ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

«Программалық инженерия» кафедрасы

Сейдалы Гүлзира Асылханқызы

«QuizEmployee» проектісінің мобильді қосымшасын құру

Дипломдық жобаға
ТҮСІНІКТЕМЕЛІК ЖАЗБА

5В060200 – «Ақпараттану» мамандығы

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

«Программалық инженерия» кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ

ПИ кафедра меңгерушісі,
техника ғылымдарының кандидаты,
ассистент-профессор
_____ Р. Юнусов
«17» мая 2019 ж.

Дипломдық жобаға
ТҮСІНІКТЕМЕЛІК ЖАЗБА

«QuizEmployee» проектісінің мобильді қосымшасын құру

5В060200 – «Ақпараттану» мамандығы

Орындаған Сейдалы Г.

Ғылыми жетекші
Сениор-лектор
_____ С. Қалдыбеков
«17» мамыр 2019 ж.

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

«Программалық инженерия» кафедрасы

5В060200 – «Ақпараттану»

БЕКІТЕМІН
ПИ кафедра меңгерушісі,
техника ғылымдарының кандидаты,
ассистент-профессор
_____ Р. Юнусов
«__» _____ 2019 ж.

Дипломдық жоба орындауға
ТАПСЫРМА

Білім алушы *Сейдалы Гүлзира Асылханқызы*
Тақырыбы *«QuizEmployee» проектісінің мобильді қосымшасын құру*
Академиялық мәселелер жөніндегі проректорының *2018 жылғы «16» қазанның № 1162-б* бұйрығымен бекітілген
Аяқталған жобаны тапсыру мерзімі *2019 жылғы «21» мамыр*
Дипломдық жобаның бастапқы берілістері *Ұсынылатын дипломдық жобада «QuizEmployee» проектісінің мобильді қосымшасын құру. Rest технологиясы арқылы веб бетпен байланыс орнату*
Дипломдық жобада қарастырылатын мәселелер тізімі:
а) Негізгі бөлім;
б) Жобалау бөлімі;
е) Қолданылған бағдарламалық қамтамалар тізімі;
ж)Мобильді қосымшаны құру және оны іске асыру.

Сызба материалдарының тізімі (міндетті сызбалар дәл көрсетілуі тиіс)
Жобаның презентациялық 18 слайды ұсынылған.
Ұсынылатын негізгі әдебиет *15 әдебиеттер тізімінен тұрады.*

Дипломдық жобаны дайындау

КЕСТЕСІ

Бөлімдер атауы, қарастырылатын мәселелер тізімі	Ғылыми жетекші мен кеңесшілерге көрсету мерзімдері	Ескерту
Жалпы бөлім	15.02.2019	
Қолданылған бағдарламалық қамтама	19.03.2019	
Жобалау бөлімі	08.04.2019	
Дипломдық жұмыстың түсініктемелік	06.05.2019	

Дипломдық жоба бөлімдерінің кеңесшілері мен норма бақылаушының аяқталған жобаға қойған

қолтаңбалары

Бөлімдер атауы	**Кеңесшілер аты, әкесінің аты, тегі (ғылыми дәрежесі, атағы)**	**Қол қойылған күн**	**Қолы**
Норма бақылау	Алғожаева Р. Сениор-лектор	16.05.2019	
Бағдарламалық бөлімі	Сман Н.О. Ассистент	16.05.2019	

Ғылыми жетекші _____ С. Қалдыбеков

Тапсырманы орындауға алған білім алушы _____ Г. Сейдалы

Күні «17» мамыр 2019 ж.

АҢДАТПА

Қазіргі XXI- ғасыр даму мен инновация заманында ақпараттық технологияның,компьютерлер мен гаджеттердің адам өмірінде алатын орны ерекше. Жаңа технологиялар бізге жылдам қарым-қатынас үшін,білім саласында айтарлықтай жетістіктерге жетуге, жаңалықтар ашуда ыңғайлы болып отыр. Қазірдің өзінде смартфондар мен планшеттік компьютерлер адам өмір сүру салтын артарлықтай өзгертті. Қызмет көрсету саласы,әлеуметтік үкімет,денсаулық сақтау саласында да жылдан-жылға қарқынды дамуда.

Сонымен қатар адамдардың көпшілігі тым жиі ұялы телефондарды пайдаланады. Кез келген уақытта сізді қызықтыратын ақпаратты осы ықшам құрылғымен, әріптестеріңізбен, туыстарыңызбен және достарыңызбен таба аласыз. Барлық ноутбуктер, ұялы телефондар әрқашан пайдаланылмайды.

Телефонда көптеген контактілер, сондай - ақ басқа да көптеген-несие карталарының нөмірлері, күнделікті естеліктер, түрлі файлдар, адамдардың идеялары мен ойлары сақталған. Бұл телефон дереу ноутбук пен флешканы ауыстырады дегенді білдіреді. Осы себепті ұсынылған жоба Android платформасында әзірленді.

”QuizEmployee” мобильді қосымшасы – бұл жұмысқа қабылданушылардың түйіндемесінің интерактивті жүйесі болып табылады. Қосымшаны Қ.И Сәтбаев атындағы университетке жұмысқа тұруға үміткер маман PlayMarket бағдарламалар дүкенінен жүктеп,тіркеліп өзіне қажетті бөлім кадрларын таңдап,түйіндемесін толтырып, арнайы талаптарды орындау арқылы өз түйіндемесін жібере алады.

Жоба бойынша, Android мобильді қосымшасы клиент ретінде, ал веб-бет сервер ретінде, клиент-сервер технологиясын іске асырады.

REST сервис тек қана Android платформасына ғана емес, кез-келген құрылғыға, яғни Android, web-бет, Windows операциялық жүйелерінің барлығына веб-бетті қолдануға мүмкіндік береді.

АННОТАЦИЯ

В современном XXI веке в эпоху развития и инноваций информационные технологии, компьютеры и гаджеты занимают особое место в жизни человека. Новые технологии позволяют нам быстро общаться,добиваться значительных успехов в сфере образования, открывать новые открытия. Уже сейчас смартфоны и планшетные компьютеры значительно изменили образ жизни человека. Из года в год динамично развивается сфера услуг,социального правительства,здравоохранения.

Кроме того, большинство людей слишком часто используют мобильные телефоны. В любое время вы можете найти интересующую вас информацию с этим компактным устройством, коллегами, родственниками и друзьями. Все ноутбуки, мобильные телефоны не всегда используются.

В телефоне сохранилось множество контактов, а также множество других - номера кредитных карт, ежедневные воспоминания, различные файлы, идеи и мысли людей. Это означает, что телефон немедленно заменяет ноутбук и флешку. По этой причине предлагаемый проект разработан на платформе Android.

Мобильное приложение " QuizEmployee " - это интерактивная система резюме поступающих на работу. Приложение Специалист,претендующий на трудоустройство в университет им.и. Сатпаева,может загрузить в магазине программ PlayMarket, зарегистрироваться и выбрать необходимые для себя кадры отдела, заполнить резюме и отправить свое резюме, выполнив специальные требования.

По проекту, мобильное приложение Android, как клиент, а веб-страница, как сервер, реализует технологию клиент-сервер.

REST сервис позволяет использовать веб-страницу не только на платформе Android, но и на любом устройстве, включая все операционные системы Android, web-страница, Windows.

ANNOTATION

In the modern XXI century in the era of development and innovation information technologies, computers and gadgets occupy a special place in human life. New technologies allow us to communicate quickly,to achieve significant success in the field of education, to open new discoveries. Even now, smartphones and planetary computers have significantly changed the way of human life. From year to year dynamically developing services,social government,health.

In addition, most people use mobile phones too often. At any time you can find the information you need with this compact device, colleagues, relatives and friends. All laptops, mobile phones are not always used.

The phone has a lot of contacts, as well as many others - credit card numbers, daily memories, various files, ideas and thoughts of people. This means that the phone immediately replaces the laptop and flash drive. For this reason, the proposed project is developed on the Android platform.

Mobile application "QuizEmployee" is an interactive system of resumes of applicants. Application Specialist applying for employment at the University.I. Satpayeva,can download in the playmarket software store, register and select the necessary personnel of the Department, fill out a resume and send your resume, fulfilling special requirements.

According to the project, the Android mobile application as a client and the web page as a server implements the client-server technology.

REST service allows you to use a web page not only on the Android platform, but also on any device, including all operating systems Android, web-page, Windows.

МАЗМҰНЫ

	КІРІСПЕ	8
1	Негізгі бөлім	9
1.1	Жобаның артықшылығы	9
1.2	Мобильді қосымша серверіне қойылатын талаптар	9
1.3	REST API сервистер құрылымы	10
1.4	Сервер логикасы	14
2	Жобалау бөлімі	16
2.1	Унифицирлен модельдеу тілі	16
2.2	Жүйенің мінез-құлқын модельдеу	17
2.3	Жүйенің құрылымын модельдеу	18
2.4	Жүйенің архитектурасын модельдеу	19
3	Қолданылған бағдарламалық қамтамалар	20
3.1	PostgreSQL технологиясы	20
3.2	Android платформасы.Android Studio даму ортасы	21
3.3	Java программалау тілі	25
3.4	Django REST тірегі	28
3.5	Retrofit 2.0 кітапханасы	33
	ҚОРЫТЫНДЫ	34
	ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ	35
	А ҚОСЫМШАСЫ	36
	Б ҚОСЫМШАСЫ	39
	СПЕЦИФИКАЦИЯ	46

КІРІСПЕ

Дипломдық жоба идеясы дайын құрылған API арқылы жұмысқа тұру үшін түйіндеме жіберетін мамандарға арналған Android платформасында мобильді қосымша құру болып табылады. Жобаның қызметі дайын API деректер қорымен қауіпсіз байланысқа түсіп, сұрауларға уақытылы жауап қайтару.

Қосымша аясы жұмысқа кадр қабылдайтын белгілі бір мекемеге,ұйымға,университетке арналған,сонымен қатар қосымша жоғарғы технологияларға сәйкес,тартымды дизайнға және ыңғайлы әрі түсінікті интерфейске ие болу қажет.

“QuizEmployee” проектісінің түпкі мақсаты, Қ.И.Сәтбаев атындағы университетіне кадр қабылдауға (оқытушылар,асистент,лектор) арналған жылдам әрі ыңғайлы,қағаз бастылықтан арылту мақсатындағы мобильді қосымша болып табылады. Жобаның жүзеге асу барысында барлық тапсырмалар мен талаптар әділ түрде орындалып,барлығы базада сақталынады. “QuizEmployee” проектісінде қолданылатын технологиялар:

- Java программалау тілі;
- PostgreSQL деректер қорымен басқару;
- RetroFit 2.0 кітапханасы;
- Android платформасында;

“QuizEmployee” проектісі ыңғайлы әрі жылдам онлайн түйіндеме өткізіп,өз білімін дәлелдеп заманауи үлгідегі бәсекеге қабілеттіктің арнасы. “QUIZ-тест” проектісі мобильді қосымша мен веб беттен тұрады. Мобильді қосымша мен бет бет арасындағы байланыс REST сервис, JSON форматы және HTTP сұраныстары арқылы жүзеге асырылады.

1 Негізгі бөлім

1.1 Дипломдық жоба артықшылығы

Мобильді қосымша веб-беттің клиент-сервер технологиясындағы клиенттің рөлін атқарады және жекелей қосымша бола алады.Қосымша көп функционалды,пошта арқылы тіркелу, түйіндеме толтыру,жылдам тест тапсыру және дайындалу,жеке кабинет те қарастырылған. Қосымша аудиториясы Қ.И.Сәтбаев университетіне жұмсықа түйіндеме жіберуші үміткер мамандарға арналады. Дипломдық жобаның өзектілігі онлайн тест тапсыру технологиясының бар болуында, және веб-бетпен REST технологиясының ауқымын кеңінен пайдалана отырып байланысқа түсе алуында. Қосымша Android studio бағдарламасында Java тілінде жазылған, қазіргі кезде мобильді бағдарлама кеңінен таралғандықтан қолайлылық танытады.

Сонымен қатар, қазіргі кезде барлық адамдар мобильді телефонды өте көп қолданады. Әрдайым, кез келген уақытта, осы ықшам құрылғымен бізді қызықтыратын ақпаратты табуға, әріптестерін, олардың туыстары мен достары байланыстыра алуға болады. Барлығы ноутбук, ұялы телефонды әрдайым қолданады.

Телефонда сақталған контактілер, сонымен қатар көптеген тағы басқа да ақпарат көп және несиелік карта нөмірлерін, күнделікті естелік, түрлі файлдар, адамдардың идеялары мен ойлары жазылады.

“QuizEmployee” проектісі ыңғайлы әрі жылдам онлайн түйіндеме өткізіп,өз білімін дәлелдеп заманауи үлгідегі бәсекеге қабілеттіктің арнасы. “ QuizEmployee ” проектісі мобильді қосымша мен веб беттен тұрады. Мобильді қосымша мен бет бет арасындағы байланыс REST сервис, JSON форматы және HTTP сұраныстары арқылы жүзеге асырылады.

1.2 Мобильді қосымша серверіне қойылатын талаптар

Мобильді қосымшалар серверін құру алдында келесі талаптарға ерекше назар аударылды:

1) Серверде REST API сервисімен дұрыс байланыс болуы керек.

2) Жүйе үшін қажетті барлық деректерді сақтауға арналған сервердің деректер қоймасы болуы тиіс.

3) Мобильді қосымша әрқашан пайдаланушыға қол жетімді, яғни 24 сағат,аптасына 7 күн бойы тұру жұмыс істейді. Пайдаланушылар, яғни үміткерлер жүйеге кіріп, кез келген уақытта тестілеуден өтуге немесе дайындалуға мүмкіндік алуы тиіс. Жүйеге тұрақты түрде байланыса алу үшін интернетке байланысы бар Android операциялық жүйесін қамтамасыз ететін гаджет болуы тиіс.

4) Деректер тұтастығы мен қауіпсіздігі үшін қосымшаны пайдаланушылар үшін аутентификация және авторизациялandыру мүмкіндіктерін жасау. Серверде мамандар кандидаттары туралы маңызды жеке ақпарат сақталатындықтан. Бұл жүйенің дұрыс жұмыс істемеуіне әкелуі мүмкін. Кез келген пайдаланушының жүйеге қол жеткізбеуін ұйымдастыру қажет,тек арнайы логин мен пароль сияқты деректері бар пайдаланушылар жүйеге қол жеткізуге құқылы болуы тиіс.

5) Сервердің веб-интерфейсі пайдаланушы үшін түсінікті және ыңғайлы белгілі бір нәтижеге қол жеткізу үшін пайдаланушылар қабылдайтын қадамдардың ең аз саны қажет.

6) Серверде сақталған деректерді пайдалану және пайдалану

нәтижелі әрекеттерді орындау үшін API пайдаланушы интерфейсінің интерфейсін, яғни мобильді қосымшалар үшін пайдаланушы интерфейсін пайдалану мүмкіндігі көзделуі тиіс.

1.3 REST API сервистер құрылымы

Дипломдық жобаның негізгі мақсаты веб-серверге негізделген мобильді қосымша құру болып табылады. Веб-сервер мен қосымша арасындағы байланыс API, REST API сервистер құрылымы көмегімен жүзеге асырылады. Ал, API (ағылш. application programming interface, API) бір компьютерлік бағдарлама басқа бағдарламамен өзара іс-қимыл жасай алатын тәсілдердің сипаттамасы (кластар, рәсімдер, функциялар, құрылымдар немесе константалар жиынтығы). Әдетте қандай да бір интернет протоколының (мысалы, RFC), бағдарламалық қаңқаның (фреймворк) немесе операциялық жүйе функцияларының қоңырау стандартының сипаттамасына кіреді. Жиі жеке бағдарламалық кітапхана немесе операциялық жүйе сервисі арқылы жүзеге асырылады. Бағдарламашылар әр түрлі қосымшаларды жазуда қолданады.

API API бағдарламасы (модуль, кітапхана) қамтамасыз ететін функционалдылықты анықтайды және API бұл функцияның қалай жүзеге асатынын дәлелдеуге мүмкіндік береді.

Егер бағдарлама (модуль, кітапхана) қара жәшік деп қарастырылса, онда API - осы қораптың пайдаланушыға қол жетімді және ол айналдыра және тарта алатын «тұтқалар» жиынтығы.

Бағдарламалық құралдың құрамдас бөліктері API арқылы бір-бірімен өзара әрекеттеседі. Бұл жағдайда компоненттер әдетте иерархияны қалыптастырады - жоғары деңгейлі компоненттер төмен деңгейлі құрамдас бөліктердің API-ін пайдаланады және өз кезегінде төменгі деңгейлі компоненттердің API-ын пайдаланады.

Осы принцип бойынша Интернет арқылы деректерді беру хаттамалары жасалды. Стандартты хаттама стекі (OSI желісінің үлгісі) 7 қабаттан (HTTP және IMAP протоколдарына ұқсас физикалық бит-тасымал қабатын қосымшалар

хаттамасының қабатына дейін) қамтиды. Әрбір деңгей деректерді берудің алдыңғы («төменгі») деңгейінің функционалдығын пайдаланады және өз кезегінде келесі («жоғарғы») деңгейге қажетті функционалдылықты қамтамасыз етеді.

Протоколдың тұжырымдамасы API тұжырымдамасына мағынаға жақын екенін атап өту маңызды. Екеуі де функционалдылықтың абстракциясы, тек бірінші жағдайда деректерді беру туралы, екіншісі - қолданбалардың өзара әрекеті туралы.

REST (өкілдігін жіберуге арналған қысқа) - желіде таратылатын қосымшаның компоненттерінің өзара әрекеттесуінің сәулет стилі. REST бөлінген гипермедиа жүйесін жобалау кезінде ескерілетін шектеулер жиынтығы болып табылады. Кейбір жағдайларда (онлайн-дүкендер, іздеу жүйесі, деректерге негізделген басқа жүйелер) бұл өнімділікті арттыруға және жеңілдетілген сәулетке әкеледі. Кең мағынада REST құрамдас бөліктері клиенттер мен серверлердің Дүниежүзілік торда өзара әрекеттесуіне ұқсас түрде өзара әрекеттеседі. REST - RPC-ге балама.

Интернетте қашықтан қоңырау шалу рәсімі қарапайым HTTP сұрауы болуы мүмкін (әдетте «GET» немесе «POST», бұл сұрау «REST сұрауы» деп аталады) және қажетті деректер сұрау параметрлері ретінде беріледі.

REST негізіндегі веб-қызметтер үшін (яғни ол қолданатын шектеулерді бұзбайды) «RESTful» термині қолданылады.

SOAP-негізделген веб-қызметтерден (веб-қызметтер) айырмашылығы, RESTful веб-API үшін «ресми» стандарт жоқ. Өйткені, REST - сәулет стилі, ал SOAP - хаттама. REST өздігінен стандартты болмаса да, RESTful қолданбаларының көбісі HTTP, URL, JSON және XML сияқты стандартты пайдаланады.

REST API әзірлеушілер сұрау жасай алатын және жауап ала алатын функциялар жиынтығын анықтайды. Өзара іс-қимыл HTTP протоколы бойынша жүреді. Мұндай тәсілдің артықшылығы HTTP протоколын кең тарату болып табылады, сондықтан REST API кез келген бағдарламалау тілінен іс жүзінде қолдануға болады.

REST API негізінен серверлерге сыртқы серверлерден сұрау үшін арналған. Веб-клиенттерден — әлеуметтік қосымшаның клиенттік бөлігінен немесе сайттан сұраулар үшін-JS API және Flash-кітапхана бар,олар пайдалануға ыңғайлы және оңай.

Интернетте қашықтағы сұраныстарды, әдеттегі HTTP-сұратуды (әдетте "Post" немесе "GET", мұндай сұрау "REST-сұрау" деп аталады) жіберуге болады және талап етілетін деректер сұрау параметрі ретінде беріледі.

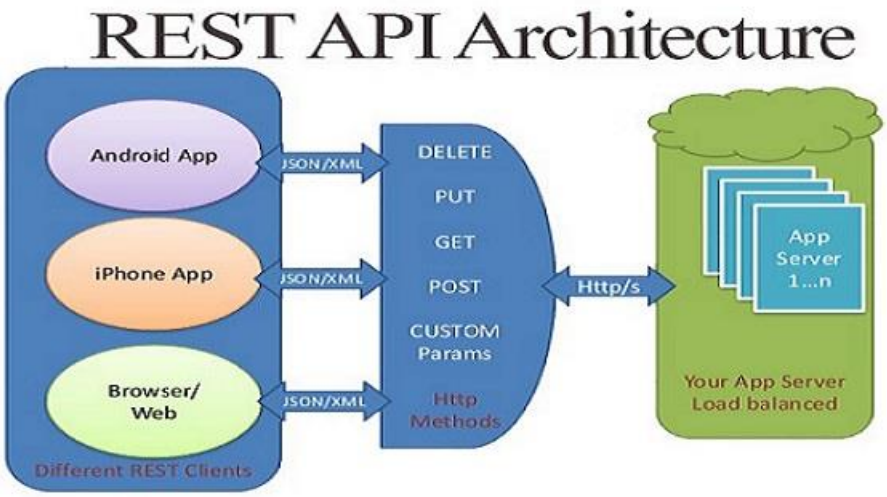
REST – (Representational state transfer)-бұл World Wide Web сияқты таратылған жүйелер үшін бағдарламалық қамтамасыз ету архитектурасының стилі. REST термині 2000 жылы http-Протокол авторларының бірі Роем Филдингпен енгізілді. REST қолдайтын жүйелер RESTful-жүйелер деп аталады.

RESTful API-деректерді алу, алу, орналастыру және жою үшін HTTP-сұрауларды пайдаланатын қолданбалы бағдарламалық интерфейс (API).

RESTful API-интерфейсі, сондай-ақ RESTful веб-қызметі деп аталатын, көріністер күйін беру технологиясына (REST), сәулет стиліне және Веб-қызметтерді әзірлеу кезінде жиі пайдаланылатын коммуникацияларға жақындауға негізделген.

REST технологиясы, әдетте, сенімді SOAP технологиясы, өйткені REST аз өткізу қабілетін пайдаланады, бұл оны Интернетте пайдалануға жарамды етеді. Веб-сайтқа арналған API-бұл екі бағдарламаға бір-бірімен өзара әрекеттесуге мүмкіндік беретін код. API әзірлеушілерге операциялық жүйеден немесе басқа қосымшадан қызметтерді сұрайтын бағдарламаны жазудың дұрыс тәсілін ұсынады.

REST API архитектурасы 1.1 - суретте көрсетілген.



1.1 - сурет – REST API архитектурасы

RESTful API-интерфейсі, сондай-ақ RESTful веб-қызметі деп аталатын, көріністер күйін беру технологиясына (REST), сәулет стиліне және Веб-қызметтерді әзірлеу кезінде жиі пайдаланылатын коммуникацияларға жақындауға негізделген.

REST технологиясы, әдетте, сенімді SOAP технологиясы, өйткені REST аз өткізу қабілетін пайдаланады, бұл оны Интернетте пайдалануға жарамды етеді. Веб-сайтқа арналған API-бұл екі бағдарламаға бір-бірімен өзара әрекеттесуге мүмкіндік беретін код. API әзірлеушілерге операциялық жүйеден немесе басқа қосымшадан қызметтерді сұрайтын бағдарламаны жазудың дұрыс тәсілін ұсынады.

Браузерлер пайдаланатын REST Интернет тілі ретінде қарастыруға болады. Бұлттарды пайдаланудың өсуіне қарай веб-сервистерді ұсыну үшін API-интерфейстер пайда болады. REST-бұл пайдаланушыларға бұлтты сервистермен қосылуға және өзара әрекеттесуге мүмкіндік беретін API құру үшін логикалық таңдау. RESTful API Amazon, Google, LinkedIn және Twitter сияқты сайттарда қолданылады.

API RESTful шағын модульдер сериясын құру арқылы транзакцияны бұзады. Әрбір модуль транзакцияның белгілі бір базалық бөлігіне жүгінеді. Бұл модуль әзірлеушілерге үлкен икемділік береді, бірақ әзірлеушілер үшін нөлден әзірлеу оңай емес. Қазіргі уақытта Amazon Simple Storage Service сервисі, бұлтты деректерді басқару интерфейсі және OpenStack Swift ұсынатын модельдер ең танымал болып табылады.

RESTful API RFC 2616 хаттамасымен анықталған HTTP әдістемелерінің артықшылықтарын пайдаланады. Олар ресурсты алу үшін GET қолданады; put объект, файл немесе блок болуы мүмкін ресурсты жаңарту немесе күйін өзгерту үшін; post осы ресурсты құру үшін; және оны жою үшін жою үшін.

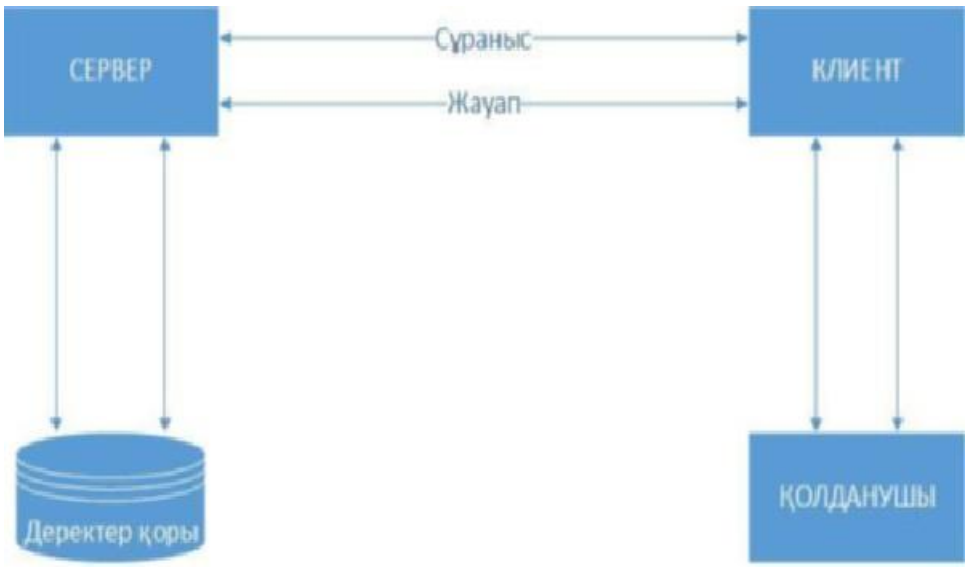
REST - пен желілік компоненттер - бұл сіз қол жеткізуді сұрайтын ресурс-қара жәшік, оны жүзеге асыру бөлшектері түсініксіз. Барлық азаматтығы жоқ қоңыраулар деп болжанады; ештеңе орындалған арасында RESTful сервисі сақталуы мүмкін емес.

Қоңырау күйі сақталмайтындықтан, REST бұлт қолданбаларында пайдалы. Күйсіз компоненттер істен шыққан жағдайда еркін қайта бөлінуі мүмкін және олар жүктеменің өзгеруін ескере отырып масштабталуы мүмкін. Бұл кез-келген сұрау компоненттің кез-келген данасына жіберілуі мүмкін. Бұл REST веб-пайдалану үшін қолайлы етеді, бірақ RESTful моделі бұлтты сервистерде де пайдалы, өйткені API арқылы сервисті байланыстыру - бұл URL декодтау басқару мәселесі. Бұлттық есептеулер мен микросервисттер болашақта RESTful API-ді жетілдіреді.

1.4 Сервер логикасы

Дипломдық жоба бойынша, Android мобильді қосымшасы клиент ретінде, ал веб-бет сервер ретінде, клиент-сервер технологиясын іске асырады.
Осы тарауда, жалпы сервер логикасына ,байланысына тоқталып өтіледі.

Мобильді қосымша серверінің жұмыс істеу принципі клиент-серверлік архитектураға негізделген. (1.2-суретте көрсетілген)



1.3 - сурет – Клиент-сервер технологиясы

Сервер-қашықтан компьютерде жұмыс істейтін және клиент-қосымшалармен "қарым-қатынас" функционалын іске асыратын бағдарламаны білдіреді (сұраныстарды тыңдайды, берілген параметрлер мен мәндерді таниды, оларға дұрыс жауап береді).

Клиент-біздің жағдайда, серверге түсінікті сұраныс қалыптастыра алатын және алынған жауапты оқи алатын мобильді құрылғыдағы бағдарлама.

Өзара іс - қимыл интерфейсі-екі тараптың сұрау салуларын/жауаптарын беру/алу форматы мен тәсілі. Бұл элементтердің кез келгені қалай жүзеге асырылғаны маңызды емес, олар кез келген жағдайда.

Серверлік және клиенттік қосымшалар үшін талаптар қалыптасуының ерекшелігі болып табылады, олар бірқатар жағдайларда өзара байланысты. Бастау үшін деректермен алмасу механизмі контекстіндегі базалық талаптарды жазамын:

– клиенттің кроссплатформалығы: көбінесе әр түрлі платформаларды Android, iOS, Windows Phone және т.б. қолдауды қамтамасыз ету маңызды;

– жылдамдық: workflow үшін жеткілікті жұмыс жылдамдығы, пайдалаушының графикалық интерфейсінде жайлы жауап болуы керек;

– қарапайымдылық: хаттама API оңай болған сайын, кодты іске асыру мен қолдауға аз уақыт кетеді, әзірлеушінің біліктілігі аз болуы мүмкін;

– тиімділік: Хаттаманы іске асыру қиын болған сайын, шектеулі мобильді құрылғы ресурстары көп тұтынылады.

– қосымша талаптар қосымшаның ерекшелігіне байланысты:

– сервердің масштабталуы-SaaS үшін, ең дұрысы келушілердің үлкен ағыны күтілетін әлеуметтік қосымшалар, бұл шарт міндетті. Пайдаланушылар саны бойынша шектеулер бар немесе Саны болжанатын бизнес қосымшалар үшін бұл сипат талап етілмейді;

– интерактивтілік: бірқатар қосымшалар нотификация механизмімен қамтамасыз ету керек-қосымшаға (пайдаланушыға) белгілі бір оқиғалардың

басталғандығы туралы хабарлау, пайдаланушыға хабар беру. Бұл қасиетке, мысалы, биржалық жүйе немесе автоматты такси диспетчері ие болуы керек.

– ашық API: бөгде әзірлеушілер құжатталған хаттама арқылы жүйенің функционалын пайдалана алады деп болжанады. Өйткені клиент мобильді және сыртқы серверлік қосымша болуы мүмкін.

2 Жобалау бөлімі

2.1 Унифицирлен модельдеу тілі

Unified Modeling Language-UML (Unified Modeling Language-UML) - бұл бағдарламалық жүйелерді, сондай-ақ бизнес модельдерін және басқа да бағдарламалық емес жүйелерді модельдеу, жобалау және құжаттау үшін қажет тіл. UML күрделі және аса ауқымды жүйелерді модельдеуде бұрыннан пайдаланылатын озық инженерлік технологиялардың үйлесімін көрсетеді. UML ажырамас бөлігі - OCL (Object Constraint Language).

Модель дегеніміз – жүйені жақсырақ түсіну үшін құрылатын нақтылықтың қарапайымдандырылған түрі,абстракция. Ал, жүйе дегеніміз – қандай да бір мақсатқа жету үшін ұйымдасқан элементтер жиыны. Жүйе бірнеше ішкі жүйелерден тұруы мүмкін. Сонымен қатар, жүйе оны мүмкіндігінше әр түрлі жағынан қарастыратын модельдердің көмегімен сипаттайды. Модельдердің құрастырушы бөліктері болып класстар,интерфейстер,компоненттер және түйіндер сияқты болмыстар болып табылады. UML тіліндегі модельдер осындай жүйелерді абстрактілі түрде сипаттауға қолданылады. Модельдер бізге құрылатын жүйенің құрылымын анық көрсетуге,жүйенің қалай жақсы жұмыс істейтінін түсінуге және оны UML тіліндегі диаграммалар көмегімен көрсетуге көмектеседі.

Сонымен қатар, UML тілінде модель әр түрлі диаграммалар көмегімен құрылады. Диаграмма – бұл бір-бірімен байланысқан төбелер (болмыстар) және қабырғалар (байланыстар) жиынтығынан тұратын граф түріндегі графикалық көрініс. Жүйені модельдеу барысында әр түрлі диаграммалар сызылады. Әрбір жеке алынған диаграмма жүйенің белгілі бір бөлігін ғана көрсететін проекциясы болып саналады.

Дипломдық жобада хабарлама алмасу жүйесін модельдеу үшін IBM компаниясының Rational Rose Enterprise Case – құралының көмегімен сызатын боламыз.

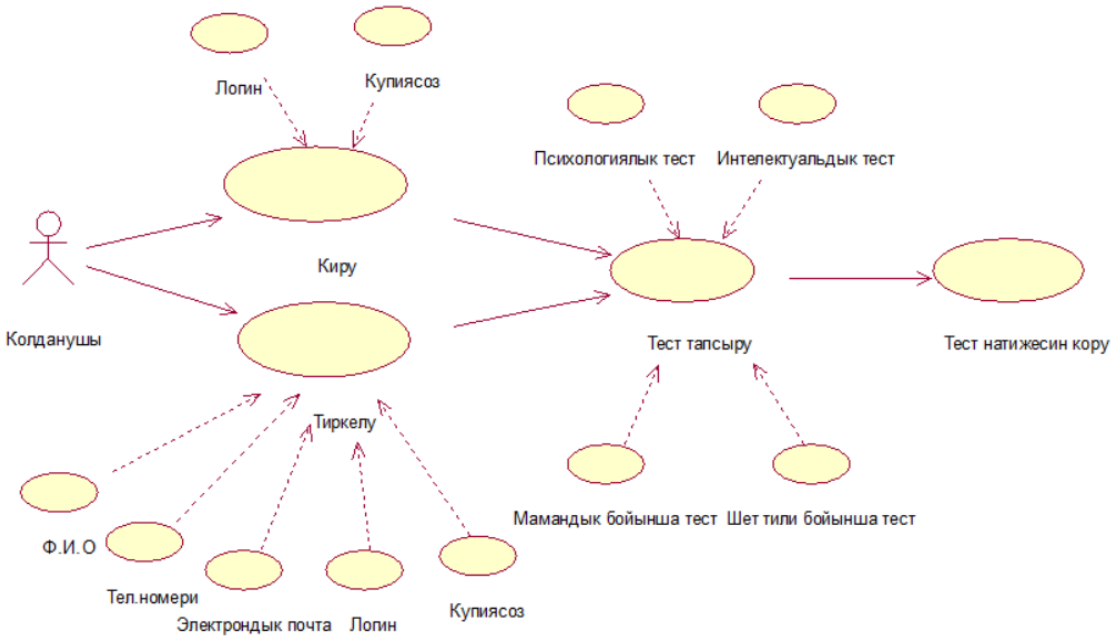
Дипломдық жұмыста құрылатын жүйенің моделі үш түрлі сызбада қарастырылды:

1) Жүйенің мінез-құлқын модельдеу. Мұнда жүйенің моделін пайдалану нұсқаларының диаграммасы (Use Case Diagram) және қызмет диаграммасы (Activity Diagram) арқылы көрсетіледі.
2) Жүйенің құрылымын модельдеу. Мұнда жүйенің моделі класстар диаграммасының (Class Diagram) көмегімен көрсетіледі.
3) Жүйенің архитектурасын модельдеу. Мұнда жүйенің моделі компоненттер диаграммасы (Component Diagram) арқылы көрсетіледі.

2.2 Жүйенің мінез-құлқын модельдеу

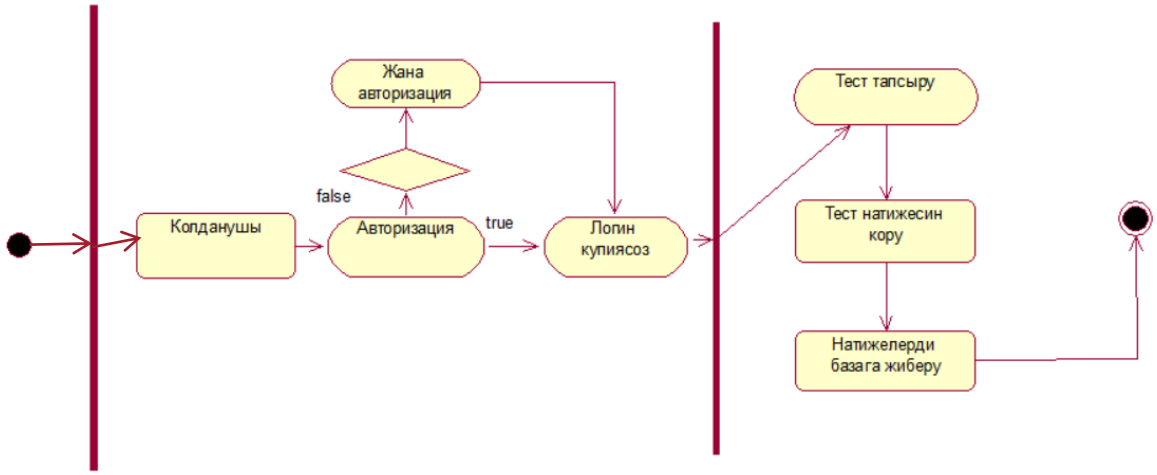
Модельдеудің бұл түрі жүйенің динамикалық аспектілерін сипаттау үшін көбінесе пайдалану нұсқаларының диаграммасы (Use Case Diagram) көмегімен 2.1 – суретте көрсетіледі. Ол диаграмма бағдарламалық жүйе жалпы алғанда «не істейді?» деген сұраққа жауап береді. Пайдалану нұсқаларының диаграммасы – жүйенің динамикалық аспектілерін модельдеуге арналған UML тіліндегі диаграммалардың бір түрі және жүйенің мінез-құлқын модельдеудегі диаграммалардың негізгі түрі. Диаграммада актерлер және пайдалану нұсқалары бейнеленеді. Актер дегеніміз – жүйемен қарым-қатынасқа түсетін сыртқы объектілер. Актер адам немесе сыртқы бағдарлама, яғни жүйеге әсер ете алатын кез келген болмыс бола алады. Ал пайдалану нұсқалары дегеніміз – актерлердің жүйемен қарым-қатынасқа түсу барысында істейтін әрекеттері. Ол әрекеттерді орындау барысында қандай да бір өзгеріс пайда болуы керек. Хабарлама алмасу жүйесінде мобильді қосымша серверімен қарым-қатынасқа түсетін актерлер:

– қолданушы үміткер мамандар.
– админ.



2.1 – сурет Пайдалану нұсқаларының диаграммасы

Жүйенің динамикалық аспектілерін модельдеу үшін пайдаланылуы мүмкін тағы бір диаграмма түрі- қызмет диаграммасы 2.2- суретте көрсетілген (Activity Diagram). Әдетте, іс-әрекет диаграммасы блок-схема болып табылады, ол басқару ағыны сервистің бір түрінен екіншісіне ауысатынын көрсетеді. Қарапайым блок-схемаға қарағанда, сервистік схемада сіз параллельді және басқару ағынының тармағын жақсы таба аласыз. Қызмет - бұл қазіргі уақытта машина жасауда (автоматтандыру) енгізілетін іс-қимылдар жиынтығы. Қызметті орындау нәтижесінде белгілі бір операциялар немесе жүйеде және оның жай-күйіндегі өзгерістер орындалады.

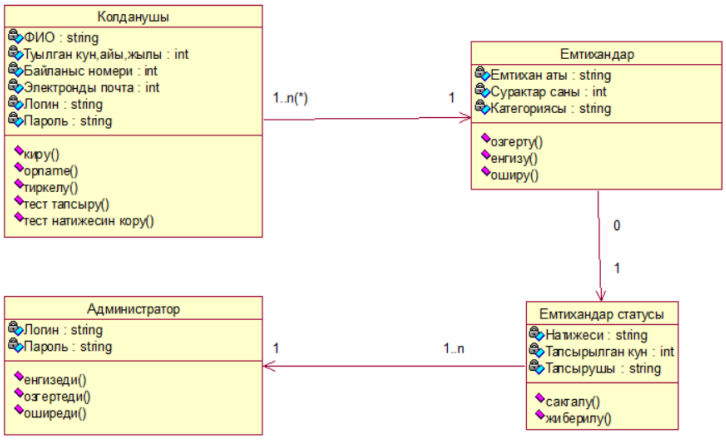


2.2-сурет Қызмет диаграммасы

Кез келген модельдеуші элемент үшін қызмет көрсету диаграммасы динамикалық аспектілерді модельдеу үшін пайдаланылуы мүмкін. Қызмет көрсету диаграммасы іс-әрекет сценарийін модельдеу бойынша операцияларға және нысандар жиынтығының динамикалық аспектілерін модельдеу бойынша операцияларға жиі қолданылады. Сіз көріп отырған нұсқада сервистік схеманы өрістеткен кезде, орындау ортасы жұмыс процестерінің ағынын іске қосады. Қызмет диаграмманың негізгі функциясы жұмыс парағын орындау кестесін көрсету болып табылады. Диаграммада сопақ формалар мен тікбұрыштар арқылы жүйенің күйі көрсетілген. Параллель жұмыс қалың көлденең және тік сызықтар арқылы орындалады. Тармақталу белгілері ромб ретінде белгіленген. Сонымен қатар, бұл операцияның бастапқы және соңғы позициялары дөңгелек.

2.3 Жүйенің құрылымын модельдеу

Модельдеудің осы түрін сипаттау үшін класс диаграммасы (Class Diagram) 2.3 – суретте көрсетілген. Жүйені модельдеу үшін пәндік саланың ерекшеліктерін анықтау қажет. Сыныптар жүйедегі негізгі ұғымдар болып табылады және жүйелік сөздік құрайды.



2.3 – сурет Класстар диаграммасы

Класс-бірдей атрибуттары, операциялары, сілтемелері және мәндері бар объектілер жиынтығы. Класта атрибуттар мен операциялар сияқты атрибуттар бар. Атрибут-сынып нысандары үшін қосымша белгілерді көрсететін атрибут. Операция-класс объектілерімен орындалуы мүмкін әрекеттер.

Класс диаграммасы-бұл объектілі-бағытталған жүйелерді моделдеу кезінде жасалатын диаграмма түрі. Класстар, интерфейстерді және бірлескен жұмыстарды және олардың арасындағы байланыстарды көрсетеді. Класстар диаграммасы статикалық жүйе көрінісін моделдеу үшін пайдаланылады. Яғни сөздік жүйесін көрсету.

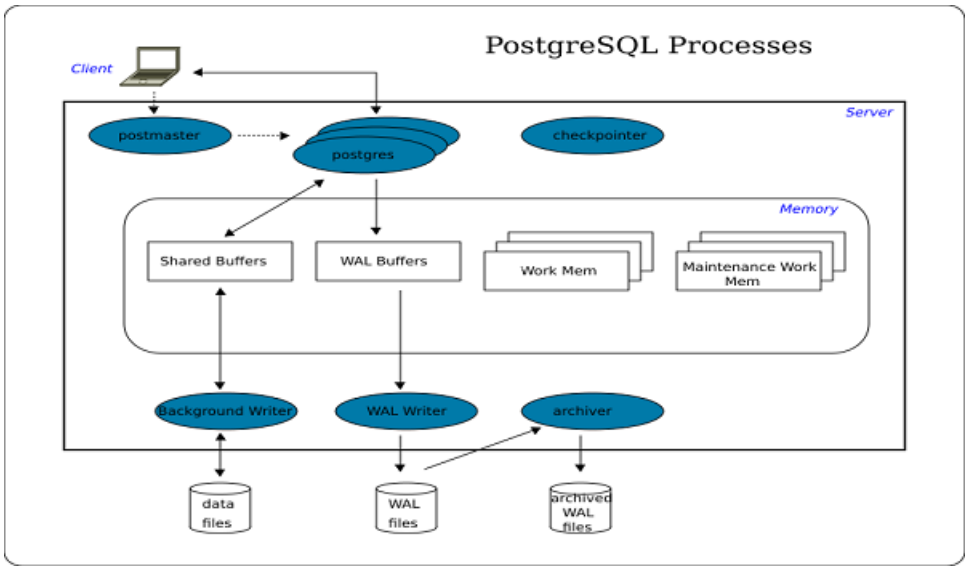
Диаграмма нүктелік сызықтардың тәуелділігін көрсетеді. Өйткені кейбір класстар басқа класс атрибуттарын пайдаланады.

3 Қолданылған бағдарламалық қамтамалар

3.1 PostgreSQL технологиясы

Мобильді қосымша серверінің деректер қорын басқару жүйесі ретінде PostgreSQL пайдаланылды. Сервердің мәліметтер қорында жүйенің дұрыс жұмыс істеуіне қажетті барлық ақпарат сақталады. Онда барлық мама үміткерлердің жеке ақпараттары,дайындық бөлімінде жиналған баллдары сақталынатын болады.

PostgreSQL ("пост-МАЭС-Кью-Эл"деп аталады) — әлемдегі ашық ДБЖ неғұрлым дамыған және коммерциялық дерекқорлардың шынайы баламасы болып табылатын деректер қорын басқарудың еркін таратылатын объектілік-реляциялық жүйесі. PostgreSQL архитектурасы 3.1-суретте көрсетілген.



3.1 - сурет -PostgreSQL архитектурасы

PostgreSQL - Берклиде жазылған түпнұсқалық кодтың ұрпағы болып табылатын ашық бастапқы өнім болып табылады. PostgreSQL SQL стандартының көпшілігін қолдайды және көптеген жетілдірілген мүмкіндіктерді ұсынады:

- күрделі сұраулар;
- шетелдік кілттер;
- триггерлер;
- пікірлер;
- транзакциялық тұтастық;
- көп нұсқалы параллельді кіруді басқару;

Сондай-ақ, PostgreSQL мүмкіндіктері жаңа пайдаланушы қосу арқылы кеңейтілуі мүмкін:

– деректер түрлері;
– функциялары;
– операторлар;
– жиынтық функциялар;
– индекстік әдістер;
– процедуралық тілдер;

Сонымен қатар, PostgreSQL либералды лицензия бойынша босатылғандықтан, оны кез-келген мақсатта, соның ішінде жеке, коммерциялық немесе академиялық түрде пайдалануға, өзгертуге және таратуға болады.

PostgreSQL тек реляциялық емес, объектілік-реляциялық ДББЖ. Бұл MySQL, MariaDB және Firebird сияқты ашық бастапқы коды бар басқа SQL деректер базасынан кейбір артықшылықтарды береді.

Объектілі-реляциялық деректер базасының іргелі сипаттамасы - бұл пайдаланушы объектілерді және олардың мінез-құлқын, оның ішінде деректер типтерін, функцияларын, операцияларын, домендерді және индекстерді қолдау. Бұл Postgres керемет икемді және сенімді етеді. Сонымен қатар, ол деректердің күрделі құрылымын жасай, сақтай және шығара алады. Төменде кейбір мысалдарда сіз стандартты РСУБД қолдау емес, салынған және құрамдас конструкцияларды көресіз.

Postgres-тің қолдайтын деректер түрлерінің кең тізімі бар. Сандық, құбылмалы нүктелі, мәтіндік, буль және басқа да күтілетін деректер типтерінен (сондай-ақ олардың көптеген нұсқалары) басқа, PostgreSQL uuid, ақшалай, аударылатын, геометриялық, бинарлық типтерді, желілік адрестерді, биттік жолдарды, мәтіндік іздестіруді, xml, json, массивтерді, композиттік типтерді және диапазондарды, сондай-ақ объектілерді сәйкестендіру үшін кейбір ішкі типтерді қолдаумен мақтана алады. Әділдік үшін MySQL деп айтуға тұрарлық, MariaDB және Firebird де осы деректер түрлерінің кейбір бар, бірақ тек Postgres оларды барлық қолдайды.

PostgreSQL әр түрлі желілік адрестердің сақталуын қамтамасыз етеді. CIDR деректер түрі (интернет Домен, Classless Internet Domain Routing кластық емес маршруттау) IPv4 және IPv6 желілік адрестеріне келісім береді. Міне, бірнеше мысал:

– 192.168.100.128 / 25
– 10.1.2.3 / 32
– 2001:4f8:3:ba:2e0:81ff:fe22:d1f1/128
– :ffff:1.2.3 / 128

Сондай-ақ, желілік адрестерді сақтау үшін IPv4 және IPv6 хосты үшін пайдаланылатын INET деректер түрі бар, онда қосалқы желілер міндетті емес. MACADDR деректер түрі 08-00-2b сияқты жабдықты идентификациялау үшін MAC адрестерін сақтау үшін пайдаланылуы мүмкін-01-02-03 ескерту.

MySQL және MariaDB-да желілік адрестерді айырбастау үшін INET функциялары бар, бірақ олар желілік адрестерді Ішкі сақтау үшін деректер түрлерін ұсынбайды. Firebird-де желілік мекенжайларды сақтау үшін түрі жоқ.

Postgres объект-реляциялық дерекқор болғандықтан, мәндер массивтері бар деректер түрлерінің көбінде сақталуы мүмкін. Бұл баған үшін деректер түрінің сипаттамасына тік жақшалар немесе ARRAY өрнегін пайдалану арқылы жасалуы мүмкін. Массивтің өлшемі көрсетілуі мүмкін, бірақ бұл міндетті емес.

Орналасқан жер туралы деректер көптеген қосымшаларға тез бейімделуде. PostgreSQL көп нүктелер, сызықтар, шеңберлер және көпбұрыш сияқты көптеген геометриялық деректер түрлерін қолдады. Мұндай түрлердің бірі PATH болып есептелінеді, ол дәйекті нүктелер жиынтығынан тұрады және ашық болуы мүмкін (бастау және аяқталу нүктелері қосылмаған) немесе жабық (бастау және аяқталу нүктелері қосылған). Туристік бағытты мысалға келтірейік. Бұл жағдайда, жаяу серуендеу цикл болып табылады, сондықтан бастапқы және аяқталу нүктелері біріктіріледі, бұл менің жолым жабық дегенді білдіреді. Координаттар жиынтығының айналасындағы жақша жабық жолды көрсетеді, ал тік жақшалар ашық екенін көрсетеді.

PostgreSQL сенімділігі тексерілген және дәлелденген факт болып табылады және келесі мүмкіндіктермен қамтамасыз етіледі:

– ACID - атомдық, қарама-қайшы болмаушылық, оқшаулану,деректердің сақталуы принциптеріне толық сәйкестік;

– atomicity-транзакция бірыңғай логикалық бірлік ретінде қарастырады, оның барлық өзгерістері немесе толығымен сақталады немесе толығымен жойылады;

– Consistency-транзакция деректер базасын бір қарама-қайшы емес күйден (транзакцияны бастау сәтінен) басқа қарама-қайшы емес күйге (транзакцияны аяқтау сәтіне) аударады. Деректер базасының физикалық және логикалық тұтастығының барлық шектеулері орындалатын базаның жай-күйі қарама-қайшы емес деп есептеледі, бұл ретте транзакция ішінде тұтастықтың шектеулерін бұзуға жол беріледі, бірақ бүтіндіктің барлық шектеулері аяқталған сәтте физикалық да, логикалық да сақталуы тиіс;

– Isolation - бәсекелес транзакциялар кезінде деректерді өзгерту нұсқалық жүйе негізінде бір-бірінен оқшауланған;

– Durability-PostgreSQL сәтті транзакциялардың нәтижелері аппаратураның істен шығуына қарамастан қатты дискіге сақталуына қамқорлық жасайды;

PostgreSQL деректерді көп өңдей алады. Жарияланған шектеулер төменде 3.2 - кестеде келтірілген:

Деректер базасының ең үлкен мөлшері	шексіз
Кестелердің ең үлкен өлшемі	32 TB
Жолдардың ең үлкен өлшемі	1.6 TB
Максималды өріс өлшемі	1 GB
Кестедегі жолдардың максималды көлемі	шексіз

Кестедегі бағандардың максималды көлемі	250-1600 баған түріне байланысты
Кестедегі индекстердің максималды көлемі	шексіз

3.2-кесте – Жарияланған шектеулер

Құрастыру кезінде деректерді көбейту туралы алаңдамау үшін автоматты түрде орнатуды шоғырландырамыз. Бірақ, кез-келген дерекқор әкімшісі білетіндіктен, тым үлкен және шектеусіз мүмкіндіктері бар. Кестелерді жасау және индекстерді қосу кезінде сізге дұрыс ойды қолдануға кеңес береміз.

Салыстыру үшін, MySQL және MariaDB 65,535 байт жолының өлшемін шектеу үшін атақты. Firebird сонымен қатар максималды сызық өлшемі ретінде тек 64K ұсынады. Әдетте, деректер көлемі операциялық жүйенің максималды файл өлшемімен шектеледі. PostgreSQL бірнеше кішірек файлдарда кестелік деректерді сақтай алатындықтан, бұл шектеуді айналып өтуге болады. Бірақ тым көп файлдар өнімділікке кері әсерін тигізуі мүмкін. MySQL және MariaDB, PostgreSQL-ке қарағанда, кестедегі көп бағандарды (деректер түріне байланысты 4,096 дейін) және үлкен жеке кестелердің өлшемдерін қолдайды, бірақ Postgrese қолданыстағы шектеулерден асып кету қажеттілігі өте сирек кездеседі.

PostgreSQL көпсалалық (Multiversion Concurrency Control,MVCC) бәсекелі жағдайларда деректердің келісімділігін қолдау үшін пайдаланылады, сонымен қатар, дәстүрлі деректер базасында блоктау үшін қолданылады. MVCC әрбір транзакция дерекқордың көшірмесін (деректер қорының нұсқасын) транзакцияның басталу уақытына, базаның жай-күйі өзгеруі мүмкін екеніне қарамастан, көретінін білдіреді. Бұл транзакцияны бәсекелес транзакция тудыруы мүмкін келісілмеген деректердің өзгерістерінен қорғайды және транзакцияларды оқшаулауды қамтамасыз етеді. Бұғаттаумен салыстырғанда MVCC пайдалануының негізгі ұтысы-MVCC оқу үшін қоятын бұғаттау жазбаға бұғаттаумен қақтығыспайды, сондықтан оқу ешқашан жазбаны бұғаттамайды және керісінше. Бәсекелестік операциялар жазбалары бір жазбамен жұмыс істеген кезде ғана бір-біріне "кедергі жасайды".

PostgreSQL-дің тамаша ерекшеліктерінің бірі жалпыланған іздеу ағашы немесе GiST (жобаның басты беті) болып табылады, ол нақты білім саласындағы мамандарға арнайы деректер түрлерін жасауға мүмкіндік туғызады және деректер қоры саласындағы сарапшылар болмай оларға индекстік қолжетімділікті қамтамасыз етеді. GiST аналогы-қазіргі кезде IBM иеленетін DataBlade технологиясы.

GiST идеясын профессор Беркли Джозеф Хеллерстейн(Joseph m.Hellerstein) ойлап тапты және Generalized Search Trees for Database Systems мақаласында жарияланды. GiST түпнұсқа нұсқасы POSTGRES патч ретінде Беркли әзірленген және кейінірек PostgreSQL-да инкорпорацияланған. Кейінірек, 2001 жылы код ауыспалы ұзындықтың кілттерін, көп атрибутты индекстерді қолдау және NULL-мен қауіпсіз жұмыс істеу үшін қатты

модификацияланған, сондай-ақ бірнеше қате түзетілді. Қазіргі уақытта GiST негізінде көптеген қызықты кеңейтулер жазылған, соның ішінде:

– tsearch2 толық мәтінді іздеу модулі;

– иерархиялық деректермен жұмыс істеуге арналған модуль (tree-like);

– intarray бүтін сандар массивімен жұмыс істеуге арналған модуль;

Contrib / subdirectory ішіндегі PostgreSQL үлестірімі толық мәтінді іздеу, XML-мен жұмыс жасау, математикалық статистика функциялары, қателермен іздеу, криптографиялық модульдер және т.б. сияқты әртүрлі қосымша функционалдылықты іске асыратын көптеген шамамен ұзын саны 80-нен астам модульдер бар. Сондай-ақ, әкімшілік жұмыс үшін mysql, oracle арқылы көші-қонды жеңілдететін коммуналдық қызметтері де қарастырылған.

PostgreSQL деректер түрлерінің бай жиынтығы:

– SQL стандарттарында белгіленгендей символдық типтері (character(n)) және іс жүзінде шексіз ұзындығы бар text түрі;

– Numeric түрі ғылыми және қаржылық қосымшаларда өте қажет еркін дәлдікті қолдайды;

– SQL:2003 стандартына сәйкес массивтер;

– үлкен нысандар (Large Objects) деректер қорында 2GB өлшемдегі екілік деректерді сақтауға мүмкіндік береді;

– геометриялық түрлері (point, line, circle,polygon, box) жазықтықта кеңістіктік деректермен жұмыс жасауға мүмкіндік береді;

– GIS (GIS) PostgreSQL типтері PostgreSQL кеңейтілуінің дәлелі болып табылады және үш өлшемді деректермен тиімді жұмыс істеуге мүмкіндік береді;

– желі түрлері (Network types) IPv4, IPV6 және cidr (Classless Internet Domain Routing) блоктары және macaddr үшін inet деректер түрлерін қолдайды;

– композиттік түрлер (composite types) бір немесе бірнеше қарапайым түрлерді біріктіріп, пайдаланушыларға күрделі объектілермен айла-шарғы жасауға мүмкіндік береді;

– уақыт түрлері (timestamp, interval, date, time) өте үлкен дәлдікпен жүзеге асырылған;

– Serial және bigserial псевдотиптері бүтін сандардың реттілігін ұйымдастыруға мүмкіндік береді;

Оңай пайдалану әрқашан әзірлеушілер үшін маңызды фактор болып табылады. Psql утилитасы (дистрибутивке кіреді) деректер базасымен жұмыс істеу үшін ыңғайлы интерфейсті ұсынады, SQL бойынша қысқаша анықтамалықты қамтиды, командаларды енгізуді жеңілдетеді (қайталау үшін көрсеткілерді және кеңейту үшін табуляторды пайдалана отырып), сұраулардың тарихы мен буферін қолдайды, сондай-ақ интерактивті режимде де, ағын режимінде де жұмыс істеуге мүмкіндік береді.

– phpPgAdmin (GPL лицензиясы) веб шолғыш арқылы PostgreSQL кластерін басқару мүмкіндігін ұсынады.

– pgAdmin III (GNU Artistic license) PostgreSQL мәліметтер базасымен жұмыс істеу үшін ыңғайлы интерфейсті ұсынады және Linux, FreeBSD және Windows 2000/XP астында жұмыс істейді.

– PgEdit-бағдарламаларды әзірлеу үшін бағдарламалық орта және SQL редактор, Windows және Mac үшін қол жетімді.

Деректердің сақталуы және оның қауіпсіздігі кез келген ДББЖ маңызды аспектісі болып табылады. PostgreSQL ол 4 қауіпсіздік деңгейлері қамтамасыз етеді:

– PostgreSQL артықшылықты пайдаланушымен іске қосуға болмайды жүйелік контекст.

– SSL,SSH клиент пен сервер арасында трафикті шифрлау-желілік контекст.

– хост немесе IP мекенжайы/ішкі желі деңгейінде күрделі аутентификация жүйесі. Аутентификация жүйесі парольдерді, шифрланған парольдерді, Kerberos, IDENT және қосылатын аутентификациялық модульдердің тетігін пайдалана отырып қосыла алатын басқа да жүйелерді қолдайды.

– Әрбір пайдаланушы үшін объектілердің атауларын оқшаулауды қамтамасыз ететін схемамен бірге, PostgreSQL бай және икемді инфрақұрылым ұсынады.

PostgreSQL көптеген нұсқалары бар. Нысан-реляциялық модельді қолдану арқылы жасалған, ол күрделі құрылымдарды және бекітілген және пайдаланушы анықтайтын деректер түрлерінің кең ауқымын қолдайды. Бұл деректердің өнімділігін арттырады және деректер тұтастығына қатысты сенімділікке ие. Бұл мақалада зерттелетін барлық қосымша деректерді сақтау функцияларының қажет болуы мүмкін емес, бірақ мұқтаждық тез өсетіндіктен, барлығын сіздің қолыңызда ұстаудың сөзсіз артықшылығы бар.

3.2 Android платформасы. Android studio ортасы

Бүгін өзін "ақылды" техникамен қоршауды ұнатпайтын адамды табу қиын. "Телефонсыз-қолсыз" сөзі жиі естіледі, ал плеер, ноутбук немесе басқа да танымал гаджетсіз өмірді елестету мүмкін емес. Сондықтан қазіргі заманғы электроника нарығында пайда болатын жаңалықтар туралы әркім білуі тиіс. Міне, мысалы, телефондағы плей маркет пайдаланатын Android платформасы екені туралы әркім біле білмейді.

Толығырақ тоқтала кететін болсақ, Android - бұл ұялы құрылғыны (телефон, планшет, смартфон) өңдей алатын операциялық жүйе. Атап айтқанда, Android операциялық жүйелерге арналған бағдарламалық стек, аралық деңгейдегі бағдарламалық қамтамасыз ету (middleware), сондай-ақ базалық қосымшалардан тұратын мобильді қосымшалар болып табылады.

Android платформасы Linux ядросына негізделген. Бұл 2005 жылы Google компаниясы өзінің еншілес компаниясы бола отырып, Android Inc сатып алды және мобильді құрылғылар үшін сол атау платформаларын шығара бастады. Содан бері платформа тек дамып келеді. Android көбінесе өзінің бағдарламасының жаңа нұсқаларын шығарады. Әр жаңа нұсқаның атауының

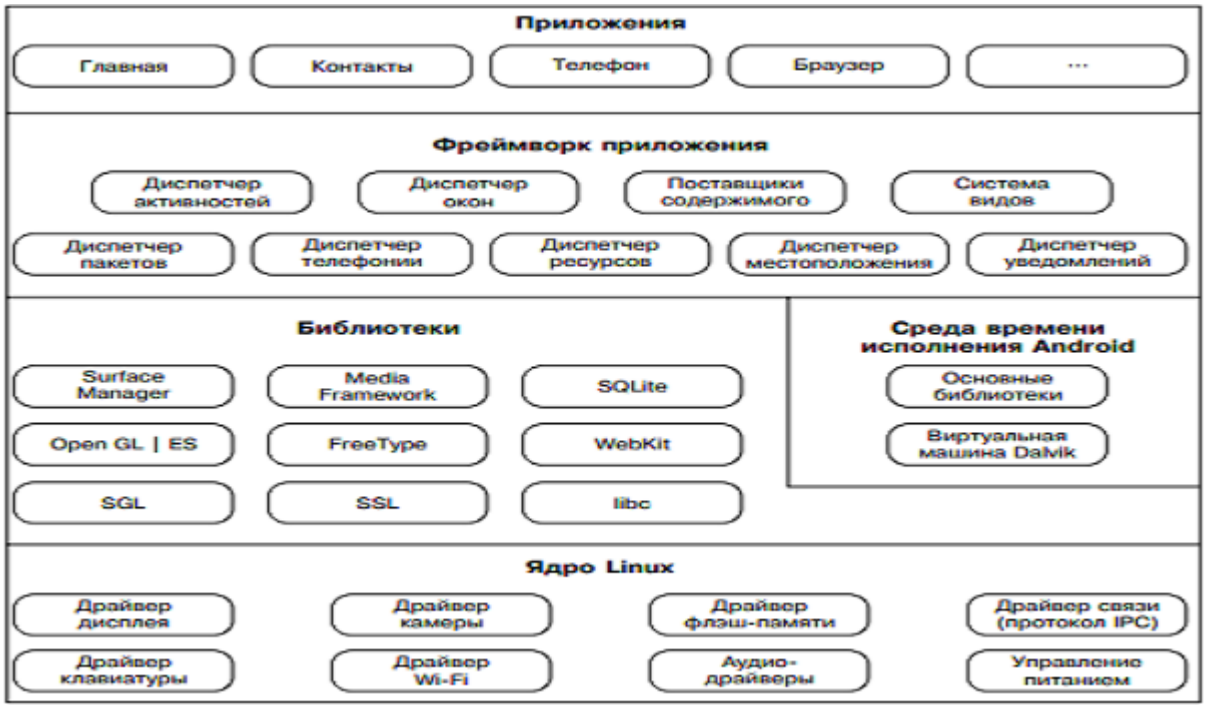
алғашқы әріптері латын әліпбиінің әріптеріне сәйкес келеді. Бүгінгі таңда Android платформасы әлемдегі танымалдылыққа ие, iPhone үшін әзірленген iOS операциялық жүйесінен кейін ғана.

Андроид платформасы дегеніміз не екенін түсіну оңай, енді ол не үшін қажет екенін түсіну керек. Операциялық жүйе-бұл адамның барлық командаларын орындау үшін қажет кез келген "электрондық құрылғының миы". Тиісінше, Android-бұл ұялы құрылғының ішінде отырған виртуалды робот, ол осы құрылғының ішінде болып жатқан барлық процестерді орындауға жауап береді. Бұл платформаның артықшылығы Android ыңғайлы және өте түсінікті интерфейсі, сондай-ақ бірнеше қосымшаларды бірден іске қосуға және баптаулармен тәжірибе жасауға мүмкіндік беретін икемді және мультизатты жүйесі бар. Бұдан басқа, Android платформасы үшін арнайы жасалған көптеген қосымшалар осы жүйенің бақытты пайдаланушыларына шын мәнінде шексіз мүмкіндіктер береді. Бұл қосымшалардың көмегімен планшетті сатып алу үшін төлеуге, фотосуреттер жасауға, фильмдер көруге немесе кітаптарды оқуға болады. Бұл платформа шығармашылық адамдар үшін жасалған деген қорытынды жасауға болады, себебі екі бірдей ұялы құрылғылар мүлдем басқаша көрінуі мүмкін. Андроид адам өмірін барынша жайлы ете отырып, әрқашан өзімен бірге барлық – жеке фитнес-жаттықтырушыны, дәрігерді, ойыншықты немесе теледидарды алып жүруге мүмкіндік береді.

Жалпы Android архитектурасын алты деңгейге бөлуге болады:

– жабдық деңгейі;
– Linux ядросының деңгейі;
– нативті кітапханалар деңгейі;
– Android орындау ортасының деңгейі;
– қосымшалар қаңқасының деңгейі (Application Framework);
– қосымшалар деңгейі.

3.3 - суретте Android платформасының архитектурасы көрсетілген. Linux ядросы және драйверлер жиынтықтары Android платформасының орталығы операциялық жүйенің негізгі міндеттерін шешуге және жабдықтармен өзара іс - қимыл жасауға жауап беретін Linux операциялық жүйесінің ядросы (Android нұсқасына байланысты 2.6.25-3.0.31 нұсқасы) болып табылады.



3.3 – сурет - Android платформасының архитектурасы

Бұл деңгейде процестерді басқару және жадты тарату, файлдық жүйе мен қауіпсіздікті басқару үшін негізгі қызметтер, сондай-ақ желілік функцияларды басқару үшін қызметтер (желілік стек) орналасқан. Тек ядро деңгейінде ғана драйверлерді теру арқылы құрылғы жабдықтарымен тікелей өзара әрекеттеседі. Ядро деңгейінің негізгі компоненттері:

– процессаралық өзара іс-қимыл драйвері (IPC Driver).

– қуат тұтынуды басқару драйвері (Android Power Management).

– мобильді құрылғының құрамына кіретін жабдықтарға арналған драйверлер жиынтығы.

Android операциялық жүйесі Linux ядросына негізделген болса да, олардың арасында (таза түрде Android және Linux) кейбір айырмашылықтар бар. Мысалы, Android жады бөлу механизмдерінің, процестер арасындағы өзара әрекеттесулері және т.б. бар.

Дипломдық жоба Android платформасында Android studio даму ортасында Java программалау тілінде жазылғандықтан Android studio даму ортасына да осы тарауда тоқталайық. Ал, программалау тілі туралы жеке тарауда қарастырылған жөн.

Android Studio-бұл 2013 жылдың 16 мамырында Google I/O конференциясында анонсталған Android платформасымен жұмыс істеу үшін біріктірілген даму ортасы (IDE).

IDE 2013 мамыр айында жарияланған 0.1 нұсқасынан бастап, содан кейін 2014 жылдың маусым айында шығарылған 0.8 нұсқасынан бастап бета-тестілеу сатысына өтті. 1.0 бірінші тұрақты нұсқасы 2014 жылдың желтоқсанында шығарылды, сонда Eclipse үшін Android Development Tools (ADT) плагинін қолдау тоқтатылды.

JetBrains компаниясынан IntelliJ IDEA бағдарламалық жасақтамасына негізделген Android Studio - Android қосымшаларын әзірлеудің ресми құралы. Бұл жұмыс ортасы Windows, OS X және Linux үшін қол жетімді. 17 мамыр 2017, Google I / O жыл сайынғы конференциясында, Google Java және C++қосу Android платформасына арналған ресми бағдарламалау тілі ретінде Android Studio-да қолданылатын Kotlin тілін қолдауды жариялады.

Жаңа функциялар Android Studio бағдарламасының әрбір жаңа нұсқасында пайда болады. Қазіргі уақытта келесі функцияларға қол жеткізу мүмкіндігі бар:

– Жетілдірілген орналасу редакторы: WYSIWYG, Drag-and-Drop функциясын пайдаланып UI компоненттерімен жұмыс істеу мүмкіндігі, бірнеше экрандық конфигурацияда орналасуды алдын-ала қарау функциясы.

– Grader негізіндегі қосымшалар құрастыру.

– Құрылымдардың әртүрлі түрлері және бірнеше .apk файлдарын жасау.

– Кодты рефакторинг..

– Статикалық кодын талдаушы (Lint), ол өнімділіктің, нұсқаның үйлесімділігін және т.б. мәселелерін табуға мүмкіндік береді.

– Енгізілген ProGuard және қосымшаға қол қою утилитасы.

– Негізгі жоспарлар мен Android компоненттері үшін үлгілер.

– Android Wear және Android TV үшін қосымшаларды әзірлеуге қолдау көрсету.

– Google Cloud Messaging және App Engine қызметтерімен біріктіруді қамтитын Google Cloud Platform үшін жергілікті қолдау.

Android Studio 2.1 Android N Preview SDK-ді қолдайды, яғни әзірлеушілер жаңа бағдарламалық жасақтама платформасына арналған бағдарлама жасау үшін жұмыс істей алады.

Android Studio 2.1 нұсқасының жаңартылған нұсқасы жаңартылған Джек компиляторымен жұмыс істей алады, сондай-ақ Java 8 және Improved Instant Run мүмкіндігі жақсартылған қолдауды алады .

Linux-платформа құралдарымен 23.1.0 бастап, тек 64-бит.

Android Studio 3.0 нұсқасында JetBrains IDE негізіндегі Kotlin тіл құралдары стандартты ретінде қосылады.

3.3 Java программалау тілі

Java - Android Studio дамыту ортасы қолдайтын ресми бағдарламалау тілі. Stackoverflow ресурсының жыл сайынғы сұрауының мәліметтері бойынша, 2018 жылы Java ең танымал бағдарламалау тілдерінің бестігіне кірді.

Дипломдық жоба осы негізге сүйене отырып программалау тілінің ыңғайлығын ескеріп Java программалау тілінде жазылды.

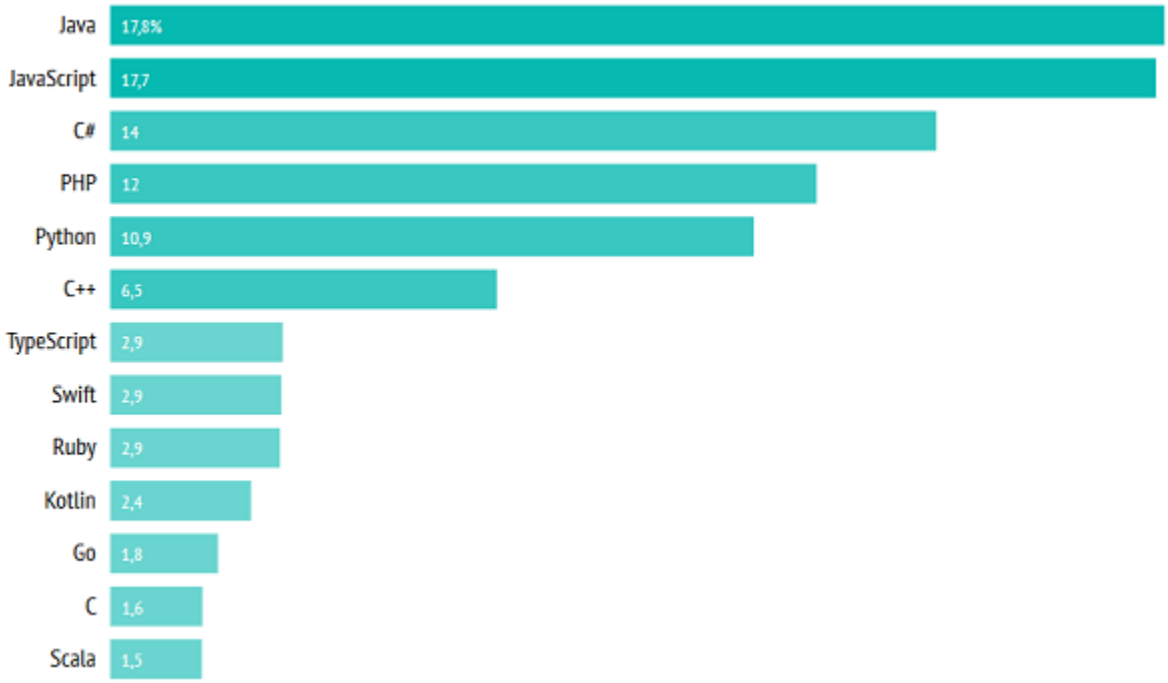
Жалпы, Java - Sun Microsystems (кейінірек Oracle сатып алған) тарапынан әзірленген объектілі-бағытталған бағдарламалау тілі. Java қосымшалары әдетте

арнайы байт кодқа айналдырылады, сондықтан Java виртуалды машинасын пайдаланып кез келген компьютерлік архитектурада жұмыс істей алады. Ресми жарияланған күні - 1995 жылғы 23 мамыр. 2019 жылы Java ең танымал бағдарламалау тілдерінің бірі болып табылады.

Java тілі Android операциялық жүйесіне арналған мобильді қосымшаларды жасау үшін белсенді қолданылады. Бұл ретте бағдарламалар Dalvik виртуалды машинасын пайдалану үшін стандартты емес байт-кодқа компиляцияланады (Android 5.0 Lollipop бастап Виртуалды машина ART-ға ауыстырылды). Мұндай компиляция үшін Google компаниясы әзірлеген Android SDK (Software Development Kit) қосымша құралы қолданылады.

Қосымшаларды әзірлеуді Android Studio, NetBeans, Eclipse ортасында, сонымен қатар Android Development Tools (ADT) немесе IntelliJ IDEA плагинін пайдалана отырып жүргізуге болады. JDK нұсқасы 5.0 немесе одан жоғары болуы керек.

2014 жылдың 8 желтоқсанында Android Studio Google компаниясы Android операциялық жүйесі арқылы әзірлеудің ресми ортасы деп танылды. 3.4 – суретте Android платформасында қолданылатын программалау тілдерінің салыстырмалы рейтингі көрсетілген.



3.4 - сурет - Программалау тілдерінің салыстырмалы рейтингі

Java бағдарламалары Java индексі арқылы Java кодтарына аударылады, ол Java-ның Virtual Machine (JVM) -а бағдарламасы арқылы жүзеге асырылады, ол байт-кодты өңдейді және аппараттық құралдарды аудармашы ретінде жібереді.

Осы бағдарламаны орындау әдісінің артықшылығы операциялық жүйеден және жабдықтан Java-қосымшаларын іске қосуға мүмкіндік беретін байт кодтың толық тәуелсіздігі болып табылады, ол үшін тиісті виртуалды машина бар. Java

технологиясының тағы бір маңызды ерекшелігі - бағдарламаны орындау виртуалды машинамен толығымен басқарылатын икемді қауіпсіздік жүйесі. Бағдарламаның белгіленген рұқсаттарынан асатын кез келген операциялар (мысалы, деректерге рұқсатсыз кіру немесе басқа компьютерге қосылу әрекеті) дереу үзіліс жасайды.

Жиі виртуалды машина тұжырымдамасының кемшіліктері өнімділіктің төмендеуін қамтиды. Бірқатар жақсартулар Java бағдарламаларын орындау жылдамдығын арттырды:

– машиналық кодта сыныптық нұсқаларын сақтау мүмкіндігі бар бағдарламаның (JIT-технологиясы) тікелей машиналық кодқа жан-жақты аудару технологиясын қолдану,

– стандартты кітапханаларда платформа кодын (тектік-код) кеңінен қолдану,

– жеделдетілген байт-кодты өңдеуді қамтамасыз ететін жабдық. (мысалы, Jazelle технологиясы, APM архитектурасының кейбір процессорлары қолдайтын)

3.4 Django REST тіреgі

Қазіргі таңда жаңа даму ортасын шығаруға көптеген кітапханалар, тіректер пайдалынады. Тірек – бұл программалық әзірлеушілер ыңғайлы,жұмысты жеңілдетуге арналады. Дипломдық жобада мобильді қосымшаға веб-API құру үшін Django REST ортасын қолданылды.

Ал, Django - бұл күрделі веб-сайттарды және Python бағдарламалау тілінде жазылған веб-қосымшаларды әзірлеу үшін қолайлы бай бағдарламалық жасақтама.

Атаудан түсінікті, Django REST Framework — бұл rest (Representational State Transfer) идеологиясын қолдайтын Django үшін жеңіл фреймворк-күйдің репрезентативті берілуі. Бұл фреймворк пайдалану деректер базасына сұраныстарды оңай стандарттауға және бір уақытта біздің сайтымыздың RESTful WEB API құруға мүмкіндік береді. Артықшылығы:

– Django-Class Based Views жаңа функциясын пайдалану арқылы ресурстарды көрсету;

– ModelResources қолдау және кіріс деректерін тексеру;

– Қосылатын парсерлердің, бейнелердің, авторландырудың және қол жеткізу құқықтарының болуы-бәрі оңай теңшеледі;

– Http-сұраулар тақырыбында Access қолдану арқылы материалдың түрін көрсету;

– Тексеру мүмкіндігі бар нысандарды қолдау;

Жалпы, Django Rest Framework (DRF) — бұл жоба үшін икемді және қуатты API құру үшін стандартты Django үлгілерімен жұмыс істейтін кітапхана. DRF API-і 3 қабаттан тұрады: сериализатор,көрініс және маршрутизатор.

– Сериализатор: деректер базасында сақталған және Django модельдерінің көмегімен анықталған ақпаратты API арқылы оңай және тиімді берілетін форматқа түрлендіреді.

– Көрініс (ViewSet): API арқылы қол жетімді болатын функцияларды (оқу, құру, жаңарту, жою) анықтайды.

– Маршрутизатор: әрбір түрге қатынайтын URL мекенжайын анықтайды. Django модельдері базада сақталған деректерді интуитивті түрде ұсынады, бірақ API ақпаратты аз күрделі құрылымда беруі тиіс. Деректер Model кластарының даналары ретінде ұсынылғанымен, оларды API арқылы жіберу үшін JSON форматына көшіру қажет.

DRF сериализаторы бұл түрлендіруді шығарады. Пайдаланушы ақпаратты (мысалы, жаңа дананы жасау) API арқылы жібергенде, сериализатор деректерді алады, оларды тексереді және Django модельдің данасына қосуы мүмкін бірдеңеге түрлендіреді. Сол сияқты, пайдаланушы API арқылы ақпаратқа жүгінгенде, тиісті даналар оларды JSON ретінде пайдаланушыға оңай берілуі мүмкін форматқа түрлендіретін сериализаторға беріледі.

Сериализаторлар DRF керемет икемді және қуатты компоненті. Модельге сериализаторды қосу ең кең тараған болса да, сериализаторлар белгілі бір параметрлерге сәйкес API арқылы кез келген Python деректер құрылымын жасау үшін пайдаланылуы мүмкін.

Сериализатор ақпаратты екі бағытта да талдайды (оқу және жазу), ал ViewSet - қол жетімді операциялар анықталған код. Ең көп таралған ViewSet-ModelViewSet, ол келесі кірістірілген операциялар бар:

– дананы жасау: create ();

– дананы алу / Оқу: retrieve ();

– дананы жаңарту (барлық немесе тек таңдалған өрістер): update () немесе partial_update ();

– дананы жою / жою: destroy ();

– даналардың тізімі (әдепкі беттерге бөлінген): list ();

Қорыта келгенде, соңында маршрутизаторлар API-дың жоғарғы деңгейін ұсынады. "Тізімдер", "бөлшектер" және "өзгерту" түріндегі шексіз URL-адрестерін жасауды болдырмау үшін DRF маршрутизаторлары әрбір ViewSet үшін бір жолға осы түрге қажетті барлық URL-адрестерін біріктіреді.

3.5 Retrofit 2.0 кітапханасы

Android даму әлемінде көптеген қызықты кітапханалар бар, олар әзірлеушілердің жұмысын жеңілдетуге ,сонымен қатар, ыңғайлы әрі икемді болып есептелінеді. Дипломдық жобада Square компаниясы ұсынатын Retrofit 2.0 версиядағы кітапханасын қолданылды. Ол клиент-серверлік қосымшаларда API - мен жұмыс істеу үшін таптырмас құрал болып табылады. 5 жыл бұрын

Android әзірлеушілері желіге кері шақырулар, AsyncTask-тармен және басқа да" төмен деңгейлі " заттары бар кодтар үшін тау төңкерердей жұмыс жасауға тура келді.Осылайша, Square компаниясы осындай тамаша кітапхананы шығарды — Retrofit.

Retrofit-Java және Android үшін REST клиент. Ол rest негізіндегі веб-сервис арқылы JSON (немесе басқа құрылымдалған деректер) оңай алуға және жүктеуге мүмкіндік береді. Retrofit-те сіз деректерді сериализациялау үшін қандай конвертерді баптайсыз. Әдетте JSON үшін GSon пайдаланылады, бірақ сіз XML немесе басқа хаттамаларды өңдеу үшін өз конвертерлерін қосуға болады. Retrofit-те http сұраулары үшін OkHttp кітапханасы қолданылады.

Retrofit жұмыс істеу үшін Сізге келесі үш сынып қажет:

– Model JSON моделі ретінде пайдаланылатын класс;

– Ықтимал HTTP операцияларды анықтайтын интерфейстер;

– Retrofit класы. Builder-http операцияларына арналған соңғы URL нүктесін анықтау үшін интерфейс және API Builder пайдаланатын дана;

Интерфейстің әрбір әдісі ықтимал API шақыруларының бірі болып табылады. Ол сұрау түрін және салыстырмалы URL мекенжайын көрсету үшін HTTP аннотациясы (GET, POST және т.б.) болуы керек. Қайтарылатын мән күтілетін нәтиже түрі бар Call-объектідегі жауапты аяқтайды.

2013 жылы әзірлеушілер үшін Volley және Retrofit тамаша кітапханалары пайда болды. Retrofit REST Api - мен жұмыс істеу үшін арнайы құрылған, ал volley-желімен жұмыс істеуге арналған жалпы жоспардағы кітапхана. Соңғы кітапхана клиент-серверлік қосымшаларды әзірлеу кезінде жалпыға танылған стандарт болды.

Retrofit, басқа құралдармен салыстырғанда, бірнеше негізгі артықшылықтарды бөлуге болады:

1) кез келген сұрауларды орындау үшін толық функционал ұсынатын өте ыңғайлы және қарапайым интерфейс;

2) икемді теңшеу — кез келген клиентті сұрау үшін, json және т. б. талдау үшін кез келген кітапхананы пайдалануға болады.;

3) JSON-а парсингін өз бетінше орындау қажеттілігінің болмауы-бұл жұмысты Gson кітапханасы (Gson ғана емес);

4) нәтиже мен қателерді ыңғайлы өңдеу;

5) Rx қолдау, бұл да бүгін маңызды фактор.

Retrofit бағдарламасын үш жолмен қосуға болады: Gradle, Maven, Jar. Төменде әрбір әдісті қосу жолдарын туралы төменде тоқталып өтіледі.

Көптеген жағдайларда Android қолданбаларын құрастыру үшін Gradle құралын пайдаланылады.

Build файлына қосылу үшін .gradle бағдарлама модульдері dependencies бөліміне жолды кірістіру:

Compile 'com. squareup.retrofit2:retrofit:2.1.0'

Егер де сіз Maven тәуелділік және жинау жүйесін пайдалансаңыз, онда тәуелділік фрагменті осылай көрінеді:

<dependency>

<groupId> com.squareup.retrofit2 </groupId>
<artifacId>retrofit</artifactId>
<version>2.1.0</version>
</dependency>

Jar әдісі онша көп танымал тәуелділік болып табылмайды, алайда кей әзірлеушілер оны қолданады. Jar-файлды (сілтемені) ресми сайттан жүктеп ,оны libs қалтасына жылжытамыз.Сонымен қатар, кітапханадан басқа JSON және RecyclerView-v7 парсерлері қажет:

compile 'com.squareup.retrofit2:converter-gson:2.1.0'
compile 'com.android.support:recyclerview-v7:25.0.0'

Retrofit post орындай алатын толық REST-клиент жасауға мүмкіндік береді, GET, PUT, DELETE. Сұраудың түрі мен басқа да аспектілерін белгілеу үшін аннотациялар қолданылады. Мысалы, GET сұрау талап етілетіндігін белгілеу үшін, бізге POST сұрау үшін GET әдісінің алдында жазу керек,және т. б.

ҚОРЫТЫНДЫ

Дипломдық жоба орындау барысында, қазіргі заманауи мобильді технологиялардың адам өмірінің әртүрлі салаларына қарқынды түрде еніп жатқаны және мобильді қосымшаларды қолдану айтарлықтай тиімді екенін баса көрсетіліп,тоқталып өтілді. Сонымен қатар,мобильді телефондар мен гаджеттерді өндіруші компаниялардың басым бөлігі Android операциялық жүйесіне арнап өндіретіндігі, тұтыну бағасы қолжетімділігі мен ыңғайлы екендігін алға тарта отырып, маңыздылығы айқындалды. Осыған орай, Қ.И.Сатбаев атындағы университетке жаңа бөлімдік кадр мамандар қабылдауға арналған Android платформасында, Java программалау тілінде мобильді қосымша және серверге API құрылды. Қосымша веб-сервермен байланысы Django REST API көмегімен жүзеге асырылды.

Мобильді қосымшаның жұмыс істеу принциптерін көрсету үшін пәндік аймақ зерттеліп, UML тілінде жүйе жұмысының жалпы моделі құрылды және жалпы сервер мәліметтер қоры жобаланды.

Мобильді қосымшаға қойылған талаптарға сәйкес қолданушыларға түсінікті әрі ыңғайлы интерфейс құрылды. Қауіпсіздікті қамтамасыз ету үшін қолданушыларға аутентификация және авторизация жүргізу жүзеге асырылды. Сонымен қатар,веб-серверден қажетті мәліметтерді алу үшін API арқылы мобилді қосымшаның серверге сұраныс жасау мәселелері де жүзеге асырылды.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТЕР

1 Бағдарлама құрушыларға арналған ақпараттық сайт [Электрондық ресурс]. – URL:https://developer.android.com

2 Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя, 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс,2006. – 496 с.: ил. – С. 28-444

3 Кузин А.В. Базы данных : учеб.пособие для студ. высш. учеб.заведений / А.В.Кузин, С.В.Левонисова. – 5-е изд., испр. – М.: Издательский центр «Академия»,2012.

4 Ричардсон Л., Амундсен М. RESTful Web APIs 1-е изд.: USA, 2013.

5 Dar U., Jim M., Kirk R., PostgreSQL Server Programming, Second Edition, 2013.

6 Professional Postgres // Сайттың электрондық нұсқасы https://postgrespro.ru/education/courses/DBA1 /

7 Маклин Д., Хэшими С. Google Android: программирование для мобильных устройств 2012г.

8 Сатия Коматинени, Дэйв Маклин. Android 4 для профессионалов. Создание приложений для планшетных компьютеров и смартфонов = Pro Android 4

9 Википедия // Сайттың электронды нұсқасы https://ru.wikipedia.org/wiki/API /

10 Википедия // Сайттың электронды нұсқасы https://ru.wikipedia.org/wiki/PostgreSQL

11 Википедия // Сайттың электронды нұсқасы https://ru.wikipedia.org/wiki/Android

12 Эйблсон Ф., Android платформасы үшін өңдеулерге кіріспе. (Электронды ресурс). – Мақалаға қол жеткізу тәртібі:

13 Android 2. Программирование приложений для планшетных компьютеров и смартфонов / Рето Майер . - СПб.: Санкт-Петербург, 2011. - 672 с.

А қосымшасы
(міндетті)

«QuizEmployee» мобильді қосымшасын құруға арналған
техникалық тапсырма

А.1 Кіріспе

Бүгінгі таңда көптеген адамдар ақпарат алмасу үшін ЖК немесе ноутбуктар емес, керісінше осы аталған құрылғыларға тең келетін замануи ұялы телефондарды қолданады. Нәтижесінде, мобильді қосымшаларды құру қазіргі таңда маңыздырақ болуда – бүгінде компьютерлік жүйелерге бағдарламалық қамтамасыздандыруды құруға қарағанда, мобильді қосымшаларды құру өте танымал және де көптеген адамдар осы салада қызмет етуді жөн көреді. Сондықтан оқу орнына қажетті мобильді қосымша құру қазіргі заманның өзекті мәселесі. «QuizEmployee» мобильді қосымшасы толығымен Қ.И.Сәтбаев атындағы университетіне кадр қабылдауға арналып жасалынды.

А.1.1 Жобаның мақсаты мен міндеті

«“QuizEmployee” проектісінің түпкі мақсаты, Қ.И.Сәтбаев атындағы университетіне кадр қабылдауға (оқытушылар,асистент,лектор) арналған жылдам әрі ыңғайлы,қағаз бастылықтан арылту мақсатындағы мобильді қосымша болып табылады. Жобаның жүзеге асу барысында барлық тапсырмалар мен талаптар әділ түрде орындалып,барлығы базада сақталынады. “QuizEmployee” проектісінде қолданылатын технологиялар:

- Java программалау тілі;
- PostgreSQL деректер қорымен басқару;
- RetroFit 2.0 кітапханасы;
- Android платформасында;

А.1.2 Қолдану саласы

Қолдану саласы – ұялы телефонмен қолдана білетін кез-келген адам бұл мобильді қосымшаны қолдана алады.

А.1.3 Анықтамалар, терминдер және қысқартулар

Анықтамалар, терминдер және қысқартулар А.1 – кестеде көрсетілген.

А.1-кесте – Анықтамалар, терминдер және қысқартулар

Терминдер немесе қысқартулар	Анықтамалар
Интернет	Интернет сөзі ағылшынша (байланысқан желілер) екі сөздің туындысы. Интернет ұғымы *ақпараттық және есептеу техникалары ресурстарының әлемдік телекоммуникациялық желісі д*егенді білдіреді.
Java	бірнеше компьютер бағдарламалық жасақтама өнімдерінен құралған өнім және Sun Microsystems мекемесі жазған спецификация (қазір Oracle Corporation мекемесімен бірге), барлығы қосыла келіп қосымшалар бағдарламалық жасақтарын дайындауға қолданысын табатын жүйені құрайды, бұл жүйе кросс-платформалы есептеу орталықтарында қолданысын таба береді.
Android	Android ықшамды (желілі) операциялық жүйе. Ол Linux ядросының негізінде жасалған және коммуникаторлар, планшетті компьютерлерге, санды ойнатқыштарға, қолсағаттарға, нетбуктар мен смартбуктарға арнап жасалынған.
UML	UML – бұл бағдарламалық жүйелерді ерекшелендіру, бұрыштама қою, конструкциялау және құжаттамалау, сондай-ақ модельдер бизнесі мен өзге де бағдарламалық емес жүйелердің тілі болып табылады.

А.2 Жалпы сипаттамасы

Дерекқор жұмысы клиент-сервер технологиясына негізделген. Мобильді қосымша қолданушы бөлімінен тұрады. Қолданушы мобильді қосымшаларды көріп, қажетті жүйелерді жүргізе алады.

А.2.1 Пайдаланушы интерфейстер

Ақпараттық жүйенің ыңғайлы және түсінікті бағдарламалық интерфейстен тұруы мобильді қосымша үшін маңызды рөлге ие. Қолданушылардың қосымшаны қолдануы осы бағдарламалық интерфейске тікелей қатысты.

А.2.2 Аппаратттық интерфейстер

Компьютерлерге қойылатын жалпы талаптар:
– процессор – P4 және жоғары, Celeron;
– оперативті жады – 512 MB;
– диск тұлғалы кеңістік – ~ 52 MB + қолданушылар файлдарын сақтауға қажет орын;
– Internet желісіне қатынау;
– қатты диск 40 Gb; 1024×768 рұқсатта жұмыс қолдайтын видеокарта;
– пернетақта, манипулятор тышқан.

А.2.3 Программалық интерфейстер

Жобаға қатысты бағдарламалық компоненттер:

– Windows 8,10 операциялық жүйесі;
– Android SDK бағдарламасы ;
– PostgreSQL бағдарламасы;
– Java бағдарламасы.

Қолданушы бағдарлама компоненттер: кез келген операциялық жүйе, ұялы телефон, интернет желісіне қосыла алу, әр түрлі браузер.

А.2.4 Коммуникациялық интерфейстер

Пайдаланушылардың компьютерлерінде интернетке қатынауға бар модем құралдары немесе жылдамдығы 100 Мб/с кем емес желілі карта болуы тиіс. Сервер мен клиенттерді байланыстыратын TCP/IP протоколы қолданылады.

А.2.5 Жады бойынша шектеулер

Серверлік және клиенттік машиналар үшін жалпы жады бойынша қойылатын талаптар, оларда орналасқан жұмысқа қажетті программалық компоненттердің қойылатын талаптармен анықталады.

А.2.6 Адаптация бойынша талаптар

«QuizEmployee» мобильді қосымшасы Android және Windows платформаларында жұмыс істей алады.

Б қосымшасы
(міндетті)

Бағдарламаның мәтіні

1. *API.java беті*

```
import kz.seidali.suquiz.Model.Kategory;
import kz.seidali.suquiz.Model.LoginResponse;
import kz.seidali.suquiz.Model.Questions;
import kz.seidali.suquiz.Model.RegisterResponse;
import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.POST;
public interface Api {
   @FormUrlEncoded
   @POST("auth/login/")
   Call<LoginResponse> login(
         @Field("username") String username,
         @Field("password") String password
   );
   @FormUrlEncoded
   @POST("users/register/")
   Call<RegisterResponse> createUser(
         @Field("first_name") String firstName,
         @Field("last_name") String lastName,
         @Field("username") String username,
         @Field("email") String email,
         @Field("password") String password
   );
   @GET("categories/")
   Call<Kategory> getAllCategories();
   @GET("questions/")
   Call<Questions> getAllQuestions();
}
```

2. *RETROFIT.java беті*

```
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
public class RetrofitClient {
```

```
    public static final String BASE_URL = "https://suquiz.pythonanywhere.com/api/";
    private static RetrofitClient mInstance;
    private Retrofit mRetrofit;
    public RetrofitClient() {
        mRetrofit = new Retrofit.Builder()
                .baseUrl(BASE_URL)
                .addConverterFactory(GsonConverterFactory.create())
                .build();
    }
    public static synchronized RetrofitClient getInstance(){
        if (mInstance == null) mInstance = new RetrofitClient();
        return mInstance;
    }
    public Api getApi() {
        return mRetrofit.create(Api.class);
    }
}
```

3. *RegisterResponse.java беті*

```
import com.google.gson.annotations.SerializedName;
public class RegisterResponse {
    @SerializedName("username")
    private String username;
    public String getUsername() {
        return username;
    }
}
```

4. *LoginResponse.java беті*

```
import com.google.gson.annotations.SerializedName;
public class LoginResponse {
    @SerializedName("key")
    private String token;
    @SerializedName("user")
    private User mUser;
    public String getToken() {
        return token;
    }
    public User getUser() {
        return mUser; }
```

```
 public LoginResponse(String token, User user) {
        this.token = token;
        mUser = user;
    }
}
```

Б қосымшасының жалғасы

5. *MainActivity.java* беті

```
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import kz.seidali.suquiz.API.RetrofitClient;
import kz.seidali.suquiz.Model.LoginResponse;
import kz.seidali.suquiz.R;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
public class MainActivity extends AppCompatActivity implements View.OnClick-
Listener {
    private EditText usernameEditText;
    private EditText passwordEditText;
    private Button login;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        usernameEditText = findViewById(R.id.loginEditText);
        passwordEditText = findViewById(R.id.passwordEditText);
        findViewById(R.id.signInBtn).setOnClickListener(this);
        findViewById(R.id.registerBtn).setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {

        switch (v.getId()) {
            case R.id.signInBtn:
```

```
            userLogin();
            break;
        case R.id.registerBtn:
            //TODO kelesi betke koshu
            Log.w("LoginActivity", " =======> Register btn");
            Intent intent = new Intent(MainActivity.this, RegisterActivity.class);
            startActivity(intent);
            break;
        default:
            Log.w("LoginActivity", "Basylgat batyrma tabylmday");
            break;
    }
```

Б қосымшасының жалғасы

```
}

private void userLogin() {

    String username = usernameEditText.getText().toString().trim();
    String password = passwordEditText.getText().toString().trim();

    if (username.isEmpty()) {
        usernameEditText.setError("Username is required");
        usernameEditText.requestFocus();

        return;
    }

    if (password.isEmpty()) {
        passwordEditText.setError("Password is required");
        passwordEditText.requestFocus();
        return;
    }
    loginApiCall(username, password);

}

private void loginApiCall(String username, String password) {
    final Call<LoginResponse> loginResponseCall = RetrofitClient
            .getInstance()
            .getApi()
            .login(username, password);
loginResponseCall.enqueue(new Callback<LoginResponse>() {
```

```
        @Override
        public void onResponse(Call<LoginResponse> call, Response<LoginRe-
sponse> response) {
```

Б қосымшасының жалғасы

```
            if (response.isSuccessful()) {
                LoginResponse loginResponse = response.body();

                Intent intent = new Intent(MainActivity.this, CategoryActivity.class);
                intent.putExtra("username", loginResponse.getUser().getUsername());
                intent.putExtra("email", loginResponse.getUser().getEmail());
                intent.putExtra("fName", loginResponse.getUser().getFirstName());
                intent.putExtra("lName", loginResponse.getUser().getLastName());
                startActivity(intent);

//                Log.d("U R HERE => TOKEN", loginResponse.getToken());
//                Log.d("U R HERE => TOKEN", loginRe-
sponse.getUser().getUsername());

                Toast.makeText(MainActivity.this, "Salem, " + loginRe-
sponse.getUser().getUsername().toUpperCase(), Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(MainActivity.this, "Error! Response from api wasn't
successfull ", Toast.LENGTH_LONG).show();
            }
        }
        @Override
        public void onFailure(Call<LoginResponse> call, Throwable t) {
            Toast.makeText(MainActivity.this, t.getMessage() ,
Toast.LENGTH_LONG).show();
        }
    });
  }
}
```

Отчет подобия

Университет:	Satbayev University
Название:	Сейдалы Гульзира
Автор:	Сейдалы Гульзира
Координатор:	Саламат Қалдыбеков
Дата отчета:	2019-05-06 12:33:01
Коэффициент подобия № 1: ?	**10,2%**
Коэффициент подобия № 2: ?	**2,8%**
Длина фразы для коэффициента подобия № 2: ?	**25**
Количество слов:	6 192
Число знаков:	50 033
Адреса пропущенные при проверке:	
Количество завершенных проверок: ?	46

! **К вашему сведению, некоторые слова в этом документе содержат буквы из других алфавитов. Возможно - это попытка скрыть позаимствованный текст. Документ был проверен путем замещения этих букв латинским эквивалентом. Пожалуйста, уделите особое внимание этим частям отчета. Они выделены соответственно.**
Количество выделенных слов 7

Ғылыми жетекші _____С. Қалдыбеков