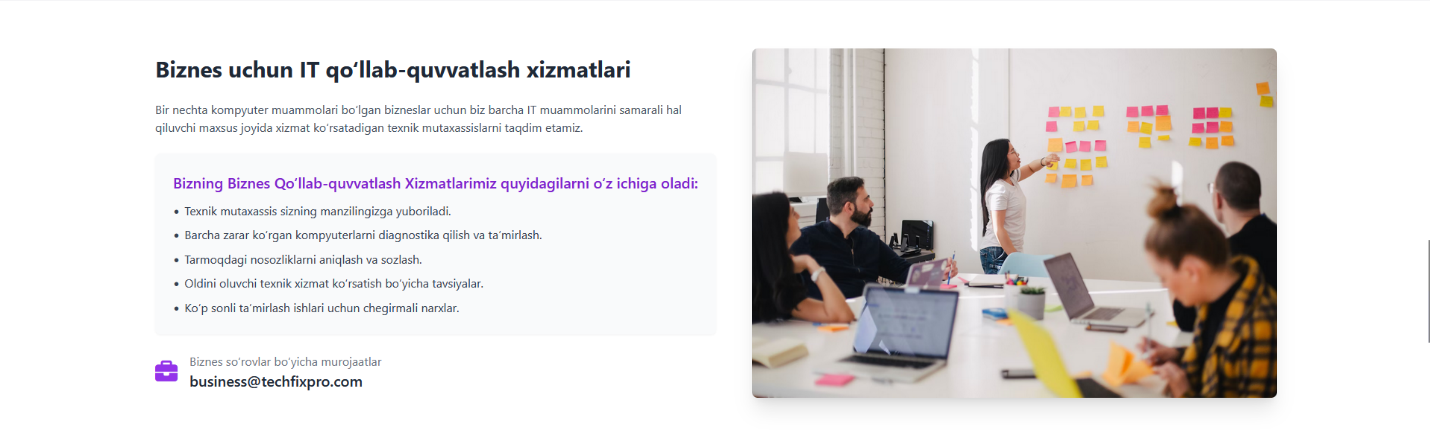
1. **Qurilmani olib kelish** – foydalanuvchi o‘zi olib keladi.
2. **Texnik mutaxassis joyiga boradi** – biz manzilga boramiz.
3. **Olib ketish va qaytarish** – kompyuterni olib ketamiz, ta’mirdan so‘ng qaytaramiz.

Bu xizmatlar turlicha ehtiyojga ega foydalanuvchilar uchun maksimal moslashtirilgan. Men buni ishlab chiqishda “faqat texnik xizmat emas, balki logistika ham muhim” degan tamoyil asosida harakat qildim.

Bu sahifa faqat aloqa uchun emas, balki foydalanuvchining muammosini qayerda bo‘lishidan qat’i nazar, tez va qulay hal qilish uchun yaratilgan. Bu yechimlarning har biri real hayotdagi ehtiyojdan kelib chiqqan. Foydalanuvchi uchun har bir bosqich aniq va ishonchli ko‘rinishda ishlab chiqilgan. Shu sababli, bu sahifa men loyihamda eng ko‘p foydalanuvchi interaktsiyasi kutilgan bo‘limlardan biridir.

**“Biznes uchun IT qo‘llab-quvvatlash xizmatlari”**

Mening loyihamdagi eng strategik bo‘limlardan biri hisoblanadi. Aynan ushbu qism orqali men yirik tashkilotlar yoki kichik biznes egalariga TechRepService xizmatining korporativ darajadagi imkoniyatlarini ko‘rsatishni maqsad qilganman.



Matnda yozilganidek, **bir nechta kompyuter muammolari bo‘lgan bizneslar uchun** maxsus xizmat turi taklif qilinadi. Bu orqali men shaxsiy mijozlar bilan cheklanmasdan, xizmat doiramizni kengroq auditoriyaga — ya’ni tashkilotlarga, ofislarga va korxonalarga qaratishga erishdim.

Bo‘lim ichida men xizmatlarning konkret elementlarini ro‘yxat shaklida sanab chiqdim:

* Texnik mutaxassisni manzilga yuborish — bu korxonalarda muammolar joyida hal etilishini ta’minlaydi.
* Zarar ko‘rgan qurilmalarni diagnostika qilish va ta’mirlash — bu texnik jihatdan murakkab holatlarni o‘z ichiga oladi.
* Tarmoq konfiguratsiyasi va nosozliklarini aniqlash — ayniqsa Wi-Fi yoki LAN muammolari biznes uchun juda muhim.
* Oldini oluvchi texnik xizmat tavsiyalari — bu esa doimiy xizmatlar uchun yillik xizmat ko‘rsatish rejalarini bildiradi.
* Va albatta, ko‘p sonli buyurtmalar uchun chegirmali narxlar — bu esa byudjetni nazorat qilayotgan tashkilotlar uchun eng jozibali nuqtadir.

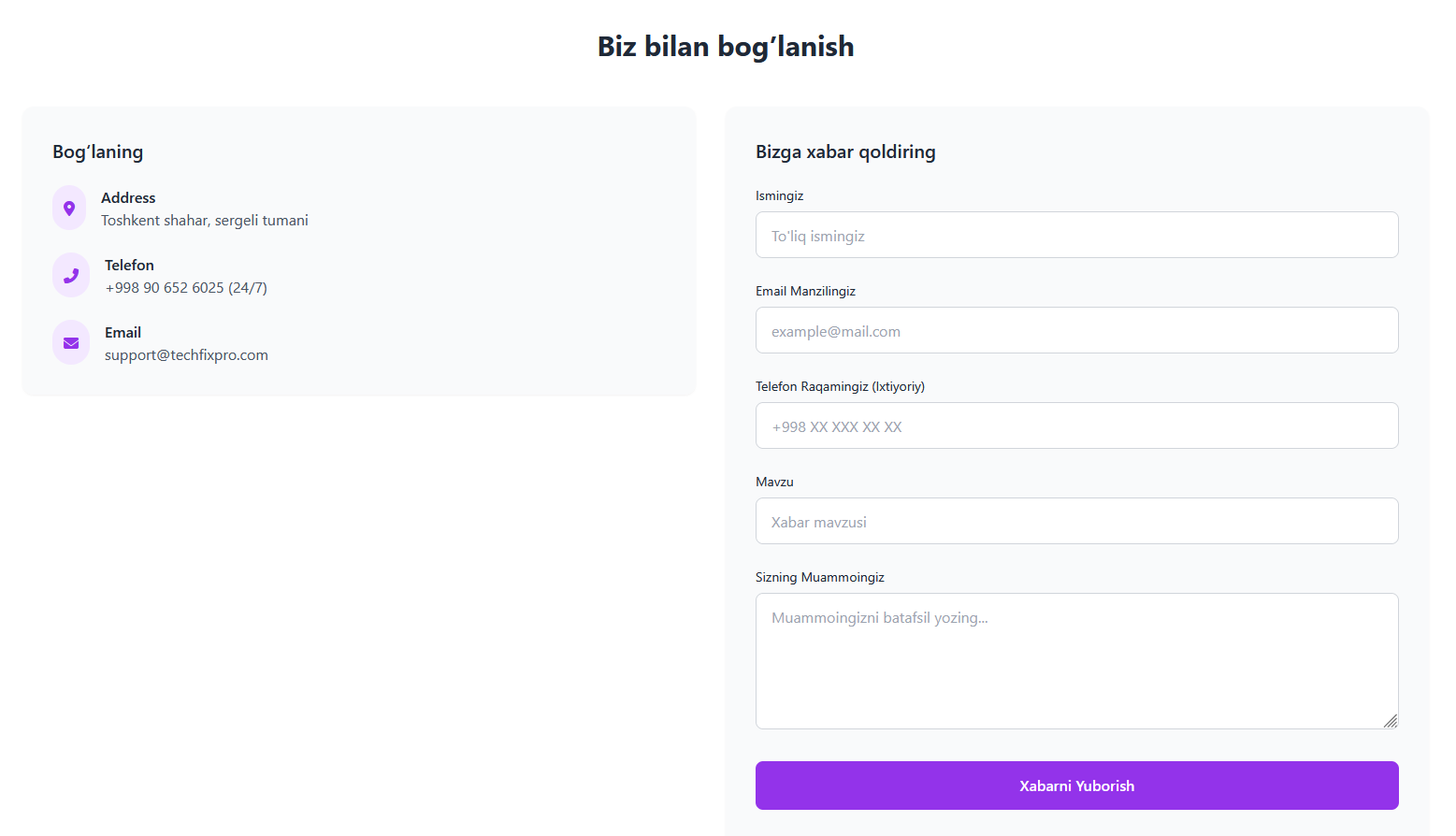
Men ushbu xizmatlarni taklif etayotganda mijoz ehtiyojini emas, balki **korxona ish faoliyatining barqarorligini** oldinga qo‘ydim. Bu yondashuv menga foydalanuvchiga oddiy yordam emas, balki ish samaradorligini oshiruvchi strategik yechim taklif qilish imkonini berdi.

Bo‘lim ostida elektron pochta manzili (business@techfixpro.com) bilan biznes so‘rovlari uchun alohida kontakt maydoni qo‘shilgan. Bu korporativ mijozlar bilan aloqani yanada soddalashtiradi. Ya’ni, korxona vakillari bu yerda alohida murojaat kanalini ko‘radi va bu ularning ishonchini orttiradi.

Umuman olganda, bu sahifani ishlab chiqishda men shunchaki xizmatlarni sanab o‘tish emas, balki biznes foydalanuvchisining bosh og‘rig‘ini tushunish va unga moslashtirilgan echimni vizual va matn orqali aniq yetkazishga harakat qildim. Shu sababli, bu qism mening loyihamdagi eng professional, eng ishonch uyg‘otuvchi segmentlardan biridir.

**“Biz bilan bog‘lanish”**

Mening butun loyiha konsepsiyamda foydalanuvchi bilan bevosita aloqa o‘rnatiladigan eng muhim interfeys qismlaridan biridir. Bu sahifa foydalanuvchilarga muammolarini tushuntirish, savollar berish va xizmat buyurtma qilish imkonini beradi.



Sahifa ikki asosiy ustunga ajratilgan: chap tarafda **aloqa ma’lumotlari**, o‘ng tarafda esa **xabar yuborish formasi** joylashgan.

**Chap tomonda:**

Bu yerda men foydalanuvchiga **real va tezkor bog‘lanish imkoniyatlari**ni taqdim etdim:

* **Manzil:** Toshkent shahar, Sergeli tumani — bu ishonch hissini kuchaytiradi, chunki foydalanuvchi real manzilni ko‘radi.
* **Telefon:** 24/7 ishlaydigan telefon raqam (→ bu xizmat doimiy ochiqligini bildiradi).
* **Email:** support@techfixpro.com orqali esa yozma murojaat qilish mumkin.

Bu ma’lumotlar dizayn jihatdan sodda, ammo zamonaviy ikonkalar bilan kuchaytirilgan, bu esa foydalanuvchiga oson navigatsiya beradi.

**O‘ng tomonda:**

Bu qismda foydalanuvchining bizga murojaat qilish imkoniyati mavjud:

* To‘liq ism
* Email manzili
* Telefon raqami (ixtiyoriy qilingan)
* Mavzu
* Xabar matni (muammo tavsifi)

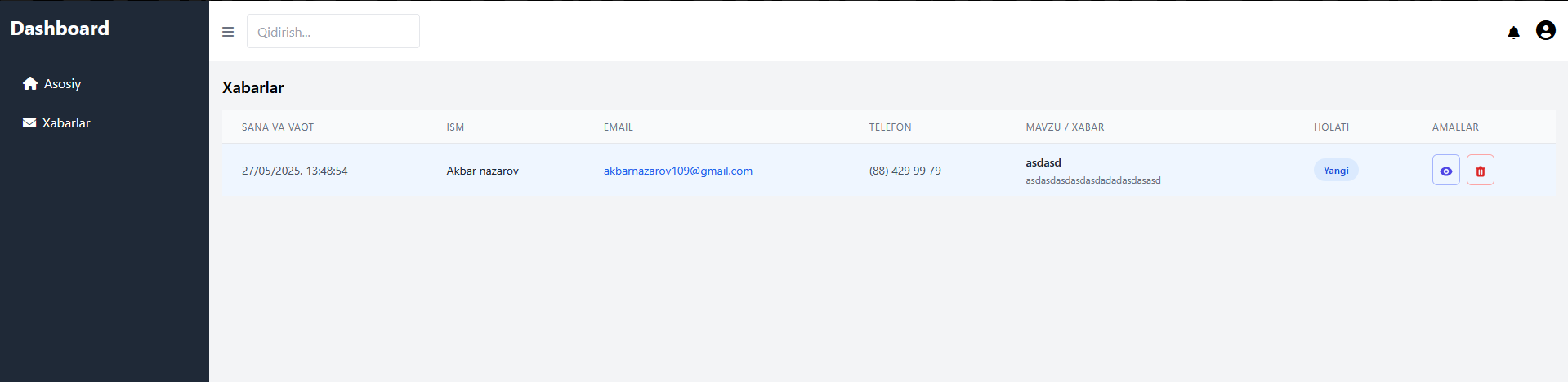
Formaning oxirida joylashgan **"Xabarni yuborish"** tugmasi ochiq, diqqatni tortuvchi va foydalanuvchiga aniq harakat qilish imkonini beradi. Rang tanlovi (yorqin binafsha) esa tugmani ajratib ko‘rsatadi.

Men bu bo‘limni ishlab chiqayotganda shuni nazarda tutdimki, har qanday zamonaviy xizmat sayti uchun bog‘lanish oson, soddalashtirilgan va foydalanuvchi tushunadigan darajada bo‘lishi kerak. Aynan shu maqsadda men **interaktiv, minimal, ammo kuchli aloqa kanali**ni yaratdim.

Bu sahifa orqali foydalanuvchi o‘z muammosini hech qanday ortiqcha bosqichsiz taqdim etadi — bu esa TechRepService xizmatining **foydalanuvchi-yo‘naltirilganligini** ko‘rsatadi. Shu bois, bu qismni ham vizual, ham funksional jihatdan eng muhim sahifa deb hisoblayman.

**Ro’yxatdan o’tish, kirish va admin dashboard qismi**

|  |  |
| --- | --- |
|  |  |

Shundan so‘ng, admin uchun yaratilgan dashboard’ni ishlab chiqdim. Boshida bu sahifa faqatgina xabar matnlaridan iborat oddiy jadval edi. Fikr bildirganlar bu sahifaga “ko‘rish” va “o‘chirish” tugmalari, holat ko‘rsatkichi va sanani qo‘shish kerakligini aytishdi. Shu asosda men “Yangi” deb belgilanadigan status badge’larini qo‘shdim, va foydalanuvchining barcha ma’lumotlari (ism, email, telefon, xabari) to‘liq aks ettirilishini ta’minladim.  


Krsdoshim ushbu sahifalarga navigatsiya paneli va qidiruv maydonchasini qo‘shish bo‘yicha ham maslahat bergan edi. Bu foydalanuvchilar soni ko‘payganda muhim ahamiyatga ega bo‘ladi. Shuningdek, responsive dizayn qo‘shilishi bilan bu sahifalar mobil qurilmalarda ham qulay ishlaydigan holga keltirildi.

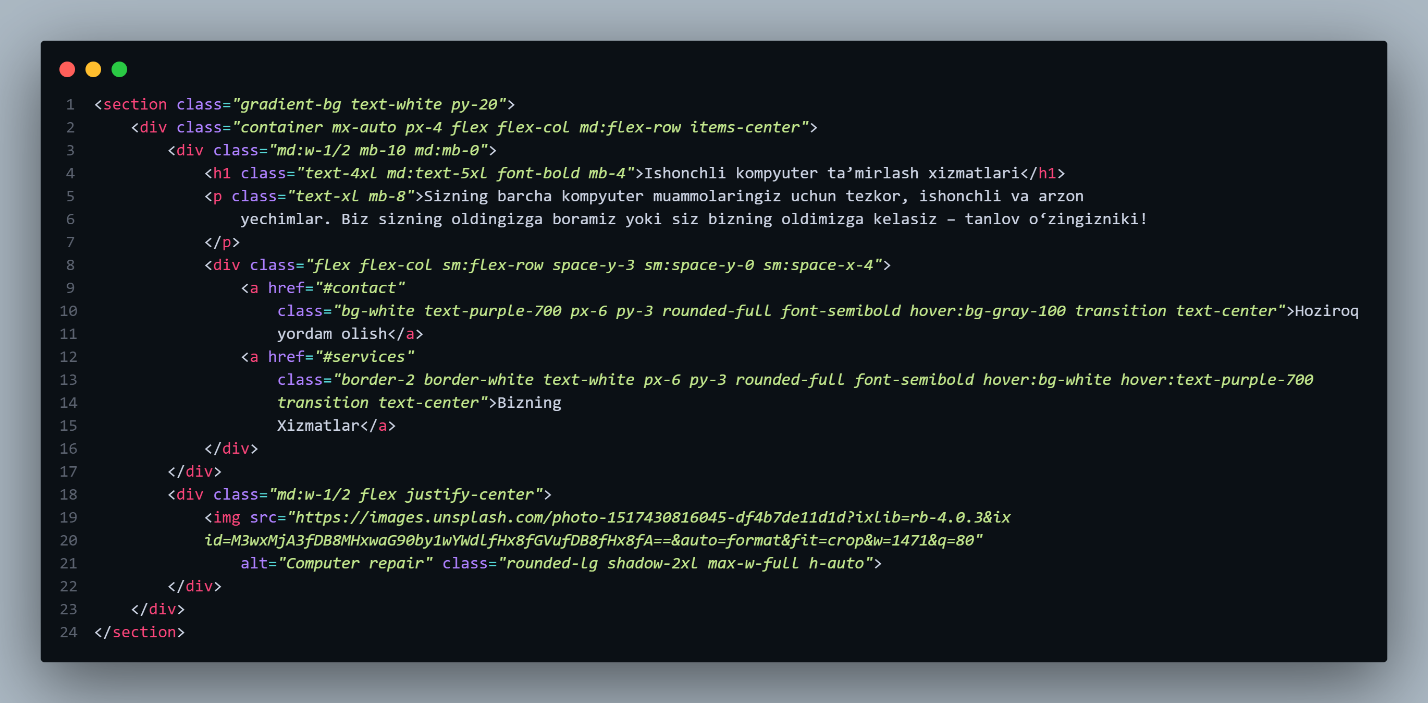
Har bir bu o‘zgarish foydalanuvchi tajribasini yaxshilashga qaratilgan bo‘ldi. Men uchun eng muhim narsa – loyiha texnik jihatdan emas, aynan foydalanishda qulay, toza va intuitiv ko‘rinishda bo‘lishi edi. Aynan o‘z fikrim bilan emas, boshqalar aytgan tavsiyalarni amalda qo‘llaganim sababli, dizaynlarim ko‘p tomonlama boyidi.

Shuningdek, har bir sahifaning kod qismini tozalab, xatoliklarni yo‘qotdim va komponentlar orasida yagona uslubni saqlab qoldim. Masalan, butun loyiha davomida bir xil ranglar, shriftlar, input border’lar va tugma effektlaridan foydalandim. Bu dizaynni bir butun holatga olib keldi.

Umuman olganda, men yaratgan fullstack loyiha dizayni doimiy ravishda takomillashtirib borildi. Har bir bosqichda bildirilgan fikrlar asosida yangiliklar qo‘shdim, keraksiz elementlar olib tashlandi, interfeys esa toza va professional shaklga keltirildi. Shu orqali loyiha nafaqat texnik, balki vizual va amaliy jihatdan ham talab darajasiga yetdi.

**3-TOPSHIRIQ**

Loyihamni ishlab chiqish jarayonida foydalanuvchining e’tiborini tortuvchi va ishonch uyg‘otuvchi **asosiy sahifa (hero section)** ni yaratish muhim ahamiyatga ega deb bildim. Ushbu qismda men Tailwind css dan samarali foydalandim. Shu sababli, index.html faylida yuqoridagi <section class="gradient-bg text-white py-20"> blokini ishlab chiqdim.



Bu bo‘lim orqali men foydalanuvchini saytga kirgan zahoti xizmatlarim bilan tanishtirish va tezda harakatga undashni maqsad qilganman.

Kodda birinchi navbatda **mobil moslashuvchanlikni** inobatga olgan holda, flex flex-col md:flex-row sinflari orqali bloklarni moslashuvchan qilib joylashtirdim. Ya’ni, kichik ekranlarda kontent ustma-ust chiqadi, katta ekranlarda esa yonma-yon bo‘ladi. Bu dizayn foydalanuvchi tajribasini yaxshilashga xizmat qiladi.

Chap tomonda joylashgan matn blokida, men **asosiy sarlavhani** (<h1>) katta shrift bilan va **ishonch uyg‘otadigan jumlalar bilan** yozdim. Masalan: “Ishonchli kompyuter ta’mirlash xizmatlari” degan sarlavha orqali foydalanuvchiga birinchi daqiqadanoq nima taklif qilayotganimni ochiq-oydin ko‘rsatdim.

Undan keyingi <p> elementda xizmatimning asosiy afzalliklarini qisqacha bayon qildim: “Tezkor, ishonchli va arzon”. Shuningdek, foydalanuvchiga qulaylik berish uchun “Biz sizning oldingizga boramiz yoki siz bizning oldimizga kelasiz” degan jumlani qo‘shdim — bu yondashuv xizmatning moslashuvchanligini ko‘rsatadi.

Keyin esa, foydalanuvchining amaliy harakat qilishini rag‘batlantiradigan **ikki muhim tugma (CTA)** joylashtirdim:

* “Hoziroq yordam olish” — bu tugma foydalanuvchini #contact sahifasiga olib boradi.
* “Bizning xizmatlar” — bu esa #services bo‘limiga yo‘naltiradi.

Har ikki tugma rounded-full, hover:bg-white, transition kabi klasslar yordamida zamonaviy va interaktiv ko‘rinishda bo‘lishi uchun yaratilgan. Ayniqsa, birinchi tugmani oq fonda binafsha matn bilan ajratdim — bu foydalanuvchining e’tiborini kuchli jalb qiladi.

Bo‘limning o‘ng tomoniga esa men yuqori sifatli va mazmunli tasvirni joylashtirdim: img src="...". Ushbu rasm texnik xizmat ko‘rsatilayotgan kompyuterlarni aks ettiradi va matn bilan vizual uyg‘unlik hosil qiladi. Rasmga rounded-lg va shadow-2xl sinflari yordamida chuqurlik berilib, u sahifaga nafislik qo‘shadi.

Ushbu bo‘limni ishlab chiqishda men foydalanuvchini o‘z xizmatimga ishontirish va sayt ichida harakatlantirish maqsadini ko‘zlaganman. Dizayn jihatdan esa, bu bo‘lim butun veb-ilovaning ishonchlilik, soddalik va zamonaviylik tamoyillariga asoslanganligini aks ettiradi. Shu jihatdan, men bu kod orqali saytimning ilk taassurotini samarali shakllantira oldim.

**Script.js**

Backend qismida men foydalanuvchi tomonidan yuborilgan kontakt formasini backendga yuborish uchun JavaScript orqali asinxron funksiyani yaratdim.



Bu yerda contactForm elementiga submit hodisasini qo‘shdim va sahifaning default (standart) yangilanishini e.preventDefault() orqali to‘xtatdim. Shundan so‘ng, foydalanuvchining forma orqali kiritgan barcha ma’lumotlarini FormData yordamida yig‘ib oldim va uni Object.fromEntries orqali oddiy obyektga aylantirdim. Bu obyektni JSON.stringify() orqali JSON formatga o‘tkazib, fetch() funksiyasi yordamida POST metodi orqali /api/messages endpointiga yubordim.

Yuborish jarayoni tugagach, agar serverdan ijobiy javob (res.ok) qaytsa, foydalanuvchiga "Xabar yuborildi!" degan alert chiqadi. Aks holda, “Xatolik yuz berdi.” degan alert bilan xatolik haqida ogohlantiraman. Bu kod orqali foydalanuvchi xabarlari aniq va qulay tarzda serverga uzatilishini ta’minladim. Shu bilan birga, foydalanuvchi interfeysini soddalashtirishga va foydalanuvchi tajribasini yaxshilashga erishdim.

Men foydalanuvchining mobil menyusini boshqarish uchun oddiy, ammo samarali JavaScript funksiyasini yozdim.



Bu yerda mobile-menu-button identifikatoriga ega tugmani bosganda, mobile-menu nomli menyu elementining hidden klassini toggle() orqali yoqib-o‘chiriladigan qilib sozladim. Ya’ni, foydalanuvchi tugmani bosganida menyu ochiladi yoki yopiladi.

Bu yondashuv foydalanuvchi interfeysini mobil qurilmalarda yanada qulayroq qilishga yordam beradi. Kod qisqa va samarali bo‘lib, faqat kerakli elementlar mavjud bo‘lsa (?. optional chaining bilan) ishga tushadi, bu esa xatoliklarning oldini oladi. Shu tarzda men loyihamga mobil moslashuvchanlik qo‘shdim.

Foydalanuvchi tizimga kirganmi yoki yo‘qligini aniqlab, sahifadagi foydalanuvchi nomini avtomatik ko‘rsatish funksiyasini yaratdim.



updateNavigation() funksiyasi orqali localStoragedan "loggedInUser" degan elementni olib, uni JSON.parse yordamida JavaScript obyektiga aylantirdim. Agar bu foydalanuvchi obyekti ichida fullName mavjud bo‘lsa, unda user-auth-section identifikatoriga ega HTML elementga foydalanuvchining to‘liq ismini chiqaraman.

Bu funksiya saytning navigatsion qismida foydalanuvchining tizimga kirgan holatini real vaqt rejimida aks ettiradi. Ya’ni foydalanuvchi tizimga kirgandan keyin, sahifaning yuqori qismida uning ismi chiqib turadi. Bu foydalanuvchiga shaxsiylashtirilgan tajriba beradi va interaktivlikni oshiradi.

**Server.js**

Men bu loyihaning server qismini yaratishda **Node.js** va **Express.js** texnologiyalaridan foydalandim. Maqsadim foydalanuvchilarni ro‘yxatdan o‘tkazish, tizimga kiritish, kontakt orqali yuborilgan xabarlarni qabul qilish va ularni faylga saqlash kabi backend funksiyalarni bajarish edi. Loyihamga kirishishdan oldin kerakli modullarni, ya’ni **express, fs, path** va **uuid** ni chaqirdim. Ayniqsa **uuid** moduli orqali har bir foydalanuvchi va xabarga noyob ID berishni yo‘lga qo‘ydim. Ma’lumotlarni oddiy JSON fayllarda saqladim – buning uchun users.json va messages.json dan foydalanildi. Agar bu fayllar mavjud bo‘lmasa, server ishga tushganda ular avtomatik tarzda yaratiladi.

Frontenddagi sahifalarni, masalan, index.html, auth.html va dashboard.html ni to‘g‘ridan-to‘g‘ri brauzerda ko‘rish uchun Express marshrutlaridan foydalandim. Kontakt formasidan yuborilgan xabarlarni /api/messages marshruti orqali qabul qilib, messages.json fayliga yozib qo‘ydim. Har bir xabarni ID orqali o‘chirish yoki o‘qilgan deb belgilash imkoniyati ham mavjud. Bundan tashqari, foydalanuvchilarni ro‘yxatdan o‘tkazish va tizimga kirish uchun POST /api/register va POST /api/login marshrutlarini yozdim. Tizimga kirishda foydalanuvchining roli ham tekshirilib, noto‘g‘ri rol bilan kirishga uringanlarga aniq xatolik xabari qaytariladi. Bu loyihamda men JSON fayllar bilan ishlashda o‘zim yozgan yordamchi funksiyalar – readJsonFile() va writeJsonFile() yordamida ma’lumotlarni o‘qish va yozishni boshqardim. Server ishga tushganda initializeFiles() funksiyasi yordamida barcha fayllar tayyor holatga keltiriladi.

Oxirida esa serverni listen orqali ishga tushirib, port raqamida foydalanuvchilar bilan aloqa qilish imkonini ochdim. Umuman olganda, bu backend qismi yordamida men to‘liq ishlaydigan, oddiy, ammo foydali tizim yaratdim. Bu loyiha orqali foydalanuvchi tajribasini oshirish, xavfsizlikni ta'minlash va har bir yuborilgan xabarni nazorat qilish imkoniyatini berdi. Hammasini o‘z qo‘lim bilan yozdim va har bir satrni tizimning ehtiyojlariga moslashtirdim.

**Express serverni sozlash va marshrutlarni yaratish**

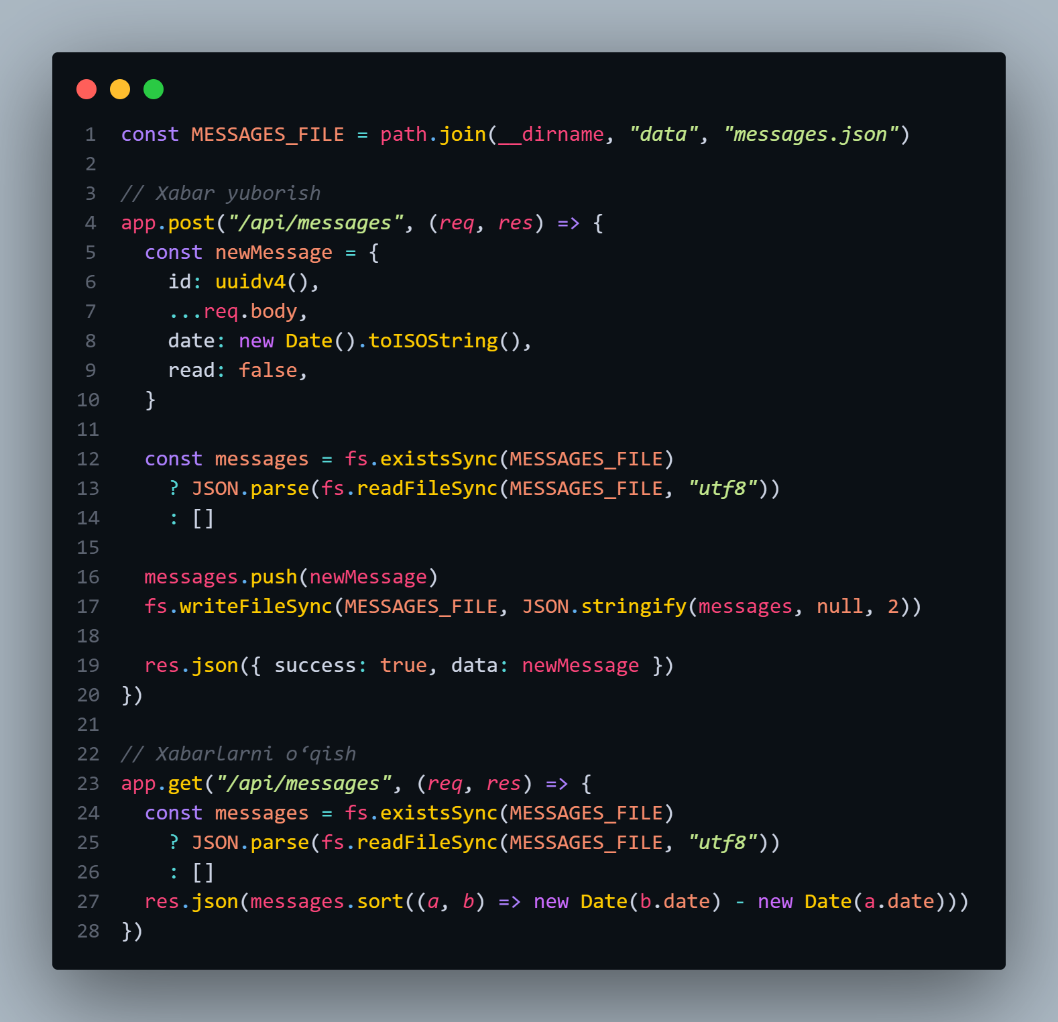
Men loyihamda Express.js kutubxonasidan foydalandim, chunki u sodda va kuchli backend platforma bo‘lib, REST API’lar yaratishda juda qulay. Dastlab serverni ishga tushurish uchun kerakli express, path, fs va uuid paketlarini import qildim.



So‘ng express() funksiyasi orqali app obyektini yaratdim va PORT raqamini belgilab oldim. HTML sahifalarni xizmatga taqdim etish uchun app.get() metodlaridan foydalandim. Ular yordamida foydalanuvchilar /, /auth.html va /dashboard.html yo‘llari orqali sahifalarga kira olishadi. Shuningdek, public papkasini statik fayllar uchun ochib qo‘ydim.

**Xabarlarni yuborish, o‘qish va o‘chirish uchun API yo‘llari.**

Mijozlardan keladigan xabarlarni qabul qilish va ularni JSON faylga saqlash uchun POST /api/messages yo‘lini yozdim.



Bu yerda foydalanuvchi ism, email, telefon, mavzu va xabar matnini yuboradi. Men har bir xabarga uuid orqali noyob ID berdim va xabarni vaqt belgisi bilan saqladim. GET /api/messages orqali esa barcha xabarlarni o‘qib, vaqt bo‘yicha saralab foydalanuvchiga ko‘rsatdim. Bundan tashqari, POST /api/messages/:id/read orqali xabar holatini “O‘qilgan” deb belgilayman va DELETE /api/messages/:id yo‘li orqali xabarni o‘chiraman. Bu ishlarni bajarishda fs moduli orqali JSON fayllarni o‘qib-yozish funksiyalarini yaratdim.

**Foydalanuvchi ro‘yxatdan o‘tishi va kirishi uchun autentifikatsiya.**

Men foydalanuvchilarning ro‘yxatdan o‘tishi uchun POST /api/register endpointini yaratdim.



Bu joyda foydalanuvchidan username, parol, ism va roli (admin yoki user) olinadi. Yangi foydalanuvchi users.json fayliga saqlanadi. Agar foydalanuvchi allaqachon mavjud bo‘lsa, men unga xatolik xabarini yuboraman. Kirish (login) uchun esa POST /api/login endpointini yozdim.



Unda foydalanuvchining username, parol va roli tekshiriladi. Rol noto‘g‘ri kiritilgan bo‘lsa, foydalanuvchiga aniq tushuntirish beriladi. Bu jarayonlarda xavfsizlikni ta’minlash uchun foydalanuvchi rolini ham tekshirib qo‘ydim, ya’ni admin bo‘lmagan foydalanuvchi dashboard sahifasiga kira olmaydi.

**DASHBOARD QISMI: XABARLARNI OLISH VA JADVALGA CHIQARISH FUNKSIYASI**

Foydalanuvchi qoldirgan xabarni Admin bo’limiga ya’ni dashboard yuborisih uchun men ushbu funksiyani yozdim:



Bu funksiya orqali men dashboard sahifasida foydalanuvchilardan kelgan xabarlarni jadval shaklida chiqarishni tashkil qildim. Avval HTML-dagi <tbody id="messagesTableBody"> elementini tanlab oldim, chunki barcha xabarlar shu qismga joylashtiriladi.

So‘ngra fetch funksiyasi yordamida /api/messages endpointiga so‘rov yuboraman va undan qaytgan JSON ma’lumotlarni olib, messages nomli massivga saqlayman. Har safar sahifa yangilansa, eski xabarlar tozalansin deb messagesTableBody.innerHTML = '' qilib tozalab qo‘yaman.

Agar hech qanday xabar bo‘lmasa, foydalanuvchiga "Xabarlar topilmadi" degan xabar chiqadi.

Aks holda, har bir msg (ya’ni har bir xabar) uchun HTML jadvaliga yangi satr (row) qo‘shaman. Bu satrda:

* **Vaqti** — timestamp ni foydalanuvchiga qulay ko‘rinishda chiqaraman.
* **Ismi**, **emaili**, **telefon raqami** — bo‘sh bo‘lsa "-" ko‘rsatiladi.
* **Mavzusi va xabari** — xabarning dastlabki 70 belgisi ko‘rsatiladi.
* **Statusi** — msg.status qiymatida Yangi yoki O'qilgan deb chiqadi.
* **Amallar** — ikki tugma bor: biri "O'qildi" deb belgilash uchun, ikkinchisi esa xabarni o‘chirish uchun yozilgan markMessageAsRead(id) va deleteMessage(id) funksiyalarini chaqirad

**Xabarni o‘qilgan deb belgilash.**

Dashboard sahifasida xabar o‘qilganda, uning holatini "O‘qildi" deb belgilash uchun yozdim.



Funksiya id parametrini qabul qiladi — bu har bir xabarning noyob identifikatori.

Funksiya ichida fetch yordamida serverga POST metodida so‘rov yuboraman. Yo‘l /api/messages/${id}/read bo‘lib, bu orqali aynan shu idga ega xabarni o‘qilgan deb belgilaydi. Agar bu amaliyot muvaffaqiyatli bajarilsa, fetchAndDisplayMessages() funksiyasini qayta chaqiraman, bu esa jadvalni yangilab, xabar holatini avtomatik ko‘rsatadi.

Agar biron bir xatolik yuz bersa, console.error yordamida xatoni konsolga chiqarib ko’rishim mumkin. Shu kichik, lekin muhim funksiya orqali men admin panelda xabarlarni "O‘qildi" deb belgilab, ularni holatiga qarab boshqarish imkonini yaratdim. Bu tizimni yanada interaktiv qiladi va foydalanuvchi tajribasini oshiradi.

**XABARNI O’CHIRISH.**

Men foydalanuvchi tomonidan yuborilgan xabarni o‘chirish imkoniyatini yaratish uchun quyidagi funksiyani yozdim:

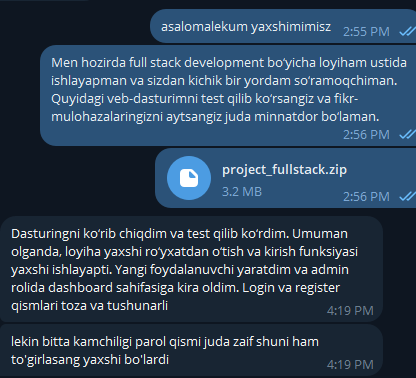


Bu yerda foydalanuvchi xabarni o‘chirmoqchi bo‘lsa, dastlab confirm() orqali tasdiq so‘rayman. Agar foydalanuvchi “OK” tugmasini bossagina DELETE metodida serverga so‘rov yuboriladi. Ushbu so‘rov id bo‘yicha aniq bir xabarni aniqlab, users.json yoki messages.json faylidan uni olib tashlaydi. Jarayon tugagach, fetchAndDisplayMessages() funksiyasini chaqirib, jadvalni avtomatik ravishda yangilayman.

Agar biror xatolik yuz bersa, uni console.error() orqali brauzer konsolida chiqaraman. Shu bilan men xabarlarni oddiy, tez va xavfsiz tarzda boshqarish imkoniyatini qo‘shdim.

**4-TOPSHIRIQ**

Men tayyorlagan proyektimni mendan kuchliroq dasturchi akamga yubordim va uning aytishida dasturda xavfsizlikka doir masalada zaifliklar bor ekan.

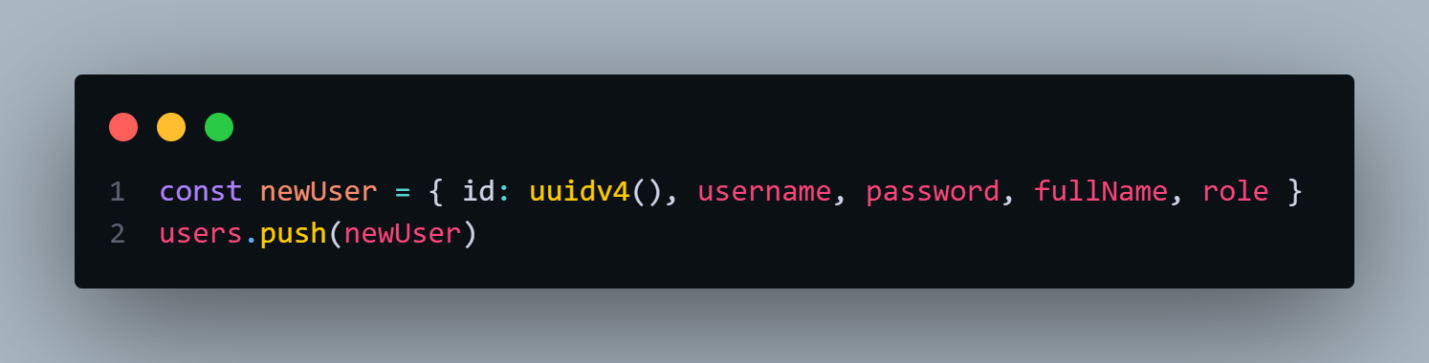


Ya’ni fikr bildiruvchi menga sening server.js faylingdagi POST /api/register endpointida foydalanuvchi username, password, fullName va role bilan ro‘yxatdan o‘tmoqda. Ammo **parolni hech qanday shifrlamasdan** users.json faylga yozayapsan. Bu xavfsizlik nuqtai nazaridan jiddiy muammo hisoblanadi. Deya ta’kidlab o’tgan.

Men bu muammoni inobatga olib ushbu dasturimni quyidagi kodlar bilan xavfsizligini taminladim:



Oldingi kodim esa shunchaki ma’lumotlarni shifrlamasdan saqlab ketar edi.



Foydalanuvchi ro‘yxatdan o‘tganida, men uni tizimga yangi foydalanuvchi sifatida qo‘shdim. Buning uchun avval uuidv4() funksiyasi yordamida har bir foydalanuvchi uchun noyob (takrorlanmas) identifikator yaratdim — bu id maydoniga yozildi. So‘ngra foydalanuvchi tomonidan yuborilgan username, password, fullName va role qiymatlarini olib, yangi newUser obyektini shakllantirgan edim.

Bu newUser obyektini users massiviga push() metodi orqali qo‘shdim. Ya’ni, foydalanuvchi tizimdagi mavjud users.json faylga avtomatik tarzda qo‘shiladi. Shundan so‘ng bu foydalanuvchi boshqa sahifalarda tan olinadi va dashboardga kira oladi.

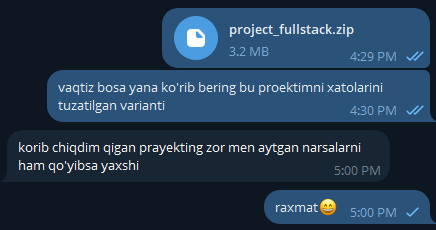
**YANGILANGAN XAVFSIZLIK**

Yangilangan kodimda esa men foydalanuvchi ma’lumotlarini xavfsiz saqlash uchun bcryptjs kutubxonasidan foydalandim. Foydalanuvchi parolini ochiq matn ko‘rinishida emas, balki shifrlangan (hashlangan) holda saqlashni maqsad qildim. Shu sabab, foydalanuvchi register sahifasida parolni kiritgach, uni bcrypt.hashSync(password, 10) funksiyasi orqali 10 martalik "salt" bilan shifrladim va bu qiymatni hashedPassword nomli o‘zgaruvchiga yozdim.

So‘ng, uuidv4() orqali foydalanuvchiga noyob identifikator (ID) yaratdim va yangi foydalanuvchini username, fullName, role va endi shifrlangan password bilan newUser obyektiga joyladim. Bu newUser obyektini users massiviga qo‘shdim, ya’ni saqlashga tayyor holatga keltirdim.

Bu yondashuv xavfsizlikni ancha oshirdi — foydalanuvchi parollari endi JSON faylda shifrlangan holda saqlanadi, bu esa tizimni buzilish yoki ma’lumotlar sizib chiqishidan himoya qiladi.

Keyin yaxshilangan dasturimni yana akamga tashlab ulardan quyidagicha izoh oldim:



Bu fikr loyihamning umumiy ishlashidan ijobiy taassurot qoldirganini bildiradi va qo‘shimcha tavsiyalarni ham e’tiborga olishim kerakligini ko‘rsatadi. Foydalanuvchi takliflariga asoslanib, keyingi bosqichda interfeysda yana biroz qulayliklar va funksiyalarni yaxshilashni rejalashtirdim.

**XULOSA**

TechRepService deb nomlangan ushbu loyiha davomida men nafaqat texnik bilimlarimni, balki muammolarga echim topishdagi mantiqiy fikrlash qobiliyatimni ham amalda qo‘lladim. Men tanlagan full stack yondashuv orqali frontend va backend qismlarini yagona tizim sifatida boshqarishga erishdim. Loyihamning boshida kompaniyaning amaldagi muammolari: telefon orqali xizmat buyurtmasi, foydalanuvchi bilan cheklangan aloqa, xizmatlar monitoringining yo‘qligi kabi masalalarni chuqur o‘rganib chiqdim. Shundan so‘ng ushbu muammolarni zamonaviy texnologiyalar orqali yechish rejasini tuzdim.

Frontend qismini yaratishda HTML, Tailwind CSS va JavaScript texnologiyalaridan foydalandim. Bu texnologiyalar menga oddiy, lekin zamonaviy va moslashuvchan interfeys yaratishda yordam berdi. Ayniqsa, foydalanuvchi tajribasini hisobga olgan holda har bir sahifani mobil moslashuvchanlik bilan ishlab chiqdim. Mijoz ro‘yxatdan o‘tishi, tizimga kirishi, xabar yuborishi va xizmat haqida to‘liq ma’lumot olishi mumkin bo‘lgan sahifalar ishlab chiqildi. Har bir tugma, rasm, matn va forma foydalanuvchining harakatini osonlashtirish uchun ehtiyotkorlik bilan tanlandi.

Backend qismida esa Node.js va Express.js asosida server tuzdim. Ma’lumotlar bilan ishlash uchun JSON fayllarni tanladim, bu menga loyihaning dastlabki bosqichlarida oddiy va samarali saqlash usulini berdi. Kirish va ro‘yxatdan o‘tish marshrutlari bilan bir qatorda, foydalanuvchidan keladigan xabarlarni olish, ularni holatiga qarab yangilash yoki o‘chirish imkonini beruvchi marshrutlar yozdim.

Loyihani yanada xavfsiz qilish uchun foydalanuvchi parollarini ochiq matnda emas, bcryptjs kutubxonasi yordamida shifrlangan holda saqlashni yo‘lga qo‘ydim. Bu o‘zgarish loyihani professional xavfsizlik talablariga yaqinlashtirdi. Admin panel (dashboard) orqali foydalanuvchi xabarlari ko‘rsatiladi, ularni o‘qilgan deb belgilash yoki o‘chirish funksiyalari mavjud. Bularning barchasi JavaScript yordamida dinamik tarzda boshqariladi.

Loyihani yaratishda test foydalanuvchilardan fikr oldim. Jumladan, akam kabi tajribali dasturchidan loyiha xavfsizligi va dizayni borasida maslahatlar oldim. Ularning fikrlari asosida men autentifikatsiya tizimini kuchaytirdim, parollarni shifrlashni joriy qildim va interfeysda bir nechta qulayliklar qo‘shdim. Bu iteratsion yondashuv — foydalanuvchi fikri asosida doimiy takomillashtirish — mening loyihamni real ehtiyojlarga yanada yaqinlashtirdi.

Umuman olganda, men bu loyiha orqali faqat dasturlash emas, balki tizimiy fikrlash, foydalanuvchining ehtiyojini anglash, xavfsizlikni hisobga olish, va samaradorlikni oshirish bo‘yicha katta tajriba orttirdim. Men biror tayyor platformadan foydalanmadim, balki har bir kod satrini o‘zim ishlab chiqdim. Har bir funksiya — foydalanuvchining ehtiyojidan kelib chiqib yozildi. Bu esa meni nafaqat yaxshi dasturchiga, balki yechim taklif qiluvchi mutaxassisga aylantirdi. Shunday qilib, TechRepService loyihasi mening professional o‘sishimda muhim bosqich bo‘lib, kelajakdagi yanada yirik loyihalarga tayyorgarlik vazifasini bajardi.