ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

Калиев Дидар Муктарұлы

Тақырыбы: “Клиенттердің жазбаларын есепке алу үшін мобильді қосымшаны әзірлеу"

ТҮСІНДІРМЕ ЖАЗБА
дипломдық жобаға

5В070400 – «Есептеу техникасы және бағдарламалық қамтамасыз ету»

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
ПИ кафедрасының меңгерушісі
физ.-мат. ғыл.канд, профессор
_____ А.Н.Молдагулова
"25" 05 2022 ж.

ТҮСІНДІРМЕ ЖАЗБА
дипломдық жобаға

Тақырыбы: "Клиенттердің жазбаларын есепке алу үшін мобильді қосымшаны әзірлеу"

5В070400 – «Есептеу техникасы және бағдарламалық қамтамасыз ету» мамандығы

Орындаған Калиев Дидар Муктарұлы

Рецензент
физика-математика
ғылымдарының кандидаты,
доцент
_____ Мансурова М.Е
"25" 05 2022 ж.

Ғылыми жетекші
PhD, қауымдастырылған-профессор
_____ Бейсембекова Р.Н
"24" 05 2022 ж.

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

БЕКІТЕМІН
ПИ кафедрасының меңгерушісі
физ.-мат. ғыл.канд, профессор
_____ А.Н.Молдагулова
"25" 05 2022 ж.

Дипломдық жобаны орындауға
ТАПСЫРМА

Білім алушыға Калиев Дидар Муктарұлы
Тақырыбы: "Клиенттердің жазбаларын есепке алу үшін мобильді қосымшаны әзірлеу"
Академиялық мәселелер жөніндегі проректор бұйрығының №5005 "07" 04 2022 ж. шешімімен бекітілген.
Орындалған жобаның өткізу мерзімі "27" 05 2022 ж.
Дипломдық жобаның бастапқы мәліметтері:
Есеп – түсініктеме жазбаның талқылауға берілген сұрақтардың тізімі:
а) Веб қосымшаның артықшылықтары қандай
б) Пайдаланушы интерфейсін жобалау және жасау
в) Мобильді қосымшаны әзірлеу және тестілеу
Графикалық материалдар тізімі (міндетті суреттердің нақты көрсетілуімен):
Презентацияның 18 слайдпен берілген құжат түрінде ұсынылған.
Ұсынылған негізгі әдебиеттер: 20 пайдаланылған әдебиеттер тізімінен

Дипломдық жобаны орындау
КЕСТЕСІ

Бөлімдердің атаулары, зерттелген мәселелердің тізімі	**Ғылыми жетекшіге және кеңесшілерге ұсыну мерзімі**	**Ескерту**
1. Дипломдық жоба тақырыбына байланысты ақпараттарды іздеу	30.01.2022	орындалды
2. Жобаны UML диаграммалары арқылы моделдеу.	15.02.2022	орындалды
3. Дипломдық жобаның веб қосымшасын жасау	1.03.2022	орындалды
4. Дипломдық жұмысты комиссияға көрсету, ескертулерді түзеу	7.04.2022	орындалды
5. Презентация дайындау	10.05.2022	орындалды

Дипломдық жұмыс бөлімдерінің кеңесшілерінің аяқталған жұмысқа қойған **қолтаңбалары**

Бөлімдер атауы	**Кеңес берушілер (аты-жөні, тегі, ғылыми дәрежесі, атағы)**	**Қолтаңба қойылған мерзімі**	**Қолы**
Нормалық бақылаушы	Жекамбаева М.Н. PhD, қауымдастырылған-профессор	23.05.22	
Бағдарламалық бөлім	Марғұлан Қ. тех. ғыл. магистірі, лектор	23.05.22ж	

Ғылыми жетекші _____ Бейсембекова Р.Н.

Тапсырманы орындауға қабылдап алған студент _____ Калиев Д.М.

Күні «12» 11 2021 ж.

КІРІСПЕ

Қазіргі кезде әлемде бизнестің негізгі міндеті клиенттермен тиімді байланыс орнату болып табылады. Ол үшін кәсіби түрде жасалған Instagram есептік жазбалары жасалады, желіде жарнама жазылады, веб-қосымшалар жасалады. Бірақ жоғарыда аталған құралдар смартфондарға бағытталған: Instagram бастапқыда өзін мобильді қосымша ретінде көрсетті, жарнама көбінесе Instagram-да іске қосылды, ал барлық сайттарда мобильді нұсқасы бар. Қазір қосымшалар барлық сферада: ойын – сауық, денсаулық, сатып алу, бизнес және т.б., сондықтан қазір кез-келген бизнес: шағын, орта, үлкен-жеке мобильді қосымшаны құруға ұмтылады.

ҚР Ұлттық экономика министрлігі Статистика комитетінің есептік деректеріне сәйкес 2018 жылғы ЖІӨ-дегі тауарлар өндірісі мен қызметтер өндірісінің үлесі Қазақстанның еңбекке қабілетті халқының 66.1% - ы қызмет көрсету саласында жұмыс істеді [1]. Бұл пайыз жыл сайын өсіп келеді. Өсуге мобильді қосымшалар айтарлықтай үлес қосады, өйткені олар клиенттермен жұмыс істеуді және бизнесті енгізу процесін айтарлықтай жеңілдетеді. Адалдық бағдарламалары, push хабарландырулары сияқты әртүрлі құралдардың көмегімен клиенттерді өз қызметтерін пайдалануға ынталандыруға болады. Құрылғыға қарамастан, клиент қызметтің барлық функционалдығын пайдалана алатындай бағдарлама кросс-платформа болуы керек. Сондай-ақ, жеке қосымшаның бренд имиджін едәуір арттыратынына назар аударған жөн, бұл шағын бизнес үшін маңызды. Осылайша, жеке қосымшалар жыл сайын өзекті бола бастайды деп қорытынды жасауға болады.

Жоғарыда айтылғандарға сәйкес жұмыстың мақсатын тұжырымдауға болады: клиенттердің тапсырыстарын онлайн жазуға арналған платформалық мобильді қосымшаны әзірлеу. Осы мақсатқа жету үшін келесі міндеттер құрылды:

- тақырып аймағын талдау;
- клиенттердің талаптарын талдау;
- ұқсас жүйелерді талдау арқылы негізгі талаптарды анықтау;
- жүйеге қойылатын талаптар мен техникалық тапсырманы құрастыру;
- мобильді қосымшаның тұжырымдамалық моделін жобалау;
- мобильді қосымшаның интерфейсін жобалау;
- Django сияқты мобильді қосымшаны жүзеге асыруда қолданылатын технологиялар мен құралдарды зерттеңіз. Forms, SQLite және басқалары;
- қосымшаны әзірлеу және оның функционалдығын тексеру.

1 Пәндік саланы талдау

1.1 Django деген не

Django - бұл MVC дизайн шаблонын қолдана отырып, Python веб-қосымшаларына арналған негіз.
Django-дағы Сайт иеліктен шығарылатын және қосылатын бір немесе бірнеше қосымшалардан тұрады. Бұл осы шеңбердегі кейбір басқа жақтаулардан маңызды архитектуралық айырмашылықтардың бірі. Жақтаудың негізгі қағидаларының бірі - DRY (ағылш. Don't repeat yourself) [5].
Сондай-ақ, басқа жақтаулардан айырмашылығы, Django-дағы URL өңдеушілері тұрақты тіркестер арқылы нақты конфигурацияланады.
Деректер базасымен жұмыс істеу үшін Django өзінің ORM-ін қолданады, онда Деректер моделі Python сыныптарымен сипатталады және оған сәйкес мәліметтер базасының схемасы жасалады.
Django Веб-шеңбері Instagram, Disqus, Mozilla, The Washington Times, Pinterest, YouTube, Google және т. б. сайттарда қолданылады.

Django архитектурасы "модель-көрініс-Контроллер" (MVC) сияқты. Классикалық MVC моделінің контроллері Django-да көрініс деп аталатын деңгейге сәйкес келеді (ағылш. View), ал презентациялық логика Django-да Шаблондар деңгейімен жүзеге асырылады (ағылш. Template). Осыған байланысты Django деңгейлі архитектурасы көбінесе "модель-үлгі-көрініс" (MTV) деп аталады.

Django-ның жаңалықтар ресурстарын пайдалану құралы ретінде алғашқы дамуы оның архитектурасына айтарлықтай әсер етті: ол ақпараттық сипаттағы веб-сайттарды тез дамытуға көмектесетін бірқатар құралдарды ұсынады. Мысалы, әзірлеушіге сайттың әкімшілік бөлігі үшін контроллерлер мен беттер жасаудың қажеті жоқ, Django-да Django-да жасалған кез-келген сайтқа қосылатын және бір серверде бірден бірнеше сайтты басқара алатын кірістірілген мазмұнды басқару қосымшасы бар. Әкімшілік қосымша барлық жасалған әрекеттерді тіркеп, сайтты толтырудың кез келген объектілерін жасауға, өзгертуге және жоюға мүмкіндік береді және пайдаланушылар мен топтарды басқаруға арналған интерфейсті ұсынады.
Джангоның кейбір мүмкіндіктері:

-orm, транзакцияны қолдайтын дерекқорға кіру API;

-кірістірілген әкімші интерфейсі, көптеген тілдерге аудармалары бар;

-тұрақты өрнектерге негізделген URL менеджері;

-тегтермен және мұрагерлікпен шаблондардың кеңейтілген жүйесі;

-кэштеу жүйесі;

-интернационализация;

-кез-келген Django веб-сайттарына орнатуға болатын қосымшалардың қосылған архитектурасы;

-"generic views" — контроллер функциясының үлгілері;

-авторизация және аутентификация, сыртқы аутентификация модульдерін қосу: LDAP, OpenID және т.б.;

-қосымша сұрау өңдеушілерін құруға арналған сүзгі жүйесі ("middleware"), мысалы, кэштеу, қысу, URL мекен-жайларын қалыпқа келтіру және анонимді сессияларды қолдау үшін тарату сүзгілеріне енгізілген;

-формалармен жұмыс істеуге арналған кітапхана (мұрагерлік, қолданыстағы ДБ моделі бойынша формаларды құру);

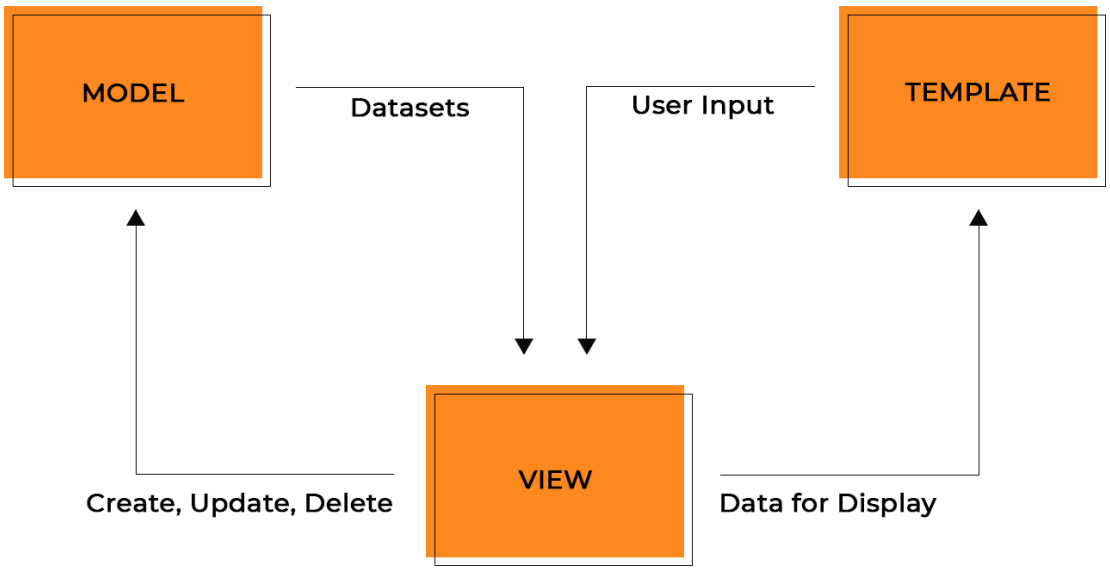
-әкімшілік бағдарлама арқылы қол жетімді шаблон тегтері мен деректер үлгілері бойынша автоматты құжаттама.

Қазіргі уақытта PostgreSQL дерекқорынан басқа, Django басқа ДҚБЖ-мен жұмыс істей алады: MySQL, SQLite, Microsoft SQL Server, DB2, Firebird, SQL Anywhere және Oracle.

Django-да әзірлеу үшін өзінің веб-сервері бар. Сервер жобаның бастапқы кодындағы файлдардағы өзгерістерді автоматты түрде анықтайды және қайта іске қосылады, бұл Python-да даму процесін тездетеді.

1.2 Неліктен MVT архитектурасы Django веб-қосымшалары үшін маңызды: деңгейлерді бөлу және тәуелсіз пайдалану

Django шеңбері Python бағдарламалау тілінде жазылған, сондықтан оның құрылымы тілдің ерекшеліктеріне сәйкес келеді. Жасаушылар Django-да MVC үлгісін енгізді және ол жақтаудың қазіргі нұсқасында қолданылады.
MVC архитектурасы әзірлеушіге қосымшаның визуалды көрінісі мен іскери логикасымен бөлек жұмыс істеуге мүмкіндік береді. Айтпақшы, Django — мен жұмыс істеу кезінде мамандар MVT-Model-View-Template немесе модель-көрініс-шаблон терминін жиі қолданады. MVT компоненттерін бір-біріне тәуелсіз пайдалануға болады 1.1-суретте көрсетілген.



1.1-сурет – “MVT архитектурасының схемасы”

Django құжаттамасы модельді (модельді) "деректердің негізгі өрістері мен мінез-құлқын қамтитын деректер көзі"ретінде анықтайды. Әдетте бір модель дерекқордағы бір кестені көрсетеді. Django PostgreSQL, MySQL, SQLite және Oracle дерекқорларын қолдайды.

Модельдерде деректер туралы ақпарат бар. Бұл деректер атрибуттармен немесе өрістермен ұсынылған. Модель қарапайым класс болғандықтан, ол Джангоның басқа деңгейлері туралы ештеңе білмейді. Деңгейлер арасындағы өзара әрекеттесу API арқылы жүреді.

Модель бизнес логикасына, әдістеріне, қасиеттеріне және деректерді басқаруға байланысты басқа элементтерге жауап береді. Сондай-ақ, модельдер әзірлеушілерге дерекқордағы нысандарды құруға, оқуға, жаңартуға және жоюға мүмкіндік береді.

View (көрініс) үш мәселені шешеді: HTTP сұрауларын қабылдайды, әдістер мен қасиеттермен анықталған бизнес логикасын жүзеге асырады, сұраныстарға жауап ретінде HTTP жауабын жібереді. Яғни, көрініс модельден деректерді алады және шаблондарға (templates) сол деректерге қол жеткізуге мүмкіндік береді немесе деректерді алдын-ала өңдейді, содан кейін шаблондарға қол жеткізуге мүмкіндік береді.

Django-да қуатты шаблон қозғалтқышы және өзіндік белгілеу тілі бар. Шаблондар-бұл деректерді көрсететін HTML коды бар файлдар. Файл мазмұны статикалық немесе динамикалық болуы мүмкін. Үлгілерде бизнес логикасы жоқ. Сондықтан олар тек деректерді көрсетеді.

Дамыған экожүйесін қолдану:

Тәжірибелі әзірлеушілер Django-ны жүйе ретінде қабылдауға кеңес береді. Бұл фреймворк әдетте көп тарапты қосымшалар. Оларды белгілі бір жобаның қажеттіліктеріне байланысты таңдауға болады.

Бұл принципті жақсы түсіну үшін LEGO конструкторын елестетіп көріңіз. Онда көптеген типтік блоктар бар. Джангода да типтік блоктар бар. Мысалы, авторизация блогы немесе ақпараттық бюллетеньге жазылу блогы кез-келген жобада қолданылады. Фреймворк көмегімен жасалған веб-қосымшалар осындай тәуелсіз блоктардан тұрады.

Әкімшілік панелін қолдану:

Бағдарлама жасалған кезде Django әкімшілік тақтасы автоматты түрде жасалады. Бұл әзірлеушіні қолмен басқару тақтасын құру қажеттілігінен құтқарады.

Үшінші тарап қосымшаларының көмегімен Django әдепкі басқару консолін жетілдіруге және жобаның қажеттіліктеріне бейімдеуге болады. Сонымен қатар, жақтау әдепкі әкімшілік панельдің интерфейсін теңшеуге мүмкіндік береді.

SEO-достық:

Python-да жазылған код тіпті дайын емес адамдар үшін оқылатын және түсінікті болып шығады. Бұл Python веб-қосымшаларын SEO-ға ыңғайлы деп санайтын факторлардың бірі. Django семантикалық URL мекен-жайларын жасайды. Оларды адам түсінетін URL мекен-жайы немесе CNC деп те атайды.

Django қосымшаларында іздеу жүйесін оңтайландыру үшін қажет басқа мүмкіндіктер оңай жүзеге асырылады.

Django-нің кеңеюі:

Django функционалдығы плагиндер арқылы кеңейеді. Бұл сайтқа қажетті функцияны тез қосуға мүмкіндік беретін бағдарламалық модульдер. Ресми каталогта sitemap сайтында іске асыруды жеңілдететін жүздеген плагиндер бар.xml, кіруді басқару, Stripe төлем жүйесін қосу және т.б. Қажет болса, бағдарламаны жобаның ағымдағы қажеттіліктеріне бейімдеу үшін плагиндерді өшіруге немесе ауыстыруға болады.

Кітапханалары:

Танымал бағдарламалау тілдерінде арнайы мәселелерді шешуге ыңғайлы кітапханалар бар. Кітапханаларда сіз дайын шешімдерді таба аласыз: функциялар, сыныптар, конфигурациялар және т.б. Осындай шешімдердің арқасында тілдің мүмкіндіктері кеңейеді, сонымен қатар қосымшаларды құру жеңілдетіледі.

Django веб-қосымшаларды әзірлеу кезінде кітапханаларды пайдалануды қолдайды. Танымал кітапханаларға мыналар кіреді:

-Django REST Framework, ол API-мен жұмыс істеуді жеңілдетеді;

-Django CMS-мазмұнды басқарудың ыңғайлы құралы;

-Django-allauth-оның көмегімен тіркеу, авторизациялау, шоттарды басқару функциялары жүзеге асырылады.

ORM жүйесі:

Django-да объектінің реляциялық дисплейі (ORM) жүзеге асырылады, ол қосымшаның мәліметтер базасымен (ДБ) өзара әрекеттесуін қамтамасыз етеді. ORM деректерді PostgreeSQL немесе MySQL сияқты дерекқордан бағдарлама кодында қолданылатын нысандарға автоматты түрде жібереді.

Неге Джангоны таңдау керек?

Әзірлеушілер келесі сипаттамалардың арқасында Джангоны таңдайды:

-бизнес логикасы мен көрнекі бөлігін сәулет деңгейінде бөлу;

-SEO-достық;

-дамыған инфрақұрылым: көптеген кітапханалар мен плагиндер.

1.3 Тұтынушы талаптарын талдау және талаптарды анықтау

Қазіргі әлемде кез-келген бизнес жеке мобильді қосымшаны құруын көздейді. Біріншіден, бұл компанияның имиджін арттырады, өйткені оның болуы жоғары қызметті көрсетеді. Сондықтан бағдарлама интерфейсі менеджердің жақсы бағдарлануы үшін жағымды дизайн мен логотипке ие болуы керек.

Екіншіден, смартфондағы қосымша оны пайдаланушылардың уақытын үнемдейді, өйткені менеджерлерге жазу үшін қосымша контактілерді, жазбаның мекен-жайы мен формасын іздеудің қажеті жоқ. Мұны істеу үшін

қосымшаны ашқан кезде менеджер белгілі бір мақсаттар үшін қайда басу керектігін түсінуі үшін ең түсінікті интерфейсті жасау керек. Ең жақсы шешім – әрқайсысының өз функционалдығы, жазу формасы және Тапсырыс күйі болатын қойындылары бар бетті пайдалану. Қосымшаның аясында менеджер қызметкерлермен немесе әкімшімен туындаған мәселелерді тез шеше алады, өйткені оның тарихы сақталады.

Үшіншіден, әр менеджер өз деректерінің құпиялылығына қызығушылық танытады, сондықтан әр пайдаланушы үшін есептік жазба жасау керек, сондықтан тіркелгі иесінен басқа ешкім жазбалардың жеке тарихын көре алмайды. Ол үшін тіркелу мүмкіндігінсіз авторизация бетін құру қажет. Сондай-ақ, өзіңіздің интерфейсіңізбен және қосымша функционалдылығыңызбен әкімші тіркелгісін жасау қажет. Ол пайдаланушыларға рөлдерді беру құқығына ие болуы керек, сондықтан қызметкер оған барлық жазбаларды көреді, ал менеджерлер тек өздері жазу. Сондай-ақ, әкімшінің барлық жазбаларды көруге мүмкіндігі болуы керек.

Клиенттердің өсуі үшін кез-келген бизнес клиенттердің есебін жүргізуі керек. Шағын бизнес үшін деректерді талдаудың ең үнемді және тиімді нұсқасы-жеткілікті үлкен талдау функциясы бар қарапайым электрондық кестелер. Сондықтан, жазбаларды ыңғайлы көру үшін қызметкерлер мен әкімші SQLlite-ді қосуы керек. Осылайша, қосымшаның өзінен басқа, бухгалтерлік есеп онлайн кестеде жүргізіледі, бұл деректерді талдауды әлдеқайда жеңілдетеді және ұқсас кесте негізінде оларды визуализациялау үшін графиктер құруға болады.

Тұтынушының талаптарын талдау арқылы ең маңызды функциялар анықталды, сонымен қатар клиенттермен жақсы қарым-қатынас жасауға көмектесетін функциялар ескерілді.

1.4 Ұқсас басқару жүйелерін салыстыру

Тұтынушылардың талаптарын анықтау үшін қызмет көрсету ұйымдары, атап айтқанда сұлулық салондары, шаштараздар, қоймалар және т.б. талданды. Олар Instagram аккаунттарын іздеп, Direct арқылы хабарласуы керек, онда хабарламалар жоғалуы мүмкін. Сондай-ақ, жұмыс телефон қоңырауларынан өтеді, бірақ қиындық-қалаған Нөмірді тез табу әрдайым мүмкін емес. Қызметкерлер жиі жағдайларда өздері есеп жүргізуі керек, яғни клиенттерді ноутбукта немесе онлайн кестеде жазған сайын, бұл уақытты қажет етеді және бұл өте ыңғайлы емес.

Ірі ұйымдар өздерінің брендтік қосымшаларына тапсырыс беріп жатыр. Мұндай салондар үлкен танымалдылыққа ие және клиенттердің саны көп, және мұндай қосымшалар бұған айтарлықтай үлес қосты. жұмысшыларға смартфонда веб-қосымшаны ашу әлдеқайда ыңғайлы, жазбаларды белгілі бір күнге қарап, жазуды бастаңыз. Егер сұрақтар туындаса, қосымшаның өзі

арқылы тауарды немесе тапсырысты жасау күні мен беру күнін көруге болады.

Сіз әрбір маман бойынша кез келген күнге тапсырыс тізімін қарай аласыз және кез-келген өнімді сатуға мүмкіндік аласыз. Бағдарлама бөлімшелер мен қоймалардың кез-келген санымен жұмыс істей алады. Барлық филиалдар интернет арқылы бірыңғай базада жұмыс істейтін болады.

Жоғарыда аталған қосымшадан басқа, ұқсас мобильді қызметтерге талдау жасалды және клиенттердің есебін жүргізетін көптеген қосымшалар тек клиенттермен жұмыс істеуге бағытталғанын атап өтуге болады. Яғни, нақты рөлдер жоқ және қолданба ішінде администратор жоқ. Олардың көпшілігінде тек стандартты функционалдылық бар (жазу формасы мен контактілер). Сонымен қатар, мұндай қызметтер авторизация сияқты маңызды құралдардан бас тартады, бұл олар үшін өте маңызды, өйткені мұндай нюанстардың арқасында менеджер деректердің құпиялылығы туралы пікір қалыптастырады.

1.5 Жүйеге қойылатын талаптарды әзірлеу

Мобильді жүйе онлайн режимінде жазу, компаниялардың имиджін және клиенттер санын арттыру үшін жасалады.

Бағдарлама кросс-платформалы болуы керек, яғни Android, IOS арқылы пайдалану мүмкіндігі болуы керек.

Қосымшаның функционалды бөлігі орындалуы керек:

- тақырып пен күнді таңдау мүмкіндігі бар жеке вкладка арқылы клиенттердің тапсырысын жазу;
- әкімшінің қызметкерлер үшін аккаунт құру және бар логин мен парольдің көмегімен деректердің құпиялылығын қамтамасыз ету үшін жүйеге кіру мүмкіндігі;
- жазба тарихын бөлек қойындыда көру мүмкіндігі;
- әкімші мен қызметкерлерге клиенттердің барлық жазбаларын бөлек қойындыда қарау мүмкіндігі;
- әкімшінің жүйенің тиімді жұмыс істеуі үшін пайдаланушыларға рөлдер беру мүмкіндігі;
- администратордың барлық процестерді толық бақылау мүмкіндігі.

Жүйе өзінің функционалы бар пайдаланушылардың 3 түрін көздеуі тиіс:

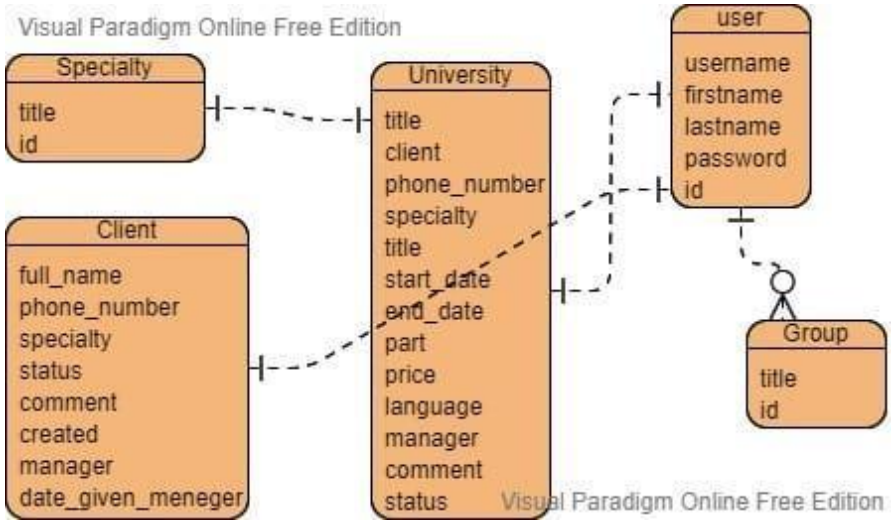
1. Менеджер. Менеджердің құқықтары тапсырыс қосуға және өз клиенттерінің барлық жазбаларын көруге мүмкіндік береді;
2. Администратор. Администратор үшін қосымша бет қарастырылуы керек, онда ол пайдаланушыларға рөлдер бере алады. Бастапқыда қосымшада қазірдің өзінде құрылған админ аккаунты бар;
3. Бас менеджер. Бас менеджер тапсырыптарды бақылап, орындалуын бақылайды. бас менеджер рөлін администратор ашып береді.

Пайдаланушы деректерін және тұтынушы жазбаларын сақтау үшін СУБД пайдалану керек.

2 Бағдарлама моделін жобалау және оны жасау үшін құралдарды таңдау

2.1 Мобильді қосымшаның моделін құру

Болашақта әрбір сәтті жоба модель құруды талап етеді. Олар бизнес қажеттіліктері негізінде құрылады. Бастапқыда әлеуетті клиенттердің сұранысы негізінде қосымшаның қажетті функционалдығын зерттеу қажет. Ұқсас қосымшаларды зерттеп, жағымсыз жақтарын түзетіңіз. Қазірдің өзінде зерттелген мәліметтер негізінде модель құрыңыз. Дұрыс жобаланған сәулет қызмет жұмыс істеп тұрған кезде қандай процестер жүретінін анықтауға мүмкіндік береді. Осылайша, даму сатысында да жүйенің осал тұстарын анықтап, олардан арылуға болады. Талаптарды зерттегеннен кейін ER диаграммасы жасалды 2.1 суретте көрсетілген.



2.1-сурет – “ER диаграмма”

Контекстік диаграмма 2.2 суретте көрсетілген.

2.2 –сурет – “Контекстік диаграмма”

2.2 Әзірлеу үшін құралдарды таңдау

Мобильді қосымшаны жазудың негізгі шарттарының бірі-оның кросс-платформасы. Әрбір қызметкер мобильді құрылғының операциялық жүйесіне қарамастан смартфон арқылы веб-қосымшаны аша алуы керек, сондықтан оны жасау үшін платформа мәтіндік редакторы таңдалды. Ол Android, IOS-та жұмыс істейтін бірыңғай кодты жазуға мүмкіндік береді[1].Осылайша, әртүрлі платформаларда пайдаланушы интерфейсінің орналасуын, кодын және логикасын бөлісу мүмкіндігі бар.

Джанго-бұл Python-да әртүрлі күрделіліктегі веб-қосымшаларды құруға арналған Ашық бастапқы негіз. Оның басты артықшылықтарының бірі-сіз болашақ веб-қосымшаның логикасына ғана назар аударуыңыз керек, қалғанын Django жасайды.

Проектте Android Studio даму ортасы ретінде қолданылады, ол көптеген параметрлер, қолдау көрсетілетін технологиялар, жылдамдық пен ыңғайлылықтың арқасында мобильді қосымшаны дамытудың ең жақсы ортасының бірі болып саналады.

Мәліметтер базасы ретінде SQLite СУБД алынды [9]. Бұл дерекқордың басты артықшылығы-бұл файл және барлық деректер бір файлда сақталады. Сондай-ақ, бұл база әзірлеу және тестілеу үшін өте қолайлы, өйткені ол өте үлкен функционалдылыққа ие. Осы деректер базасымен жұмыс істеу үшін кітапхананы орнату керек SQLlite.NET [9]. Бұл кітапхана SQL тілін пайдаланбай стандартты Python класстарының объектілері ретінде деректермен жұмыс істеуге мүмкіндік береді.

Қосымшаны тестілеу үшін Android-де Android Studio эмуляторы қолданылды, оны орнату өте оңай. Бұл эмуляторда жабдықтың үлкен таңдауы бар, сондықтан сынақтарды әртүрлі ажыратымдылықтар мен техникалық

сипаттамалары бар құрылғыларда жүргізуге болады. Акселерометр, экран бағдары, SD карталары, батареялар, GPS сияқты бірқатар сенсорлар физикалық құрылғы арқылы төсеу кезінде жиі қолданылатын сынақтарға ресурстарды үнемдейді.

Біз алғашқы қосымшамызды жасап, серверді іске қосқан кезде ,сіз " db.sqlite3 " Сіздің жобаңыздың каталогында. Файл-бұл сіз жасайтын барлық деректер сақталатын мәліметтер базасының файлы. Django серверлік орта болғандықтан, серверді пәрмен жолынан немесе терминалдан іске қосқан кезде ол компьютерді хост ретінде қарастырады.

Деректер базасы - бұл Django Framework-тегі алдын-ала анықталған сөздік, онда барлық деректер индекс ретінде сақталады.

Белгілі бір веб-сайтқа қосылған кезде ол пайдаланылған кітапхананы көрсетеді. Біз "django" файлын мәнге қосуымыз керек.db.backends.sqlite3 " - бұл Python кодын дерекқор тіліне түрлендіретін sqlite3 дерекқорына арналған Python кітапханасы.

SQLite-бұл реляциялық дерекқорды басқару жүйесін (RDBMS) қамтамасыз ететін ақысыз, ашық бастапқы бағдарламалық жасақтама. РСУБД-бұл үлкен кестелердегі пайдаланушы анықтаған жазбаларды қамтитын мәліметтер базасын басқару жүйесі. Дерекқор механизмі деректерді сақтау және басқару жүйесінде есептер мен мәліметтер жиынтығын құру үшін көптеген кестелердегі деректерді біріктіретін күрделі сұрау командаларын орындай алады.

3 Мобильді қосымшаны іске асыру

3.1 Қосымшаның графикалық интерфейсін жасау

Графикалық пайдаланушы интерфейсі (GUI) – бұл барлық басқару және навигация элементтерін қамтитын пайдаланушының жүйемен өзара әрекеттесуіне арналған құралдар жүйесі. Сапалы жобаланған GUI қосымша клиенттерді тартуға және қосымшаларды пайдалануды жеңілдетуге мүмкіндік береді, осылайша белгілі бір қызметке сұранысты арттырады.
Қосымшаны ашқан кезде бірінші бет авторизация беті ашылады. Веб қосымша болғандықтан бетті Html және CSS тілдері қолданылған.3.1 - 3.2 суреттерде көрсетілген.

```
{% load static %}
<!DOCTYPE html>
<html>
<head>
  <title></title>
    <link href="{% static 'css/bootstrap.min.css' %}" rel="stylesheet">
    <link href="{% static 'css/style.css' %}" rel="stylesheet">
</head>
<body>

{% block content %}
{% endblock %}

  <script src="{% static 'js/jquery-3.5.0.min.js' %}"></script>
  <script src="{% static 'js/popper.min.js' %}"></script>
  <script src="{% static 'js/bootstrap.min.js' %}"></script>
  <script src="{% static 'js/admin.js' %}"></script>

</body>
</html>
```

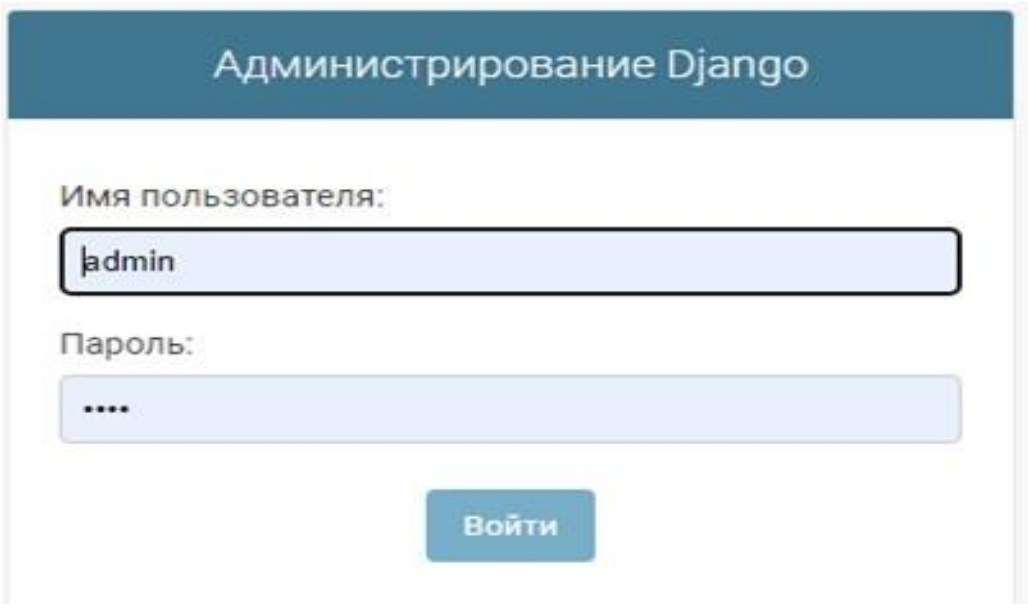
3.1 –сурет – “Base.html”

```
{% extends 'registration/base.html' %}
{% load static %}
{% load widget_tweaks %}
{% block content %}
iv class="main-wrapper">
<div class="login-page">
<div class="login-body container">
<div class="loginbox">
<div class="login-right-wrap">

<div class="login-header">
<p class="text-muted">Доступ к нашей панели управления</p>
</div>
<form class="form-horizontal form-material" action="{% url 'login' %}" method="post">
{% csrf_token %}
<div class="form-group">
<label class="control-label">Имя пользователя</label>
{{ form.username|add_class:"form-control" }}
</div>
<div class="form-group mb-4">
<label class="control-label">Пароль</label>
{{ form.password|add_class:"form-control"}}
</div>
<div class="text-center">
<button class="btn btn-primary btn-block account-btn" type="submit">Авторизоваться</button>
</div>
</form>
</div>
</div>
</div>
</div>
</div>

{% endblock %}
```

3.2-сурет – “Login.html”

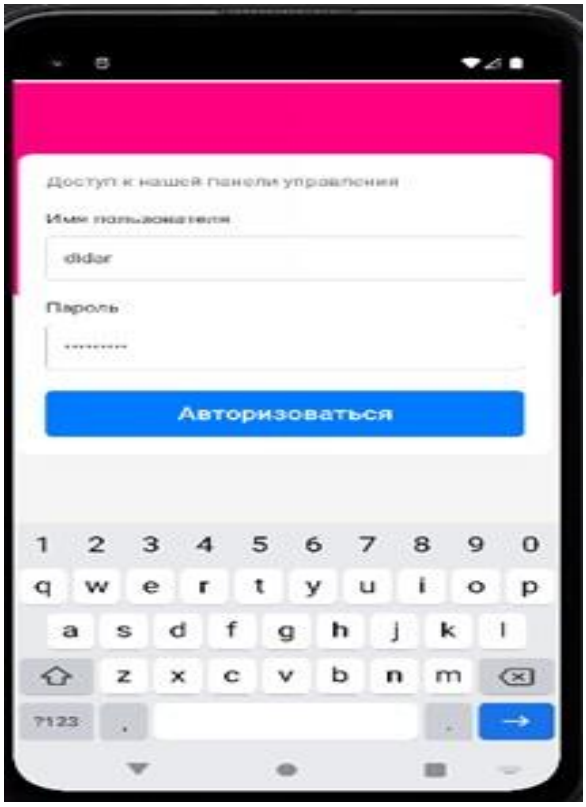
Қосымшаны 2 жолмен қолдануға болады.

1. Администратор мен бас менеджер администрация бетіне компьютермен өз аккаунтымен кіре алады 3.3 суретте көрсетілген.



3.3–сурет – “Администратор бойынша жүйеге кіру”

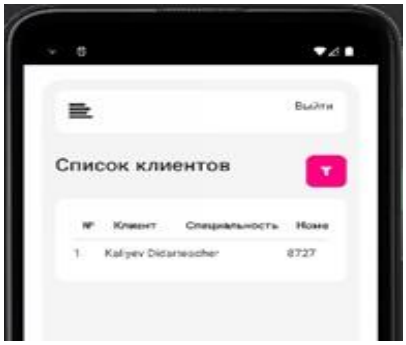
2. Менеджер тек функционалдығы шектеулі мобильді қосымшасымен қолданады 3.4 суретте көрсетілген.



3.4-сурет – “Менеджер ретінде мобильді қосымшаға тіркелу”

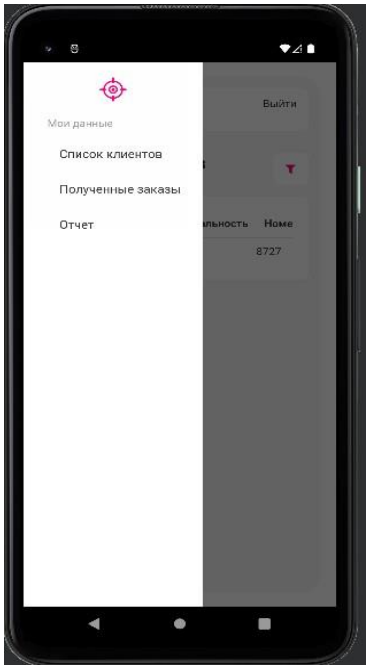
3.2 Мобильді қосымшасы арқылы жүйеге кіру

Сәтті авторизациядан өткен қолданушы үшін Clients басты беті ашылады сурет 3.5.

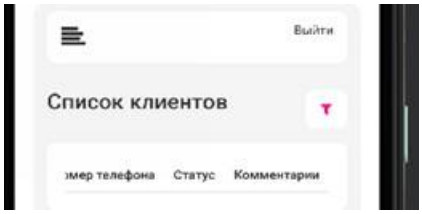


3.5-сурет – “Клиенттер тізімі”

Сол жағында навигация жолағы бар - бұл қолданбаның негізгі навигациялық мәзірін көрсететін пайдаланушы интерфейсі тақтасы. Пайдаланушы қолданбалар тақтасындағы тартпаның белгішесін түрткен кезде немесе пайдаланушы экранның сол жақ жиегінен сырғыған кезде пайда болады сурет 3.6, ал клиенттер тізімінің қосымша информациясын, солға қарай жүргізгенде көре алады сурет 3.7.

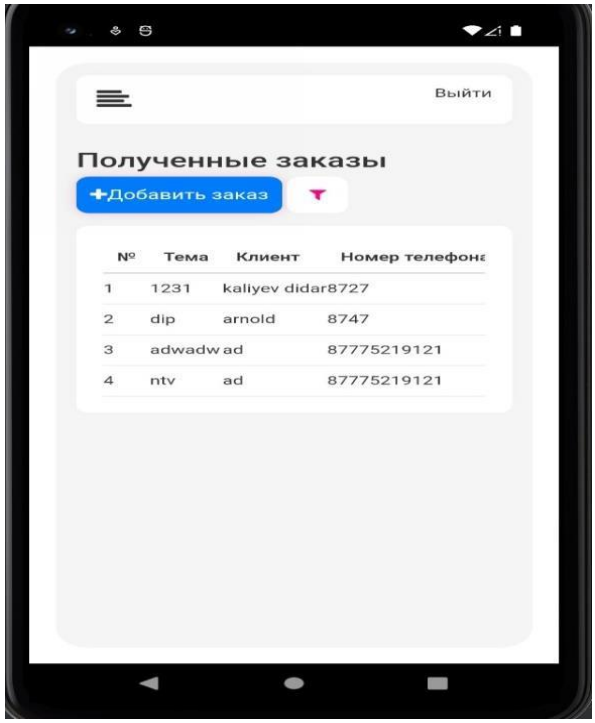


3.6-сурет – “Навигация”

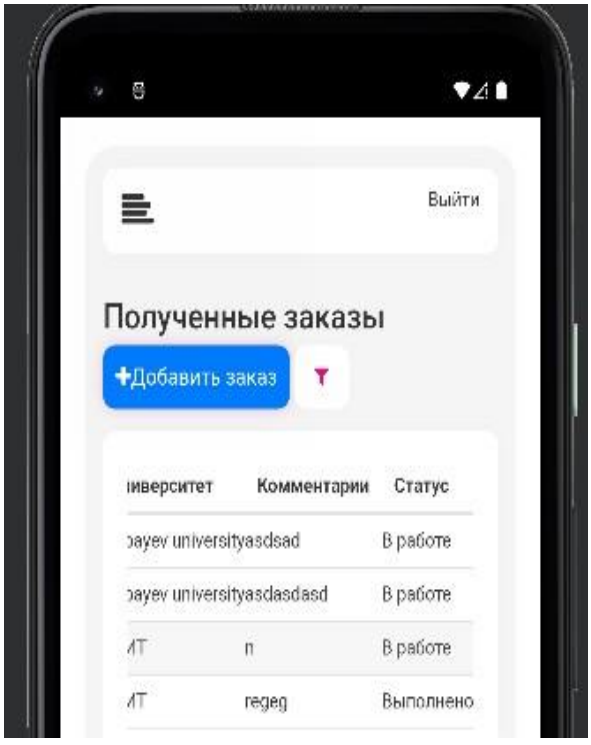


3.7–сурет – “Клиенттер тізімінің қосымша ақпараты”

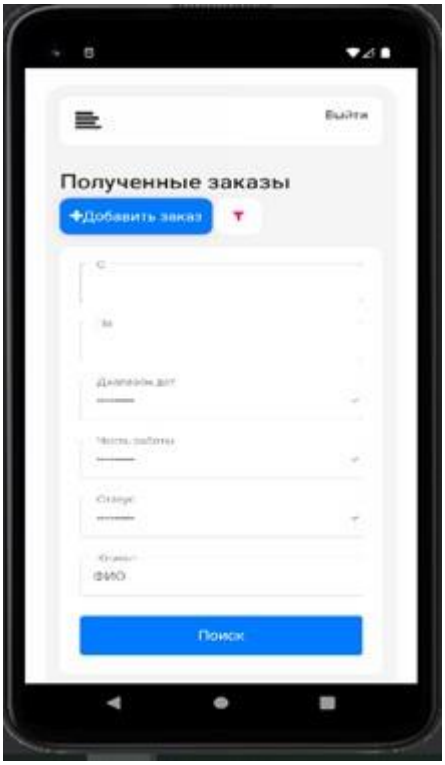
Тапсырыс алу бетінде, тапсырыс тізімі сурет 3.8, сурет 3.9, тапсырысты ыңғайлы әрі тез табу үшін фильтр қосылған сурет 3.10 және тапсырыс қосу беті 3.11 - 3.12 суреттер.



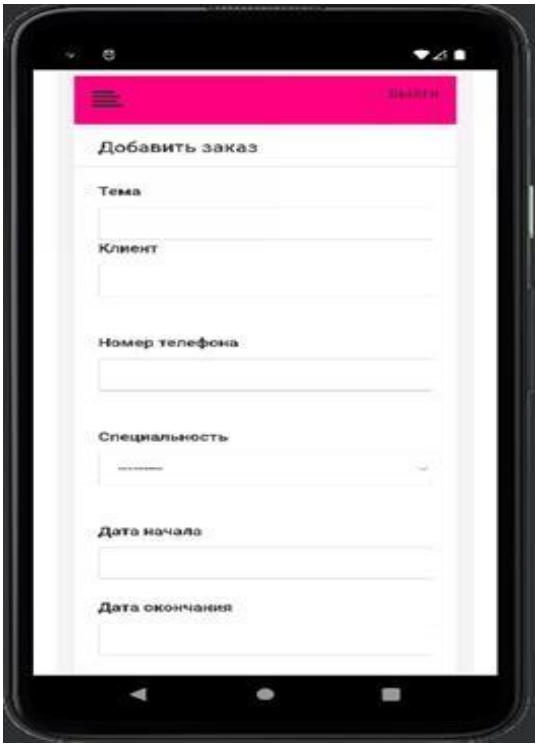
3.8-сурет – “Тапсырыс тізімі”



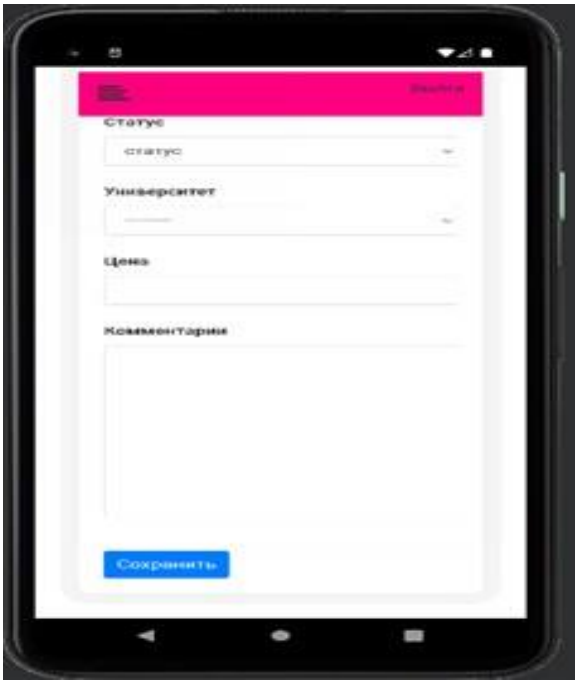
3.9-сурет – “Тапсырыс тізімі (жалғасы)”



3.10-сурет – “Клиенттер тізімінің фильтрі”

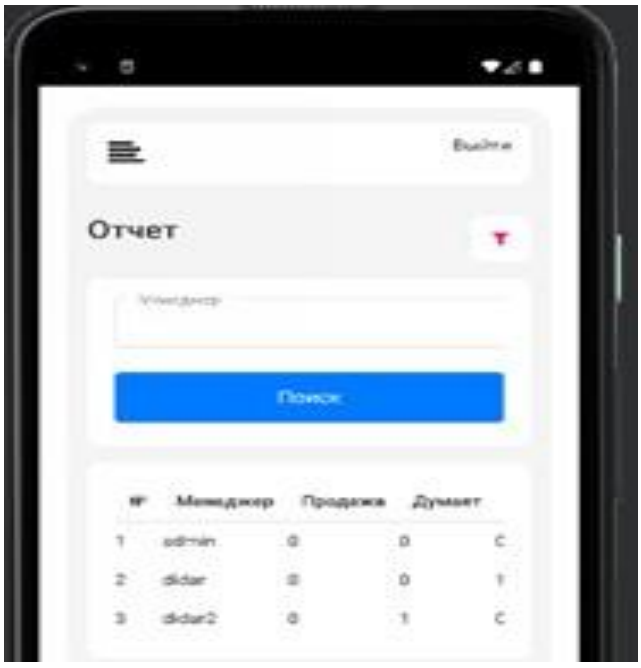


3.11-сурет – “Тапсырыс қосу беті”



3.12-сурет – “Тапсырыс қосу беті (жалғасы)”

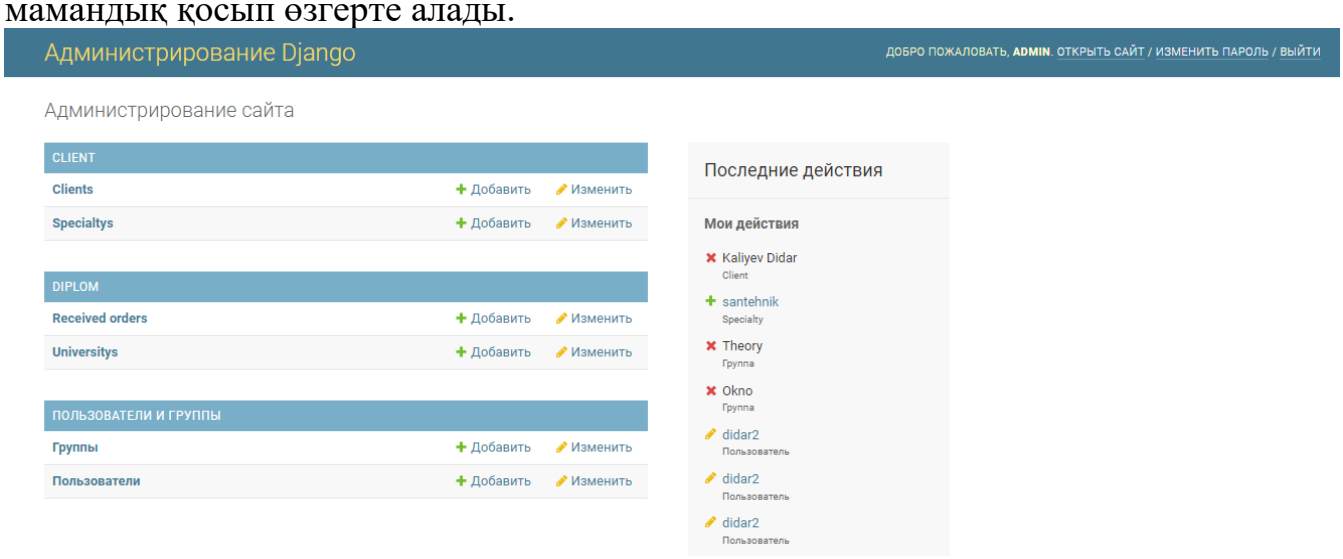
Менеджер өзінің жұмысын бағалау үшін және басқалармен салыстыру үшін есеп көрсету бетінде, менеджер бойынша фильтрлеу бар сурет 3.13.



3.13-сурет – “Есеп көрсету беті”

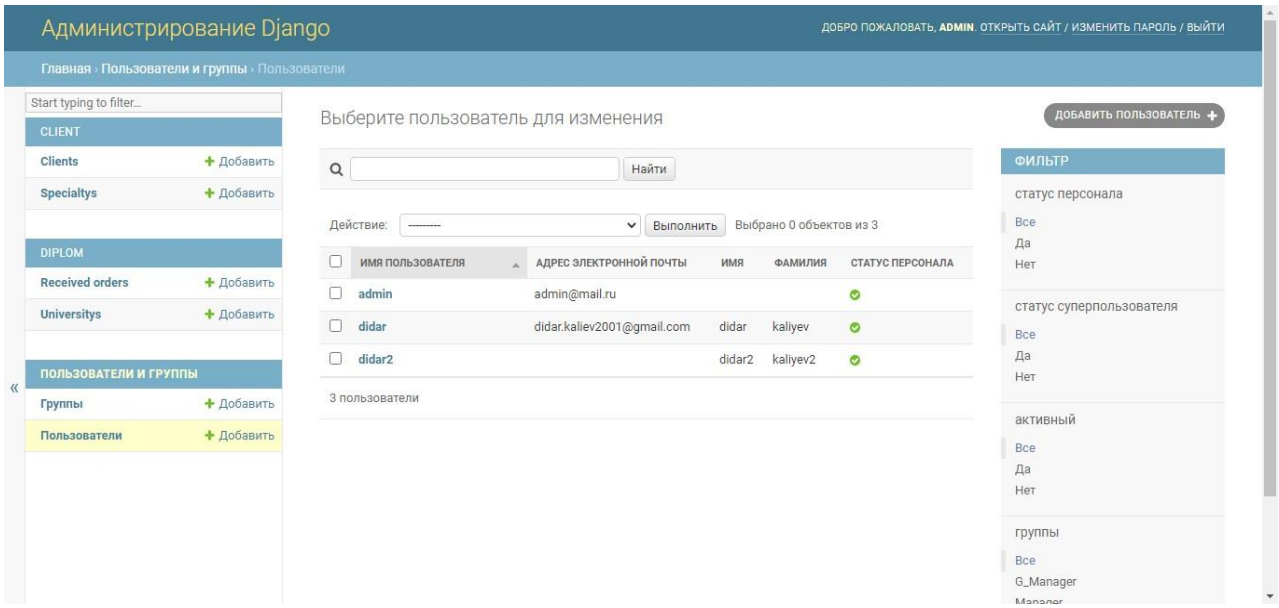
3.3 Администраторлық интерфейсін жасау

Администратордың функционалдығы көп болғаннан кейін өзіне ыңғайлы интерфейсі болу керек сурет 3.14. Администратор: менеджерлерді қосады, алады және рөлдер бере алады, тапсырыстарды, клиенттерді, топтарды және мамандық қосып өзгерте алады.



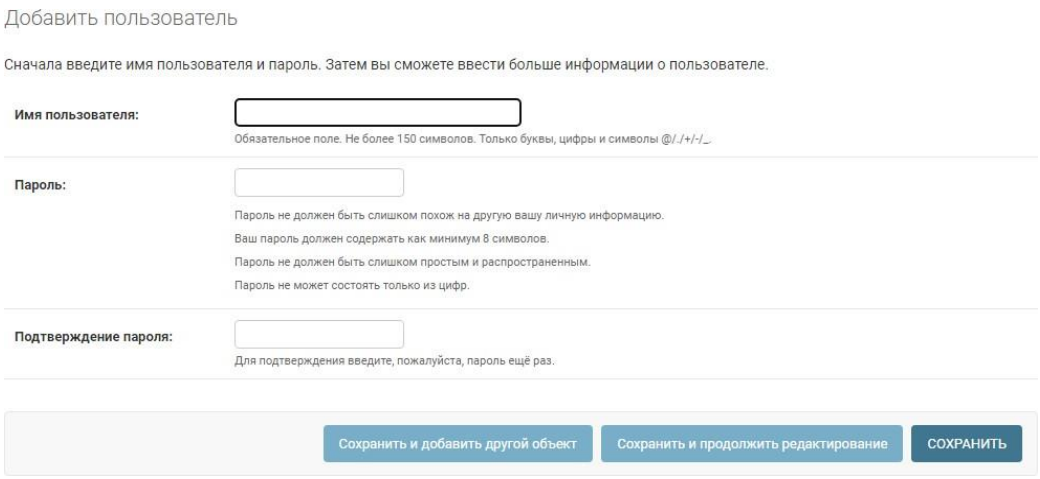
3.14-сурет – “Администратордың басты беті”

«Пользователи» вкладкасының интерфейсі сурет - 3.15 және менеджер бойынша фильтрлеу қосылған.

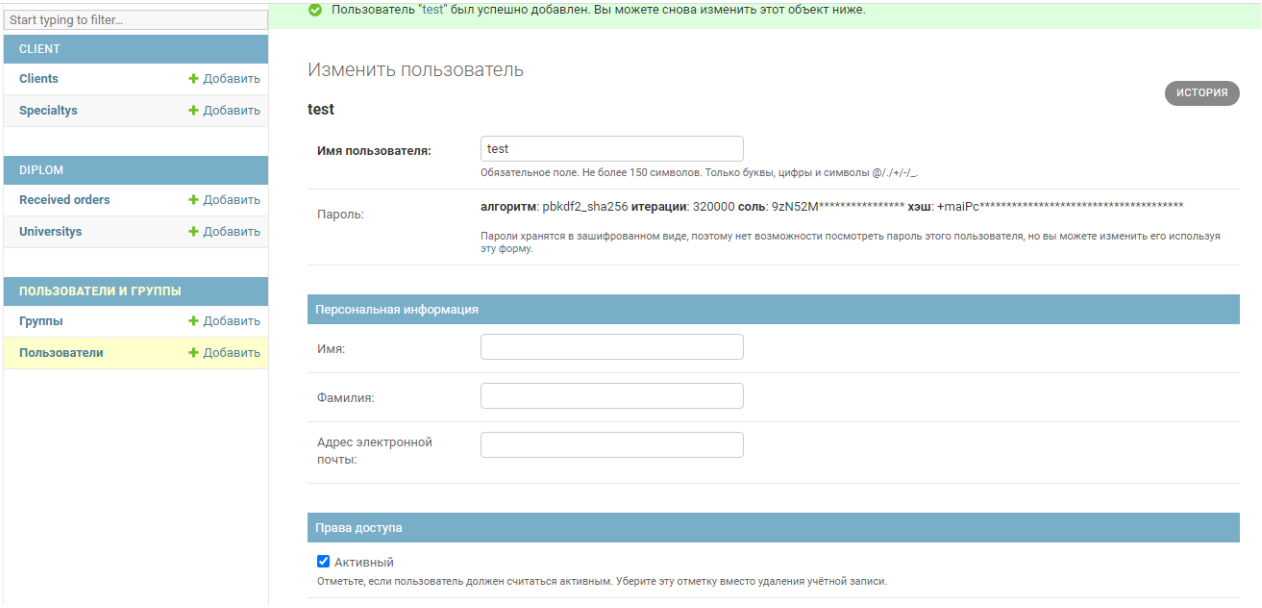


3.15-сурет – “Пользователи вкладкасының интерфейсі”

Менеджер қосу үшін: «Добавить пользователя» сурет 3.16 -> «имя пользователя» және «пароль» еңгізеді -> бос жерлерін толтырады 3.17 сурет.

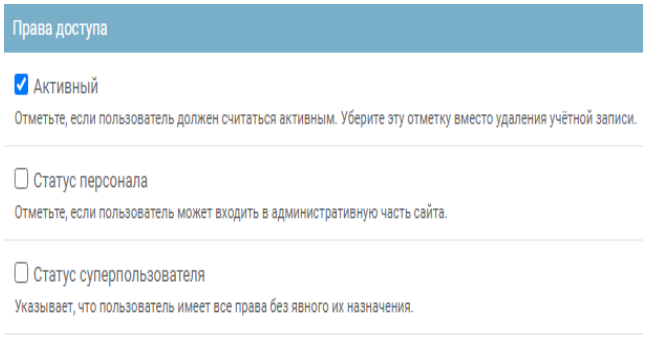


3.16-сурет – “Қолданушыны қосу беті”

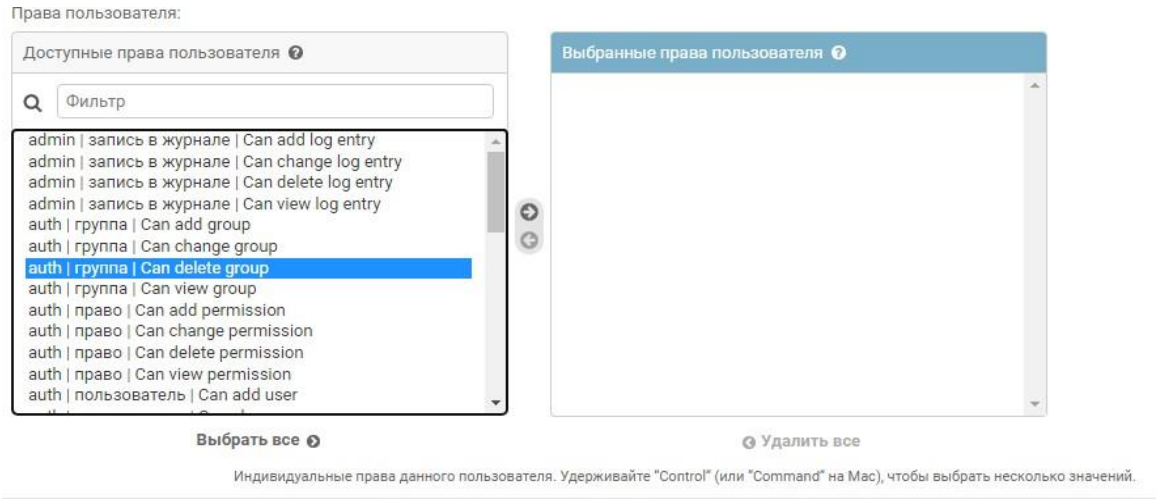


3.17-сурет – “Қолданушыны қосу беті (жалғасы)”

Бас менеджерді рөл беру кезінде тағайындайды 3.18 - 3.19 суреттер.

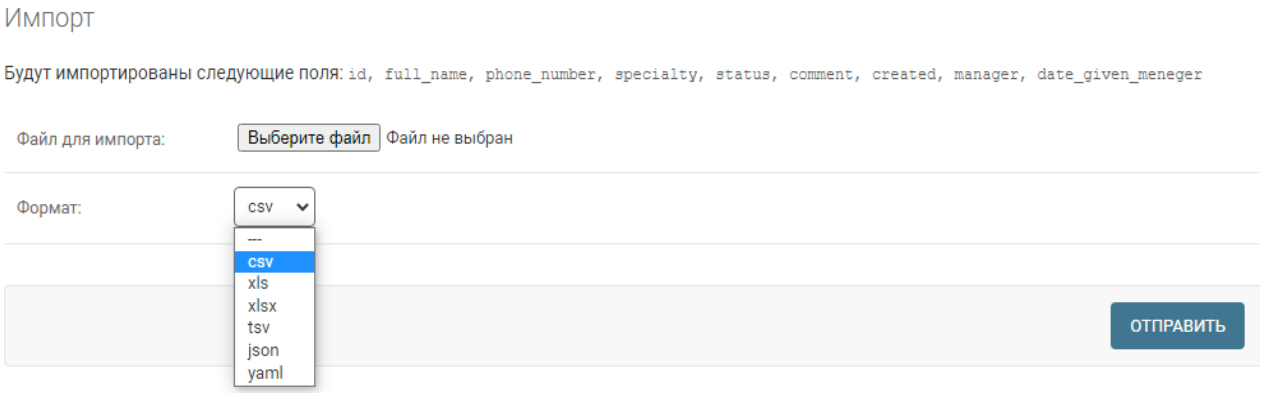


3.18-сурет – “Қолданушылардың рөлдердін беру”



3.19-сурет - “Қолданушылардың рөлдердін беру (жалғасы)"

Клиенттерді қосқан кезде әр-қайсысын теріп отырмас үшін, клиенттерді импорттау және экспорттау еңгізілді сурет 3.20.



3.20-сурет – “Клиенттерді импорттау және экспорттау”

Қорытынды

Дипломдық жұмыс аясында қызмет көрсету сервистеріне клиенттерді онлайн жазуға арналған мобильді Веб-қосымша іске асырылды. Бұл жобада келесі міндеттер шешілді: пәндік аймақ талданды, тұтынушының талаптарын талдау және ұқсас жүйелерді талдау арқылы жүйеге қойылатын негізгі талаптар анықталды, техникалық тапсырма жасалды, қызметтің ерекше моделі және ерекше пайдаланушы интерфейсі жасалды, Andriod studio платформасы зерттелді және оның негізінде клиенттерді онлайн жазуға арналған платформалық қосымша жасалды.

Іске асыру үшін Python және Django бағдарламалау тілдері қолданылды. Деректер базасы ретінде SQLite СУБД қолданылды, онда клиенттердің барлық деректері мен олардың жазбалары сақталады. Даму ортасы ретінде Android Studio қолданылды.

Бұл қосымшада түсінікті пайдаланушы интерфейсі бар ерекше модель мен дизайн бар. Жүйе клиенттердің онлайн-жазбасын жүргізуге мүмкіндік береді, бұл қызметтің жұмысын айтарлықтай жеңілдетеді және кез-келген ұйымның қызметін автоматтандыруға мүмкіндік береді. Пайдаланушылар үшін рөлдер енгізілді, бұл қосымшаны тек менеджерлерге ғана емес, сонымен қатар бас менеджерлерге де бағыттайды. Деректердің құпиялылығы үшін әр пайдаланушының жеке есептік жазбасын жасау мүмкіндігі бар.

Айта кету керек, бұл қосымша белгілі бір модификацияларда тек онлайн жазу үшін ғана қолданыла алмайды клиенттердің есебі жүргізілетін және ұқсас моделі бар кез келген басқа саладағы ұйым. Мысалы, маркетинг саласында адамдарға қоңырау шалуға және шарт немесе өнімді ұсыну.

Бітіруші біліктілік жұмысын орындау кезінде 1-кестеден мүмкіндіктер игерілді.

1-кесте

Мүмкіндік атауы	Қолданылған жері
Экономикалық білім негіздері мірдің әр түрлі салаларында қолдан білу.	Таңдалған тақырып еңбек нарығы тұрғысынан өзекті. бет. 3, абз. 1-2.
Тіршілік әрекетінің әртүрл і арында құқықтық білі егіздерін пайдалану қабілеті.	Қызметтің әртүрлі салаларындағы қатынастарды реттейтін нормативті актілерді талдауға негізделген білім айдаланылды
Әлеуметтік, этникалық, конфессиялы қ және мәдени айырмашылықтард қабылдай отырып, ұжымда жұмы істеу қабілеті.	Жұмыс барысында ғылыми жетекшіге т лық төзімділік көрсетілді.

1-кестенің жалғасы

Өзін-өзі ұйымдастыру және өзін-өзі тәрбиелеу қабілеті.	Android технологиялары әдебиеттер тізімінен қосымша әдебиеттер көмегімен дербес зерттелді.
Заманауи математикалық әдістер мен заманауи қолданбалы бағдарламалық құралдарды қолдану және заманауи бағдарламалау технологиялары меңгеру.	Платформалық мобильді дамудың заманауи технологиялары қолданылды. 2.2 Тарау.
Компьютерде практикалық есептерді шешу үшін қолданбалы бағдарламалардың стандарт пакеттерін қолдану, күйін келтіру қолданбалы бағдарламалық жасақтаманы тестілеу мүмкіндігі.	Жұмыста Android Studio кітапханалары, модульдері және функционалдығы дұрыс қолданылады. 2.2 Тарау.
Есептеу техникасын баптау және тестілеу және тексеруді жүзеге асыру қабілеті мен дайындығы.	Мобильді қосымша сауатты жобаланған және әзірленген, бұл ретте әзірлеу үшін оңтайлы құралдар таңдалған. 2 Тарау.
Қазіргі заманғы бағдарламалау тілдерін, операциялық жүйелерді офистік қосымшаларды, Интернеттің деректерді басқару тәсілдері мен тетіктерін; операциялық жүйелерді ұйымдастыру қағидаттарын, жұмыс құрамы мен схемасын білу қабілеті және дайындығы.	Заманауи Python бағдарламалау тілі және платформалық қосымшаны жасау үшін платформа қолданылды. 2.2 Тарау.
Ақпаратты басқару білімдері мен дағдыларын қолдануға дайындық	Ақпаратты басқарудың барлық процестері сауатты жүргізілді: жинау, беру, түрлендіру, өңдеу, қолдану

Қолданылған әдебиеттер тізімі

1 Новостной сервис strategy2050 [Электрондық нұсқасы]: https://strategy2050.kz/ru/news/51754.
2 Документация по SQLite [Электрондық нұсқасы]: https://docs.microsoft.com/ru-ru/django/android/data-cloud/data-access/using- sqlite-orm.
3 Создаём первое веб-приложение с Джанго [Электрондық нұсқасы]: https://tproger.ru/translations/create-your-first-django-app/
4 Веб-фреймворк Django (Python). [Электрондық нұсқасы]: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django
5 Почему Django лучший фреймворк для разработки сайтов. [Электрондық нұсқасы]: https://ru.hexlet.io/blog/posts/pochemu-django-luchshiy-freymvork-dlya-razrabotki-saytov
6 Python Django + SqlLite | Rest Api's [Электрондық нұсқасы]: https://www.youtube.com/watch?v=ntGMU4BVMtY
7 Django Database: How to Connect SQLite Database with Django Project [Электрондық нұсқасы]: https://techvidvan.com/tutorials/django-database-connectivity/
8 Документация Django 1.9 [Электрондық нұсқасы]: https://djbook.ru/rel1.9/intro/tutorial02.html
9 Синтаксис SQL Lite [Электрондық нұсқасы]: https://www.sqlite.org/index.html
10 SQL Lite [Электрондық нұсқасы]: https://ru.wikipedia.org/wiki/SQLite
11 Ендірілген SQL Lite [Электрондық нұсқасы]: https://habr.com/ru/post/149356/
12 SQL Lite кіріспе [Электрондық нұсқасы]: https://metanit.com/sql/sqlite/1.1.php
13 Веб-фреймворк Django [Электрондық нұсқасы]: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django
14 Django кіріспе [Электрондық нұсқасы]: https://tutorial.djangogirls.org/ru/django/
15 Django фреймворк [Электрондық нұсқасы]: https://web-creator.ru/articles/django
16 Django кітапханалары[Электрондық нұсқасы]: https://django.fun/
17 Django кіріспе [Электрондық нұсқасы]: https://metanit.com/python/django/
18 Django қалай жұмыс істейді [Электрондық нұсқасы]: https://blog.skillfactory.ru/glossary/django/
19 Моделді құру және дерекқорды тасымалдау [Электрондық нұсқасы]: https://metanit.com/python/django/5.1.php
20 Django [Электрондық нұсқасы] : https://django.fun/docs/django/ru/3.2/topics/db/sql/

А Қосымшасы

“Клиенттердің жазбаларын есепке алу үшін мобильді қосымшаны әзірлеу” тақырыбында веб қосымшаны құруға арналған техникалық тапсырма

А.1 Кіріспе

Дипломдық жұмыстың негізгі функциясы – Клиенттердің жазбаларын сақтап оларды ыңғайлы қолдану.

Заманауи технологияларға сай веб қосымшаны “тіркеу жүйесі”жасалған. Сондай-ақ пайдаланушыларға ыңғайлы интерфейсті қарастырылды.

Бағдарлама Python, Django, Html, CSS және Javascript тілдерінде жазылды.

А.1.1 Өңдеудің мақсаты мен өзектілігі

Дипломдық жобаның мақсаты – клиенттерді тіркеуге арналған мобильді веб қосымшасын құру.

Дипломдық жобаның өзектілігі:

- Қолдануға ынғайлылық.
- Әрбір компанияға келтіріп жасауға болады.

А.1.2 Қолдану саласы

Тез әрі ыңғайлы клиенттерді қосу, алу, нәтижие қою, пайданушылардың уақытын үнемдеуге көмектеседі.

А қосымшасының жалғасы

А.1.3 Анықтамалар, терминдер және қысқартулар

Терминдер немесе қысқартулар Анықта	мал ар
UML	Unified Modeling Language
Интерфейс	Жүйенің элементтері арасындағы әрекеттесетін құралдар, әдістер мен ережелердің жиынтығы.
Браузер	Ақпаратты компьютерлік желіден іздеуге немесе қарауға арналған бағдарлама.
SQLite	Ықшам енгізілген СУБД
СУБД	Деректер базасын басқару жүйесі (Система управления базами данных)

А.2 Мәтіндік ақпараттың көлемі мен құрамы

Дипломдық жоба 30 беттен, 3 бөлімнен тұрады.

А.2.1 Мәтіндік және графикалық ақпараттың электрондық түрдегі көлемі мен құрамы

Дипломдық жобаның көлемі 18.8 МБ тұрады

А.2.1.1 Аппаратттық интерфейстер

1. Ғаламтор желісі болу керек;
2. Браузер болу керек;
3. Компьютер;
4. Телефон Android немесе IOS болу керек;
5. Пернетақта, монитор.

А қосымшасының жалғасы

А.2.2 Мобильді қосымшаның аудиториясы

Ғаламтор желісіне қосылған ноутбуктар компьютерлер және менеджерлер.

А.2.3 Мобильді қосымшасының беттерінің саны

Веб қосымша келесі парақщалардан тұрады: клиенттер тізімі, тапсырыс тізімі, авторизация және есеп беті.

Ал администраторлық бетте: авторизация, негізгі бет, клиенттер тізімін қосу және алу, группа қосу және алу, тапсырып қосу және өзгерту

А.3 Мобильді қосымшаны құру мерзімі

Клиенттердің жазбаларын есепке алу үшін мобильді қосымшаны әзірлеу мерзімі - 2021-2022.

Б қосымшасы

Қосымшаның функционалды бөлігін жасау

Мобильді қосымшасы

MainActivity.kt

```
package com.example.myapplication
import android.annotation.TargetApi
import android.os.Build
import android.os.Bundle
import android.webkit.WebResourceRequest
import android.webkit.WebView
import android.webkit.WebViewClient
import androidx.appcompat.app.AppCompatActivity

private class MyWebViewClient : WebViewClient() {
    @TargetApi(Build.VERSION_CODES.N)
    override fun shouldOverrideUrlLoading(view: WebView, request:
WebResourceRequest): Boolean {
        view.loadUrl(request.url.toString())
        return true
    }

    // Для старых устройств
    override fun shouldOverrideUrlLoading(view: WebView, url: String): Boolean {
        view.loadUrl(url)
        return true
    }
}

class MainActivity : AppCompatActivity() {
    private lateinit var webView: WebView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        webView = findViewById(R.id.webView)
        webView.webViewClient = MyWebViewClient()
        // включаем поддержку JavaScript
        webView.getSettings().setJavaScriptEnabled(true)
        // указываем страницу загрузки
        webView.loadUrl("http://172.16.0.102:8000/")
    }

    override fun onBackPressed() {
        if (webView.canGoBack()) {
            webView.goBack()        } else {super.onBackPressed()        }    }
}
```

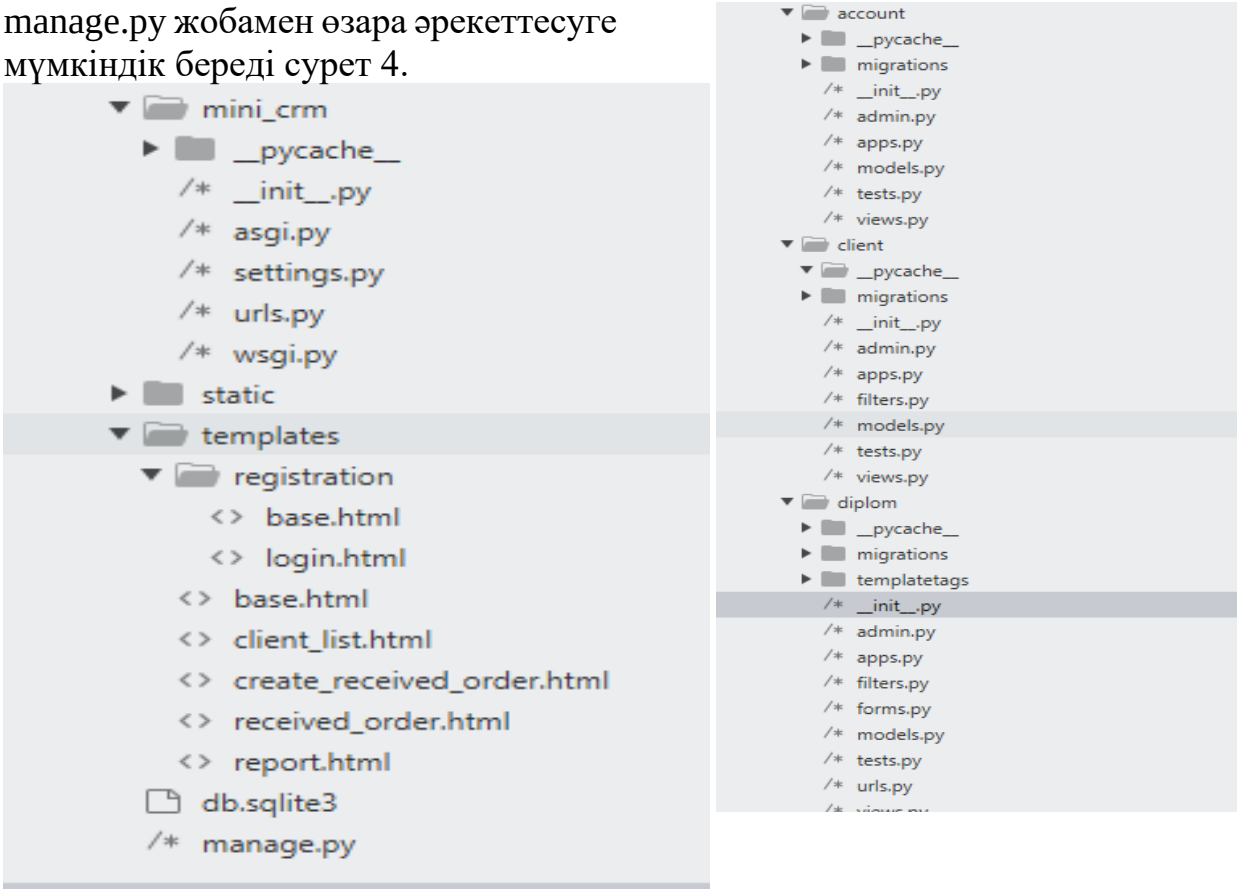
Б қосымшасының жалғасы

Activity_main.xml - разметкасы - webView арқылы жасалып тұр

```
<?xml  version="1.0"  encoding="utf-8"?>
<WebView
    android:id="@+id/webView"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

Программалық код

__init__.py -Python-ға бұл каталогты пакет ретінде қабылдау керек деп айтатын бос файл.
settings.py - біздің жобаның конфигурациясын қамтиды.
urls.py -мұнда URL мекен-жайы жарияланады.
wsgi.py -оның көмегімен қосымша WSGI протоколы бойынша веб-сервермен жұмыс істей алады.
manage.py жобамен өзара әрекеттесуге мүмкіндік береді сурет 4.



4 - сурет – “программалық кодтың папкалары”

Б қосымшасының жалғасы

manage.py

```
import os
import sys
def main():
 """Run administrative tasks."""
   os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'mini_crm.settings')
   try:
      from django.core.management import execute_from_command_line
   except ImportError as exc:
      raise ImportError(
         "Couldn't import Django. Are you sure it's installed and "
         "available on your PYTHONPATH environment variable? Did you "
         "forget to activate a virtual environment?"
      ) from exc
   execute_from_command_line(sys.argv)

if __name__ == '__main__':
   main()
```

urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.contrib.auth import views as auth_views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('diplom/', include('diplom.urls')),
    path('', auth_views.LoginView.as_view(),
         name='login'),
    path('logout', auth_views.LogoutView.as_view(),
         name='logout'),
]
```

wsgi.py

```
import os
from django.core.wsgi import get_wsgi_application
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'mini_crm.settings')
application = get_wsgi_application()
```

Б қосымшасының жалғасы

settings.py

```
import os
from pathlib import Path
BASE_DIR = Path(__file__).resolve().parent.parent
SETTINGS_PATH = os.path.dirname(os.path.dirname(__file__))
STATICFILES_DIRS = [
   os.path.join(BASE_DIR, "static"),
]

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'h%7omsmnia3#2kkj4#7qi+f^oa4ykkc$@yhg+6gfgqj%p*q(bz'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']


# Application definition

INSTALLED_APPS = [
'django.contrib.admin',
   'django.contrib.auth',
   'django.contrib.contenttypes',
   'django.contrib.sessions',
   'django.contrib.messages',
   'django.contrib.staticfiles',
   'account',
   'client',
   'diplom',
   'django_filters',
   'widget_tweaks',
   'import_export',
]

MIDDLEWARE = [
   'django.middleware.security.SecurityMiddleware',
   'django.contrib.sessions.middleware.SessionMiddleware',
   'django.middleware.common.CommonMiddleware',
   'django.middleware.csrf.CsrfViewMiddleware',
```

Б қосымшасының жалғасы

```
'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'mini_crm.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(SETTINGS_PATH, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'mini_crm.wsgi.application'
# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
```

```
  'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
{
        'NAME':
 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

# Internationalization
# https://docs.djangoproject.com/en/3.1/topics/i18n/
LANGUAGE_CODE = 'ru-ru'
TIME_ZONE = 'Asia/Almaty'
USE_I18N = True
USE_L10N = True

USE_TZ = True

# Static files (CSS, JavaScript, Images)
#  https://docs.djangoproject.com/en/3.1/howto/static-files/

STATIC_URL = '/static/'
LOGIN_URL = 'login'
LOGIN_REDIRECT_URL = 'diplom/received/order/list'
```

asgi.py

```
import os
from django.core.asgi import get_asgi_application
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'mini_crm.settings')
application = get_asgi_application()
```

filters.py - Клиенттер тізімін фильтлеу

```
import django_filters
from django_filters import DateRangeFilter,DateFilter
from .models import Client

class ClientFilter(django_filters.FilterSet):
    STATUS_CHOICES = (
```

```
 (1, "Думает"),
    (2, "Продажа"),
    (3, "Отказ"),
    (4, "Статус"),
    )
    status = django_filters.ChoiceFilter(label='Часть',
choices=STATUS_CHOICES)
    start_date = DateFilter(field_name='date_given_meneger',lookup_expr=('gt'),)
    end_date = DateFilter(field_name='date_given_meneger',lookup_expr=('lt'))
date_range = DateRangeFilter(field_name='date_given_meneger')
    full_name = django_filters.CharFilter(lookup_expr='icontains')

    class Meta:
        model = Client
        fields = ['full_name', 'specialty',]
```

Models.py - клиенттер тізімінің моделі

```
  from django.db import models
  from django.contrib.auth.models import User
  from datetime import date
  class Specialty(models.Model):
      title = models.CharField(max_length=190)

      def __str__(self):
          return self.title
  class Client(models.Model):
      STATUS_CHOICES = (
      (1, "Думает"),
      (2, "Продажа"),
      (3, "Отказ"),
      (4, "Статус"),
      )
      full_name = models.CharField(max_length=190)
      phone_number = models.CharField(max_length=11)
      specialty = models.ForeignKey(Specialty, on_delete=models.CASCADE,
  null=True, blank=True)
      status = models.IntegerField(choices=STATUS_CHOICES,
                  default=4)
      comment = models.TextField(null=True, blank=True)
      created = models.DateField(default=date.today)
      manager = models.ForeignKey(User, on_delete=models.CASCADE,
  null=True, blank=True)
```

Б қосымшасының жалғасы

```
  date_given_meneger = models.DateField(null=True, blank=True)

    def __str__(self):
        return self.full_name
```

filters.py - тапсырыс тізімін фильтрлеу

```
import django_filters
from django_filters import DateRangeFilter,DateFilter
from django.contrib.auth.models import User
from datetime import date
from .models import ReceivedOrder

class ReceivedOrderFilter(django_filters.FilterSet):
    PART_CHOICES = (
    (1, "Теория + Практика"),
    (2, "Теория"),
    (3, "Практика"),
    (4, "часть"),
    )

    STATUS_CHOICES = (

    (1, "В работе"),
    (2, "Выполнено"),
    (3, "статус"),
    )

    part = django_filters.ChoiceFilter(label='Часть', choices=PART_CHOICES)
    status = django_filters.ChoiceFilter(label='Часть',
choices=STATUS_CHOICES)
    start_date = DateFilter(field_name='start_date', lookup_expr=('gt'))
    end_date = DateFilter(field_name='start_date',lookup_expr=('lt'))
    date_range = DateRangeFilter(field_name='start_date')
    title = django_filters.CharFilter(lookup_expr='icontains')

    class Meta:
        model = ReceivedOrder
        fields = ['title', 'client']

class ReportFilter(django_filters.FilterSet):

    class Meta:
```

```
  model = User
      fields = ['username',]
```

models.py - тапсырыс тізімінің моделі

```
from django.db import models
from django.contrib.auth.models import User
from datetime import date
from client.models import Client, Specialty

class University(models.Model):
    title = models.CharField(max_length=300)

    def __str__(self):
        return self.title

class ReceivedOrder(models.Model):
    PART_CHOICES = (
    (1, "Теория + Практика"),
    (2, "Теория"),
    (3, "Практика"),
    (4, "часть"),
)
    LANGUAGE_CHOICES = (
    (1, "На русском"),
    (2, "На казахском"),
    (3, "На английском"),
    (4, "язык"),
    )
    STATUS_CHOICES = (
    (1, "В работе"),
    (2, "Выполнено"),
    (3, "статус"),
    )
    client = models.CharField(max_length=190)
    phone_number = models.CharField(max_length=11)
    specialty = models.ForeignKey(Specialty, on_delete=models.CASCADE)
    title = models.CharField(max_length=300)
    start_date = models.DateField(default=date.today)
    end_date = models.DateField(null=True, blank=True)
    part = models.IntegerField(choices=PART_CHOICES,
                default=4)
```

```
    price = models.DecimalField(max_digits = 10, decimal_places = 2)
    language = models.IntegerField(choices=LANGUAGE_CHOICES,
                default=4)
    university = models.ForeignKey(University, on_delete=models.CASCADE)
    manager = models.ForeignKey(User, on_delete=models.CASCADE)
    comment = models.TextField()
    status = models.IntegerField(choices=STATUS_CHOICES,
                default=3)
    def __str__(self):
        return self.title
```

urls.py - тапсырыс тізімінің мекен жайы

```
from django.urls import path
from . import views
app_name = 'diplom'

urlpatterns = [
    path('client/list', views.ClientListView.as_view(), name='client_list'),
    path('client/update/status', views.ClientUpdateStatus.as_view(),
name='update_client_status'),
    path('received/order/list', views.ReceivedOrderListView.as_view(),
name='received_order_list'),
   path('create/received/order/', views.CreateReceivedOrderView.as_view(),
name='create_received_order'),
    path('report', views.ReportView.as_view(), name='report'),
]
```

views.py - тапсырыс тізімінің параметрлері

```
from django.shortcuts import render, redirect
from django.views.generic import ListView
from django.views.generic.base import TemplateResponseMixin, View
from django.contrib.auth.models import User
from django.http import HttpResponse
from client.models import Client
from client.filters import ClientFilter
from .models import ReceivedOrder
from .filters import ReceivedOrderFilter, ReportFilter
from .forms import ReceivedOrderForm

class ClientListView(ListView):
    queryset = Client.objects.all()
```

Б қосымшасының жалғасы

```
    template_name = 'client_list.html'
    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['filter'] = ClientFilter(self.request.GET,
         queryset=Client.objects.filter(manager=self.request.user).order_by('-
  date_given_meneger'))
        return context
class ClientUpdateStatus(View):

        def post(self, request):
                id = request.POST.get("id")
                status = request.POST.get("status")
                try:
                        client = Client.objects.get(id=id)
                        client.status = status
                        client.save()
                        return HttpResponse("True")
                except Client.DoesNotExist:
                        return HttpResponse("False")

  class  ReceivedOrderListView(ListView):
     queryset = ReceivedOrder.objects.all()
     template_name = 'received_order.html'
  def get_context_data(self, **kwargs):
        context        =        super().get_context_data(**kwargs)
        context['filter'] = ReceivedOrderFilter(self.request.GET,
           queryset=ReceivedOrder.objects.all().order_by('-start_date'))
        return context

  class CreateReceivedOrderView(TemplateResponseMixin, View):
        template_name = 'create_received_order.html'

        def get(self, request):
                form = ReceivedOrderForm()
                return self.render_to_response({'form': form})

        def post(self, request):
                form = ReceivedOrderForm(request.POST)
                if form.is_valid():
                        form_obj = form.save(commit=False)
                        form_obj.manager = request.user
                        form_obj.save()
                        return redirect('diplom:received_order_list')
```

```
        return self.render_to_response({'form': form})

class ReportView(ListView):
    queryset = User.objects.all()
    template_name = 'report.html'
    def get_context_data(self, **kwargs):
context = super().get_context_data(**kwargs)
        context['filter'] = ReportFilter(self.request.GET,
        queryset=User.objects.filter(groups__name='Manager'))
        return context
```