**КЫРГЫЗ РЕСПУБЛИКАСЫНЫН БИЛИМ БЕРҮҮ ЖАНА ИЛИМ МИНИСТРЛИГИ**

**КЫРГЫЗ – ТҮРК «МАНАС» УНИВЕРСИТЕТИ**

**ТАБИГЫЙ ИЛИМДЕР ФАКУЛЬТЕТИ**

**КОЛДОНМО МАТЕМАТИКА ЖАНА ИНФОРМАТИКА БӨЛҮМҮ**

****

**ДИПЛОМДУК ИШ**

**ТЕМА: «БӨЛҮМДҮН ВЕБ САЙТЫН ИШТЕП ЧЫГУУ»**

**Студенттин аты-жөнү:** Идирис Мамыр уулу

**Бөлүмү:** Колдонмо математика жана информатика

**Студенттик номери:** 1512.01006

**Жетекчинин аты-жөнү:** Профессор физика-математика илимдеринин доктору Асан Өмүралиев

**БИШКЕК - 2020**

****

**КЫРГЫЗ РЕСПУБЛИКАСЫНЫН БИЛИМ БЕРҮҮ ЖАНА ИЛИМ МИНИСТРЛИГИ**

**КЫРГЫЗ – ТҮРК «МАНАС» УНИВЕРСИТЕТИ**

**ТАБИГЫЙ ИЛИМДЕР ФАКУЛЬТЕТИ**

**КОЛДОНМО МАТЕМАТИКА ЖАНА ИНФОРМАТИКА БӨЛҮМҮ**

( ) Студент дипломдук ишти жактоо сынагына кире алат.

( ) Студент дипломдук ишти жактоо сынагына кире албайт.

**Бөлүм башчысы:**

проф., докт. Асан Өмүралиев

**Датасы Колу**

**......./......./...……..**

**БАКАЛАВР ПРОГРАММАСЫНЫН ДИПЛОМДУК ИШИ**

**ТЕМА:** «Бөлүмдүн веб сайтын иштеп чыгуу ».

**Студенттин аты-жөнү:** Мамыр уулу Идирис

**Бөлүмү:** Колдонмо математика жана информатика

**Студенттик номери:**  1512.01006

**Жетекчинин аты-жөнү:** Профессор физика-математика илимдеринин доктору Асан Өмүралиев

**БИШКЕК –2020**

****

**КЫРГЫЗ РЕСПУБЛИКАСЫНЫН БИЛИМ БЕРҮҮ ЖАНА ИЛИМ МИНИСТРЛИГИ**

**КЫРГЫЗ – ТҮРК «МАНАС» УНИВЕРСИТЕТИ**

**ТАБИГЫЙ ИЛИМДЕР ФАКУЛЬТЕТИ**

**КОЛДОНМО МАТЕМАТИКА ЖАНА ИНФОРМАТИКА БӨЛҮМҮ**

( ) Студент дипломдук ишти жактоо сынагына кире алат.

( ) Студент дипломдук ишти жактоо сынагына кире албайт.

**Бөлүм башчысы:**

проф., докт. Асан Өмүралиев

**Датасы ......./......./...…….. Колу:**

**БАКАЛАВР ПРОГРАММАСЫНЫН ДИПЛОМДУК ИШИ**

**Тема:** « Бөлүмдүн веб сайтын иштеп чыгуу ».

**Студенттин аты-жөнү:** Мамыр уулу Идирис

**Бөлүмү:** Колдонмо Математика жана Информатика

**Студенттик номери:** 1512.01006

**Жетекчинин аты-жөнү:** Асан Өмүралиев

**Наамы:** Профессор , Доктор **Колу: .............................**

**Жактоо баасы: Комиссиянын төрагасы:**

**......................(сан менен) Аты-жөнү, колу**

**......................(жазуу жүзүндө) ...................................................**

**БИШКЕК – 2020**

****

**КЫРГЫЗ РЕСПУБЛИКАСЫНЫН БИЛИМ БЕРҮҮ ЖАНА ИЛИМ МИНИСТРЛИГИ**

**КЫРГЫЗ – ТҮРК «МАНАС» УНИВЕРСИТЕТИ**

**ТАБИГЫЙ ИЛИМДЕР ФАКУЛЬТЕТИ**

**КОЛДОНМО МАТЕМАТИКА ЖАНА ИНФОРМАТИКА БӨЛҮМҮ**

**ЖЕТЕКЧИНИН ПИКИРИ**

**Студенттин аты-жөнү:** Бакыт Кудайберди уулу

**ТЕМАСЫ: «** Бөлүмдүн веб сайтын иштеп чыгуу **»**

Дипломдук иш аткарууда коюлган ишти аткарууда тапшырмаларды мүмкүн болушунча өз убагында аткарууга аракет кылды.Билиминин жогору экендигин көрсөттү.Өзүнүн эмгекчил жана тырышчаак мүнөздөрү менен коюлган маселени толук чыгарууга жетишти.

**Жетекчинин аты-жөнү:**  Асан Өмүралиев

**Наамы:** Профессор , Доктор

**Иштеген жери:** Кыргыз – Түрк “Манас” университети

Табигый жана Так Илимдер факультети

Колдонмо Математика жана Информатика бөлүмү

**Датасы: ......./......./............**

**Колу: ...............................**

**БИШКЕК – 2019**

**КЫРГЫЗ-ТҮРК «МАНАС» УНИВЕРСИТЕТИНИН ТАБИГЫЙ ИЛИМДЕР ФАКУЛЬТЕТИНИН ДЕКАНАТЫНЫН ЧЕЧИМИ**

Кыргыз – Түрк «Манас» Университетинин Табигый Илимдер Факультетинин экзамендик инструкциясынын ...........– жобосуна ылайык, №..................... жайында уюштурулган комиссия, Колдонмо Математика жана Информатика бөлүмүнүн бүтүрүүчүсү Идирис Мамыр уулунун « Бөлүмдүн веб сайтын иштеп чыгуу » темасында жазган дипломдук проектин анализдеп, .../.../2019-жылы, саат ........ жактоого кабыл алды.

Бүтүрүүчү ...... мүнөт ичинде дипломдук проектин жактап, комиссия ................................ (*көпчүлүк добуштан/бир добуштан*) ................................ (*Кабыл алынбайт/Кабыл алынсын/Кайра оңдолсун*) деген чечим чыгарды.

Башчы ……………………………………….

Академиялык наамы, Аты - жөнү

Мүчө …………………………………………

Академиялык наамы, Аты - жөнү

Мүчө …………………………………………

Академиялык наамы, Аты - жөнү

Мүчө …………………………………………

Академиялык наамы, Аты - жөнү

Мүчө …………………………………………

Академиялык наамы, Аты– жөнү

**БИШКЕК – 2020**

**KIRGIZİSTAN TÜRKİYE**

**MANAS ÜNİVERSİTESİ**

**FEN FAKÜLTESİ DEKANLIĞINA**

Fen Fakültesi, Uygulamalı Matematik ve Enformatik bölümünde 1512.01006 numaralı Mamır uulu Idiris’in hazırladığı « Bölümün web sitesinin geliştirilmesi » konulu Üniversite lisansı ile ilgili tez savunma sınavı, …/.../2020 günü .......................... saatleri arasında yapılmış, sorulan sorulara alınan cevaplar sonunda adayın tezinin ……………... (*başarılı/başarısız*) olduğuna ………….... (*oybirliği/oy çokluğu*) ile karar verilmiştir.

Başkan ………………………………………

Akademik Ünvanı, Adı Soyadı

Üye …………………………………………

Akademik Ünvanı, Adı Soyadı

Üye …………………………………………

Akademik Ünvanı, Adı Soyadı

Üye …………………………………………

Akademik Ünvanı, Adı Soyadı

Üye …………………………………………

Akademik Ünvanı, Adı Soyadı

**БИШКЕК – 2020**

# АННОТАЦИЯ

Бул дипломдук иште бөлүмдүн -тиркемесинде иштеп чыгуусунда керек болгон маалыматтар,программалар жана мисалдар келтирилген.Берилген диплломдук иш киришүүдөн , үч бөлүмдөн,корутундудан жана колдонулган адабияттан турат .Киришүүдө аткарыла турган ишти изилдөөнүн актуалдуулугу ,максаты , маселеси жана практикалык баалуулугу камтылган.Биринчи бөлүмүндө теориялык негиз интернеттин колдонуусу боюнча түшүнүктөр каралган.Экинчи бөлүмүндө программалардды анализдөө жүргүзүлгөн .

Үчүнчү бөлүмүндө практикалык иш каралган.

Жыйынтыгында изилдөөнүн негизи натыйжасы жана корутундусу берилген.

**Ачкыч сөздөр**: Python, Django, веб-сайт, берилиштер базасы.

**БИШКЕК – 2020**

# ÖZET

Bu tez bir çalışmasında, online eğitim web uygulaması geliştitirken gerekli olan bilgi , program ve örnekler yazılmıştar.Bu tez bir giriş ,üç böküm ,sonuç ve kullanılan edabiyet listesınden oluşur.Çalışmada amaç, hedef ve yapılan işin praktik önemi aciliyet giriş bölümünde yer almaktadır.Ilk bölüm bilgisayar teknolojisini,gelişim tarihi ve bir bütün olarak internetin gelişimini inceledi.Ikinci bölümünde programlar analize ayrılmıştır.Üçüncü bölüm praktik çalışma ile ilgilidir.Sonunda genel sonuçtarı ve araştırmanın önemli bulguları ortaya koymaktır.

**Anahtar Kelimeler**: Python, Django, internet sitesi, veri tabanı.

**БИШКЕК – 2020**

# АBSTRACT

Infomation System of the department

In this thesis work, information, programs and examples are written that are necessary when developing the department’s web application.

This thesis consists of an introduction, three chapters, conclusion and references used.

In the introduction of the urgency of the study, the purpose, objective and practical importance of the work performed.

In the first chapter were considered computing equipment, its using by humanity and development of the Internet as a whole.

The second chapter is devoted to the analysis of technology.

The third chapter deals with practical work that is developing the web application.

Finally sets out the overall results and key findings of the study.

**Keywords**: Python, Django, web site, database.

**БИШКЕК – 2020**

# АННОТАЦИЯ

В этой дипломной работе написаны информации, программы и примеры необходимые при разработке web-приложения для кафедры.Данная дипломная работа состоит из введения, трех глав, заключения и использованные списки литературы.Во введении обоснована актуальность исследования, цель, задачи и практическая значимость выполненной работы.В первой главе были рассмотрены вычислительная техника, использование интернета в истории человечества.Вторая глава посвящена анализу технологий.Третья глава посвящена практической работе то есть разработку приложения.

В заключении изложены общие результаты и основные выводы исследования.

**Ключевые слова**: Python, Django, веб-сайт, база данных.

**БИШКЕК – 2020**

# МАЗМУНУ

[АННОТАЦИЯ i](#_Toc43431133)

[ÖZET ii](#_Toc43431134)

[АBSTRACT iii](#_Toc43431135)

[АННОТАЦИЯ iv](#_Toc43431136)

[МАЗМУНУ 1](#_Toc43431137)

[КИРИШҮҮ 2](#_Toc43431138)

[1-БӨЛҮМ.ТЕОРИЯЛЫК НЕГИЗ 4](#_Toc43431139)

[§1.1 Веб колдонмо 4](#_Toc43431140)

[§1.2 Фронтэнд же кардар тарап 4](#_Toc43431141)

[2-БӨЛҮМ.КОЛДОНУЛГАН ТЕХНОЛОГИЯЛАР 11](#_Toc43431142)

[§2.1 Python 11](#_Toc43431143)

[**2.1.1 Кыскача тарыхы** 11](#_Toc43431144)

[**2.1.2 Архитектурасы** 12](#_Toc43431145)

[§2.2 Django 12](#_Toc43431146)

[§2.3 SQLlite3 13](#_Toc43431147)

[§2.4 Bootstrap 13](#_Toc43431148)

[3-БӨЛҮМ .ИШКЕ АШЫРУУ 14](#_Toc43431149)

[§3.1 Долбоордун баяндамасы 15](#_Toc43431150)

[§3.2Системаны куруу жана орнотуу 16](#_Toc43431151)

[§3.3Долбоордун тузулушу 17](#_Toc43431152)

[§3.4 Контроллер 19](#_Toc43431153)

[§3.5 Модель 28](#_Toc43431154)

[§3.5 Шаблон 36](#_Toc43431155)

[КОРУТУНДУ 43](#_Toc43431156)

[ЫРААЗЫЧЫЛЫК БИЛДИРҮҮ 44](#_Toc43431157)

[КОЛДОНУЛГАН АДАБИЯТТАР 45](#_Toc43431158)

# КИРИШҮҮ

Жашообузда орун алып жаткан жаңы ачылыштар жана бизге чейинки ачылыштар, адамзат жашоосун улам-улам өзгөртүүдө. Алгачкы учурларда, сыналгы, радио жана учак сыяктуу күтүлбөгөн, таң калычтуу акылга келбеген ачылыштар болуп саналса, азыркы учурда адамдардын ачылыштырга карата таң калуу реакциясы дээрлик жок, көнүмүш катары кабыл алынууда. Андан сырткары, ачылыштар бир гана окумуштуулар тарабынан күтүлбөстөн, карапайым, билимге толук ээ болбогон адамдар тарабынан дагы күтүлүүдө. Албетте, бул көрүнүш адамзатынын замандан заманга карай күчтүү акыл-эс, өзгөрүү стадиясында экенине далил болуп саналат.

Азыркы учурда интернеттин пайда болуусу менен дээрлик баардык тармактарда өзгөрүүлөр орун алды . Мисалга алсак, ишканалардын товар жана кызмат сатуу маркетинги өзгөрдү, университеттерде билим берүү системалары, сатып алууда, баарлашууда, маалымат алууда ж.б.у.с. Булар интернеттин бизге берген жеңилдиктери болуп саналат. Интернеттин терс тараптары катары адамдардын кыймылы аз болуп ден соолугуна таасирин тийгизүүдө, убакыттын интернетке көп сарпталышы, туура эмес маалымат таралышы сыяктуу маселелерди ортого чыгарды.

Бул дипломдук иште интернет тармагынын эң кичине бөлүгү болгон сайт жасалмакчы. Тагыраак айтканда Колдонмо Математика жана информатика бөлүмүнүн ички муктаждыктарын канааттандыруу менен бирге студенттердин жана окутуучулардын өз ара байланышын чыңдоо үчүн веб колдонмо жасоого далалат кылынат. Мугалим жана студенттердин бири бири менен өз ара теориялык маалымат алмашуу, тапшырмаларды жарыялоо, алардын жоопторун файл аркылуу жүктөө, мугалим тарабынан баалоо жана ошондой эле бөлүм башчысынын студенттерди жана мугалимдерди көзөмөлдөө, сабактарды бөлүштүрүү функциялары каралмакчы.

# 1-БӨЛҮМ.ТЕОРИЯЛЫК НЕГИЗ

# §1.1 Веб колдонмо

***Веб колдонмо деп*** – Вебсайт же вебсайт (англисче веб-сайттан: веб - "веб, тармак" жана сайт - "жер", сөзмө-түз "жер, сегмент, тармактын бөлүгү") - бир же бир нече веб-баракчалар логикалык жактан өз ара байланыштуу; сервер мазмунунун жайгашкан жери. Адатта, Интернеттеги веб-сайт - бул байланышкан маалыматтардын массиви, ал уникалдуу дареги бар жана жалпы колдонуучулар тарабынан кабыл алынат. Веб-сайттар ушуну аташат, анткени аларга HTTP протоколун колдонсо болот . Электрондук документтердин тутуму катары веб-сайт (маалыматтар жана код файлдары) жеке адамга же уюмга таандык болушу мүмкүн жана компьютердик тармакка жалпы домендик аталышта жана IP дареги боюнча же бир компьютерде жайгашкан. "Экономика жана Укук" журналындагы макалада ар бир сайттын өз аты бар экени жана аны домендик аталыш менен чаташтырбоо керектиги айтылган . Автордук укуктун көз карашынан алганда, сайт курама чыгарма, ошондуктан сайттын аталышы башка бардык чыгармалардын аттары менен кошо корголууга тийиш. Колдонмонун логикасы кардар жана сервердин ортосунда болуп, берилиштер негизинен серверде сакталат. Берилиштер болсо тармак ичинде Hypertext Transfer Protocol (HTTP) протоколу бойунча алмашылат. Бул ыкманын бирден бир артыкчылыгы колдонуучунун иштетүү системинен же конфигурациадан көз карандысыздыгы болуп эсептелет. Мындан келди колдонмолор плтформ-аралык кызматтарды аткаруучу болуп эсептелинет.

# §1.2 Фронтэнд же кардар тарап

***Фронтэнд*** – курулушу , колдонуучу менен түздөн түз байланышкан веб колдонмонун ачык тарабын жасоо менен коштолот. Фронэнд жасоо – бул веб сайтты кооз жана кызыктуураак кылуу менен гана чектелет десек да туура болот (Codesido, 2009). Веб сайттын жасалуусунда өтө чоң орду болгон Фронтэнд өзүнө көптөгөн технологиаларды камтып кетет (Codesido, 2009). Бул процессте колдонулган өзөк куралдар: Гипертекттик Жазуу Тили (HTML), Кийишилген Стилдердин Жадыбалы (CSS) жана JavaScript (JS).

***HTML*** – веб барактын негиз таштары болуп эсептелет. Белгилөө тили болуу менен барактын бүтүндөй түзүмүн баяндайт. Ар турдүү тегдерден түзүлөт.Тегдер экилик жана жалгыз болуп экиге бөлүнүшөт. Ар бир тегтин өзүнчө милдети бар. Мисалы жадыбал, сап, тизме, сүрөттөрдү ж.б. баяндоо сыяктуу. HTML тили браузерлер тарабынан которулган соң сап калыптанат. Соңунда калыптанган сап компютер же уюлдук телефон ж.б. аспаптардын экранына чыгарылганга мүмкүн болот.

***CSS*** - веб дизайнерлер үчүн өтө оңтойлуу жана алмаштырылгыс курал. Аны менен HTML тилинде жазылган документтин ар бир компонентине каалагандай стиль берсе болот. Кийишилген стилдер жадыбалын өзүнчө файлга жазуу сунуш кылынат. Себеби бир эле стил көптөгөн барактар үчүн өтүмдүү болуусу камсыз кылынышы үчүн (Consortium W. W., 2016).

***JavaScript*** – веб барактын ар бир кыймылдуу жана жандуу ишкердүүлүгүнө башкы күнөөкөр болуп эсептелинет. Бул жеңил салмактуу жана интерпреттик тил. Колдонуучу тарабынан киргизүүлөрдү тууралоо жана баскычтоп менен чычкандын аракеттерин тескөө менен алек болот (TutorialsPoint, 2016). Эң башкысы мунун эң пайдалуу касиеттеринен Асинхрондук JavaScript жана XML

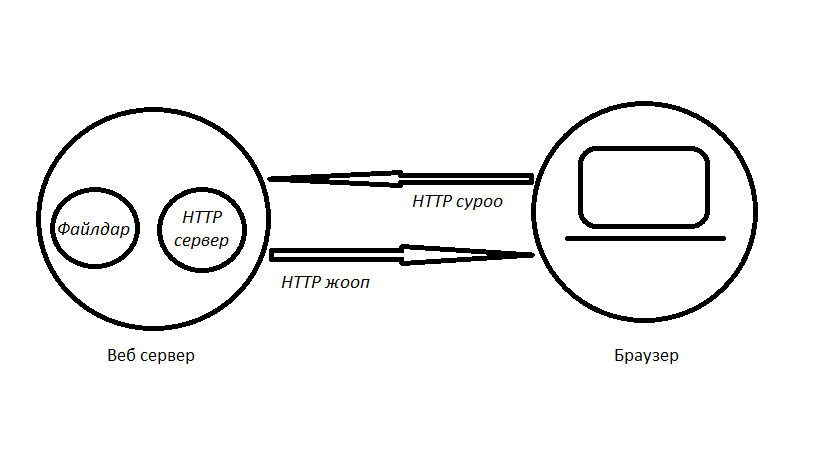
(AJAX) орун алат. Ал веб барактын кайра кайра жаңылануусуна жол бербестен керектүү гана маалымат жана берилиштерди серверге жөнөтүп же алып турат. Бул трафик үнөмдөйт жана иштөө жөндөмдүүлүктү кыйла жогорулатат.

**§ 1.2 Бэкенд**

Бэкэнд жасоодо сервер тарапты ишке ашыруу каралат. Ал биринчи кезекте веб колдонмонун логикасын башкача айтканда колдонмо кантип иштөөсүн чечет. Бул колдонмонун өзөгүн түзүү керектүү функционалдуулук менен камсыз кылуу процесси. Сервер тарап кардар тараптан келген берилиштердин үстүнөн иштеп, аны кайра кардар түшүнө алган түрдө кардар тарапка жөнөтөт. Бэкэнд негизинен үч бөлүккө бөлүнөт: сервер колдонмосу, колдонмо логикасы жана берилиштер базасы.

**§ 1.2.1 Сервер колдонмосу**

***Веб сервер колдонмосу*** – бул иштетүү системында иштеген, берилиштерди браузелер сыяктуу кардарлар үчүн тейлөө менен алек болгон програм. Ал URL жана HTTP протокол дегендер эмне экендигин түшүнүп билиши керек (MDN, 2016). HTTP протоколунун негизги принциптерин төмөндө сүрөттөн көрө алсак болот.



1-сүрөттө көрүнүп тургандай браузер суроо жөнөтөт. Качан гана тиешелүү машинага жеткенден кийин HTTP сервер суроого ылайык жоопту жөнөтөт. HTTP протоколу кандайдыр бир эрежелер жыйнагын колдонсо ал дайыма сакталат. Айта кетчү нерсе, сервер бир гана кардар жөнөткөн суроого жооп берүү менен чектелет. Тескерисинче кардарга суроо жөнөтө албайт. Экинчиден, сервер кардар тараптан келген ар бир суроого жооп берүүсү керек, жок дегенде сервер учурда бул суроого жооп бере албастыгын билдирген ката билдирүүсүн болсо дагы жөнөтүүсү керек. Акырында ар бир HTTP суроо өзүнө URL ди камтуусу керек. Ал суроо берилиши керек болгон сервердин дарегин камтыйт (Consortium, 2008).

Веб серверди тандоонун ар кандай варианттары бар. Алардын ичинде nginx, Apache HTTP server, Google Web Server жана башкалар колдонуу боюнча алдыңкы орундарда турат. Веб серверлердин базар боюнча иштеп жаткан сайттардын үлүштөрү төмөнкү сүрөттө көрсөтүлгөн. Сурамжылоо 2018-жылы августта жүргүзүлгөн.

Азыркы учурда эң эле көп колдонулган сервер Apache базардын дээрлик 33 пайызын алуу менен биринчиликте келатат. Андан кийин эле nginx базардын 26 пайыздык үлүш менен экинчиликти алуу менен өсүүдө. Веб сервелерди тандоодо иштетүү системасынын түрү, коопсуздук деңгээли жана интөө кубаттуулугу сыяктуу.бир канча белгилерги кароо керек. Ал эми Python тилинин эн эле атактуу Django фреймворгундагы өзүнүн веб сервери бар , бул веб сервер менен иштөө өтө ынгайлуу жана колдонууга оной.

**§ 1.2.1 Сервер колдонмосу**

Колдонмо логикасы – бизнес логика деп да аталат.Бул баардык суроо берилиштер жана жөнөтүлүүчү берилиштер, берилиштерди базада сактоо , берилиштерди алуу үчүн суроолорду түзүү боюнча чечимдерди кабыл алуу иш аракеттерин өзүнө камтыйт. Ал коопсуздук эрежелерин, колдонуучуларды таануу жана айырмалоо үчүн каттоо механизмдерин, серверге кандай суроолор келээрин жана кандай жооптор жөнөтүлөөрүн баяндайт. Каттоо жана таануу иш аракеттери колдонуучуларды бул же ал аракеттерге жеткиси бар же жок экендигин аныктайт (MSDN, Business Layer Guidelines, 2018).

Бизнес логика – программа катарында Python, Go lang, PHP, .Net, Node.js сыяктуу програм тилдерин колдонуу менен жасалат. Бүгүнкү күндөрдө көбүнчө программа тилдеринин даяр чечимдери болгон фрейвогдорун колдонуу менен бизнес логикасын жасоо оңой болууда. Веб фреймворгдор колдонмонун өзөк керекчелерин, кээ бир отурумду башкаруу, берилиштер базасы менен байланышты уюштуруу, чыгуучу берилишти калыптоо сыяктуу жалпы функционалдуулуктарын камтуу менен программага көзгө көрүнөөрлүк жардам берүүдө. Буларга мисал катары Django, Ruby on Rails, Express, Spring, Flask ж.б. фреймворгторун айтып кетсек болот (MSDN, Business Layer Guidelines, 2018).

Ошентип жогоруда айтылгандай колдонмо логикасы берилиштердин көрүнүшүн, жасалышын, сакталышын жана башкарылышын көзөмөлдөө менен бэкэнддин эң бир маанилүү бөлүктөрүнөн болуп эсептелинет.

**§ 1.2.1 Берилиштер базасы**

Берилиштер базасынын башкаруу системи (DBMS англ. Database Management System) веб колдонмонун айрыкча бэкэнд тараптын маанилүү бөлүгү болуп эсептелет. DBMS берилиштерди сактоо, өзгөртүү, өчүрүү мүмкүндүгүн берет. Ал чыныгы берилиштер базасынан берилиштерди суроого (алууга) андан кийинки колдонмо логикасына жараша колдонулушу үчүн мүмкүндүк берет. DBMS берилиштердин уюштурулушун, коопсуздук иш чараларын көрүү, берилиштерди интеграциалоо, жарыш эрежелерди жасоо жана көзөмөлдөө, колдонуучуларды кошуу жана мониторлоону камсыз кылат. Берилиштер базасынын эки түрү бар: *байланышкан жана байланышпаган.*

Байланышкан берилиштер базасынын башкаруу системасы (RDBMS) берилиштерди башкаруучу катары өзүн эң жакшы катары 20 жыл бою көрсөтүп келе жатат. Баардык байланыштуу берилиштер базалар берилиштерди башкаруу, өзгөртүү үчүн түзүмдөлгөн суроо тилин (SQL) колдонот. Ошондуктан байланыштуу берилиштер базалары көбүнчө түзүмдөлгөн берилиштер тил базалары деп аталат. SQL берилиштер базалары берилиштерди жадыбал түрүндө сактайт. Жадыбалдын ичинде берилиштер мамыча жана сапча түрүндө жайгашкан. Саптарда берилиштердин маңызы ал эми мамычаларда болсо берилиштердин атрибуттары жайгашкан. Берилиштер жогорку деңгээлде түзүмдөлгөн жана сактоо схемасы абдан ачык айкын болушу керек. Ар түрдүү жадыбалдар тиешелүү түрдүү логика мамиле (relationship) байланыштарын камсыз кылат. Бул түрдөгү байланыштар жадыбалдар арасында көрсөтүлгөн тартипте жасалат.Мындан келди “байланыштуу берилиштер базасы” ысмын алат (Amazon, 2019). Байланыштуу берилиштер базасы көбүнчө базанын ийилчээк болуусу арбын маалыматтын камтылышы үчүн колдонулат. Аларды бирден ашык сервелер колдоно алат. Көп таанымал RDBMS тер: Oracle, MySQL, SQL Server, PostgreSQL ж.б.

*Байланышсыз берилиштер* базасы акыркы учурда гана таанымалдуулугу арта баштады. Leavitt N.(2010) караганда 2007-жылы гана байланышсыз берилиштер базасынын этибарга алынаарлык таанылуусу ортого чыкты. Бул да Amazon дун ички колдонулушу үчүн байланышсыз Dynamo аттуу берилиштер базасын жасап колдонгондугун баяндап жазып чыккандан кийин гана маалым болду (Microsoft, 2019). Байланышсыз берилиштер базасы берилиш башкаруусу үчүн бир гана SQL ди эле колдонушпайт. Алар көбүнчө NoSQL (Not only SQL англ. Түзүмдөлгөн суроо тили гана эмес) ди да колдонушат. NoSQL түрүндөгү берилиштер базалар демейки байланыштуу берилиштер база башкаруу системдеринен кескин айырмачылыктары бар. RDBMS колдонгон берилиштерди жадыбал түрүндө сактоонун ордуна, NoSQL базалары берилишти айырмаланган ачкыч/маани, граф, кең мамычалар, JSON жана документ дагы башка форматтарда сактайт. Берилиш абдан кыймылдуу жана түзүм схеми толугу менен ийинлчээк боло алат (Microsoft, 2019). Мындай ыкманын пайдасы ар бир жазуунун өзүнчө касиети бар экенин эске алсак, берилиштер базасы колдонуп жаткан схемди өзгөртпөстөн эле оңой олтоң ишке ашуусу мүмкүн. Себеби схема берилиштер базасы менен жөнгө салынат. Ошондуктан берилиштер салыштырмалуу комплекстүү, кийишилген же жогору түзүмдөлбөгөн учурда мындай ыкманын мааниси бар. Байланышсыз берилиштер базаларына мисал катары MongoDB, Apache Cassandra, Redis, Apache Hbase жана neo4j ларды келтирсек болот.

Мен колдонгон берилиштер базасы - SQLite3 . Django фреймворгунда атайын тандалган түрдө орнотулган база. SQLite 3 дүйнөдө эң көп жайылган жана сунуш кылынган булак коддору журтка ачык, бүтүндөй С/С++ программалоо тилдеринде жазылган сервер програмы жана жасалгалоо керектөөлөрүн талап кылбаган, операциондук жана байланыштуу SQL берилиштер базасынын мотору.

# 2-БӨЛҮМ.КОЛДОНУЛГАН ТЕХНОЛОГИЯЛАР

# §2.1 Python

**Python** (IPA менен: [ˈpʌɪθ(ə)n]; **питон** аталышы жайылган) - иштеп чыгуучунун өндүрүмдүүлүгүнө жана коддун окумдуу болушуна багытталган жогорку даражалуу, жалпы милдеттерди аткарууга арналган программалоо тили. Python өзөгүнүн синтаксиси - минималдаштырылган. Ошол эле учурда, стандарттуу китепканасы көптөгөн пайдалуу функцияларды камтыйт. Python бир нече программалоо парадигмаларын колдойт. Алардын ичинде структуралык, функционалдык, императивдик, объектиге жана аспектиге багытталган парадигмалар да бар. Негизги архитектуралык өзгөчөлүктөрү - динамикалык типтештирүү, автоматтык эс башкаруу, толук интроспекция, бөтөнчөлүктөрдү иштетүү механизми, көп агымдык эсептөөнү колдоо жана жогорку даражалуу маалымат структуралары. Python'до коддор модул болуп бириге алчу иш-милдеттер жана класстар болуп уюшулат.

### **2.1.1 Кыскача тарыхы**

1990-жылы жакшыртуу үчүн Guido van Rossum тарабынан Амстердамда башталган. Элдик ишенимге жылан каршы келген Guido van Rossumдун жактырган Monty Python аттуу алты кишилик британиялык комедия тобу Monty Python’s Flying Circus шоу аттуу көрсөтүүсүнөн алынган экен. Бүгүнкү күндө, бул Python Software Foundation ыктыярчылардын күчүн чогултуу менен уланууда жана күчөөдө. Python 1.0 версиясына 1994-жылы январда жеткен. Акыркы туруктуу версиясы 2.х сериясында Python 2.7 жана 3.х сериясында Python 3.5.2 чыккан. 2008-жылы 3 – декабрдан баштап 3.х сериясы сатыка чыгып жайылып баштаган агадейре 3.х сериясы 2.х сериясы менен шайкеш келбейт.

### **2.1.2 Архитектурасы**

Python бир нече программалоо парадигмаларын колдойт. Алардын ичинде структуралык, функционалдык, императивдик, объектиге жана аспектиге багытталган парадигмалар да бар. Негизги архитектуралык өзгөчөлүктөрү - динамикалык типтештирүү, автоматтык эс башкаруу, толук интроспекция, бөтөнчөлүктөрдү иштетүү механизми, көп агымдык эсептөөнү колдоо жана жогорку даражалуу маалымат структуралары (Ziade, 2019). Python'до коддор модул болуп бириге алчу иш-милдеттер жана класстар болуп уюшулат. (модулдар өз кезегинде пакет болуп бириге алышат).

# §2.2 Django

Django (Джанго, ['dʒæŋɡoʊ]) — [Python](#_3.1_Python) тилиндеги веб-колдонмолор үчүн эркин фреймворк. MVC долбоорлоо калыбын колдонот. Долбоорду Django Software Foundation уюму колдойт.

Django'до сайт бир же бир нече колдонмодон курулат. Буларды ажыратууга же кошууга мүмкүн кылуу сунушталат. Бул анын айрымдардан олуттуу архитектуралык айырмасы болуп эсептелет (мисалы, Ruby on Rails). фреймворктун негизги принциптеринин бири - DRY (англ. Don't repeat yourself)

Башка фреймворктордон айырмаланып, URL иштеткичтери Django'до туруктуу айкаштардын жардамы менен апачык конфигурацияланат. Контроллерлердин моделинин курамынан автоматтык түрдө чыкпайт.

Django берилиштер базасы менен иштөө үчүн өз ORM'ин колдонот (Zamirbekov, 2017). Мында берилиштер модели Python класстары менен сүрөттөлөт жана буга жараша берилиштер базасынын схемасы чыгарылат. Биз да бул эмгекте ушул фреймворкту колдонууга канаат кылдык.

# §2.3 SQLlite3

SQLite 3 дүйнөдө эң көп жайылган жана сунуш кылынган булак коддору журтка ачык, бүтүндөй С/С++ программалоо тилдеринде жазылган сервер програмы жана жасалгалоо керектөөлөрүн талап кылбаган, операциондук жана байланыштуу SQL берилиштер базасынын мотору (Yalcin, 2017).

SQLite 3 ондогон программалоо тилдеринде бей капар колдонула алат. Булардан кээ бирлери: ASP, BASIC, C#, C, C++, Common Lisp, Curl, D, Delphi, D, Delphi, Python, Haskell, Java, Ruby, PHP, Smaltalk жана Visual Basic.

SQLite 3 эң башында биринчи чыгарылышы D. Richard Hipp тарабынан 2000-жылында сунулду. Ал учурда General Dynamics фирмасында Кошмо Штаттардын коргонуусунда иштеген Hipp, берилиштер базасына жана да башкаруу панелине муктаждыгы жок програм жазайын деп жатып SQLite ты таап алды. Кийинки жылдарда 2.х жана 3.х чыгарылыштары жарык көрдү (Amazon, 2019).

# §2.4 Bootstrap

**Bootstrap** (**Twitter Bootstrap** катары да белгилүү) - вебсайт жана веб-колдонмо түзүүгө арналган эркин куралдар топтому. Типографика, веб-форма, баскыч, белги, навигация блогу жана веб-интерфейстин башка компоненттерин жана Javascript -жайылууларды жасалгалоо үчүн HTML- жана CSS-калыптарды камтыйт.

Bootstrap CSS жана HTML жаатындагы заманбап иштеп чыгууларды пайдаланат. Ошондуктан, эски браузерлерди колдоодо этият болуу керек.

# 3-БӨЛҮМ .ИШКЕ АШЫРУУ

**Python** (IPA менен: [ˈpʌɪθ(ə)n]; **питон** аталышы жайылган) - иштеп чыгуучунун өндүрүмдүүлүгүнө жана коддун окумдуу болушуна багытталган жогорку даражалуу, жалпы милдеттерди аткарууга арналган программалоо тили. Python өзөгүнүн синтаксиси - минималдаштырылган. Ошол эле учурда, стандарттуу китепканасы көптөгөн пайдалуу функцияларды камтыйт. Python бир нече программалоо парадигмаларын колдойт. Алардын ичинде структуралык, функционалдык, императивдик, объектиге жана аспектиге багытталган парадигмалар да бар. Негизги архитектуралык өзгөчөлүктөрү - динамикалык типтештирүү, автоматтык эс башкаруу, толук интроспекция, бөтөнчөлүктөрдү иштетүү механизми, көп агымдык эсептөөнү колдоо жана жогорку даражалуу маалымат структуралары. Python'до коддор модул болуп бириге алчу иш-милдеттер жана класстар болуп уюшулат.

**Тарыхы**

1990-жылы жакшыртуу үчүн Guido van Rossum тарабынан Амстердамда башталган. Элдик ишенимге жылан каршы келген Guido van Rossumдун жактырган Monty Python аттуу алты кишилик британиялык комедия тобу Monty Python’s Flying Circus шоу аттуу көрсөтүүсүнөн алынган экен. Бүгүнкү күндө, бул Python Software Foundation ыктыярчылардын күчүн чогултуу менен уланууда жана күчөөдө. Python 1.0 версиясына 1994-жылы январда жеткен. Акыркы туруктуу версиясы 2.х сериясында Python 2.7 жана 3.х сериясында Python 3.5.2 чыккан. 2008-жылы 3 – декабрдан баштап 3.х сериясы сатыка чыгып жайылып баштаган агадейре 3.х сериясы 2.х сериясы менен шайкеш келбейт.

# §3.1 Долбоордун баяндамасы

Бул долбоор Колдонмо математика жана информатика бөлүмүнүн бөлум башчысы, мугалимдери жана студенттерине үчүн жеңил маалымат алмашуу, автоматташтыруу жана онлайн баалоо системасын камтыган веб система катары жасалды. Андан сырткары студенттер бат ‘https://obis.manas.edu.kg’ сайты аркылуу онлайн катталуу функциясы камтылган. Бул веб системада бөлүм башчы, мугалимдер жана студенттер үчүн төмөнкү интерфейстер каралды.

* Бөлүм башчы(админ) үчүн мугалимдерди каттоо, сабактарды каттоо жана мугалимдерге сабактарды бөлүштүрүү, башкаруу интерфейси
* Бөлүм башчы(админ) катталган студентерди башкаруу интерфейси
* Бардык катталган колдонуучулар үчүн логин интерфейси
* Мугалимдер үчүн бөлүштүрүлгөн сабактарды башкаруу, маалыматтарды, тапшырмаларды студенттер менен бөлүшүү жана тапшырмаларды баалоо, сабактын планын киргизүү интерфейси
* Студенттер үчүн мугалимдер жүктөгөн маалыматты алуу, тапшырмаларды аткарып кайра жүктөө интерфейси

Негизинен бул веб система 3 негизги чон интерфейстерди камтыйт: Бөлүм башчы(админ), мугалим, студент интерфейстери. Аталган интерфейстердин баары колдонуучулар үчүн катталгандан сон гана жеткиликтүү болот.

# §3.2Системаны куруу жана орнотуу

Бул веб долбоор Windows жана Linux операциондук системаларында иштейт. Долбоорду куруу үчүн төмөнкү программалык жабдыктар талап кылынат:

* Windows же Linux операциондук системасы
* Python 3.7X

Долбоорду Linux операциондук системасына орнотуу:

1. Python 3.7x орнотуу.

~$ sudo apt-get install python

1. Башкаруу системасы Git системасын куруу.

~$ sudo apt-get install git

1. Виртуалдык чөйрөнү куруу.

~$ pip install virtualenv

1. Виртуалдык чөйрөнү жандыруу.

~$ source /venv/bin/activate

1. Долбоорду башкаруу системасынан жүктөө

~$ git clone https://gibhub.com/idiris97/project\_ume.git

1. Бардык керек болгон модульдарды жүктөө

~$ pip install –r req.txt

Долбоорду Windows операциондук системасына орнотуу:

1. Python 3.7x орнотуу.

<https://www.python.org/downloads/release/python-377/>

1. Башкаруу системасы Git системасын куруу.

<https://git-scm.com/download/win>

1. Виртуалдык чөйрөнү куруу.

pip install virtualenv

1. Виртуалдык чөйрөнү жандыруу.

\venv\bin\activate

1. Долбоорду башкаруу системасынан жүктөө

git clone https://gibhub.com/idiris97/Ume-project.git

1. Бардык керек болгон модульдарды жүктөө

pip install –r req.txt

Курулган долбоорду локалдык серверде жандыруу:

1. Тузулгөн модельге киргизилген өзгөрүүлөрдүн негизенде жаңы түзүлгөн миграциялар.

manage.py makemigrations

1. Түзүлгөн миграцияларды базага орнотуу же өчүрүү.

manage.py migrate

1. Локалдык серверди жандыруу.

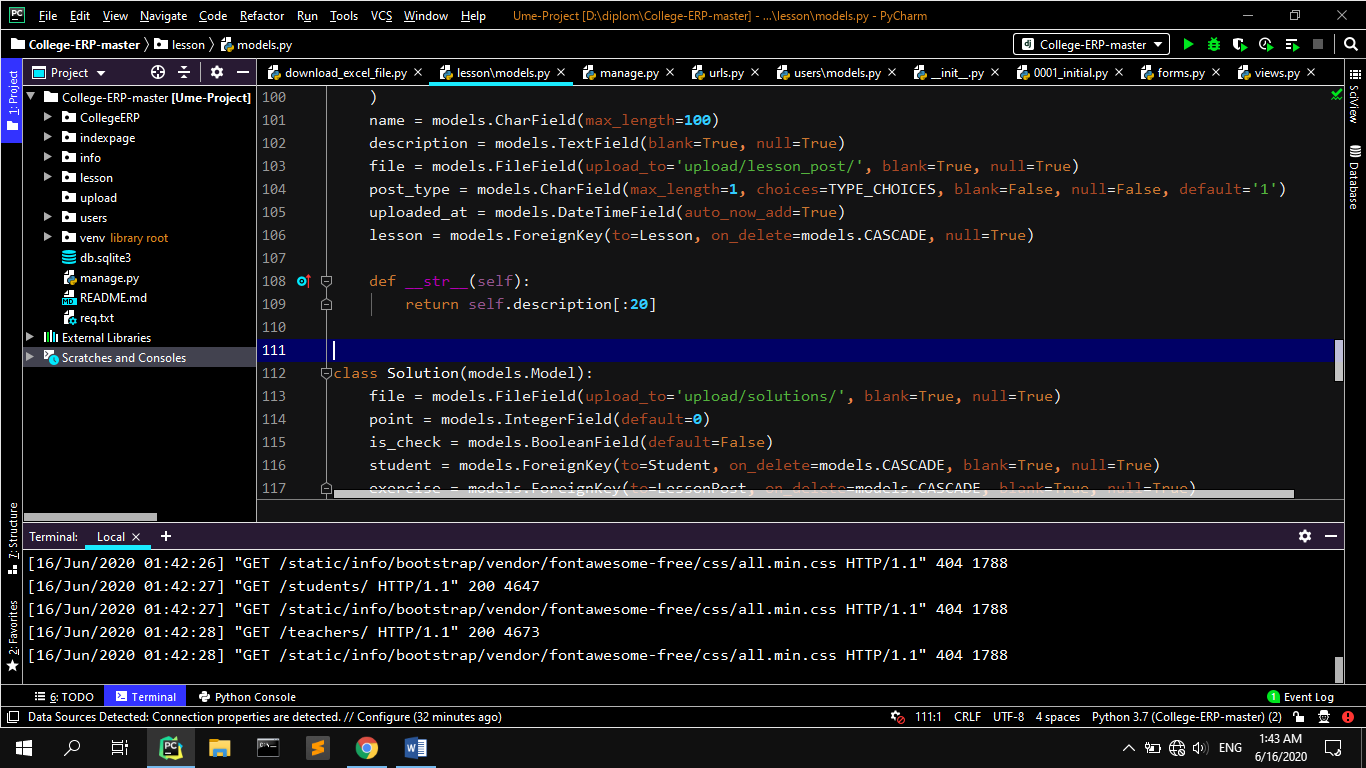
manage.py runserver

# §3.3Долбоордун тузулушу

Бул долбоор “python” программдык тилинин азыркы мезгилде белгилүү болгон “Django” веб фреймворкунун негизинде түзүлдү. “Django” веб фреймворку MVC(Model-View-Controller) шаблонуна окшош MTV(Model-Template-View) шаблонунун негизинде түзүлгөн. MTV – бул:

* Модель(Model) – бул базага жеткиликтүү болгон бөлүгү.Базадагы таблицалардын бири бири менен болгон байланышын, маалыматты салуу, текшерүү жана алмаштыруу кызматтарын аткарган бөлүк болуп саналат.
* Шаблон(Template) – бул колдонуучуга көрүнгөн бөлүгү.
* Контроллер(View) – бул бөлүк бардык логиканы өзүнө камтыйт жана модель менен шаблонду байланыштырган көпүрө болуп саналат.Башкача айтканда базадагы маалыматты шаблонго чыгарат, шаблондон киргизилген маалыматты талдап өзгөртүп кайра базага сактоо функцияларын камтыйт.

Долбоордун жалпы түзүлүшү төмөндөгү сүрөттө көрсөтүлгөн:



*Сүрөт 1.Долбоордун жалпы көрүнүшү*.

Жогоруда көрсөтүлгөн долбоордо жана бардык “Django” веб фреймворк системаларында manage.py файлы болот жана ал кирүү чекити болуп саналат. Ал серверди башкаруу жана башка функцияларды аткарууда колдонулат.

# §3.4 Контроллер

Жогоруда айтылгандай эле контроллердин кызматын views.py файлы аткарат. Жана ал долбоордо өзүнө төмөнкүдөй кодду камтыйт:

from django.contrib import messages

from django.contrib.auth import logout

from django.contrib.auth.decorators import login\_required

from django.contrib.auth.models import User

import requests

from django.contrib.auth import authenticate, login

from django.shortcuts import render, redirect

from users.forms import StudentCreationForm, StudentCreationFormObis, \

TeacherCreationForm, LoginForm

from users.models import Student, Teacher

import secrets

import string

def logoutUser(request):

logout(request)

return redirect('login')

# Student Registration View

def registrationStudent(request):

if request.method == 'POST':

form = StudentCreationForm(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

if cd['password1'] == cd['password2']:

try:

user = User.objects.create\_user(username=cd['username'], first\_name=cd['name'],

last\_name=cd['surname'],

password=cd['password1'])

student = Student(user=user)

student.save()

return redirect('login')

except:

messages.error(request, 'This user is already registered')

return redirect('registrationStudent')

else:

messages.error(request, 'The two password fields didn’t match.')

return redirect('registrationStudent')

else:

messages.error(request, 'Invalid form')

return redirect('registrationStudent')

else:

form = StudentCreationForm()

return render(request, 'info/users/registrationStudent.html', {'form': form})

# Student Registration View with 'obis.manas.edu.kg'

def registrationStudentObis(request):

if request.method == 'POST':

form = StudentCreationFormObis(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

username = cd['username']

password = cd['password']

try:

user = User.objects.get(username=username)

except Exception:

user = False

if user == False:

payload = {

"frm\_kullanici": username,

"frm\_sifre": password,

"page": "menu1"

}

session\_requests = requests.session()

login\_url = "http://obis.manas.edu.kg/"

try:

result = session\_requests.get(login\_url)

# tree = html.fromstring(result.text)

result = session\_requests.post(

login\_url,

data=payload,

headers=dict(referer=login\_url)

)

url = "http://obis.manas.edu.kg/index.php?page=bilgiler"

response = session\_requests.get(

url,

headers=dict(referer=url)

)

webstr = response.content.decode("ISO-8859-9")

if 'Sizin bilgileriniz' in webstr:

webstr1 = webstr.split("Sizin bilgileriniz")[1].split("<!-- ## CODE END -->")[0]

first\_name = webstr1.split('Adı')[1][19:].split('<')[0]

last\_name = webstr1.split('Soyadı')[1][19:].split('<')[0]

user = User.objects.create\_user(username=username, first\_name=first\_name, last\_name=last\_name,

password=password)

student = Student(user=user)

student.save()

return redirect('login')

messages.error(request, 'Invalid login or password')

return redirect('registrationStudentObis')

except:

messages.error(request, 'Invalid login or password')

return redirect('registrationStudentObis')

else:

messages.error(request, 'this user is already registered')

return redirect('registrationStudentObis')

else:

messages.error(request, 'Indvalid form')

return redirect('registrationStudentObis')

else:

form = StudentCreationFormObis()

return render(request, 'info/users/registrationStudentObis.html', {'form': form})

# Teacher Registration View

@login\_required

def registrationTeacher(request):

if request.method == 'POST':

form = TeacherCreationForm(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

alphabet = string.ascii\_letters + string.digits

password1 = ''.join(secrets.choice(alphabet) for i in range(10)) # for a 20-character password

try:

user = User.objects.create\_user(username=cd['username'], first\_name=cd['name'], last\_name=cd['surname'],

password=password1)

zvanie = cd['zvanie']

statu = cd['statu']

sira = cd['sira']

teacher = Teacher(user=user, zvanie=zvanie, statu=statu, sira=sira, password2=password1)

teacher.save()

except:

user.delete()

messages.error(request, 'invalid registration')

return redirect('registrationTeacher')

return redirect('teachers')

else:

messages.error(request, 'Invalid form')

return redirect('registrationTeacher')

else:

form = TeacherCreationForm()

return render(request, 'info/users/registrationTeacher.html', {'form': form})

# Student Login View

def loginStudent(request):

if request.method == 'POST':

form = LoginForm(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

user = authenticate(username=cd['username'], password=cd['password'])

if user is not None:

if user.is\_active:

login(request, user)

return redirect('index')

else:

messages.error(request, 'Disabled account')

return redirect('loginStudent')

else:

messages.error(request, 'Invalid login')

return redirect('loginStudent')

else:

form = LoginForm()

return render(request, 'info/users/loginStudent.html', {'form': form})

# Teacher Login View

def loginTeacher(request):

if request.method == 'POST':

form = LoginForm(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

user = authenticate(username=cd['username'], password=cd['password'])

if user is not None:

if user.is\_active:

login(request, user)

return redirect('index')

else:

messages.error(request, 'Disabled account')

return redirect('loginTeacher')

else:

messages.error(request, 'Invalid login')

return redirect('loginTeacher')

else:

form = LoginForm()

return render(request, 'info/users/loginTeacher.html', {'form': form})

# Admin Login View

def loginAdmin(request):

if request.method == 'POST':

form = LoginForm(request.POST)

if form.is\_valid():

cd = form.cleaned\_data

user = authenticate(username=cd['username'], password=cd['password'])

if user is not None:

if user.is\_active:

login(request, user)

return redirect('index')

else:

messages.error(request, 'Disabled account')

return redirect('loginAdmin')

else:

messages.error(request, 'Invalid login')

return redirect('loginAdmin')

else:

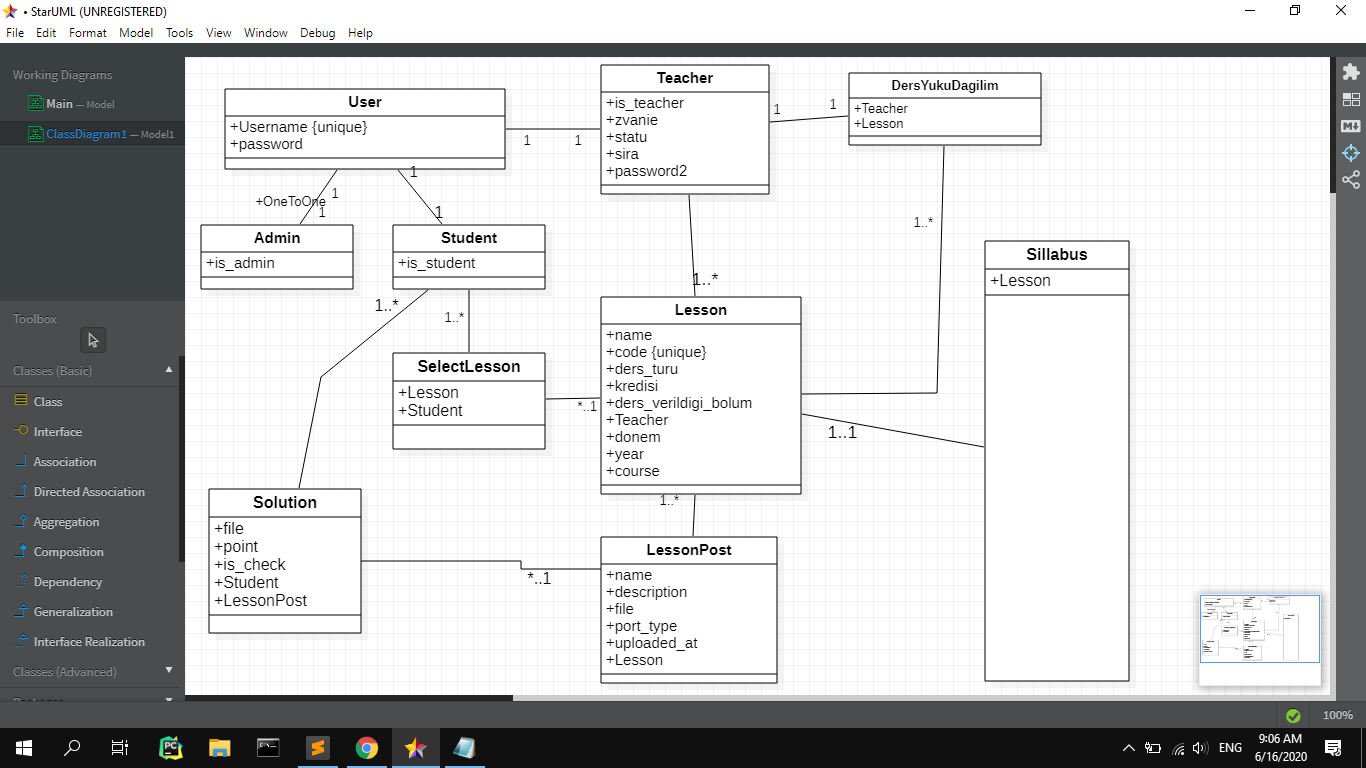
form = LoginForm()

return render(request, 'info/users/loginAdmin.html', {'form': form})

Жогоруда көрсөтүлгөн код колдонуучулар үчүн жазылган контроллер. Ал өзүнө мугалимдердин, студенттердин катталуусун жана бардык колдонуучулардын кирүү функцияларын камтыйт.

# §3.5 Модель

Төмөндө долбоордун берилиштер базасынын жалпы модели жана бири бири менен болгон байланыштары UML диаграмында чыгылдырылды:



*Сүрөт 2.Моделдин жалпы көрүнүшү жана байланышы*

Берилиштер базасындагы ар бир таблица үчүн өзүнчө класстардын модели түзүлөт. Жана бардык маалыматтар ушул берилиштер базасында сакталат. Сакталган маалыматтарга жетүү жана аларды башкаруу класстардын жардамы менен аткарылат. Төмөндө берилиштер базасы үчүн түзүлгөн класстар.

1. Колдонуучулар үчүн:

from django.contrib.auth.models import User

from django.db import models

class Student(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE, unique=True)

is\_student = models.BooleanField(blank=True, default=True)

def \_\_str\_\_(self):

return self.user.username

class Meta:

verbose\_name = 'Student'

verbose\_name\_plural = 'Students'

class Teacher(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE, unique=True)

is\_teacher = models.BooleanField(blank=True, default=True)

zvanie = models.CharField(max\_length=50)

statu = models.CharField(max\_length=50)

sira = models.IntegerField(unique=True)

password2 = models.CharField(max\_length=10)

def \_\_str\_\_(self):

return self.user.username

class AdminAdmin(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE, unique=True)

is\_admin = models.BooleanField(blank=True, default=True

def \_\_str\_\_(self):

return self.user.username

1. Сабактар үчүн:

from django.db import models

from users.models import Student, Teacher

import datetime

def current\_year():

return datetime.date.today().year

class Lesson(models.Model):

DERS\_TURU\_CHOICES = (

('Z', 'Zorunlu'),

('S', 'Seçmeli'),

)

GROUP\_CHOICES = (

('1', 'First course'),

('2', 'Second course'),

('3', 'Third course'),

('4', 'Fourth course'),

)

DONEM\_CHOICES = (

('G', 'Güz Dönemi'),

('B', 'Bahar Dönemi'),

)

name = models.CharField(max\_length=150)

code = models.CharField(max\_length=20, unique=True)

ders\_turu = models.CharField(max\_length=1, choices=DERS\_TURU\_CHOICES, blank=False, null=False, default='Z')

kredisi = models.CharField(max\_length=7)

ders\_yuku = models.IntegerField()

ders\_verildigi\_bolum = models.CharField(max\_length=50)

teacher = models.ForeignKey(to='users.Teacher', on\_delete=models.CASCADE, null=True)

donem = models.CharField(max\_length=1, choices=DONEM\_CHOICES)

year = models.IntegerField(default=current\_year)

course = models.CharField(max\_length=1, choices=GROUP\_CHOICES, blank=False, null=False)

def \_\_str\_\_(self):

return self.name

class DersYukuDagilim(models.Model):

teacher = models.OneToOneField(Teacher, on\_delete=models.CASCADE, unique=True)

lesson = models.ForeignKey(to=Lesson, on\_delete=models.CASCADE, blank=True, null=True)

# ------------SILLABUS------------

class Sillabus(models.Model):

lesson = models.OneToOneField(Lesson, on\_delete=models.CASCADE, unique=True)

ders\_adi\_kyr = models.CharField(max\_length=100)

ders\_adi\_tur = models.CharField(max\_length=100)

yari\_yil = models.IntegerField()

haftalik\_saat = models.CharField(max\_length=10)

kredisi = models.IntegerField()

akts = models.IntegerField()

seviyesi = models.CharField(max\_length=100)

ders\_veren = models.CharField(max\_length=100)

ders\_dili = models.CharField(max\_length=50)

yardimcilar = models.CharField(max\_length=150)

derslik = models.CharField(max\_length=50)

dersin\_amaci = models.CharField(max\_length=500)

saved = models.BooleanField(default=False)

# Dersin Kaynakları

class DersKaynaklary(models.Model):

kaynak = models.CharField(max\_length=250, blank=True)

kaynak\_turu = models.CharField(max\_length=50, blank=True)

sillabus = models.ForeignKey(to=Sillabus, on\_delete=models.CASCADE, null=True)

# Haftalık Ders İçeriği

class DersIcherigi(models.Model):

hafta = models.CharField(max\_length=10)

ders\_konusu = models.CharField(max\_length=150)

sillabus = models.ForeignKey(to=Sillabus, on\_delete=models.CASCADE, null=True)

# Dersin Öğrenme Çıktıları

class DersOgrenmeChytylari(models.Model):

text = models.CharField(max\_length=200)

sillabus = models.ForeignKey(to=Sillabus, on\_delete=models.CASCADE, null=True)

# Ölçme Değerlendirme Biçimi

class OlchmeBichimi(models.Model):

degerlendirme\_araci = models.CharField(max\_length=50)

adet = models.IntegerField()

katki = models.IntegerField()

katki\_yuzdesi = models.IntegerField()

sillabus = models.ForeignKey(to=Sillabus, on\_delete=models.CASCADE, null=True)

# ------------ END SILLABUS------------

class LessonPost(models.Model):

TYPE\_CHOICES = (

('1', 'Post'),

('2', 'Exercise')

)

name = models.CharField(max\_length=100)

description = models.TextField(blank=True, null=True)

file = models.FileField(upload\_to='upload/lesson\_post/', blank=True, null=True)

post\_type = models.CharField(max\_length=1, choices=TYPE\_CHOICES, blank=False, null=False, default='1')

uploaded\_at = models.DateTimeField(auto\_now\_add=True)

lesson = models.ForeignKey(to=Lesson, on\_delete=models.CASCADE, null=True)

def \_\_str\_\_(self):

return self.description[:20]

class Solution(models.Model):

file = models.FileField(upload\_to='upload/solutions/', blank=True, null=True)

point = models.IntegerField(default=0)

is\_check = models.BooleanField(default=False)

student = models.ForeignKey(to=Student, on\_delete=models.CASCADE, blank=True, null=True)

exercise = models.ForeignKey(to=LessonPost, on\_delete=models.CASCADE, blank=True, null=True)

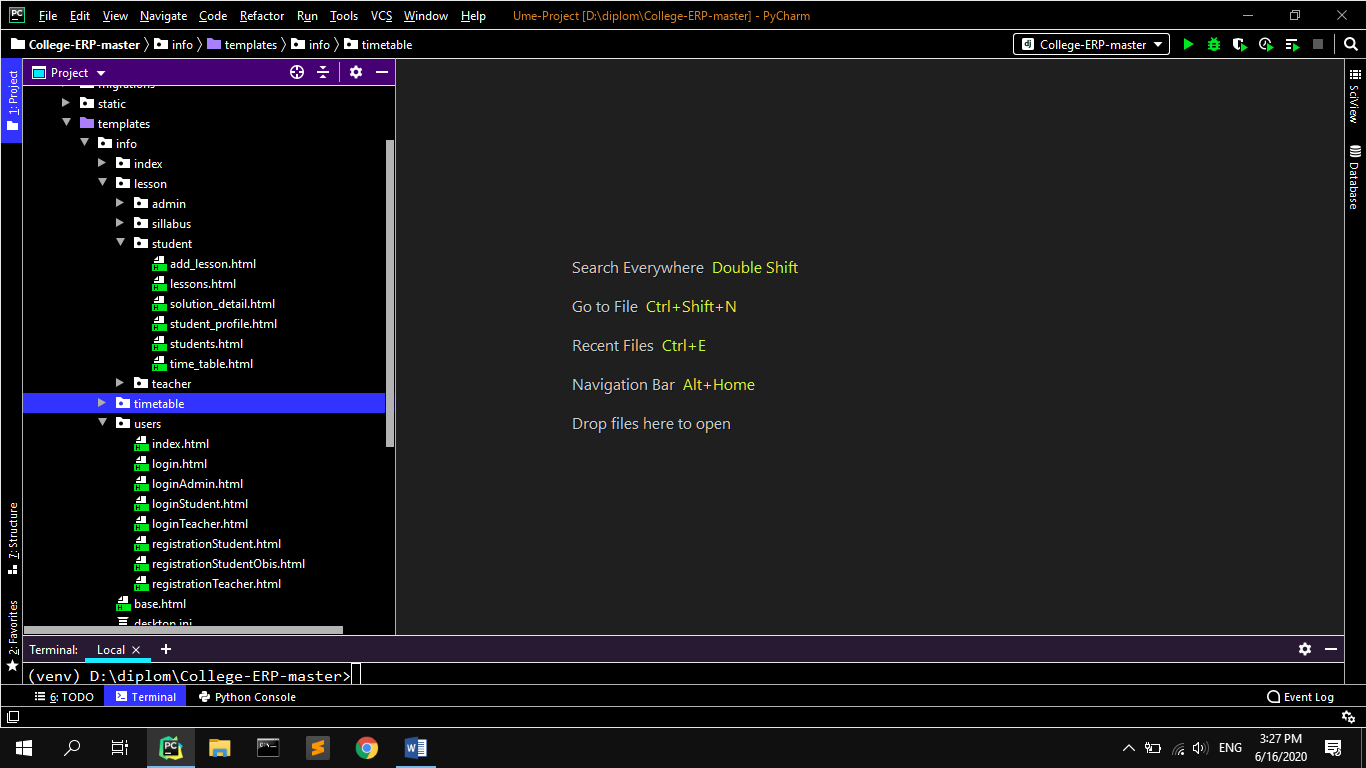
class SelectLesson(models.Model):

student = models.ForeignKey(to=Student, on\_delete=models.CASCADE, blank=True, null=True)

lesson = models.ForeignKey(to=Lesson, on\_delete=models.CASCADE, blank=True, null=True)

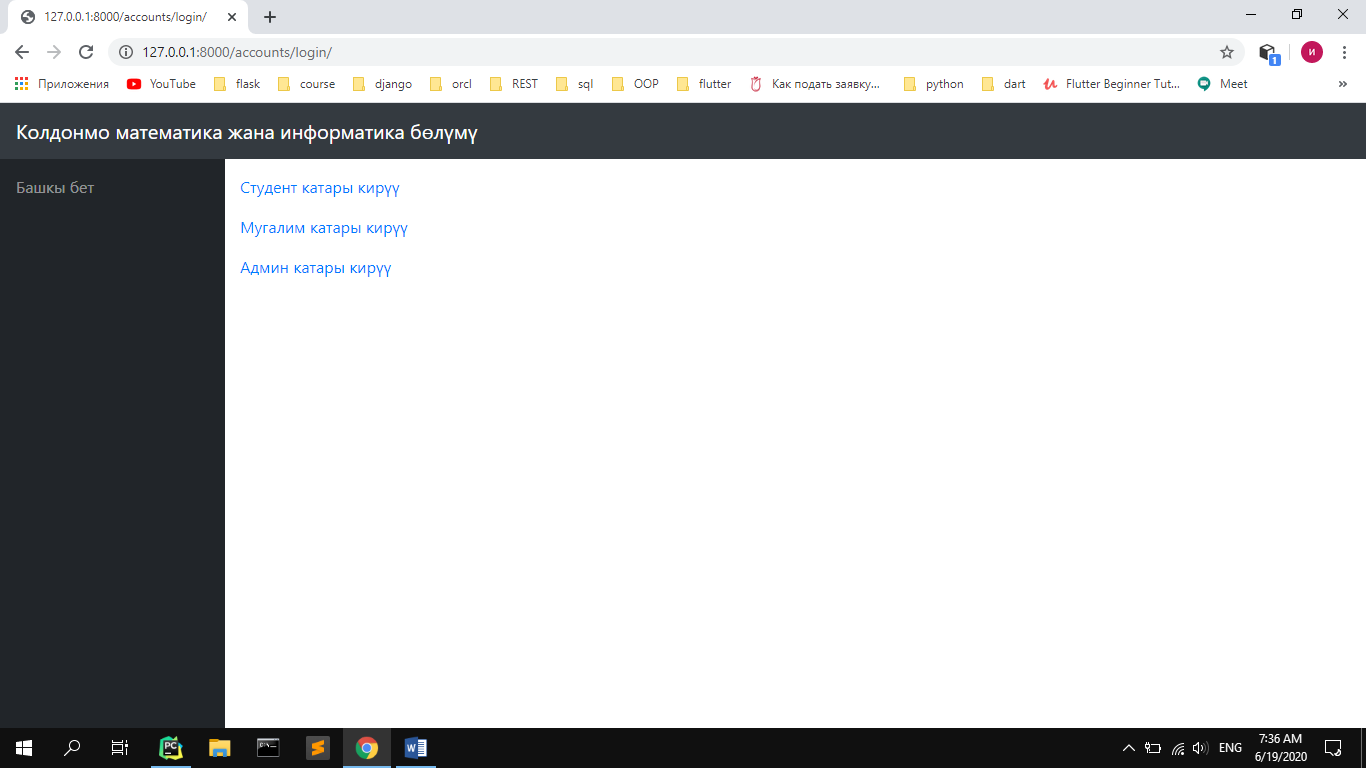
# §3.5 Шаблон

Django веб фреймворкунда шаблон катары template папкасындагы html файлдары каралат. Алар колдонуучу интерфейсин чагылдырып беришет. Ал интерфейсти кооздоо үчүн статикалык файлдар колдонулат. Алар static папкасында жайгашышкан. Шаблондун жана статикалык файлдардын долбоордогу жалпы көрүнүшү:

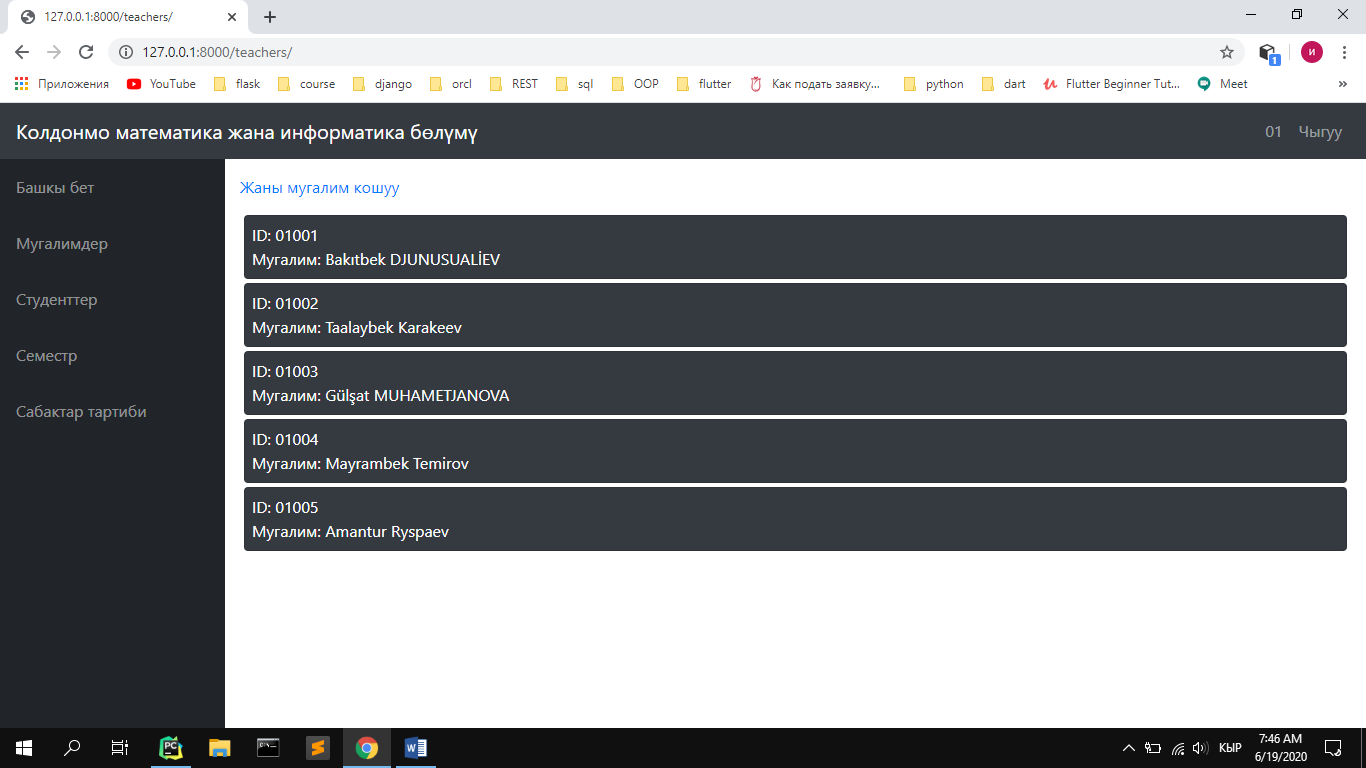


*Сүрөт 3.Шаблондордун көрүнүшү*

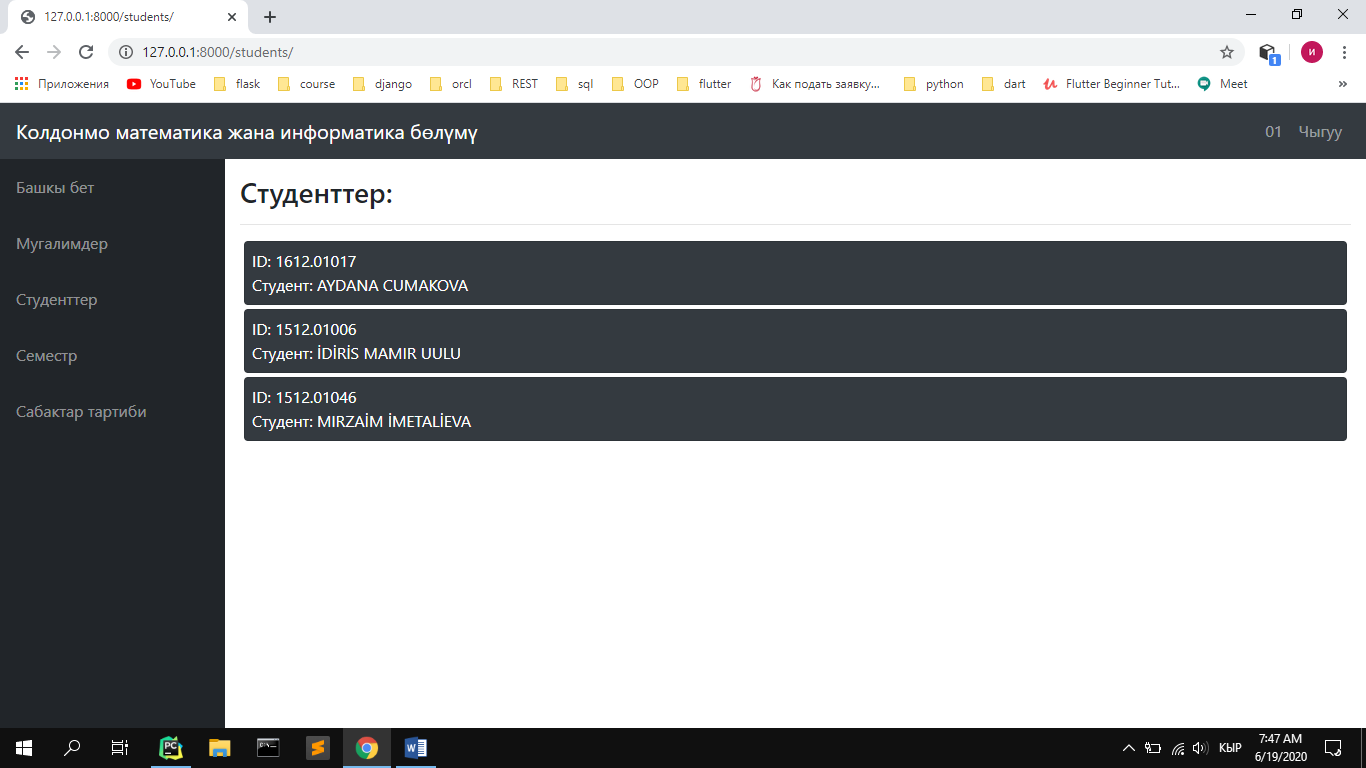
Долбоордун колдонуучуларга жеткиликтүү болгон интерфейстери:



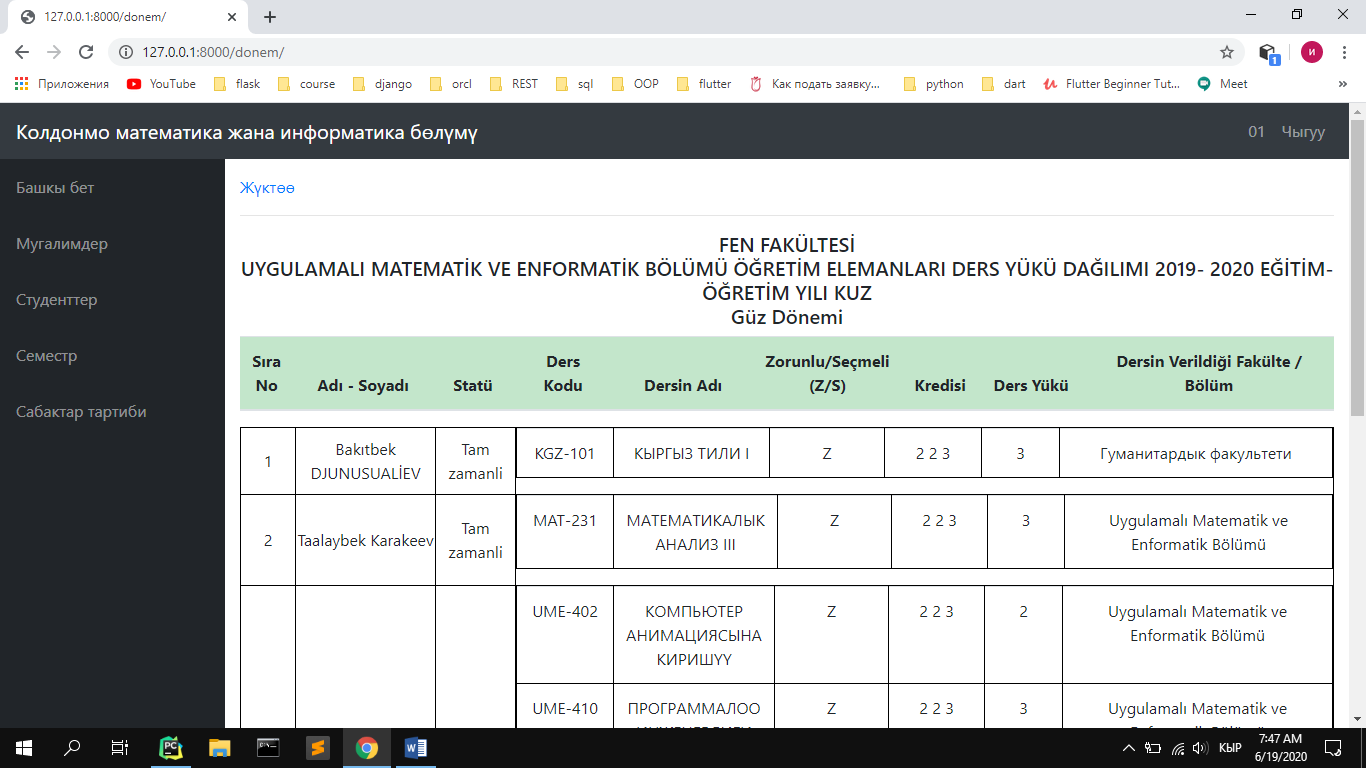
*Сүрөт 4.Башкы баракча.*



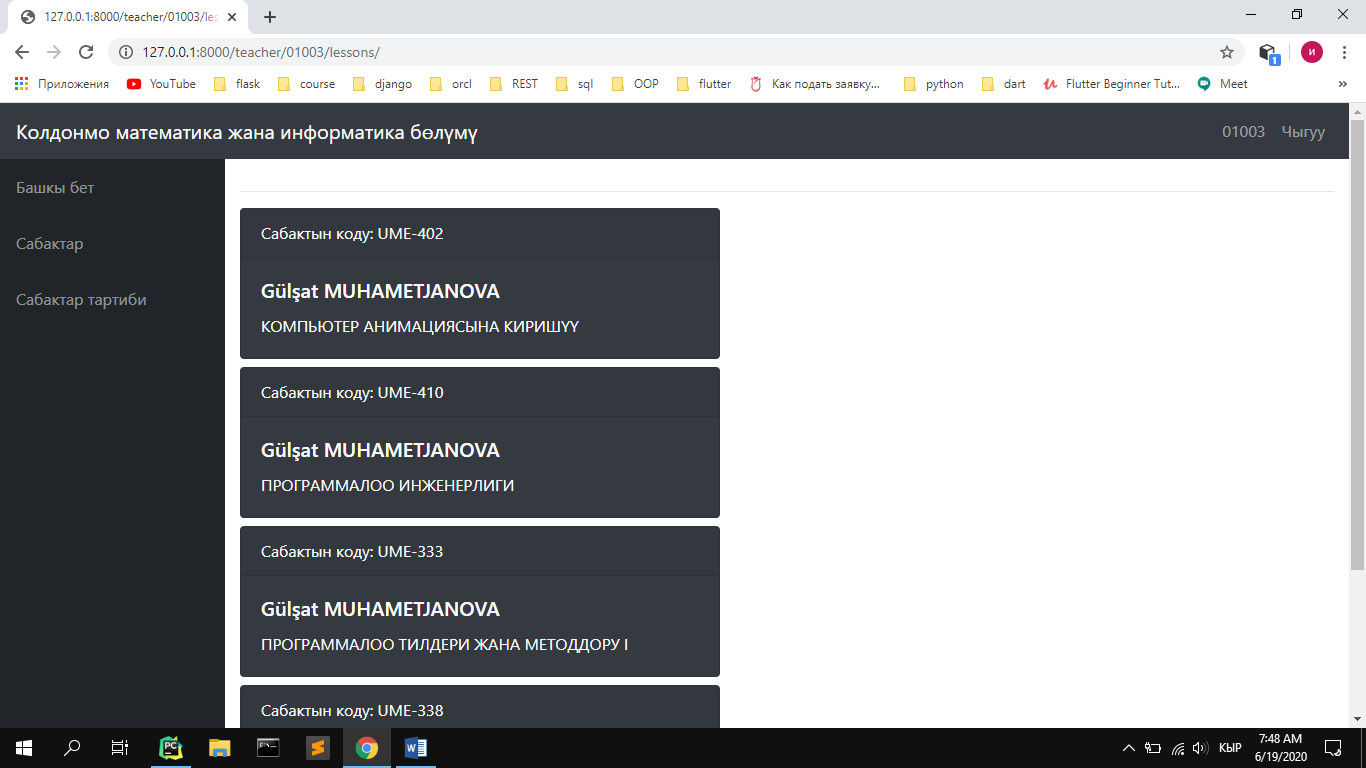
*Сүрөт 5.Админ баракчасы*



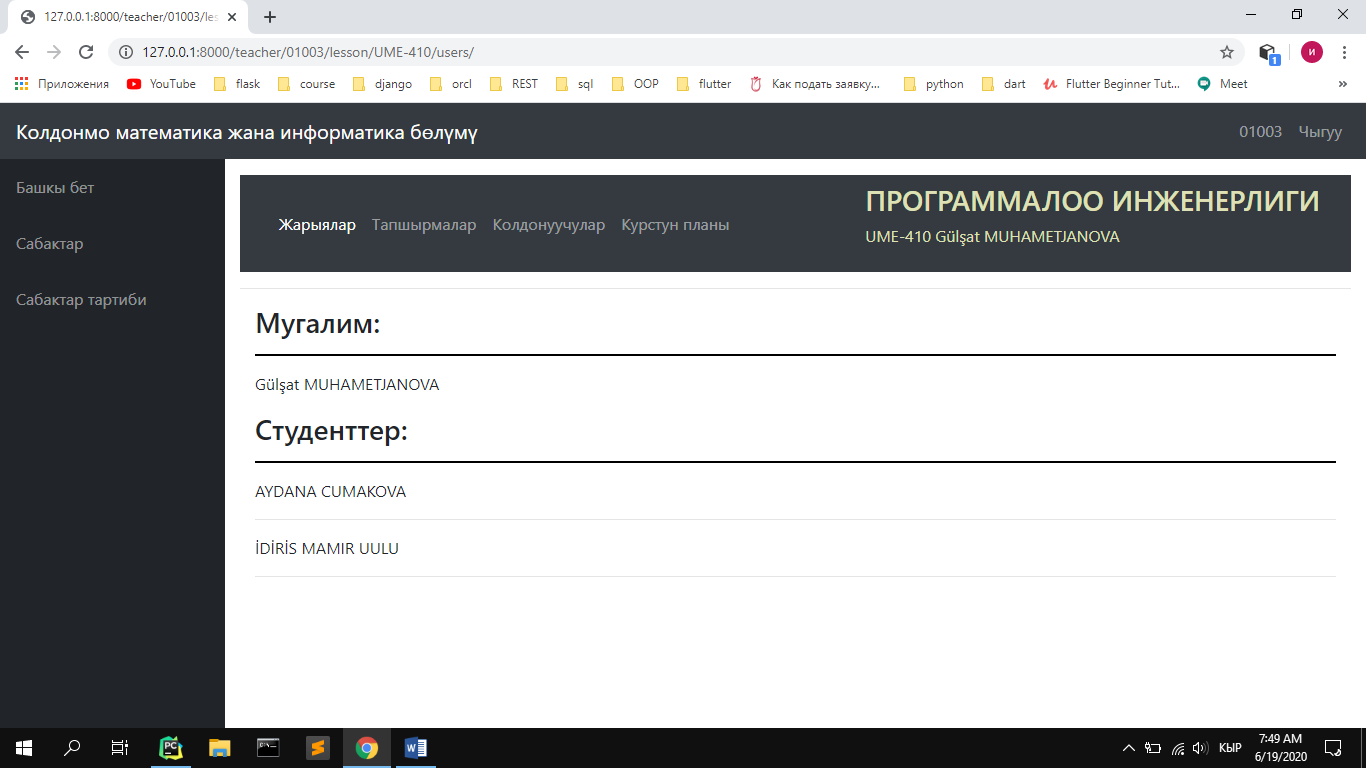
*Сүрөт 1.Админ баракчасы*



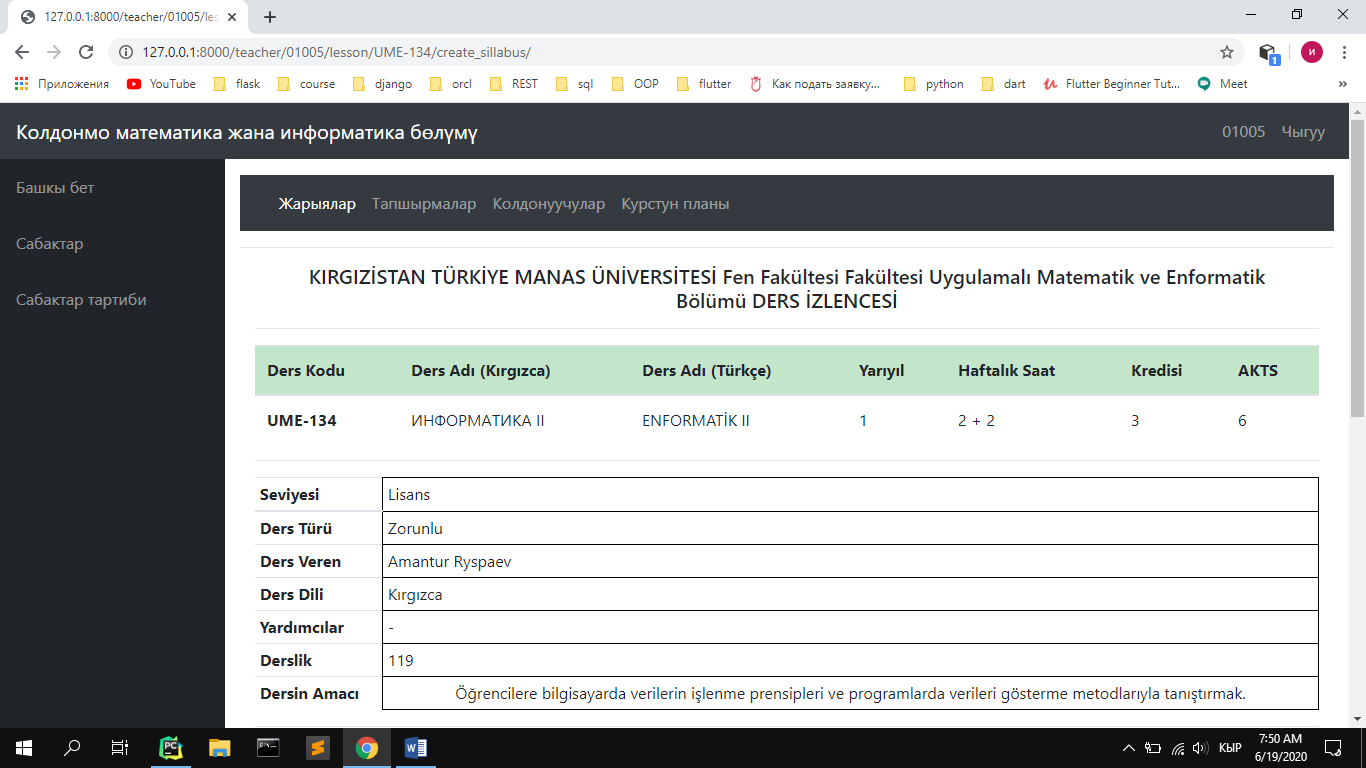
*Сүрөт 6.Админ баракчасы.*



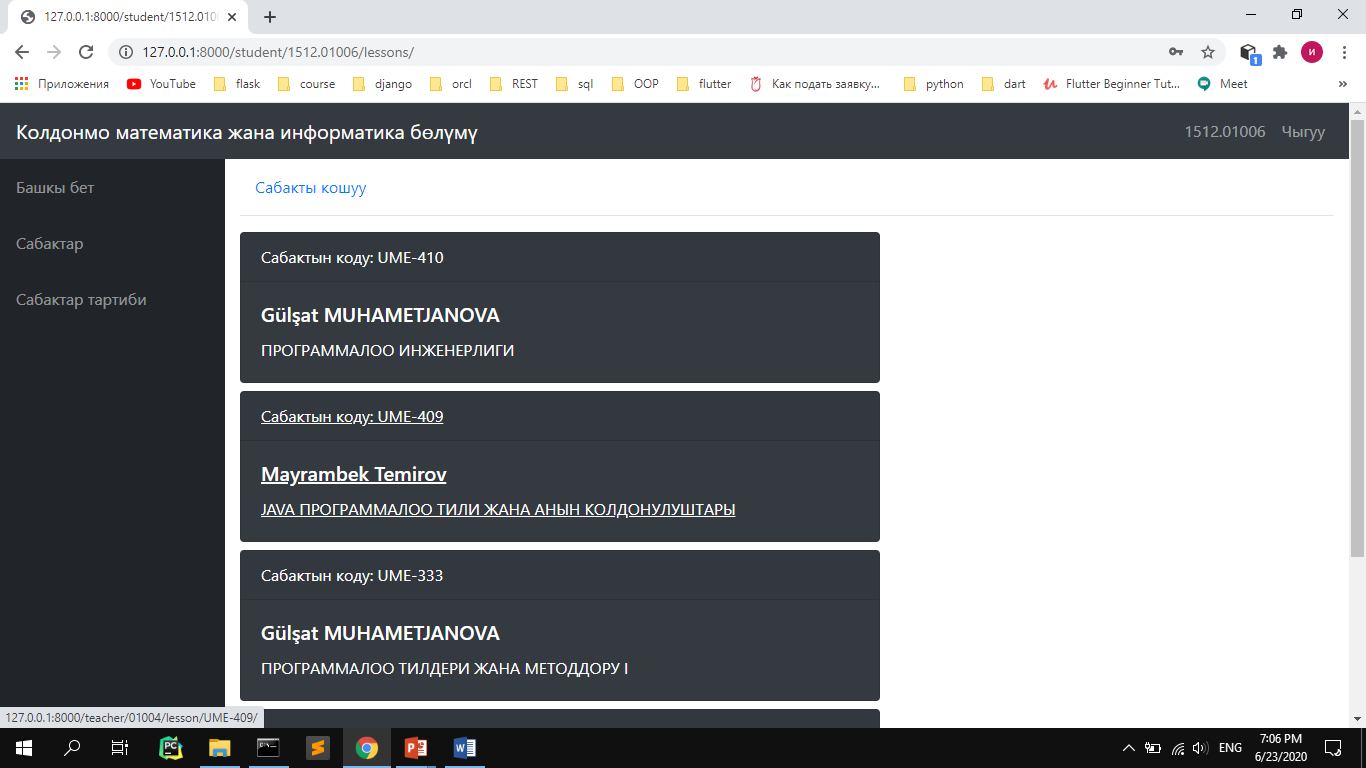
*Сүрөт 7.Мугалим баракчасы*.



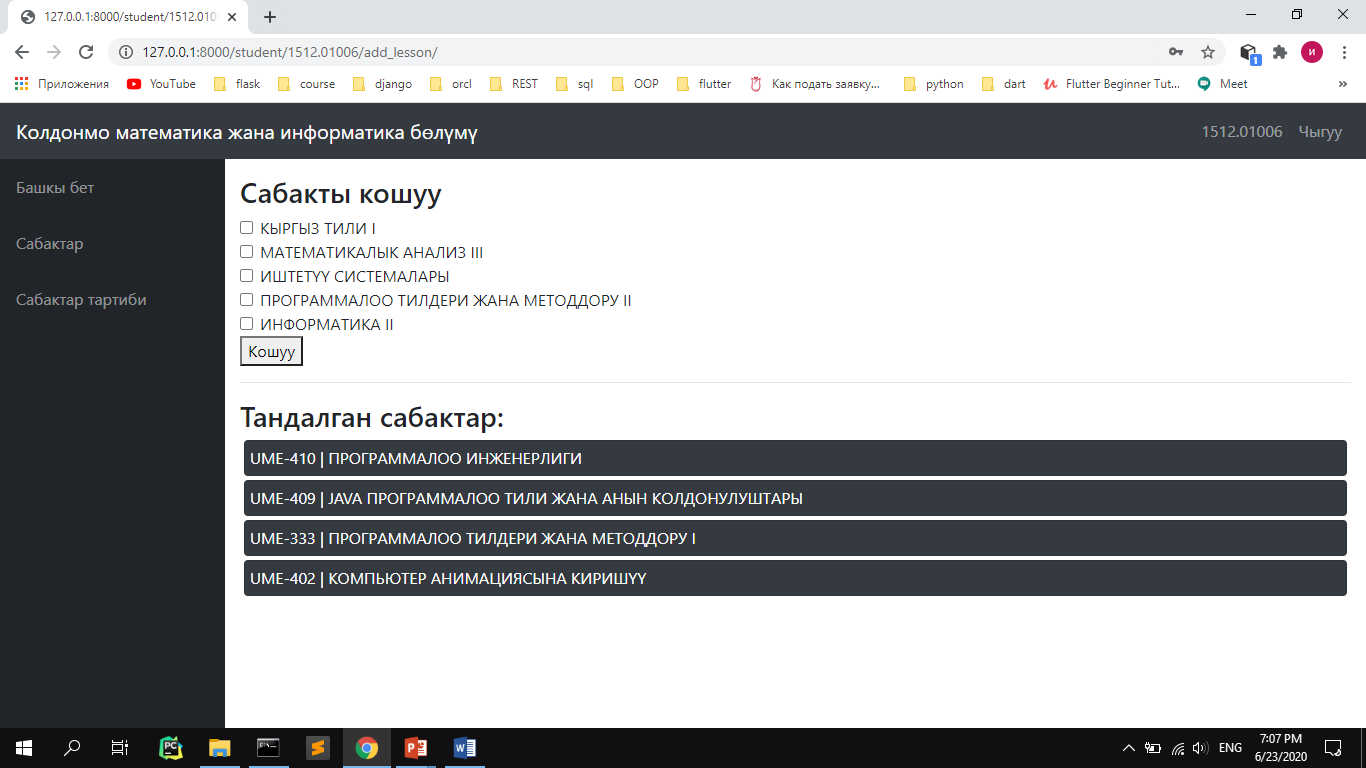
*Сүрөт 8.Мугалим баракчасы.*



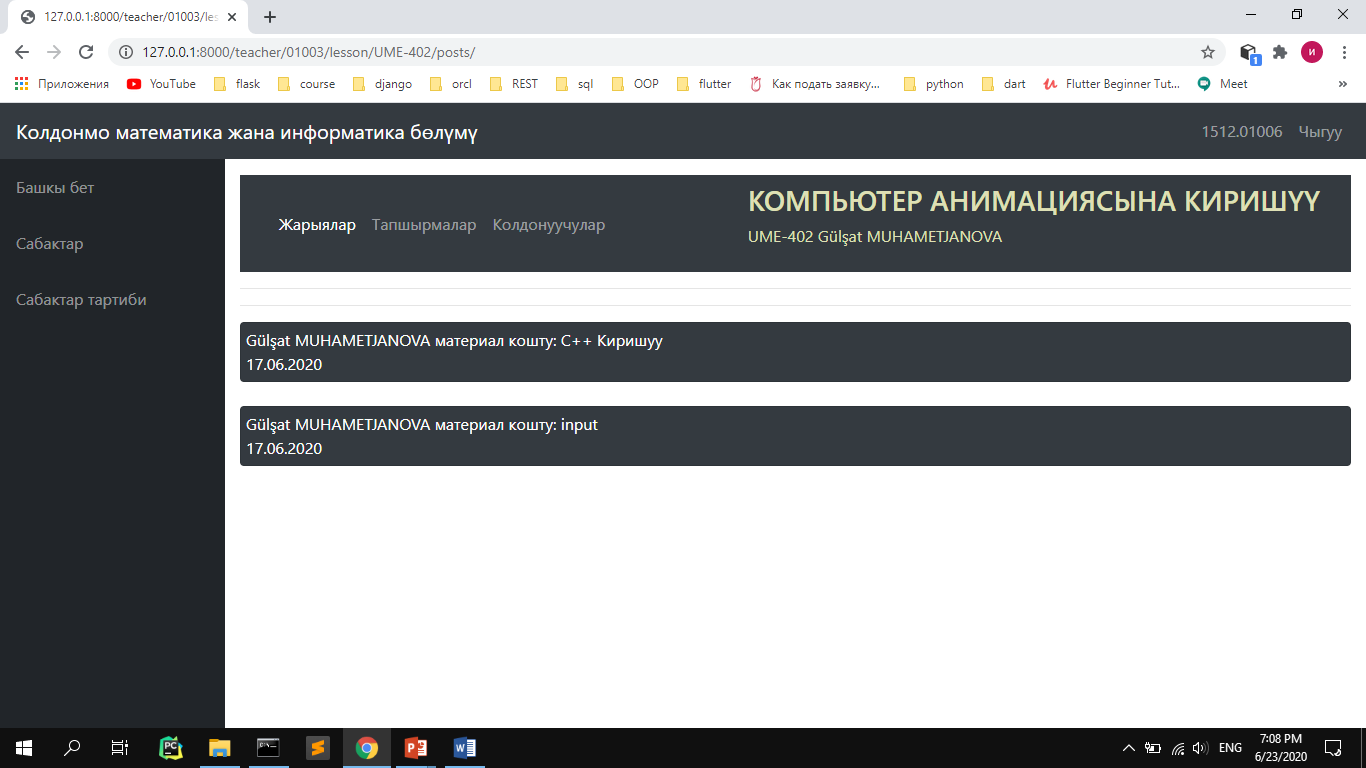
*Сүрөт 9.Мугалим баракчасы.*



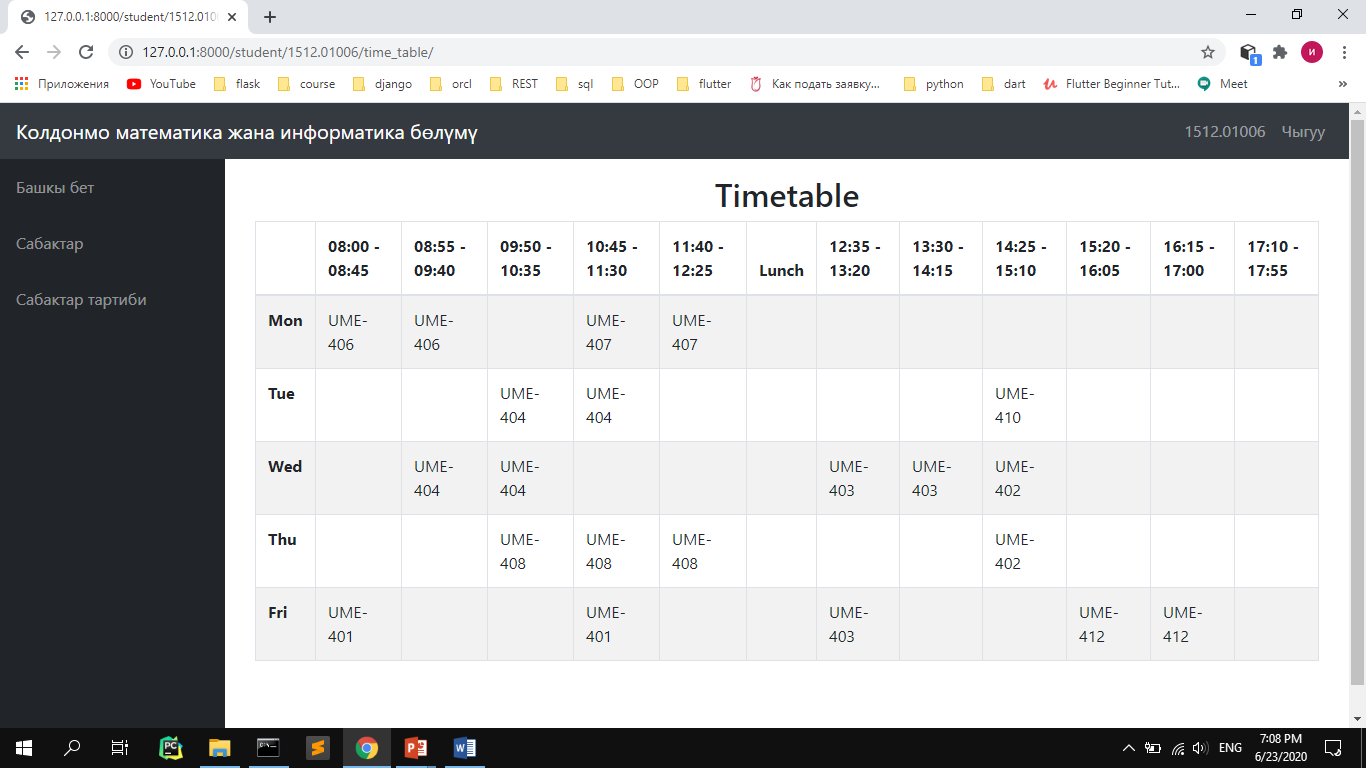
*Сүрөт 10.Студент баракчасы.*



*Сүрөт 11*.*Студент баракчасы.*



*Сүрөт 12.Студент баракчасы.*



*Сүрөт 13.Студент баракчасы*.

# КОРУТУНДУ

Бул тезис окуу жылдарында алынган билимимдин жана Python курулушунун жардамы менен ишке ашырылды.Изилдөөлөр бир канча убакытты, талыкпаган изденүүлөрдү, эмгекти талап кылды. Ошону менен бирге Python, Django, HTML, CSS, ORM, SQL, StarUML жана обьектке багытталган принциптери талданып, аларды дагы да болсо тереңирээк таанууга өбөлгө түзүлдү.

Өзүнчө бөлүп кароодон кийин “адам да дарак сыяктуу, бийиктеш үчүн тереңдеш керек” дегендей окуу планында алынган маалыматтардан сырт, көптөгөн жаңы маалыматтар менен таанышып, жумуштун практика бөлүгүн кароо да бир кыйла жеңил болду. Кылдат архитектуралык дизайн жана колдонмонун актуалдуу чечимдерин кошуу бөлүмдүн веб сайтын тамыры күчтүү башкаруу платформасына айлантты. Бул долбоордун максаты окулган предметтердин колдонулушун жана өз өзүнчө турган платформалардын биргеликте ийгиликтүү иштөө комбинациасын көрсөтүү болгон.

# ЫРААЗЫЧЫЛЫК БИЛДИРҮҮ

Бул ишти аткарууда карантин убактысына карабай өзүнүн убактысын бөлүп консультация берген жана акыл-кеңештерин айткан, ойлору менен бирге билген билими менен бөлүшкөн жана бизге түшүнүү менен мамиле жасаган биздин колдонмо математика жана информатика бөлүмүнүн бөлүм башчысы, проф.доктор Асан Өмүралиевге терен ыраазычылыгымды билдирем.Ошондой эле ар дайым ар тараптан бизге жол көрсөткөн жана кеп-кеңештерин айткан биздин бөлүмүбүздүн агай-эжейлерине да чексиз ыраазычылыгымды билдирем.

# КОЛДОНУЛГАН АДАБИЯТТАР

1. Доронов В.А (2019) . **Django 2.1 Практика СОЗАаНИЯ веб-сайтов на Python**. Санк-Петербург.
2. Codesido, I. (2009). *What is front-end development?* Guardian official website: https://www.theguardian.com/help/insideguardian/2009/sep/28/blogpost
3. MDN. (2016). *What is a web server?* MDN official website: https://developer.mozilla.org/en-US/docs/Learn/Common\_questions/What\_is\_a\_web\_server
4. MSDN. (2017). *Server-side web frameworks* . MSDN official website: <https://developer.mozilla.org/en-US/docs/Learn/Serverside/First_steps/Web_frameworks>
5. Yalcin, M. (2017). *SQLite Nedir ve Nasıl Kullanılır?* Karamelek: <http://www.karamelek.org/sqlite-nedir-ve-nasil-kullanilir/>
6. Zamirbekov, K. (2017). *Django*. Wikipedia: <https://ky.wikipedia.org/wiki/Django>
7. Ziade, T. (2019). *Python Packaging*. 2020-ж. aosabook: <https://www.aosabook.org/en/packaging.htm>
8. *Understanding the MVC pattern in Django*. 2020-ж*.*

<https://medium.com/shecodeafrica/understanding-the-mvc-pattern-in-django-edda05b9f43f>