ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

Зинеден Аңсар Сағидоллақызы

Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеу

ТҮСІНДІРМЕ ЖАЗБА
дипломдық жобаға

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

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
ПИ кафедрасының меңгерушісі
физ.-мат. ғыл.канд., профессор
_____ Молдагулова А.Н.
"____" _____2022 ж.

ТҮСІНДІРМЕ ЖАЗБА
дипломдық жобаға

Тақырыбы: «Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеу»

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

Орындаған Зинеден А.С.

Рецензент
Доктор PhD
_____ Досжанова А.А.
"____"_____ 2022 ж.

Ғылыми жетекші
қауымдастырылған профессор
_____ Мукажанов Н.К.
"____"_____ 2022 ж.

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ БІЛІМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

БЕКІТЕМІН
ПИ кафедрасының меңгерушісі
физ.-мат. ғыл.канд., профессор
_____ Молдагулова А.Н.
"____" _____2022 ж.

Дипломдық жобаны орындауға
ТАПСЫРМА

Білім алушыға Зинеден Аңсар Сағидоллақызы
Тақырыбы: «Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеу»
Академиялық мәселелер жөніндегі проректор бұйрығының №_____"___"_____2021 ж. шешімімен бекітілген.
Орындалған жобаның өткізу мерзімі " ___ " _____ 2022 ж.
Дипломдық жобаның бастапқы мәліметтері: Жобаның төлқұжаты, технология бойынша техникалық құжаттама, техникалық тапсырма, жоба диаграммлары түрінде ақпаратты жинау, деректер қорына сақтау, тестілеу, тексеруге арналған программалық қамтамаларды жасау жүргізілген.
Есеп – түсініктеме жазбаның талқылауға берілген сұрақтардың тізімі:
а) тақырып бойынша талдау және есептің қойылымын жасау;
b) жобаны жобалау және пәндік сала бойынша талдау _
в) пайдаланушы интерфейсін жобалау және дамыту;
г) бағдарламаны құру, кітапханаларды қосу, деректерді қосу және тестілеу;
Графикалық материалдар тізімі (міндетті суреттердің нақты көрсетілуімен):
презентацияның 20 слайдпен берілген құжат түрінде ұсынылған.
Ұсынылған негізгі әдебиеттер: 25 пайдаланылған әдебиеттер тізімінен

Дипломдық жобаны орындау
КЕСТЕСІ

Бөлімдердің атаулары, зерттелген мәселелердің тізімі	**Ғылыми жетекшіге және кеңесшілерге ұсыну мерзімі**	**Ескерту**
1. Дипломдық жобаның жоспарын құру	17.01.2022	орындалды
2. Тапсырма қойылымы және программалау ортасын таңдау	24.01.2022	орындалды
3. Зерттеу тақырыбы бойынша ғылыми теориялық материалдарды жинау және негізгі бөлім бойынша есеп беру жазбасын дайындау	10.02.2022	орындалды
4. Дипломның екінші және үшінші бөлімдерін, жобалау сызбаларын дайындау	11.03.2022	орындалды
5. Жобаның веб-қосымшасын тестілеуден өткізу	06.04.2022	орындалды
6. Дипломдық жобаға түсіндірме жазба жазуды аяқтау	27.04.2022	орындалды

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

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

Ғылыми жетекші _____ Мукажанов Н.К.

Тапсырманы орындауға қабылдап алған студент _____ Зинеден А.С.

Күні «__» _____2021 ж.

АҢДАТПА

Бұл дипломдық жұмыс жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеуге арналған. Ұсынылған шешім студент пен кітапхана арасындағы байланысты жеңілдетуге, оқырман үшін тез іздеу және қол жеткізу процесстерін жылдамдатуға мүмкіндік береді. Оқырман үшін кітапхана кеңсесіне келместен, қашықтан қол жеткізу мүмкіндіктері ұйымдастырылады.

Бұл дипломдық жобада кітапхананың жұмыс барысы туралы деректерді жинау және сақтау бизнес-процестерді автоматтандыру мәселелері қаралады. Пайдаланушылардың рөлі мен мүмкіндіктерін визуалды түрде көрсетуге Use Case, BPMN диаграммалары пайдаланылады. Жүйе интерфейсін жасауға Figma құралдары пайдаланылады. Деректер базасын жобалау үшін SQLite пайдаланылды, Django фреймворкі негізінде веб-қосымша іске асырылады.

АННОТАЦИЯ

Данная дипломная работа посвящена разработке библиотечного репозитория для вузов. Предлагаемое решение позволяет упростить связь между студентом и библиотекой, ускорить процесс быстрого поиска и доступа для читателя. Для читателя организуются возможности удаленного доступа без посещения библиотечного офиса.

В данном дипломном проекте рассматриваются вопросы автоматизации бизнес-процессов сбора и хранения данных о ходе работы библиотеки. Для визуального отображения роли и возможностей пользователей используются диаграммы Use Case, BPMN. Для создания интерфейса системы используются инструменты Figma. Для проектирования базы данных использовался SQLite, на базе фреймворка Django реализовано веб-приложение.

ANNOTATION

This thesis is devoted to the development of a library repository for the university. The proposed solution makes it possible to simplify the communication between the student and the library, speed up the process of quick search and access for the reader. Remote access opportunities are organized for the reader without visiting the library office.

This diploma project deals with the automation of business processes for collecting and storing data on the progress of the library. Use Case and BPMN diagrams are used to visually display the role and capabilities of users. Figma tools are used to create the system interface. SQLite was used to design the database, a web application was implemented based on the Django framework.

МАЗМҰНЫ

	Кіріспе	9
1	Пәндік саланы талдау	11
1.1	Пәндік саланың сипаттамасы	11
1.2	Кітапхана репозиторийі бойынша шолу	13
1.3	Кітапхана ісінде қолданылатын типтік ақпараттық жүйелерге шолу	14
1.4	Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеуге арналған техникалық тапсырма	17
2	Арнайы бөлім	22
2.1	Жүйенің функционалдарын анықтау	22
2.2	Жүйенің бизнес процесстері	22
2.3	Бағдарламалық жасақтама құрылымы	24
3	Кітапхана репозиторийін әзірлеу	28
3.1	UML тілін пайдаланып жобалау	28
3.2	Кітапхана репозиторийінің интерфейсімен танысу	34
	Қорытынды	40
	Пайдаланылған әдебиеттер тізімі	41
	Қосымша А	43
	Қосымша Б	45

КІРІСПЕ

Кітапхана қызметін дамыту, оқырмандарға барынша сапалы қызмет көрсету үшін озық технологиялар мен процестерді қолданудан басталмақ. Автоматтандырылған ақпараттық жүйелерді енгізу – қазіргі заман тенденциясы ғана емес, кітапхана қызметкерлерінің еңбек өнімділігі мен сапасын арттыруға жақсы негіз, оқырмандарға қажетті деректер мен ақпаратты дер кезінде жеткізудің тиімді жолы. Бұл дипломдық жұмыстың тақырыбы ЖОО кітапханасының электронды кітаптар қоймасын жасау болып табылады. Дипломдық жұмыстың мақсаты кітапхана бизнес-процестерінің бір бөлігін автоматтандыратын ақпараттық жүйені енгізу, электронды нұсқаларын қолжетімді түрге келтіру болып табылады.

Жұмыс тақырыбының өзектілігі - қоғам дамуының қазіргі кезеңінің ерекшелігі, яғни адамдардың ақпаратты дәстүрлі баспа түрінде де, электронды түрде де қолдануы. Сонымен қатар, заманауи ақпараттық технологиялар осы уақыққа дейін адамзат жинаған ақпаратты электронды түрге кеңінен аударуға ғана емес, сонымен бірге бірден электронды түрде көптеген жаңа ақпараттық ресурстарды құруға мүмкіндік берді. Ақпаратты ұсынудың бұл формасы байланыс процестерін едәуір жеделдетумен қатар, ақпаратты өндіру, сақтау және тарату процестерін сапалы жаңа деңгейде ұйымдастыруға мүмкіндік береді.

Ақпараттық әдістер мен технологиялар іс-әрекеттің барлық түрлеріне тереңірек енеді. Ақпараттандыру, ақпараттық, коммуникациялық технологиялардың және мультимедианың конвергенциясы, ғылым мен білім беру саласында қазіргі заманғы ақпараттық жүйелерді қолдануға көшу білімді алу мен жинақтаудың, оларды тарату мен пайдаланудың түбегейлі жаңа деңгейін қамтамасыз етеді [1].

Пайдаланушылардың электрондық ақпараттық ресурстарға қашықтан қол жеткізуін қамтамасыз ету ғылымға, білім мен мәдениетке ақпараттық қызмет көрсетудің бірінші кезектегі міндеттерінің бірі болып табылады. Қазіргі жағдайда, тек баспа құралдарына негізделген ақпараттық қызмет көрсету тұжырымдамасы ескіргенін және оны басқасымен алмастырғанын дәлелдеуді қажет етпейді - пайдаланушының уақыты мен орналасқан жеріне қарамастан, шексіз мөлшерде таратылатын және ғаламдық деректер желілері арқылы лезде қол жетімді әр түрлі ақпараттың электронды ұсынылуына негізделген.

Осыған байланысты ақпараттың негізгі қоймасы және таратушысы кітапхана әлеуметтік институттың рөлі мен функциялары айтарлықтай өзгереді.

Ғылым, мәдениет және білім беру қажеттіліктері үшін ақпараттық - кітапханалық қызмет көрсетудің бүгінгі күні электрондық құжаттар коллекцияларынан ғана емес, сонымен қатар ғаламдық компьютерлік желілерді қолдана отырып іздеу, талдау және оған қол жеткізу мақсатында әртүрлі ақпаратты өндіруге, сақтауға және ұйымдастыруға бірыңғай көзқарасты жүзеге асыратын жүйені білдіретін электронды кітапханаларды (digital libraries) құру арқылы тиімді екендігі жалпыға белгілі.

Дәстүрлі кітапхана мен электрондық кітапхананың басты айырмашылығы - пайдаланушы құжаттарға сілтеме жасау үшін ғана емес, қажетті ақпарат алу үшін соңғы қызметтерге жүгінеді. Электрондық кітапхананың ерекшелігі - әртүрлі іздеу механизмдері мен электронды деректердің жинақтарына қол жеткізу құралдарын параллель пайдалану мүмкіндігі. Электрондық кітапхана сұранысына жауап ретінде пайдаланушыға жалғыз электрондық құжат ұсынылуы мүмкін және мәтіндік түрде міндетті емес болғандықтан, нысандар коллекциясындағы ақпаратты тиімді кешенді іздеу мен талдауды қамтамасыз ететін ақпараттық жүйелер қажет.

Электрондық кітапханаларда ақпараттық ресурстарды ұйымдастыру ерекшеліктері мен міндеті:

- электрондық ресурстарды ұйымдастырудың жалпы мәселелерінің сипаттамасы;
- интернет ресурстардың өзара іс-қимылы және электрондық басылымдар каталогтарын қалыптастыру процесіне шолу;
- электрондық кітапханалардың ұйымдық-технологиялық негіздерінің даму үрдістерін қадағалау;
- электрондық кітапханаларды құру тұжырымдамасын және олардағы ақпараттық ресурстарды ұйымдастыру принциптерін талдау.

Дипломдық жұмыс кіріспе теориялық, арнайы және практикалық бөлімдерінен, қорытындыдан, пайдаланылған әдебиеттер тізімінен және қосымшалардан тұрады. Кіріспеде өзектілігі негізделіп, жұмыстың объектісі, пәндік аймақ, жұмыстың мақсаты мен міндеттері айқындалады. Бірінші бөлімде кітапхананың теориялық негіздері қарастырылды, кітапхана ісінде қолданылатын типтік ақпараттық жүйелерге шолу және кітапхана автоматтандыру объектісі ретінде қарастырылды. Екінші бөлімде кітапхана қызметін ұйымдастыру ерекшеліктері қарастырылды, тапсырманы қою үшін ақпараттық жүйеге қойылатын бастапқы талаптар айқындалды және қажетті программалық жабдықтар сипатталды. Үшінші бөлімде ақпараттық жүйені жобалау, жүйені енгізу сипатталды.

1 Пәндік саланы талдау

1.1 Пәндік саланың сипаттамасы

Кітапхана (грек тілінен аударғанда – кітап және қойма, сыйымдылық, қорап) – жалпыға ортақ пайдалану үшін және жазба еңбектерді жинайтын, сақтайтын, сондай-ақ анықтамалық-библиографиялық жұмыстарды жүзеге асыратын мекеме.

Заңды мағынада кітапхана деп әдетте қайталанатын құжаттардың ұйымдастырылған қоры бар және оларды жеке және заңды тұлғаларға уақытша пайдалануға беретін ақпараттық, мәдени, білім беру мекемесі деп түсіндіріледі.

Негізгі ұғымдар:

Кітапханашы – арнайы кітапханалық білімі бар және кітаптарды уақытша беруге және қайтаруға, оқырмандардың картотекасын жүргізу және тағы да басқа бірқатар кәсіби кітапханалық операцияларды орындайтын қызметкер.

Кітапхана ісі – білім беру, мәдени-ағартушылық және ақпараттық қызметтің бағыты. Оның міндеттеріне мыналар кіреді:

– кітапханалар желісін құру және дамыту;
– кітапхана қорын қалыптастыру, жинақтау және өңдеу;
– кітапхана пайдаланушыларына кітапханалық-ақпараттық, сондай-ақ, анықтамалық-библиографиялық қызмет көрсетуді ұйымдастыру;
– кітапхана қызметкерлерін оқыту;
– кітапханалардың ғылыми-әдістемелік дамуын қамтамасыз ету.

Кітапхана қызметі адамның шығармашылық және басқа да интеллектуалдық қызметін жүзеге асырудың негізін құрайды.

Құжат - кітапхананың қарапайым пайдаланушысы негізінен ақпараттың белгілі бір түрде құрылымдалған және материалданған бөлігі. Құжат ақпараттандырудың ең кең тараған түрі болып табылады.

Ақпарат - бұл ұғымның көптеген анықтамалары бар. Кең және құқықтық мағынада ақпаратты тұлғалар, объектілер, фактілер, оқиғалар, құбылыстар мен процестер туралы олардың берілу формасына қарамастан ақпарат ретінде анықтауға болады [13].

Кітапхана оқырманы (абонент, пайдаланушы, ақпаратты тұтынушы) – белгіленген құжаттардағы ресми жазба негізінде кітапхана қызметін алатын тұлға. Кітапханатану үшін бұл ұғымның негізгі мәні оқырман кез келген кітапхана қызметінің негізгі объектісі болып табылады. Барлық кітапханалардың жұмысы оны жүргізу міндеттеріне бағынады [13].

Кітапхананы жүйе ретінде қарастырғанда келесі ішкі жүйелерді бөліп көрсетуге болады: кітапхана қоры (КҚ), жазылушылар контингенті (ЖК), кітапхана қызметкерлері (КП) және материалдық-техникалық базасы (МТБ).

Кез келген кітапхананың үш негізгі қызметі бар:

– кітап қорын жинақтау;
– кітап қорын ұйымдастыру;

– оқырманға қызмет көрсету.

Кітапхана қорын жинақтау (қалыптастыру) кітапханаға қажетті басылымдарды анықтаудан және оларды иемденуден тұрады. Оқырмандарға қызмет көрсету деңгейі кітапхананы қалыптастырудың уақытылы және толықтығына айтарлықтай тәуелді [3].

Кітап қорын ұйымдастыру әдебиеттерді есепке алу, реттеу, сақтау және оны оқырманға ұсыну мәселелерімен айналысады. Қорды сауатты ұйымдастыру оқырманның әдебиетті пайдалануын жеңілдетеді, ал кітапханашы оқырман талабын тез әрі сапалы орындауға мүмкіндік береді, сонымен қатар қорлардың қоғамдық меншік ретінде сақталуын қамтамасыз етеді [3]. Кітапханалардың оқырмандарға қызмет көрсету қызметі екі негізгі нысанда жүзеге асады. Оқырманға жазылу пайдаланушыға өзінің толық иелігінде белгілі бір мерзімге кітапханадан баспа басылымын алу құқығын береді. Екінші жағдайда оқырман кітаппен кітапхананың арнайы бөлінген бөлмесі – оқу залында ғана таныса алады.

Оқырмандарға қызмет көрсету негізгі нысандардан басқа мыналарды қамтуы мүмкін:

– жеке оқырмандар мен мекемелерге қажетті әдебиеттерді таңдауға көмектесу;
– кітапхана каталогтары жүйесі арқылы кітапхананың кітап қорын ашу;
– әртүрлі типтегі ақпараттық-библиографиялық нұсқаулықтарды құрастыру;
– мәдениет, әдебиет тұрғысынан ерекше құндылықты насихаттау;
– оқырмандардың сұранысы бойынша мәтіндерді қайта шығару және т.б.

Кез келген кітапхананың маңызды сипаттамасы оның қорының құрылымы болып табылады. Әдетте, оқырмандар сұранысына ие басылымдардың бір бөлігі көпшілікке арналған және оқырман онымен тікелей кітап сөресінде таныса алады, ал басқа, сұранысы аз басылымдар кітап репозитарийінде орналасқан. Бірқатар жағдайларда оқырманның арнайы сұрауын немесе рұқсатын талап ететін ерекше сирек, бүлінген немесе мемлекеттік құпия басылымдарды шығаруға шектеулер қойылады. Әрбір кітапқа өзінің инвентарлық нөмірі беріледі. Бұл кітапхана қорының есебін жүргізу үшін қажет. Әр оқырман үшін әдебиеттерді беру үшін есеп құжаты ретінде қызмет ететін бланк толтырылады. Ол оқырман туралы мәліметтер мен шығарылған басылымдар туралы ақпаратты тасымалдайды.

Жазылуды ұйымдастыру техникасы. Кітапхана абонементі оқырмандарға белгілі бір мерзімге кітап беруді қамтамасыз етеді. Кітапхана басылымдарын пайдалану мерзімі «Кітапхананы пайдалану ережесімен» белгіленеді [1]. Жазылым бойынша жұмыс жалпы қызмет көрсету негізінде құрылады. Жалпы жазылыммен оқырмандар бір шығарылым бөлімінде әдебиет алады. Оқырмандарға үйден берілетін барлық басылымдар төмендегі ақпараттарды көрсете отырып, бланкіге жазылады: шығарылған күні, инвентарлық нөмірі, авторы және басылым атауы. Жазылушылар басылымдарды алуға қол қояды [4]. Оқырмандардың бланкілері көбінесе қайтару кезеңіне сәйкес орналасады және

қайтару кезеңінде олар оқырман нөмірлері бойынша немесе тегі бойынша әліпби бойынша сұрыпталады. Басылымды қайтарған кезде кітапханашы оқырманның қатысуымен оның оқырман бланкісінде жазылған түбіртегін сызып тастауға міндетті.

Оқу залын ұйымдастыру техникасы. Кітапханаларда әдетте жалпы оқу залы болады. Бөлек оқу залы жоқ кітапханаларда үй ішінде жұмыс істеуге арналған үстелдер орнатылады [4]. Оқу залында шығарылған басылымдарды оқырман бланкілеріне жазу міндетті емес. Оқырман бланкісінде шығарылған күні мен оқырман қолы көрсетілген басылымдардың бланкілері қоса беріледі. Оқырман нысанындағы шығару шығарылған басылымдардың жалпы санын көрсету арқылы ресімделеді [5]. Оқырмандардың оқу залындағы бланкілері, әдетте, тегі бойынша алфавиттік ретпен орналасады. Оқырман өзіне алған басылымды белгіленген мерзімде қайтармаса, ол борышкер болып саналады. Жазушы, сондай-ақ кітаптарды кешіктіріп қайтармағаны үшін кітапхананы пайдалану қағидаларында белгіленген мерзімге кітапхананы пайдалану құқығынан айырылуы мүмкін. Оқырман қандай да бір себептермен басылымын жоғалтса, кітапханашы одан жоғалғанның орнына бірдей басылымды немесе оған теңестірілген басылымды талап етуге құқылы.

Оқырмандардың кітапхана қорын пайдалану белсенділігі, сонымен қатар кітапхана атқарған жалпы жұмыстың өнімділігі, қызметкерлер жұмысының көлемі мен тиімділігі «кітапты несиелендіру» сияқты көрсеткішпен сипатталады. Бұл көрсеткіш статистиканы жүргізу үшін өте маңызды [1].

Кітапхана ісіндегі тағы бір маңызды көрсеткіш – сабаққа қатысу. Ол кітапхананың атқарған жұмысының көлемін де, оқырман белсенділігін де сипаттайды.

Әрине, әрбір кітапхананың өзіндік ерекшеліктері, құрылымы мен оқырмандарға қызмет көрсету тәсілі бар екенін естен шығармауымыз керек, бірақ олардың барлығы да адамдардың ақпараттық, мәдени-ағартушылық және демалыс қажеттіліктерін қанағаттандыруға бағытталған.

1.2 Кітапхана репозиторийі бойынша шолу

Кітапхана қызметін дамыту оқырмандарға барынша сапалы қызмет көрсету үшін озық технологиялар мен процестерді енгізуден ажырата алмайды.

Кітапхананы автоматтандыру қол еңбегін көп қажет ететін процестерді автоматты түрде орындау және жүргізу үшін техникалық құралдарды пайдалану қажеттілігімен анықталады, осылайша кітапхана қызметкерінің уақытын үнемдейді. Бұл пайдаланушыларға қажетті ақпаратты жылдам беруге мүмкіндік береді.

Заманауи әлемдегі кітапхана – бұл әр түрлі элементтерді қамтитын күрделі механизм, олар бір жерде жинақталған қызмет, ақпарат, материалдық-

техникалық және адами ресурстарды жедел және сапалы қамтамасыз ету мәселелерін шешу үшін біріктіреді. пайдаланушыларға қажетті ақпарат.

Кітапхана – ақпаратты тұтынушыларға (пайдаланушыларға) бағытталған жүйе. Оған қызмет көрсететін персонал, кітапхананың өзіндегі және одан тыс жерде де әртүрлі көздерде (мысалы, электронды) сақталатын қорларға орналастырылған ақпараттар кіреді. Сонымен бірге кітапханадағы қызмет тиісті материалдық-техникалық құралдарды пайдалана отырып, іздеу жүйесі қызметін атқара алады [3].

Автоматтандырылған ақпараттық жүйелерді енгізу – заман тенденциясы ғана емес, кітапхана қызметкерлерінің еңбек өнімділігі мен сапасын арттыруға жақсы негіз, оқырмандарға қажетті деректер мен ақпаратты дер кезінде жеткізудің тиімді жолы.

Кітапханаларда заманауи есептеуіш құралдарды пайдалану қызмет көрсету уақытын айтарлықтай қысқартуға мүмкіндік береді. Автоматтандыру монотонды және қайталантын операциялардың көпшілігін жоюға ықпал етеді, кітапхана қызметкерлерінің ыңғайлылығы мен өнімділігін арттырады, пайдаланушыларға ақпаратпен жұмыс істеудің заманауи тәсілдерін ұсынады [4].

Автоматтандыру процесіне ғылыми-техникалық жетістіктерді пайдалану, кітапханаларда электронды каталогтар мен мәліметтер қорын құру, сондай-ақ телекоммуникацияларды, интернетті кеңінен пайдалану арқылы қол жеткізіледі, сол арқылы пайдаланушыларға әртүрлі ақпараттық және басқа да қызметтерді қалыптастырады.

Кітапханалық процестердің маңызды бөлігін автоматтандыруға болатындығын түсіну маңызды. Автоматтандыру құралдары бірден бірнеше тапсырманы орындауға мүмкіндік беретінін атап өткен жөн. Мысалы, қорды алу процесін автоматтандыру мыналарға мүмкіндік береді:

- кітапхана қызметкерлерін, мысалы, мерзімді басылымдар мен әдебиеттерге тапсырыс беру орындарына бару және сатып алу қажеттілігімен байланысты уақытты айтарлықтай шығындаудан босату;
- жалпы қызмет көрсету мәдениетін арттыру;
- тапсырыс берілген құжаттарды кітапханаға жеткізуге байланысты физикалық шығындарды азайту;
- жазылушының кітапты күту уақытын қысқарту, бұл да жұмысты орындау тиімділігіне қол жеткізуге мүмкіндік береді [4].

Автоматтандыру құралдарын пайдаланудың артықшылықтары мен мүмкіндіктері кітапхананың АЖ енгізу бойынша кейінгі іс-шараларға негіз болды.

1.3 Кітапхана ісінде қолданылатын типтік ақпараттық жүйелерге шолу

Ақпараттық жүйе (АЖ) өзі тиесілі ұйымдағы ақпаратты жинайтын, өңдейтін және тарататын процедуралардың, адамдар мен ресурстардың жиынтығы ретінде түсініледі [6].

Ақпараттық жүйе, кез келген динамикалық жүйе сияқты, үш негізгі құрамдас бөліктен тұрады:

- енгізу (деректерді қабылдайды және жинақтайды);
- өңдеу (енгізілген деректерді түрлендіреді, сол арқылы ақпараттық өнімдерді жасайды);
- шығару (алынған нәтижелерді қамтамасыз етеді). енгізілген деректерді өңдеу) [6].

Автоматтандырылған АЖ (ААЖ) бар деректерді ақпараттық өнімге түрлендіру үшін компьютерлік технологияларды, бағдарламалық қамтамасыз етуді, телекоммуникацияларды және ақпараттық технологиялардың (АТ) басқа түрлерін пайдаланады. Ақпараттық жүйелер операцияны тиімді орындауға және тұтастай ұйымды басқаруға көмектесетін ақпаратты қамтамасыз ету үшін қолданылады. Кітапхана саласында қолданылатын ААЖ автоматтандырылған кітапханалық-ақпараттық жүйе (АКАЖ) деп аталады [5].

Автоматтандырылған кітапханалық - ақпараттық жүйе классификациясы айтарлықтай қиын, бұл классификацияны құрастыру кезінде жалпы байланыстар мен түсініктерді қалыптастырудың әртүрлі тәсілдерін қолдануға болатындығымен, сондай-ақ олардың күрделілігі мен ауқымдылығымен түсіндіріледі. Фасеттік классификация ең қолайлы болып табылады, ол автоматтандырылған кітапханалық - ақпараттық жүйе мүмкіндіктерінің әртүрлі топтарын анықтауға негізделген [5].

Мысалы, автоматтандырылған кітапханалық-ақпараттық жүйені келесі топтарға бөлуге болады:

- Енгізілетін және өңделетін ақпараттың түрі мен сипаты бойынша:
  - библиографиялық;
  - фактілік;
  - толық мәтінді;
  - анықтамалық;
  - аралас.
- Масштаб бойынша автоматтандырудың 3 түрін бөлуге болады. Олар:
  - күрделі (максималды) , үлкен және өте үлкен кітапханаларға тән, сыртқы ақпараттық ресурстарға қолжетімділікті қамтамасыз етеді;
  - ішінара (орта көлемдегі кітапханаларда қолданылады, ол кітапхананың негізгі кітапханалық-ақпараттық қызметін қамтиды – қорды есепке алу және өңдеу, оқырмандарға қызмет көрсету және т.б.);

- минималды (шағын кітапханаларда қолданылады, оның негізгі мазмұны сыртқы ақпараттық ресурстарға қолжетімділікті қамтамасыз ету болып табылады) [3].

– Құжаттардың ақпараттық массивтерін алу профилі бойынша:
  – әмбебап;
  – әртараптандырылған;
  – терең тақырыптар.
– Автоматтандырылған кітапханалық - ақпараттық жүйе ішкі жүйелерінің құрамына және міндеттеріне және оларды жүйеге енгізу сипатына қарай:
  – интеграцияланған;
  – біріктірілмеген.

Бірінші нұсқада автоматтандырылған кітапханалық - ақпараттық жүйе кітапханалық-ақпараттық қызметтің барлық негізгі операцияларын қамтиды және оның барлық элементтері бір-бірімен тығыз байланыста және өзара әрекетте болады.

– Функционалдық мақсаты бойынша:
  – Бір пайдаланушы;
  – Желі.
– Қолданылатын желілік технологиялар архитектурасы бойынша:
  – файл-сервер;
  – клиент-сервер.
– Жұмыс ауқымы бойынша:
  – Автономды (жергілікті);
  – таратылған (корпоративтік);
  – ғаламдық ақпараттық жүйелердің технологияларын қолдану (Интернет-технологиялар).
– Бейімделу қабілеті бойынша (өзгерту қабілеті):
  – ашық;
  – жабық;
  – икемді.

Автоматтандырылған кітапханалық - ақпараттық жүйе функционалдық құрылымын объектілер мен олардың арасындағы байланыстардың жиынтығы ретінде көрсетуге болады. Автоматтандырылған кітапханалық - ақпараттық жүйенің ішкі жүйелері осындай объектілер ретінде әрекет етеді. Ішкі жүйе, жалпы ғылыми тілмен айтқанда, тұқымдық- түрлік сипаттамасына, тіршілік ету жағдайларына, мақсатына, өзара әрекеттесуі мен қызмет етуіне қарай одан оқшауланған кез келген жүйенің бөлігі ретінде анықталады [2].

Ішкі жүйе жүйенің бір немесе бірнеше негізгі немесе қосымша функцияларын орындай алады. Ішкі жүйе өзінің негізгі белгілері бойынша басқа күрделірек жүйеге кіретін жүйе де бола алады. Автоматтандырылған кітапханалық - ақпараттық жүйе тұтас жүйенің құрамдас бөліктері ретінде бірнеше нақты функционалдық ішкі жүйелерден тұрады, олар кітапханалық

технологияларға ұқсас кейбір функцияларды жүзеге асырады, мысалы, құжаттарды алу, өңдеу, сақтау, жеткізу, есепке алу, басқару [3].

Функционалды ішкі жүйелердің өзара байланыстары бар, автоматтандырылған кітапханалық - ақпараттық жүйе оларды бір бүтінге біріктіреді және олар бір-бірінен тәуелсіз жұмыс істейді.

1.4 Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеуге арналған техникалық тапсырма

1.1-кесте – Техникалық тапсырма кестесі

№	Позиция	Түсіндіру	Қосымша ақпарат
1	Бұл бағдарламалық жасақтаманың негізгі міндеті электронды және баспа кітаптарға тапсырысты беру, іздеу және қолдау болып табылатын жалпы жүйеге арналған модуль болып табылады		
Бағдарламалық қамтамасыз ету функционалдығы			
1	Бұл модуль барлық қолданушыларға электронды түрде кітаптарды көруге мүмкіндік беруге арналған.	Әдебиеті іздеу, көру	
2	Бұл модуль әдебиеттерді есепке алу және сақтау процесін автоматтандыруға арналған	Әдебиеттердің есебін жүргізу; Ақпараттық әдебиеттерді толықтыру; Әдебиет туралы ақпаратты өңдеу; Әдебиетті іздеу; Әдебиетті жою	

1.1-кесте – жалғасы

№	Позиция	Түсіндіру	Қосымша ақпарат
3	Бұл модуль тұтынушы картасын іздеуді жеңілдетуге арналған	Баспагерлерді тіркеу; Іздеу; Өңдеу; Жою	
4	Бұл модуль сұранысқа қызмет көрсетуді автоматтандыруға арналған	Іздеу; Тапсырыспен ағымдағы әрекеттер; Сәтсіздік және аяқталу болғаны туралы хабарлама	Сұраныс құру; Өңдеу; Жою
5	Оқырмандар қоры бөлімінің пайдаланушыларының карточкалары	Жеке мәліметтерді толықтыру бөлімі	
Функционалдық талаптар			
1	Әдебиет есебі	Әдебиеттерді толықтыру; Әдебиет туралы ақпаратты өңдеу; Әдебиет іздеу; Әдебиетті жою	
2	Баспагерлерді есепке алу	Қосу; Өңдеу; Баспа үйін және оның тапсырыстарын іздеу; Жою	
3	Тапсырысты басқару	Қосу; Тапсырыспен ағымдағы әрекеттер; Сәтсіздік және орындалу туралы хабарлама	
4	Оқулықтарды іздеу	Тапсырыс жасау; Іздеу	
Пайдаланушылар			
1	Қызметкерлер бөлімі	Кітаптарды, баспаларды, анықтамаларды редакциялау; Жою; қосу	
2	Студенттер	Кітаптарға, баспаларға және басқа да баспа өнімдерін көру, сұраныс жасау.	

1.1-кесте – жалғасы

№	Позиция	Түсіндіру	Қосымша ақпарат
3	Әкімші (администратор)	Кітаптарды, баспаларды, анықтамаларды редакциялау; Оқырмандар қорының толықтыру бөлімінің карточкаларындағы адамдарды жою; қосу; өңдеу және жою	
Қолдану шегі мен ережелері			
1	Сұранысты өңдеу	Сұрауды өңдеу тұтынушыға жіберілгенге дейін мүмкін болады	Яғни, тұтынушы сұраған баспаларды қосу, өзгерту немесе жою
Болжамдар мен тәуелділіктер			
1	Белгілі бір кітапты сұраудан бас тарту саны		
2	Кітапты белгілі бір уақыт аралығында пайдалану		
Бағдарламалауға және пайдаланушы интерфейстеріне қойылатын талаптар			
1	Әрбір пайдаланушы рөлінде өзінің интерфейсі болуы керек	Интерфейс қарапайым және пайдаланушыға түсінікті болуы керек	
2	Негізгі түймелердің стандартты орналасуы	Интерфейс қарапайым және пайдаланушыға түсінікті болуы керек	

1.1-кесте – жалғасы

№	Позиция	Түсіндіру	Қосымша ақпарат
3	Сенімділік,ақауларға төзімділік	Бағдарлама бұзылмауы және күтпеген нәтижелер болмауы керек	
4	Жылдам іздеу	Кітапты табуға аз уақыт бөлу	
Деректерді сақтау талаптары			
1	Әзірленген бағдарламалық қамтамасыз ету тұтынушылар, пайдаланушылар және әдебиеттер туралы деректердің сақталуын қамтамасыз етуі керек		
Жүйелік қауіпсіздік талаптары			
1	Әзірленген бағдарламалық қамтамасыз ету тұтынушы мен пайдаланушылар туралы әдебиет деректерінің қауіпсіз сақталуын қамтамасыз етуі керек		

2 Арнайы бөлім

2.1 Жүйенің функционалдарын анықтау

Жоғары оқу орындары мен зерттеу институттарында репозиторийлердің кең таралуы ғылыми коммуникациялар жүйесін қолдайтын таратылған, жаһандық желілік инфрақұрылым үшін негіз жасайды. Қазіргі уақытта репозиторийдің бағдарламалық жасақтамасын таңдауға, оны орнатуға және конфигурациялауға арналған көптеген әдебиеттер бар.

Репозиторийлердің платформалары жиырма жыл бұрынғы, тіпті интернет, әлеуметтік медиа, веб және мобильді құрылғылар пайда болғанға дейін жасалған технологиялар мен хаттамаларды қолданады. Репозиторийлер желісінің құндылығын арттыру үшін олардың функционалдық мүмкіндіктерін кеңейту қажет.Бұл дипломдық жобада қоршаған ғылыми - ақпараттық инфрақұрылыммен белсенді өзара іс-қимыл жасауға есептелген қазіргі заманғы репозиторийді жобалауға қолданылуы мүмкін тәсілдер сипатталған.

Экономикалық тұрғыдан алғанда, жазылым бағасы да, мақаланы жариялауға дайындау үшін төлем мөлшері де жоғары және қолайсыз қарқынмен өсуді жалғастыруы мүмкін. Сонымен қатар, халықаралық баспа жүйесінде қол жетімділік пен қатысу тұрғысынан айтарлықтай қиындық бар. Зерттеушілерді дәстүрлі баспа орталықтарында жариялауға мәжбүр ететін жүйеге енгізілген ынталандыру тек осы мәселелерді күшейтеді. Біз жаһандық таратылған репозиторийлер желісін ғылыми зерттеулердің нәтижелерімен алмасудың тұрақты жүйесін құру үшін пайдалануға болады деп санаймыз. Репозиторийлер бүкіл әлем бойынша зерттеу нәтижелеріне қол жеткізуді қамтамасыз ете алады, сондай-ақ әрбір ғалым мен институтқа ғылыми зерттеулердің жаһандық желісіне қатысуға мүмкіндік береді. Зерттеушілердің пайдалану көрсеткіштерін бағалауға, рецензиялауға және бір-бірімен байланысуға арналған қосымша қызметтерді құру ғылыми ортадағы репозиторийлердің маңыздылығы мен танымалдылығын арттырады. Репозиторийлер ғылыми коммуникациялар жүйесінің жұмыс істеуін қолдайтын жаһандық желілік инфрақұрылымның негізіне айналуға тиіс. Ғылыми қоғамдастықтың өзі басқаратын осы инфрақұрылым аясында зерттеулерді одан әрі ынталандыру және инновацияларды енгізу үшін қосымша қызметтердің толық спектрін жасауға болады.

2.2 Жүйенің бизнес процесстері

Ұйымдстырылған бақылау немесе ғылыми ресурстарды (препринттер, постпринттер, зерттеу деректері, қосалқы бағдарламалық қамтамасыз ету және т.б.) және ғылыми инфрақұрылымды басқару — репозиторийді құру кезінде

маңызды қағидат болып табылады. Репозиторийді құру кезінде әр түрлі аймақтар мен ғылым салаларының әртүрлі қажеттіліктері мен жағдайларын ескеру қажет. Зияткерлік ашықтық және қолжетімділік ғылыми ресурстар мүмкіндігінше ыңғайлы форматта ұсынылуға және олардың құндылығын арттыратын және ғылыми қоғамдастықтың да, бүкіл қоғамның да мүддесіне кеңінен қолдануға мүмкіндік беретін баршаға қолжетімді болуға тиіс. Тұрақтылық мекемелер мен ғылыми-зерттеу ұйымдары ресурстардың ұзақ мерзімді тұрақтылығына ықпал ететін жаһандық желінің негізгі қатысушылары болуы керек. Функционалдық үйлесімділік репозиторийлер институттар арасындағы өзара іс-қимылды қамтамасыз ететін және оларға сыртқы қызмет жеткізушілерін бірлесіп пайдалануға мүмкіндік беретін дамудың жалпы бағыттарын, функционалдық мүмкіндіктер мен стандарттарды ұстануға тиіс.

Репозиторийді жобалаудың негізгі принциптері. Ғылыми ресурстар мен олардың метадеректері URI мекен-жайлары бар тең веб-ресурстар ретінде қарастырылатын тәсіл олардың арасында тиімді байланыс орнатуға мүмкіндік береді. Біз барынша кеңінен қолданылатын технологияларды, шешімдерді және мысалдарды таңдаймыз. Іс жүзінде бұл біз мүмкін болған жерде стандартты веб-технологияларды қолдануды жөн көреміз дегенді білдіреді. Эволюция революцияның орнына қолданыстағы инфрақұрылымды тиімді пайдалану үшін бүкіл әлемде кеңінен қолданылатын бағдарламалық жасақтама мен жүйелерді түзету арқылы шешімдерді жасаған жөн. Халықаралық стандарттарға сүйене отырып, бай, күрделі және әр түрлі шешімдерді қолданудың орнына, мүмкін болған жерде танылған конвенциялар мен стандарттарды барынша пайдалануға тырысу керек. Жаңа стандарттарды сыртқы жүйелермен өзара әрекеттесудің мүмкін қиындықтарын азайту үшін нақты қажеттіліктер туындаған кезде ғана енгізу керек. Пайдаланушылармен қай жерде болса да өзара әрекеттесу пайдаланушылардан өздері жұмыс істейтін ортаны тастап, біздің жүйелеріміздің бірімен жұмыс істеуді емес, өз құралдарымызды пайдаланушылар бұрыннан бар орталар мен жүйелерге біріктіргіміз келеді.

Репозиторий әр түрлі ресурстарға қол жетімділікті қамтамасыз етуі керек, соның ішінде жарияланған мақалалар, басып шығарылған ақпарат, мәліметтер жиынтығы, жұмыс құжаттары, суреттер, бағдарламалық жасақтама және т.б. құрылған репозиторий осы сипаттамаларға ие болуы керек. Репозиторийдің ресурсқа бағдарлануы тиіс, өйткені оның ресурстары сервистер мен инфрақұрылымның орталығына айналады. Әрбір жеке ресурсты жеке пайдалануға болады немесе басқа ресурстармен байланысты. Бұл ресурстардың барлық жиынтығы шолу, әлеуметтік медианы пайдалану, ұсыныстар жасау, пайдалануды есепке алу және т.б. сияқты қосымша қызметтерді дамытуға негіз болатын мазмұн деңгейін құрайды.

Репозиторийді жобалау кезінде онда мынадай технологияларды пайдалану мүмкіндігін көздеу керек:

- ресурстың біріздендірілген сәйкестендіргіштерін қолдану;
- пайдаланылатын лицензия туралы ақпаратты орналастыру;
- іздеу рәсімін жеңілдету үшін навигация;

– пайдаланушылармен өзара әрекеттесу (аннотация, түсініктеме беру, рецензиялау);
– мазмұнды беру;
– ресурсқа қатысудың статистикалық көрсеткіштері;
– пайдаланушыларды сәйкестендіру;
– пайдаланушылардың аутентификациясы;
– ресурсты пайдаланудың стандартталған метрикасы;
– мұрағаттау.

Ғылыми объектілерге қолжетімділік оларды одан әрі қолдануға қандай да бір шектеулерсіз берілуге тиіс. Алайда, іс жүзінде бәрі басқаша және көп жағдайда авторлық құқық иелері өз өнімдерін пайдалану шарттарын реттейді. Мұндай ережелер ғылыми объектінің бөлігі болып табылатын кез-келген веб-ресурс үшін нақты тұжырымдалуы керек және оларды адам да, машина да оңай табуы керек.

Іздеу процедурасын жеңілдету үшін навигация ғылыми ресурсты желіде әрқайсысы өз URL мекен-жайы бар объектілердің бүкіл жиынтығымен ұсынуға болады. Мысалы, PDF немесе HTML форматындағы ғылыми мақала файлы, бір немесе бірнеше деректік блоктары, бір немесе бірнеше форматтағы ресурстың библиографиялық сипаттамасы және т. б. болуы мүмкін, ал қолданушы сол ресурстар арасында оңай ауысып, олардың сол ғылыми ресурсқа жататындығын тани алады.

Интерактивті мүмкіндіктер (аннотация, түсініктеме беру және шолу), егер пайдаланушыларға бұрыннан бар ресурстарға өз ақпаратын қосуға мүмкіндік берілсе, репозиторийдің мәні көбірек болады. Бұл аннотация, түсініктеме беру және сараптамалық шолу сияқты әрекеттер болуы мүмкін. Кез — келген қолданушы үшін әріптестерінің ғылыми жұмыстарына түсініктеме беру және шолу жасау мүмкіндігі болуы керек, ал олардың пікірлері мен шолулары барлығына қол жетімді болуы керек-осылайша жұмыс сапасын объективті бағалау мүмкін болады. Зерттеуші үшін әр түрлі репозиторийлерден ақпаратты біріктіре білу өте маңызды, бұл әр түрлі сандық нысандарды біріктіру негізінде жүргізілген зерттеулердің жалпы көрінісін жасайды.

2.3 Бағдарламалық жасақтама құрылымы

Python әлемдегі ең танымал бағдарламалау тілдерінің бірі болып табылады және 2017 жылы Github жобасында екінші ең танымал тіл болды. Қазіргі уақытта веб-қосымшалардың беделі артып, жаңа қосымшаларды үйрену және олармен жұмыс істеуді кеңінен таралуда. Бұл жұмыста веб технологияны түсіну үшін Django-ны егжей-тегжейлі қарастырылды.

Django - бұл веб-қосымшаларды құру үшін қолданылатын фреймворк. Сенімділік пен қарапайымдылықты қолдвйтын фреймворк веб-әзірлеушілерге таза, тиімді және қуатты код жазуға көмектеседі. Бұл әлемдегі ең танымал веб-

шеңберлердің бірі, сонымен қатар ең көп қолданылатын шеңберлердің бірі. Ол Instagram, YouTube, Google және тіпті NASA-да қолданылады.

Django моделі дерекқорлармен және мәліметтермен жұмыс істеуді жеңілдететін және даму процесін тездететін қуатты басқару қабатын қолданады. Егер нысандарды салыстырса әзірлеуші кестені өзі жасап, сұранысты немесе процедураны анықтауы керек. Бағдарламада Java кодымен бағдарламалардың санын есептейді, сценарийдің негізгі форматын жинақтайды және тексерілмеген орындалатын мәтіндік файлдарды табады. Шын мәнінде, әзірлеушілер күрделі сұрауларды немесе түрлендірулерді білудің қажеті жоқ.

Бұл модельдік көрініс үлгілерін білдіреді. Қатаң түрді, математикалық тұрғыдан дұрыс иерархияны анықтайды. Django сервері сұранысты алған кезде, маршрутизатор сұранысты оның көрінісімен салыстырады. Содан кейін көрініс модельден деректерді шығарады, үлгіні толтырады және оны пайдаланушыға жібереді.

Басқа жақтаулардан айырмашылығы, барлық модельдер әдетте бір файлда болады models.py. Бірақ бұл ірі жобалар үшін ақпараттың толып кетуіне әкелуі мүмкін.

Django көптеген дерекқор жүйелерін қолдайды. Бұл қатаң түрді, математикалық иерархияны анықтайды. SQLite тестілеу және дамыту үшін өте жақсы, өйткені оны қосымша бағдарламалық жасақтаманы орнатпай-ақ қораптан тікелей пайдалануға болады. Өндіріс үшін MYSQL немесе PostgreSQL-ді қолдана аласыз, егер NoSQL дерекқорын керек болған жағдайда, Django-мен MongoDB-ны қолдануға болады.

Іріктеме деңгейі іс жүзінде деректерді пайдаланушының оларды қалай ұсынатынынан және қарауынан бөлу үшін қолданылады.

Үлгі деңгейі MVC ұсыну деңгейіне ұқсас. Егер қолданушы басқа тілдердегі шаблонмен таныс болса, онда Django-мен танысу қиындыққа соқпайды.

Django шаблоны - дизайнның негізгі қағидаларының бірі, бұл "қайталанбау"дегенді білдіретін дизайн үлгісі. Бұдан шығатыны, көп жағдайда бұл кодты көшірудің қажеті жоқ дегенді білдіреді. Оның орнына шаблонды шарлау тақтасы, әдепкі шарлау тақтасы, беттің жоғарғы және төменгі колонтитулдары және т.б. сияқты қайта пайдаланылатын компоненттерге бөлу керек, бұл қайталануды азайтады және тиімді және таза кодты жазуға мүмкіндік береді.

Django-да Python кодын шаблондардан бастау және айнымалыларды тағайындау мүмкін емес. Қарапайым тілмен айтқанда, Django кодты шаблон деңгейінде орындауға тыйым салады және көптеген веб-осалдықтардың қарапайым, бірақ жоғары тиімді шешімі болып табылатын тек дисплей логикасына қол жеткізуге мүмкіндік береді.

Django көрінісі бизнес-логика деңгейінде. Ол пайдаланушылардың сұрауларын өңдеуге және нақты жауаптарды қайтаруға жауап береді. Модельден деректерді ала аласыз, әр шаблон белгілі бір деректерге қол жеткізе алады немесе кейбір деректерді алдын-ала өңдей алады. Ағымдағы Django көрінісі

сұраныстарды өңдейтін және жауаптарды қайтаратын функция болуы мүмкін немесеа Laravel және rail контроллерлеріне ұқсас класстар көп болуы мүмкін.

Djangoның ерекшеліктерінің бірі-оның қауіпсіздікке қаншалықты мән беруі. Бұл шаблонды жазуға шынымен әсер етеді.

Қолдану саласына байланысты қолдануға болатын немесе болмайтын көптеген дайын материалдармен бірге келеді. Өзіңіздің жеке кодыңызды (power) жазудың орнына сіз пайдаланғыңыз келетін пакетті импорттауыңыз керек.

Бұл Django құрамына кіретін конфигурация парадигмасындағы ережелердің бөлігі және сіз әлемдік деңгейдегі сарапшылар қабылдаған шешімдерді қолдана аласыз. Django батареясы көптеген тақырыптарды қамтиды:

- Түпнұсқалықты тексеру бумасын пайдаланып тексеру;
- Әкімшілердің пакеттерінің өзара әрекеттесуі;
- Сеанстар бумасын пайдаланып сеанстарды басқару;
- Хабарлама бумасын пайдаланып уақытша хабарларды немесе сеанс негізіндегі хабарламаларды басқару;
- Sitemaps бумасын пайдаланып Google sitemap XML файлын жасау;
- Postgres пакетімен арнайы Postgres мүмкіндіктері.

Django Python-да қолданатындықтан, ол Python-ның күшінің бір бөлігін өз пайдасына пайдаланады. Python жаңадан бастаушылар үшін қарапайым бағдарламалау тілдерінің бірі бола алады және бұл оңай емес, сонымен қатар бүкіл әлем бойынша кіріспе информатика курстарында өте танымал. 2017 жылғы әзірлеушілердің сауалнамасы Python қазіргі уақытта PHP-ге қарағанда кең таралғандығын және Python мүмкіндіктері C# және C++қарағанда кең тарап жатқанын көрсетті.

Django қауымдастығы - бұл ең жақсы мүмкіндіктердің бірі, ол жаңа мүмкіндіктерді қосу арқылы жақтауды ыңғайлы ету және жақтауды тұрақтандыру үшін пайдалы және белсенді жұмыс істейді. Django құжаттамасы өте дәл және жеке нұсқаулық ретінде пайдалы және оны негізгі ақпарат көзі ретінде пайдалану үшін әртүрлі мүмкіндіктерді түсінуге көмектеседі.

Көптеген әзірлеушілер жақтауды таңдау туралы ойланғанда, өз таңдауында болашақты жоспарлайды. Сондықтан масштабталатын жақтауды таңдау көптеген адамдар үшін өте маңызды, ал Django дәл солай. Бұл масштабталуға қатысты көптеген түрлі әрекеттерді орындауға мүмкіндік береді, мысалы, мәліметтер базасы, медиа және қосымшаның өзі үшін жеке серверлерді іске қосу немесе тіпті бірнеше серверлерге тарату үшін кластерлеуді немесе жүктемені теңдестіруді пайдалану.

Дипломдық жұмыста деректер қорын құруға SQLite дерекқоры пайдаланылды.

SQLite дерекқорлары әртүрлі қолданбалы бағдарламалардың деректерін сақтауға арналған заманауи электрондық құрылғыларда кеңінен қолданылады. Сондықтан оларды талдау электронды құрылғылар мен цифрлық ақпарат тасымалдаушыларды криминалистикалық зерттеу барысында маңызды болып табылады [13].

SQLite дерекқорлары бір файл ретінде сақталады. Транзакциялар орындалған кезде кері қайтару журналы немесе жазбалар туралы ақпаратты сақтайтын журнал файлы бар екінші файл жасалады. Бұл файл бүлінген жағдайда (мысалы, құрылғы бұзылған кезде, транзакцияларды орындау кезінде) мәліметтер базасын қалпына келтіру үшін қолданылады.

Мәліметтер қорының негізгі файлы бір немесе бірнеше беттен тұрады. Бір дерекқордағы барлық беттердің өлшемі бірдей, ол 512-ден 65536 байтқа дейін болуы мүмкін. Дерекқор файлының бет өлшемі дерекқор файлының басынан 16 байт ығысуы бар 2 байт бүтін санмен анықталады.

Бір уақытта бірнеше процестер немесе ағындар мәліметтер қорынан деректерді еш қиындықсыз көре алады. Мәліметтер қорына жазу қазіргі уақытта өзге сұраныстарға қызмет көрсетілмеген жағдайда ғана жүзеге асырылады. Оқиғаларды дамытудың тағы бір түрі - белгіленген уақыт үшін сұраныстардың өздігінен қайталануы.

SQLite-тің ескі нұсқалары ешқандай шектеусіз жасалған, жалғыз шарт - бұл барлық есептеулер 32 биттік бүтін сандармен жүргізілген жадқа сәйкес келуі. Жоғарғы шектер анықталмағандықтан және сәйкесінше дұрыс тексерілгендіктен, SQLite-ті өте төтенше жағдайларда пайдалану кезінде қателер жиі пайда болды. Сонымен, SQLite-тің жаңа нұсқаларында шектеулер енгізілді, олар қазір жалпы тест жиынтығымен бірге тексеріледі [13].

3 Кітапхана репозиторийін әзірлеу

3.1 UML тілін пайдаланып жобалау

Дипломдық жобаны модельдеу барысында Unified Modeling Language (UML) тілі қолданылды. Unified (әмбебап, бірыңғай) — жобаланған бағдарламалық жүйелердің кең класына, қосымшалардың әртүрлі салаларына, ұйымдардың түрлеріне, құзыреттілік деңгейлеріне, жобалардың көлеміне сәйкестігін білдіреді [14].

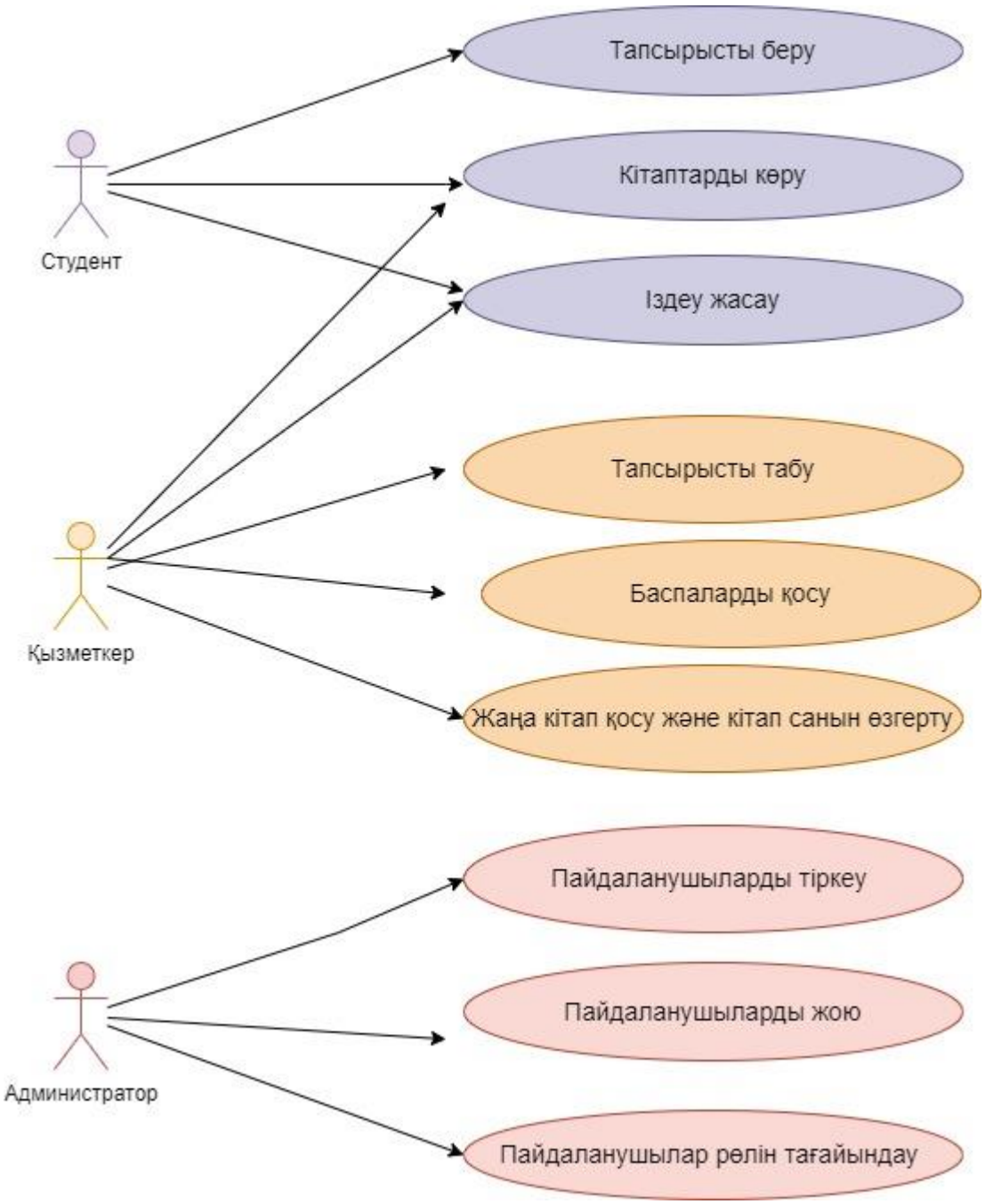
UML диаграммалары үлкен жобалардың архитектурасын модельдеуге көмектеседі, онда үлкен де, кішігірім детальдарды жинап, қолданбаның диаграммасын салуға болады.

Кез келген басқа тіл сияқты, UML-де де модельді жобалаудың өзіндік ережелері мен синтаксисі бар. UML графикалық белгісін қолдана отырып, жүйені визуализациялауға, барлық компоненттерді бір құрылымға біріктіруге, процесте модельді нақтылауға және жақсартуға болады. Жалпы деңгейде UML графикалық белгісі элементтердің 4 негізгі түрін қамтиды:

1. сандар;
2. сызықтар;
3. белгішелер;
4. жазулар.

UML белгісі бағдарламалық жасақтаманы әзірлеуде, ақпараттық жүйелер инфрақұрылымында және бизнес жүйелерінде іс жүзінде салалық стандарт болып табылады [14].

Жүйедегі негізгі 3 актер бар:студент, қызметкер және админ. Актерлардың іс-әрекет мүмкіндіктері use-case диаграммасында көрсетілген(3.1-сурет).



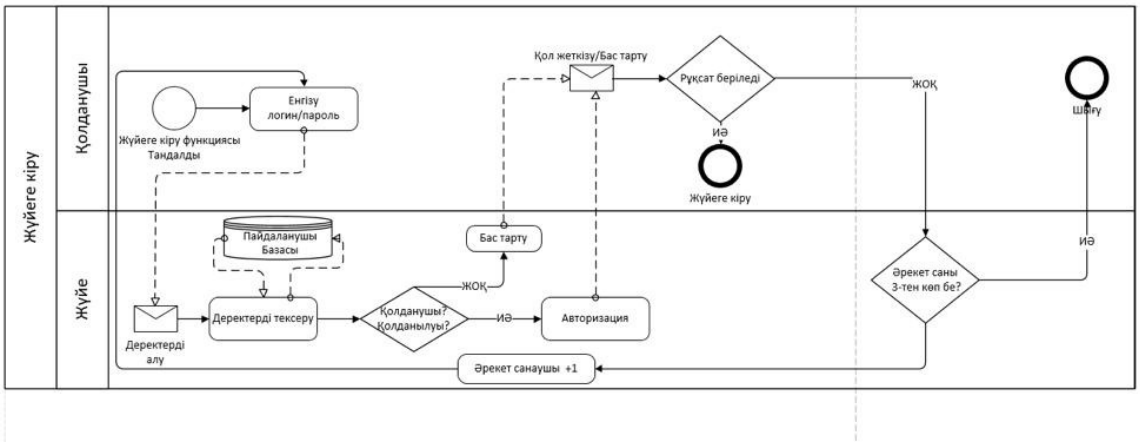
3.1-сурет – Use case диаграммасы

Жоғарғы деңгейдегі BPMN диаграммалары іске асырылатын қадамдардың қол жетімді суретін алу үшін пайдалана алатын бизнес-процесс қатысушылары мен басқа мүдделі тұлғаларға арналған. Неғұрлым егжей-тегжейлі деңгей диаграммалары процесті жүзеге асыруға тікелей қатысатын және тапсырманы орындау үшін жеткілікті ақпаратты қамтитын адамдарға арналған. BPMN диаграммалары бизнес-процестерді арнайы білім дәрежесіне қарамастан, барлық мамандарға, яғни жоба аналитиктеріне, жұмысты орындаушыларға, басқару тобына, дайындау тобына, сондай-ақ басқа жұмысшылар мен менеджерлерге баяндауды бірыңғай түрде стандартталған тілде сипаттайды. BPMN диаграммаларs бизнес-процестерді жобалаудан іске асыруға дейінгі ыңғайлы

көпірді қамтамасыз ету үшін жеткілікті егжей-тегжейлі және анық әрекеттер тізбегін көрсетуі керек.

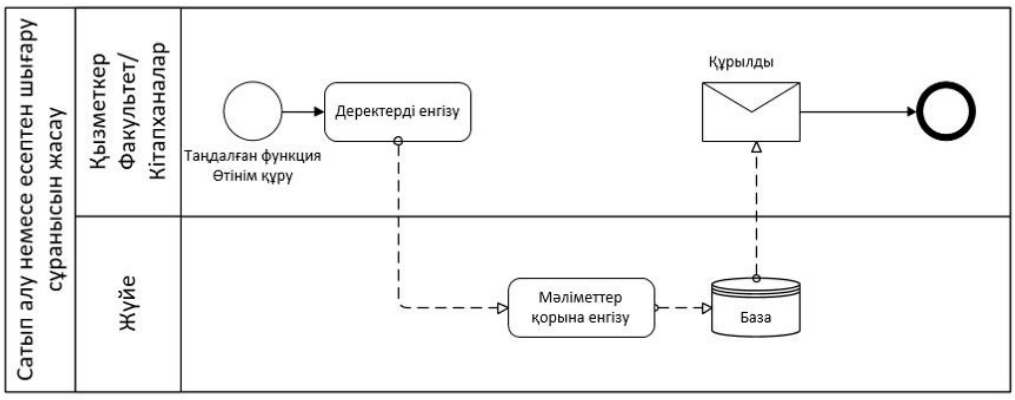
Диаграмма түрінде берілген ақпаратты мәтін түріндегі сипаттаудан гөрі түсіну оңайырақ. Схематизация ақпарат алмасуды да, сапалы нәтижеге қол жеткізу үшін тиімді процесті құру үшін бірлесіп жұмыс істеуді де жеңілдетеді. Схемалар әртүрлі процестерді орындау үшін қажетті XML құжаттарын құрастыру кезінде де талқылауды жеңілдетеді (XML - Extensible Markup Language - "extensible markup language").

Төменде жүйеге кіру BPMN диаграммасы суреттелген(3.2-сурет). Қолданушы логинді және құпиясөзді енгізеді, деректерді деректер қорынан іздейді, таба алмаса тіркелуді ұсынады, пайдаланушының аккаунты болып, деректері қате болса жүйеге кіру санын санап, мәжеден асса бұғаттайды.



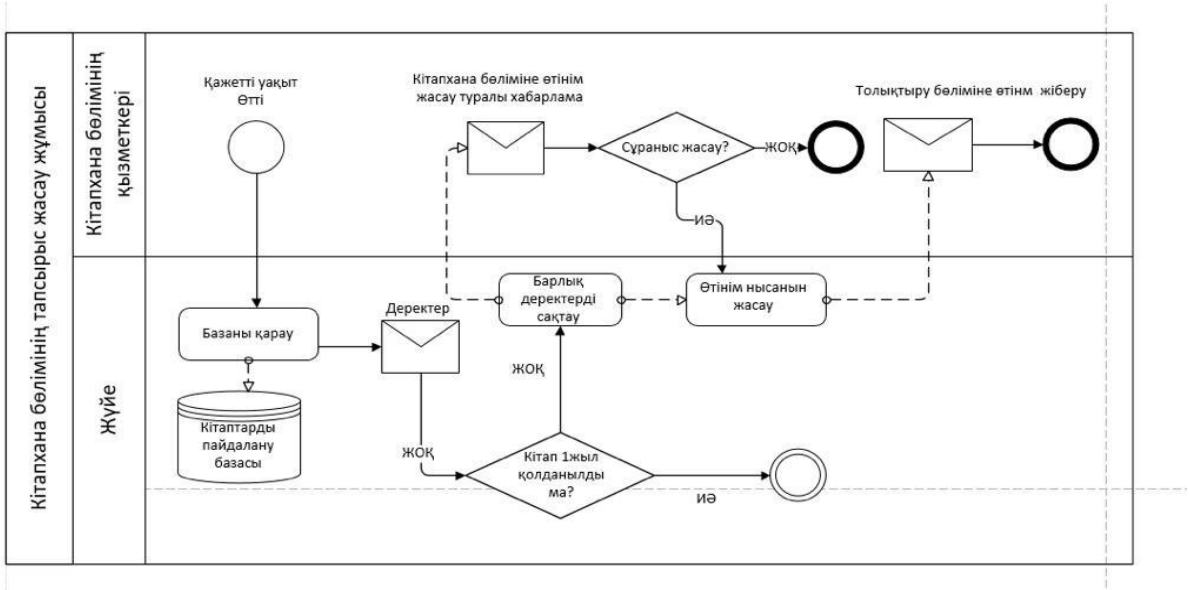
3.2-сурет – Жүйеге кіру (авторизация) бойынша BPMN диаграммасы

Сатып алу немесе есептен шығару сұранысы. Бірден деректер енгізіледі және тек хабарлама қайтарады(3.3-сурет).



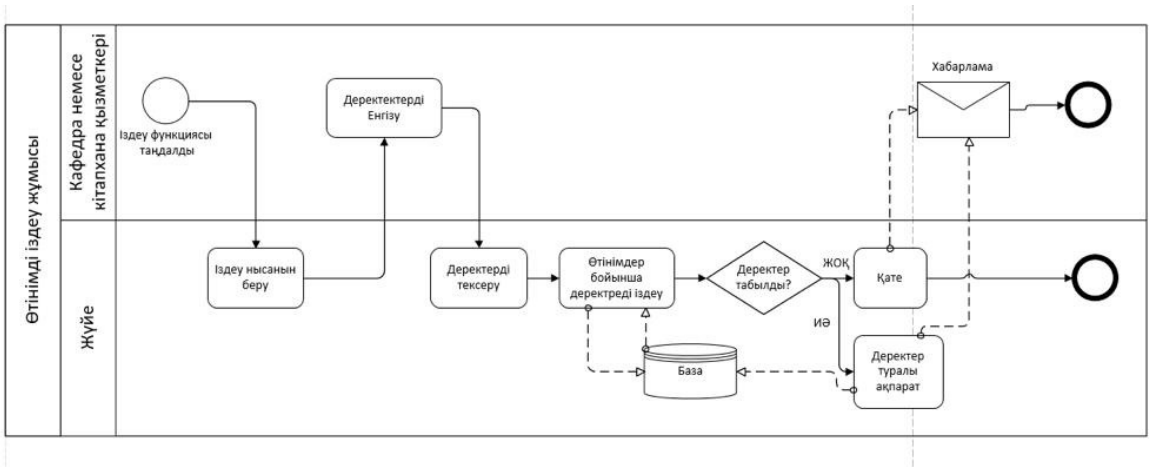
3.3-сурет– Оқулықтарды есепке алу бойынша BPMN диаграммасы

Кітапхана бөлімінің тапсырыс жасау жұмысы. Белгілі материалды тауып аламыз, егер 1 жылдан артық қолданылмаса өтінім жіберіліп, алдырылады және деректер қорына қосамыз(3.4-сурет).



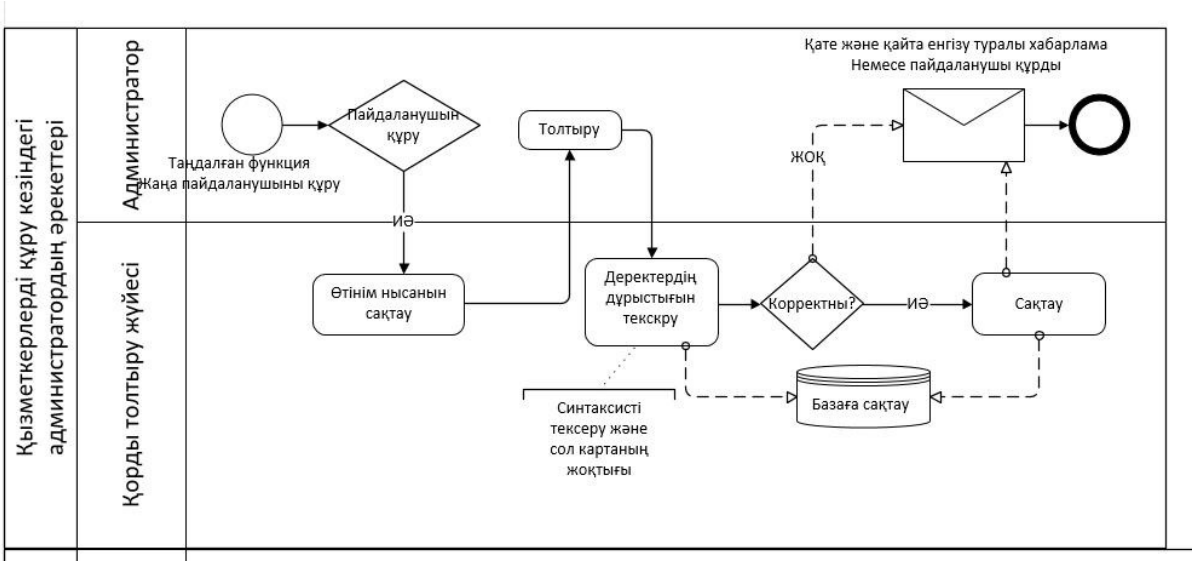
3.4-сурет – Тапсырыс жасау бойынша BPMN диаграммасы

Өтінімді іздеу. Ізделетін қажетті ақпараттарды беріледі. Өтінім бойынша деректер ізделеді және сәтті, сәтсіздігі жайлы хабарлама келеді(3.5-сурет).



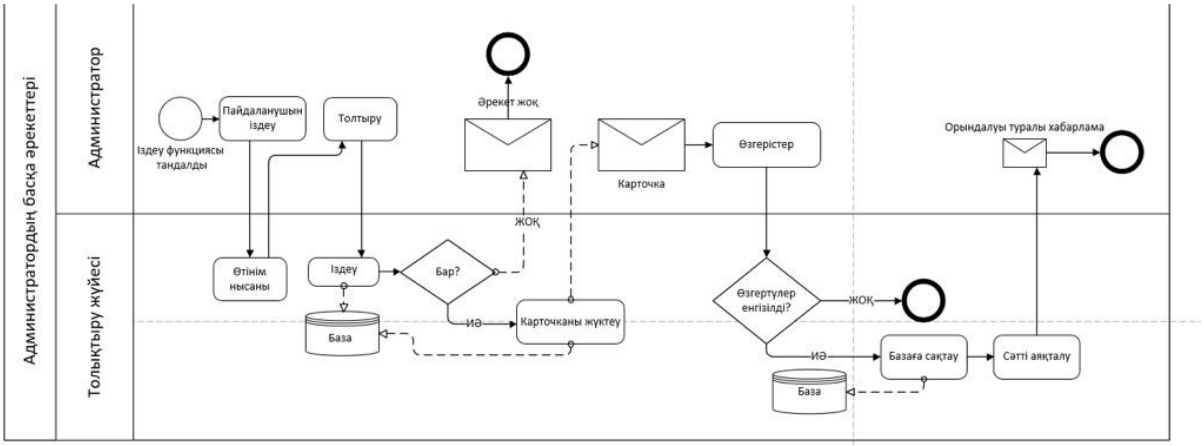
3.5-сурет – Іздеу бойынша BPMN диаграммасы

Жаңа қызыметкерлерді қосу. Деректер енгізіледі, дұрыстығы тексерілген соң сәтті болса жаңа қызыметкер косылады, қате болған жағдайда хабарлама қайтарылады(3.6-сурет).



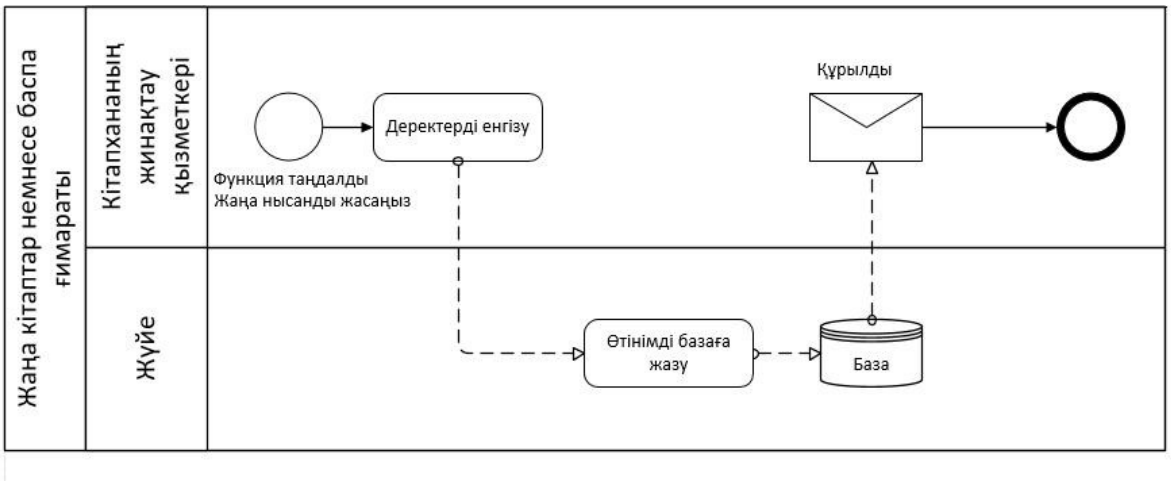
3.6-сурет – Жүйеге жаңа қолданушы тіркеу бойынша BPMN диаграммасы

Администратордың басқа әрекеттері. Пайдаланушыны іздеу, толтыру, деректерді өзгерту, жүйе өзгерістерін қабылдау(3.7-сурет).



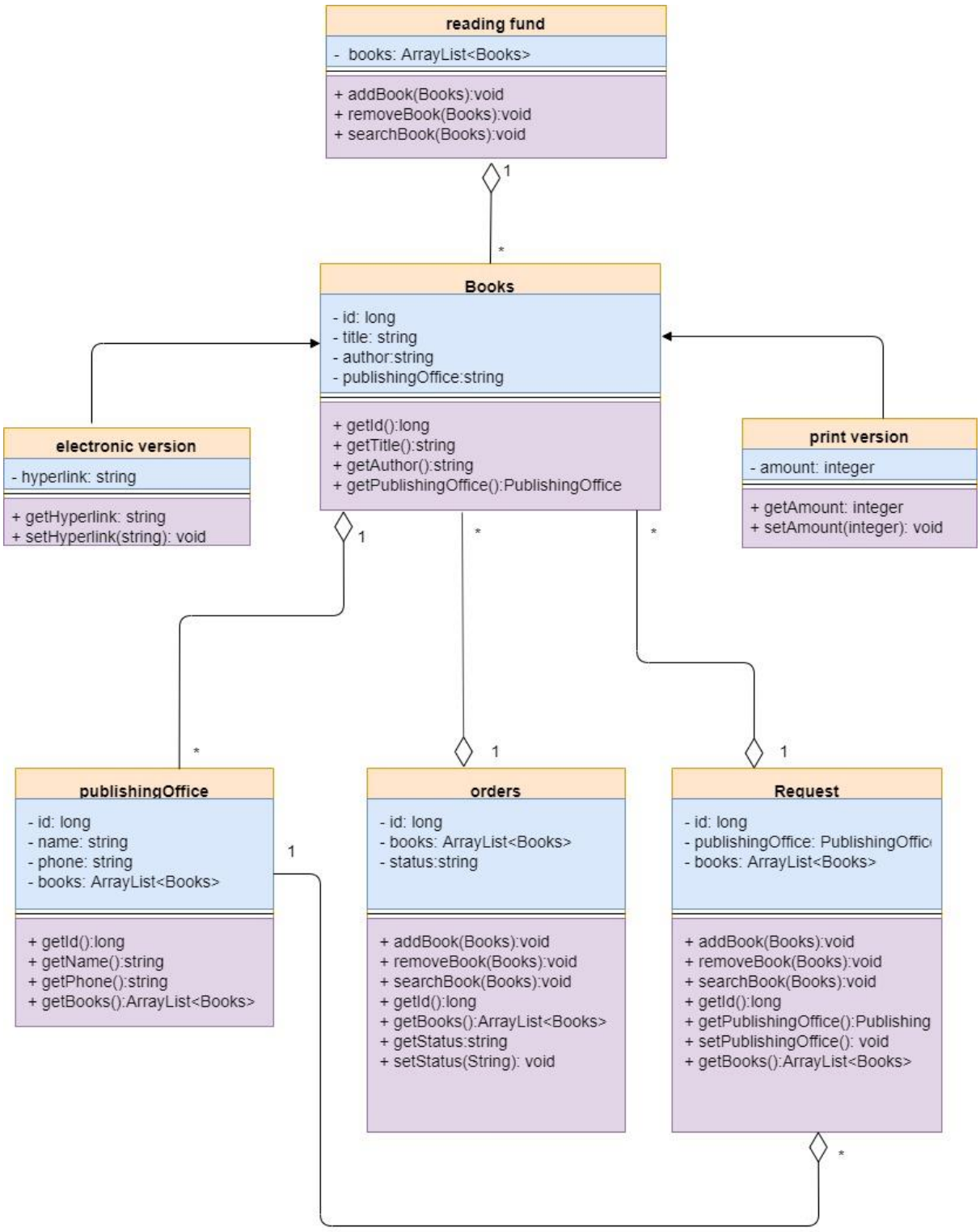
3.7-сурет – Қолданушы іздеу бойынша BPMN диаграммасы

Жаңа кітаптар немесе баспа ғимараты. Тек өтінімді құрады және қабылдайды(3.8-сурет).



3.8-сурет – Жаңа кітаптарды енгізу бойынша BPMN диаграммасы

ER-Диаграммасы тікелей класстар диаграммасындағы байланыстарға қатысты, себебі models.py-да сипатталатын класстардың байланысын деректер қорындағы кестелер байланысына келтіруге болады. Диаграммада класстар, олардың атрибуттары және функционалдарының байланысы көрсетілген (3.9-сурет).



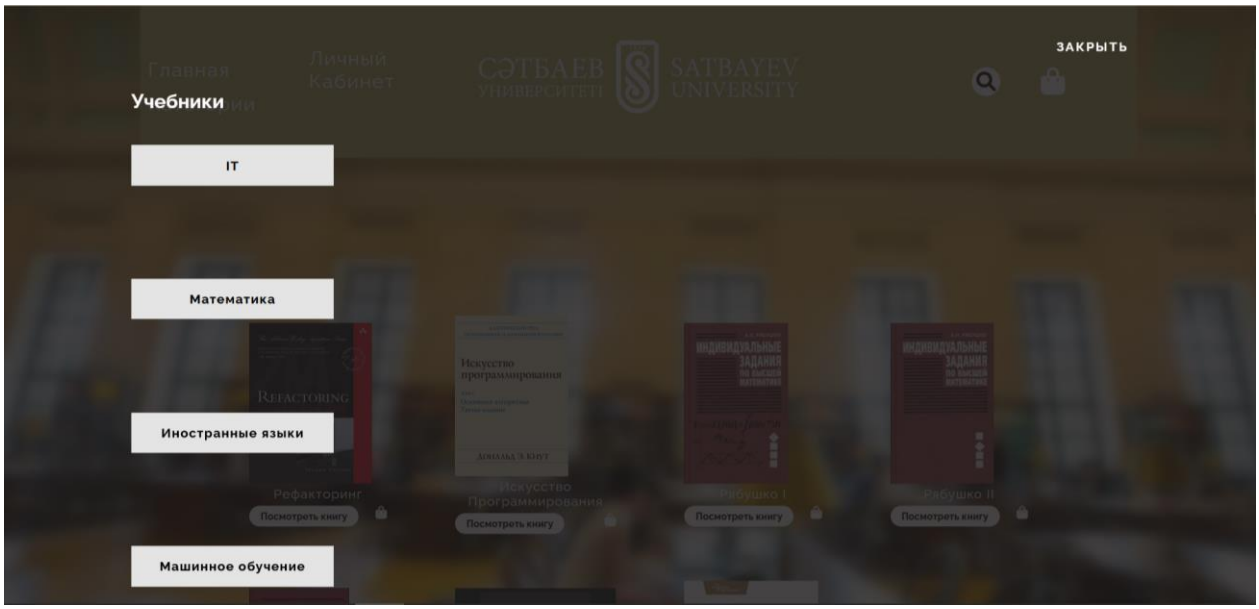
3.9-сурет – ER диаграммасы

3.2 Кітапхана репозиторийінің интерфейсімен танысу

Веб сайт интерфейсі қолданушыларға ыңғайлы, түсінікті етіліп жасалды (3.10-сурет).

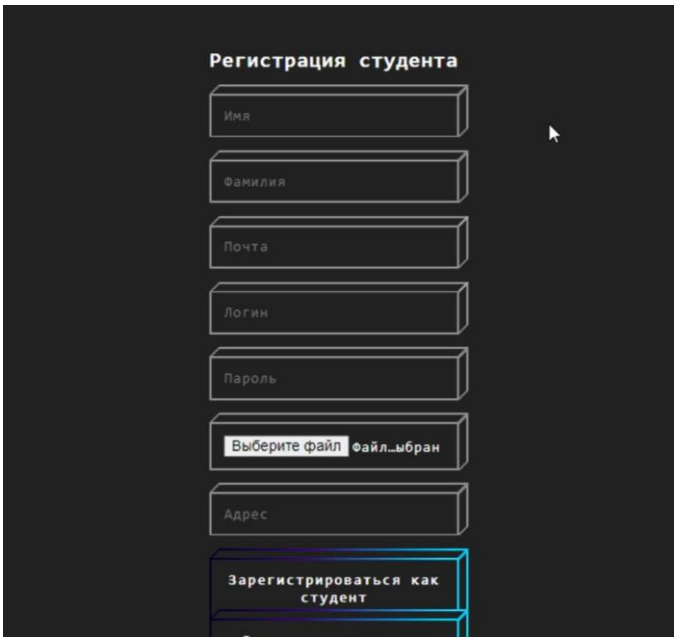
3.10-сурет – ЖОО-ға арналған кітапхана репозиторийінің интерфейсі

Оқулықты іздеуге, сұрыптауға қолайлы болу үшін «Категории» батырмасы орналастырылған. Батырманы басқан кезде төменде көрсетілгенде бірнеше категорияларға бөлінеді: оқулықтар, кітаптар, дипломдық жұмыстар мен статьялар (3.11-сурет).

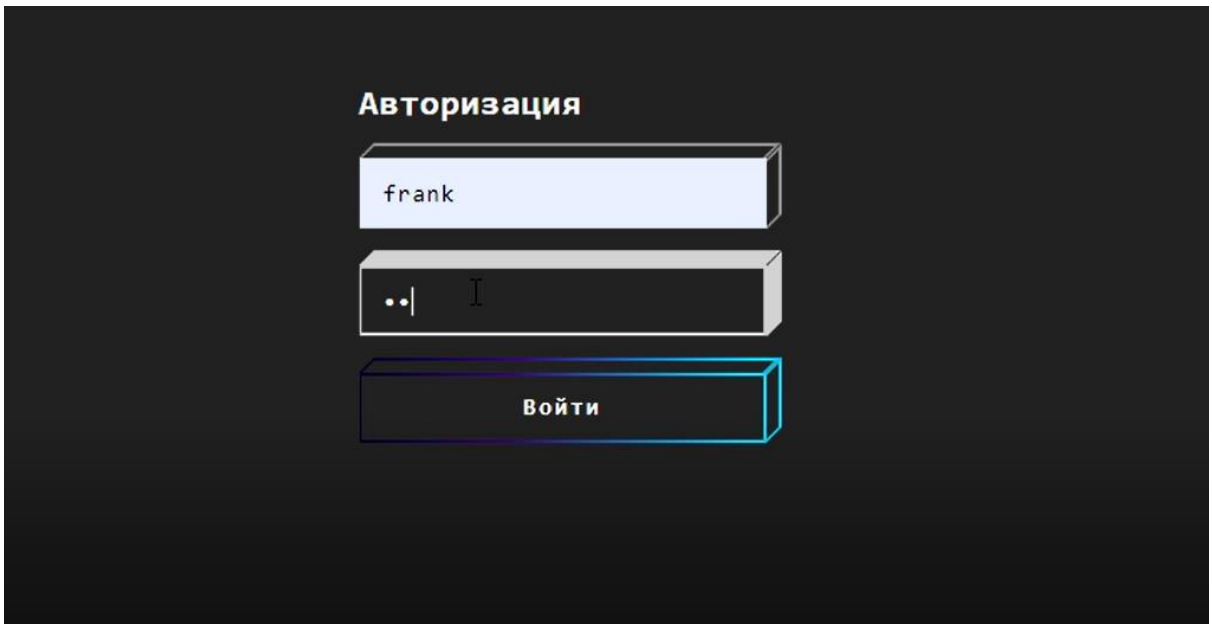


3.11-сурет – Категорияға байланысты бөлінуі

Кез келген жүйе қолданушысы сайтқа регистрация (3.12-сурет) және авторизация жасай алады (3.13-сурет). Тіркелу үшін аты-жөнін, почта, логин, пароль,адрес сияқты жеке мәліметтерін енгізу қажет.



3.12-сурет – Регистрация парағы

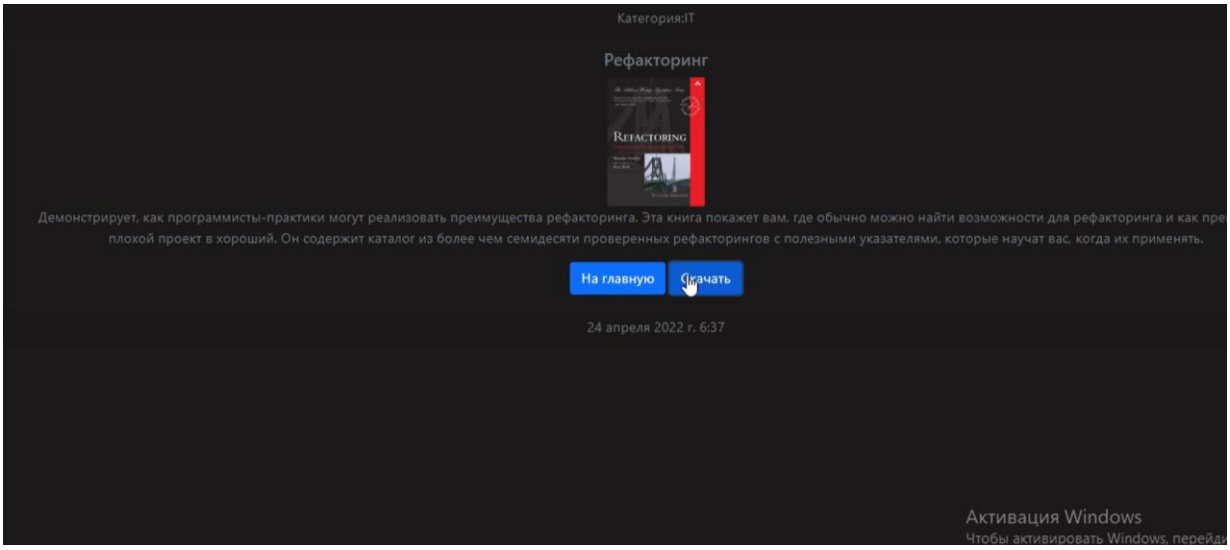


3.13-сурет – Авторизация парағы

Студент жүйеге тіркеліп, кіргеннен кейін басты бет интерфейсі өзгереді, яғни, студенттің жеке кабинеті мен іздеу, тапсырыс беру батырмалары пайда болады (3.14-сурет).

3.14-сурет – Жоғарғы оқу орнына арналған кітапхана репозиторийінің қолданушы интерфейсі

Студент өзіне қажетті оқулықтарды көшіріп ала алады (3.15-сурет).

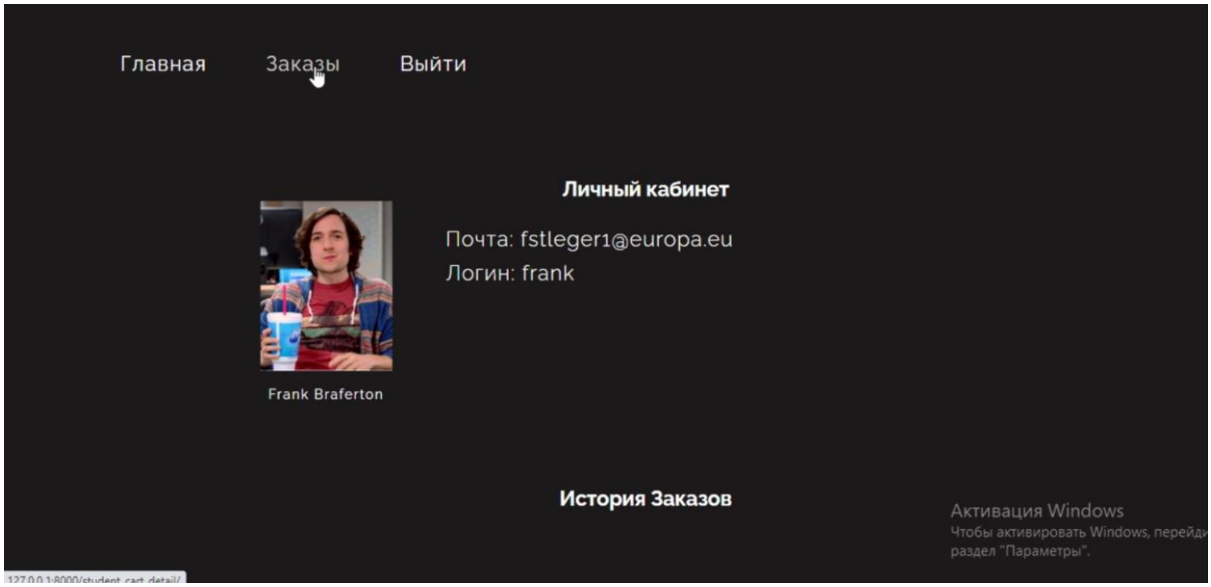


3.15-сурет – Жоғарғы оқу орнына арналған кітапхана репозиторийінің қажетті кітаптарды көшіріп алу батырмасы

Студент іздеу батырмасы арқылы қажетті оқулықты жылдам таба алады (3.16-сурет).

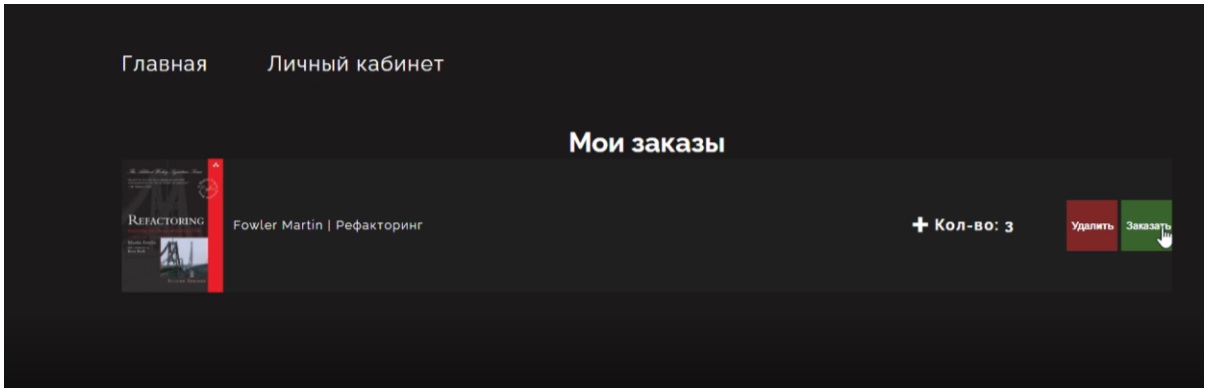
3.16-сурет – Жоғарғы оқу орнына арналған кітапхана репозиторийінің қажетті кітаптарды іздеу батырмасы

Студенттің жеке кабинетінде берген тапсырыстары мен жеке мәліметтері көрсетілген(3.17-сурет).



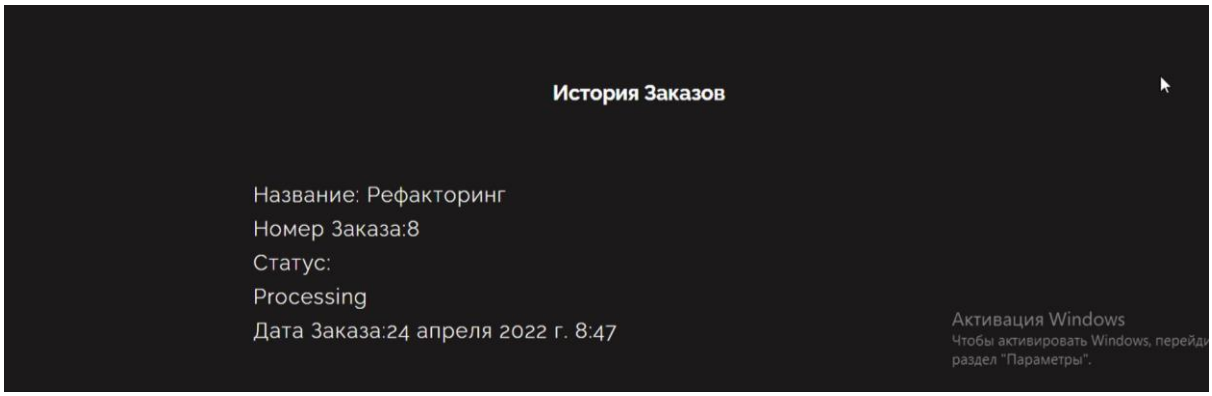
3.17-сурет – Студенттің жеке кабинеті

Студент жеке кабинетінен берген тапсырыстары туралы ақпаратты қарай алады(3.18-сурет).



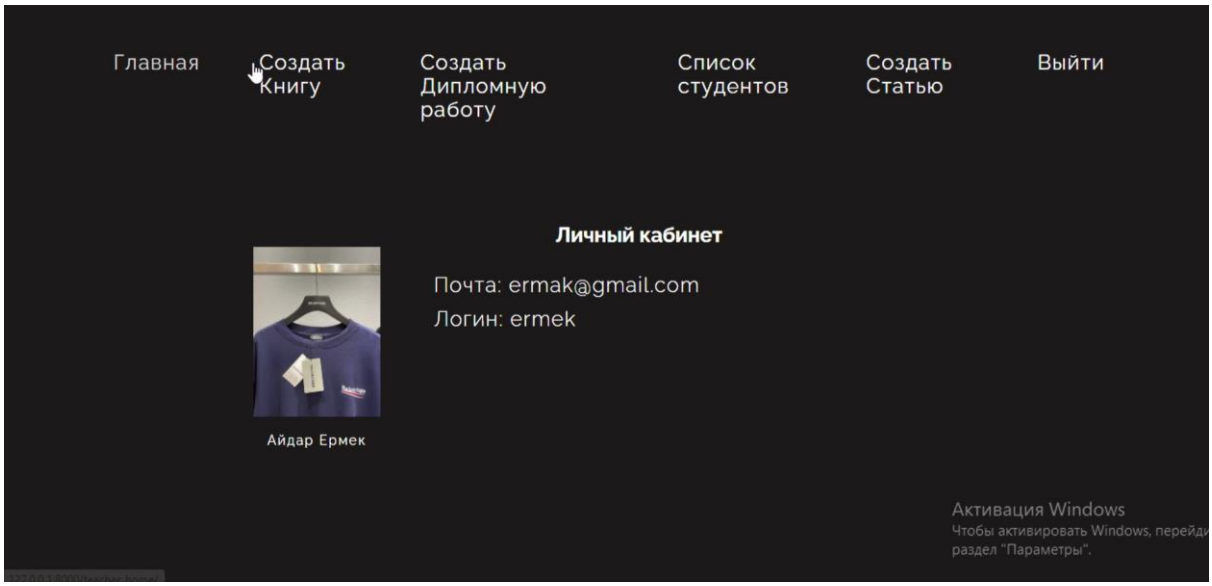
3.18-сурет – Студенттің тапсырыстары

Студент тапсырыстарының статусын көре алады(3.19-сурет)



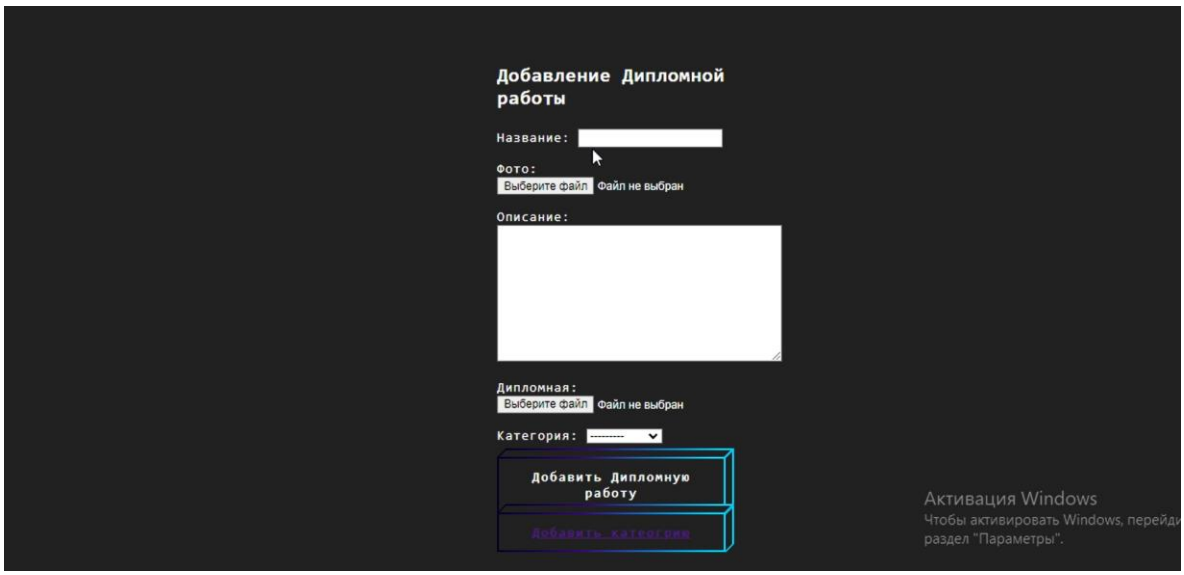
3.19-сурет – Студенттің тапсырыстарының статусын көру

Жоғарғы оқу орнына арналған кітапхана репозиторийінің қызметкері жаңадан оқулықтар, статьялар мен дипломдық жобалар қоса алады, студенттер тізімі мен тапсырыстарды көре алады (3.20-сурет).



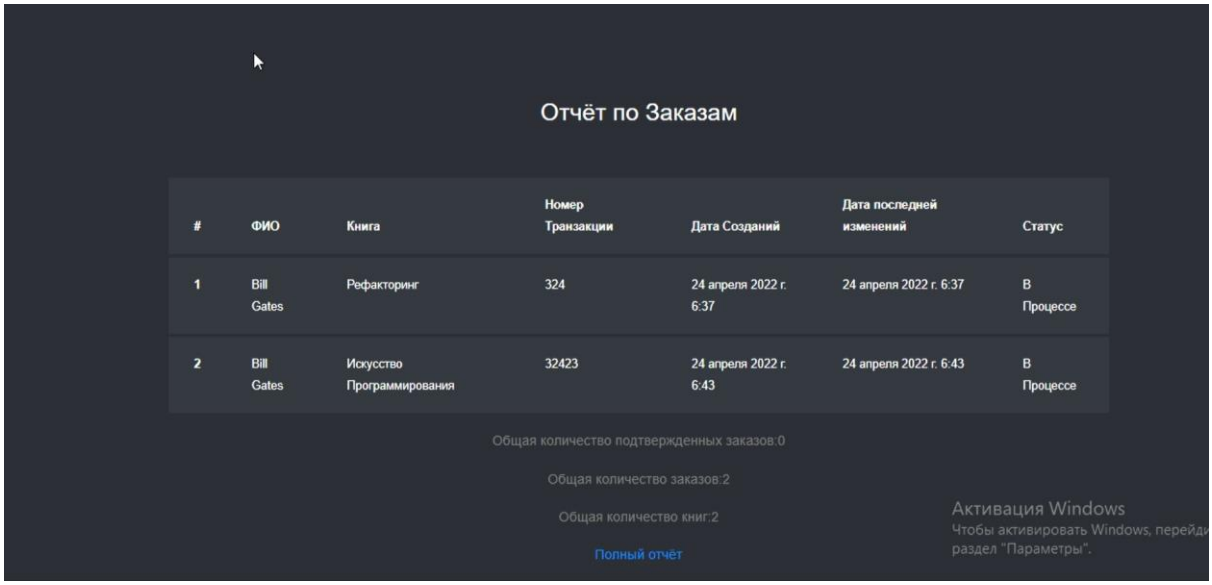
3.20-сурет – ЖОО-ға арналған кітапхана репозиторийінің қызметкер интерфейсі

Жаңа дипломдық жұмысты қосу батырмасы(3.21-сурет). Жаңа дипломдық жұмысты қосу үшін атауын,қысқаша анықтамасын жазу және категориясын белгілеу керек.



3.21-сурет – Жаңа дипломдық жұмыс қосу батырмасы

Админ тапсырыстар бойынша есепті қарай алады(3.22-сурет).



3.22-сурет – Тапсырыс бойынша есеп

ҚОРЫТЫНДЫ

Дипломдық жұмысты жобалау процесінде пәндік аймақ талданды. ЖОО-ға арналған кітапхана репозиторийін әзірлеуге арналған үшін мақсаттар мен міндеттер анықталды. Пәндік саланы талдау нәтижелері бойынша жүйенің концептуалды жобасы құрылды: пәндік саланың моделі әзірленді, концептуалды деректер моделі жасалынды.

Бұл дипломдық жобада ЖОО-ға арналған кітапхана репозиторийін әзірлеуге арналған бизнес-процестерді автоматтандыру мәселелері қаралды. Пайдаланушылардың рөлі мен мүмкіндіктерін визуалды түрде көрсетуге Use Case, BPMN диаграммалары пайдаланылды. Жүйе интерфейсін жасауға Figma құралдары пайдаланылды. Жүйесін дамыту үшін реляциялық SQLite деректер базасын жобалау жүзеге асырылды, PyCharm платформасы Django фреймворкі негізінде веб-қосымша іске асырылды.

Әзірленген бағдарламалық қамтамасыз ету келесідей функцияларды орындауға мүмкіндік береді:

- қолдаушыларды тіркеу;
- жаңа оқулықтарды, дипломдық жұмыстарды, статьяларды енгізу және сақтау;
- оқулықтарды, дипломдық жұмыстарды, статьяларды іздеу;
- студенттердің оқулықтарға, дипломдық жұмыстарға, статьяларға тапсырыс бере алуы және т.б.

Жасалған тестілеу веб сайттың техникалық тапсырма талаптарына сәйкес жұмыс істейтінін көрсетті. Бағдарламалық құралды пайдалану кітапхана жұмысын оңтайландырады және сәйкесінше оқырмандарға қызмет көрсету сапасын арттырады.

Пайдаланылған әдебиеттер тізімі

1 Аверина А.Е. Ақпараттық жүйелерді жобалау // Қазіргі ғылым мен білім беру мәселелері. – 2015. – жоқ. 12. - 83 б.

2 Коваленко, В.В. Ақпараттық жүйелерді жобалау: оқулық. ЖОО үшін оқулық / В.В. Коваленко. – М.: Форум, 2012 ж.

3 Кобайло А.С. Ақпараттық жүйелерді жобалау. – 2014 ж.

4 Kim S. N., Chan K. H. кәсіпорынды модельдеу үшін өнімділікті талдау және жобалау // Индустриалды экономиканың халықаралық журналы. - 2002. - Т. 76. - 2. - С. 121-133.

5 Репин В.В., Элиферов В.Г. Басқарудағы процестік көзқарас // Бизнес-процестерді модельдеу. - 2004. - Т. 20.

6 «Автоматтандырылған жүйені құрудың техникалық тапсырмалары» / - М .: ГОСТ 34.602 - 89, 1990 ж.

7 Butch G., Jacobson I., Rambo D. Language UML. Пайдаланушы нұсқаулығы. – Литер, 2017 ж.

8 Губина Е.А., Ирзаев Г.Х., Адеева М.Г. CASE құралдары мен реляциялық мәліметтер базасын байланыстыруға негізделген ақпараттық жүйені жобалау. – 2014. – жоқ. 4. - С. 75-79.

9 Елиферов В. Г., Репин в. В. Бизнес-процестер. Регламенттеу және басқару. -М.: Инфра-М, 2009. – 320 б.

10 Козлов А.С. бизнес-процестерді жобалау және зерттеу. – М.: Флинт, 2006. – 272 б.

11 Марк Д. А., МакГоуэн к. құрылымдық талдау және дизайн әдістемесі. – М.: МетаТехнология, 2003. – 274 б.

12 Django фреймворкі туралы [Электрондық ресурс]. – Қол жеткізу режимі: Writing your first Django app, part 1 | Django documentation | Django (djangoproject.com)

13 SQLlite туралы [Электрондық ресурс]. – Қол жеткізу режимі: SQLite Tutorial - Learn SQLite basic to advanced concepts (sqlitetutorials.com)

14 Use Case туралы [Электрондық ресурс]. – Қол жеткізу режимі: Use Case Diagram Tutorial (Guide with Examples) - Creately Blog

15 Абагрин, А.аңдау жаһандық желіге негізделген// Библиотека.- 2000.- № 12.- С. 26-28.Таңдау жаһандық желіге негізделген

16 Автоматтандырылған ақпараттық кітапхана жүйесі. Түйінді сөзді тағайындау және өңдеу, Flashback енгізу: пайдаланушы нұсқаулығы / ИНЕАК. – Мн., 2000.

17 Автоматтандырылған ақпараттық кітапхана жүйесі. Жаңа сатып алуларды тіркеу, түгендеу кітаптарын жүргізу: пайдаланушы нұсқаулығы / ИНЕАК. – Мн., 2000

18 Алёшин, Л.И.Кітапханадағы автоматтандыру:оқу құралы. Ч.1./ МГУК; Л.И. Алёшин. – М.: Профиздат, 2001. – 176 с.

19 Дворкина, М.Я.Кітапхана қызметтері: жаңа шындық / М.Я. Дворкина. – М.: МГУКИ, 2001. – 48 с.

20 Дворкина, М.Я. Кітапхана қызметі: теориялық аспект/ М.Я. Дворкина. – М.: Либерея, 1993. – 248 с.

21 Дзюба, Н. И. Балалар кітапханасы және соңғы ақпараттық технологиялар / мәселеге жауапты.С.Кобзаренко; ред. Л.І.Стахурська. — К.: ДБУ для дітей, 2001. — 40 с.

22 Дзямешка, Л.А. Кітапхана пайдаланушыларына қызмет көрсету нысандарын дамыту [Тэкст] / Л.А. Дзямешка //Беларусь мемлекеттік мәдениет университетінің хабаршысы. – 2002. - № 1. – С. 80 – 84.

23 Екимова, Н.Г. Электрондық тапсырма бөлімі/ Наталья Георгиевна Екимова // Библиотека. – 2003. - №2. - С. 42-44

24 Левин, Г.Л. Библиографиялық ізденіс: теориялық аспект// Библиотековедение. – 2000. - №4. – С. 47-50.\

25 Леонтьев, В.Дербес компьютер: әмбебап пайдаланушы нұсқаулығы 2000 / Виталий Леонтьев. – М: ОЛМА-ПРЕСС, 2000. – 590 с.

А Қосымшасы
(міндетті)

Техникалық тапсырма

А.1 Жалпы сипаттама

Жоба атауы – “Жоғарғы оқу орнына арналған кітапхана репозиторийін әзірлеу ”. Мақсаты ЖОО-дағы кітапхана қызметін автоматтандыру, процесстерді жеңілдету, қызмет сапасын арттыру, ақпарат көздерін цифрландыру болып табылады.

Жұмыс кітапхана процессінің бір бөлігін автоматтандырғандықтан кітапханадан бөлек циклда өмір сүре алмайды, демек, тек толықтырады. Оның мықты жақтары:

- Кітапхана жұмысын жетілдіреді;
- Қол жетімділікті арттырады;
- Қолданушыларды орталықтандырады;
- Жүйе тек белгілі қолдаушыларға қол жетімді бола алады;
- Қолданушылардың деректерін жинау арқылы болашақта үлкен деректерді енгізуге болады.

Әлсіз жақтары:

- Кітапхана жұмысын автоматтандырғанымен қолданушы мен кітапханашы арасында күрделі байланыс пайды болады;
- Адами факторлар кесірінен жүйенің келіспеуі;

А.1.2 Мақсаты

ЖОО-ның ақпарат көздерін(оқулық, дипломдық жұмыс, мақалаларды) орталықтандырып, сақтау және тек арнаулы қолданушыларға қол жетімді қылу. Бұл амалдар жұмысты жеңілдетуге және уақытты үнемдеуде өзінің оң нәтижесін береді.

А.1.3 Бағдарлама талаптары

А.1.3.1 Бағдарлама коды

Бағдарлама коды интуитивті түсінікті, оқуға ыңғайлы, программалау тілінің стандарттарына сәйкес болып жазылуы тиіс.

А.1.3.2 Функционалдық сипаттама

Қолданушыларға:
- Тек тіркелген қолданушыға қол жетімділік;
- Қолданушының білім беруші, білім алушы, админ типтерінің болуы;
- Қолданушылардың құпиясөздерінің шифрлануы;

Кесте А.1.3.2.1 – Қолданушылар мүмкіндігі кестесі

Қолданушы типі\мүмкіндіктер	Білім беруші	Білім алушы	Админ
Материалдарды оқу	✓	✓	✓
Материалдарды жою	✓	✕	✓
Материалдарды өзгерту	✓	✕	✓
Қолданушыларды өзгерту	✕	✕	✓
Қолданушыларды қосу	✕	✕	✓
Қолданушыларды жою	✕	✕	✓

Жүйеге:
- Интуитивті түсінікті интерфейс;
- Материалдардың қол жетімділігі.

А.1.4 Техникалық талаптар

Жүйе кроссплатформалы болуы тиіс. Ең минималды конфигурация: – процессор түрі – Pentium және одан жоғары; – жад сыйымдылығы – 64 МБ және одан жоғары.

Б Қосымшасы
(міндетті)

Бағдарлама коды

Urls.py

```
from unicodedata import name
from django.contrib.auth.views import LogoutView, LoginView
from django.urls import path
from .views import *

urlpatterns = [
    path("detail/<int:id>/",detail,name="detail"),
    path("detail_thesis/<int:id>/", detail_thesis, name="detail_thesis"),
    path("detail_article/<int:id>/", detail_article, name="detail_article"),
    path('category_book/<int:id>/', category_book, name='category_book'),
    path('category_article/<int:id>/', category_article, name='category_article'),
    path('category_thesis/<int:id>/', category_thesis, name='category_thesis'),


    path("",index,name="home"),
    path("searches/",searches,name="searches"),
    path("teacher_searches/", teacher_searches, name="teacher_searches"),

    path("student_registraion/",student_registration,name="student_registration"),
    path("studentindex/",student_index,name="student_home"),
    path("student_userpage/",student_userpage,name="student_userpages"),
    path("studenorder_create/",student_order_create,name="student_order_create"),
    path("student_order_list/",student_order_list,name="student_order_list"),
    path("student_cart_add/<int:bookid>/",student_cart_add,name="student_cart_add"),
    path("student_cart_detail/",student_cart_details,name="student_cart_detail"),
    path("student_cart_remove<int:bookid>/",student_cart_remove,name="student_cart_remove"),
    path("student_create_thesis/",student_create_thesis,name="student_create_thesis"),

    path('student_category_book/<int:id>/',                    student_show_category_book,
name='student_show_category_book'),
    path('student_category_article/<int:id>/',                 student_show_category_article,
name='student_show_category_article'),
    path('student_category_thesis/<int:id>/',                  student_show_category_thesis,
name='student_show_category_thesis'),
    path("student_detail_article/<int:id>/",student_detail_article,name="student_detail_article"),
    path("student_detail_thesis/<int:id>/", student_detail_thesis, name="student_detail_thesis"),
    path("student_detail_book/<int:id>/",student_detail_book,name="student_detail_book"),

    #
    path('studentlogin', all_login,name="login"),
    path("teacher_registration/", teacher_registration, name="teacher_registration"),
    path("exit/",LogoutView.as_view(),name="exit"),
    path('afterlogin', afterlogin_view,name='afterlogin'),
```

Б қосымшасының жалғасы

```
    #teacher
    path("teacher_detail_article/<int:id>/",teacher_detail_article,name="teacher_detail_article"),
    path("teacher_detail_thesis/<int:id>/", teacher_detail_thesis, name="teacher_detail_thesis"),
    path("teacher_detail_book/<int:id>/",teacher_detail_book,name="teacher_detail_book"),

path("teacher_show_category/<int:id>/",teacher_show_category,name="teacher_show_category"),

    path("teacher_bookcategory/",teacher_add_category_book,name="teacher_category"),
    path("teacher_addbook/",teacher_addbook,name="teacher_addbook"),
    path("teacher_home/",teacher_dashboard,name="teacher_home"),
    path("teacher_userpage/",teacher_userpage,name="teacher_userpage"),
    path("teacher_book_author/",teacher_add_author,name="teacher_addauthor"),
    path("teacher_thesis_create/",teacher_thesis_create,name="teacher_thesis_create"),
    path("teacher_thesis_category/",teacher_thesis_category,name="teacher_thesis_category"),
    path("teacher_showstudent/",teacher_showstudent,name="teacher_showstudent"),

path("teacher_show_category_thesis/<int:id>/",teacher_show_category_thesis,name="teacher_sho
w_category_thesis"),

path("teacher_show_category_article/<int:id>/",teacher_show_category_article,name="teacher_sho
w_category_article"),

    path("teacher_cart_add/<int:bookid>/", teacher_cart_add, name="teacher_cart_add"),
    path("teacher_card_detail",teacher_cart_details,name="teacher_card_detail"),
    path("teacher_cart_remove<int:bookid>/", teacher_cart_remove, name="teacher_cart_remove"),
    path("teacher_order_create/",teacher_order_create,name="teacher_order_create"),
    path("teache_order_list/",teacher_order_list,name="teacher_order_list"),
    path("teacher_article/",teacher_article_create,name="teacher_article_create"),
    path("teacher_article_cat/",teacher_article_category,name="teacher_article_category"),

    path("teacher_report_book/",teacher_report_book,name="teacher_report_book"),
    path("teacher_report_thesis/", teacher_report_thesis, name="teacher_report_thesis"),
    path("teacher_report_article/", teacher_report_article, name="teacher_report_article"),
    path("teacher_report_order/", teacher_report_orderItem, name="teacher_report_order"),

    #admin
    path("admin_report_all_article/", admin_report_article_all, name="admin_report_all_article"),
    path("admin_report_all_order/", admin_report_all_orderItem, name="admin_report_all_order"),
    path("admin_report_all_book/", admin_report_all_book, name="admin_report_all_book"),
    path("admin_report_all_thesis/", admin_report_thesis_all, name="admin_report_all_thesis"),

    path("admin_report_book/",admin_report_book,name="admin_report_book"),
    path("admin_report_thesis/", admin_report_thesis, name="admin_report_thesis"),
    path("admin_report_article/", admin_report_article, name="admin_report_article"),
    path("admin_report_order/", admin_report_orderItem, name="admin_report_order"),
    path("admin_cart_add/<int:bookid>/", admin_cart_add, name="admin_cart_add"),
    path("admin_card_detail", admin_cart_details, name="admin_card_detail"),
    path("admin_cart_remove<int:bookid>/", admin_cart_remove, name="admin_cart_remove"),
    path("admin_order_create/", admin_order_create, name="admin_order_create"),
```

```
    path("admin_order_list/", admin_order_list, name="admin_order_list"),
    path("admin_article/", admin_article_create, name="admin_article_create"),
    path("admin_article_cat/", admin_article_category, name="admin_article_category"),

    path('admin_category_book/<int:id>/', admin_show_category_book,
name='admin_show_category_book'),
    path('admin_category_article/<int:id>/', admin_show_category_article,
name='admin_show_category_article'),
    path('admin_category_thesis/<int:id>/', admin_show_category_thesis,
name='admin_show_category_thesis'),
    path("admin_detail_article/<int:id>/",admin_detail_article,name="admin_detail_article"),
    path("admin_detail_thesis/<int:id>/", admin_detail_thesis, name="admin_detail_thesis"),
    path("admin_detail_book/<int:id>/",admin_detail_book,name="admin_detail_book"),
    path("admin_regis/", admin_registration, name="admin_registration"),

    path("admin_bookcategory/", admin_add_category_book, name="admin_category"),
    path("admin_addbook/", admin_addbook, name="admin_addbook"),
    path("admin_home/", admin_dashboard, name="admin_home"),
    path("admin_userpage/", admin_userpage, name="admin_userpage"),
    path("admin_book_author/", admin_add_author, name="admin_addauthor"),
    path("admin_thesis_create/", admin_thesis_create, name="admin_thesis_create"),
    path("admin_thesis_category/", admin_thesis_category, name="admin_thesis_category"),
    path("admin_showstudent/", admin_showstudent, name="admin_showstudent"),
    path("admin_show_category_thesis/<int:id>/", admin_show_category_thesis,
name="admin_show_category_thesis"),
    path("admin_show_category_article/<int:id>/",
admin_show_category_article,name="admin_show_category_article"),
    path("admin_searches/",admin_searches,name="admin_searches"),
    path("admin_show_teacher/",admin_showteacher,name="admin_show_teacher"),
    path("admin_edit_profile/",admin_edit_profile,name="admin_edit_profile"),
    path("admin_del_other/<int:id>/",admin_del_other,name="admin_del_other"),
    path("admin_del_other_teachers/<int:id>/", admin_del_other_teachers,
name="admin_del_other_teachers"),
    path("admin_add_other/", admin_create_other, name="admin_create_other"),
    path("admin_add_other_teachers/", admin_create_other_teachers,
name="admin_create_other_teachers"),
    path("admin_order_lists",admin_order_lists,name="admin_order_lists"),
path("admin_create_order_edit/<int:id>/",admin_create_order_edit,name="admin_create_order_edi
t"),
]
```

Models.py

```
from django.db import models
from django.contrib.auth.models import User
from django.db.models.signals import post_save
from django.dispatch import receiver
from django.urls import reverse


TYPE_CHOICES = (
```

```
    ("Книга","Книга"),
    ("Дипломная работа","Дипломная работа"),
    ("Статья","Статья")
)

class AdminProfile(models.Model):
    photo = models.ImageField(upload_to="users/%Y/%m/%d/",  verbose_name="Фото")
    user = models.OneToOneField(User,on_delete=models.CASCADE,
verbose_name="Пользыватель")

    def __str__(self):
        return self.user.username

class TeacherProfile(models.Model):
    photo = models.ImageField(upload_to="users/%Y/%m/%d/",  verbose_name="Фото")
    user = models.OneToOneField(User,on_delete=models.CASCADE,
verbose_name="Пользыватель")
    address = models.CharField(max_length=255, verbose_name="Адрес")

    def __str__(self):
        return self.user.username

class StudentProfile(models.Model):
    photo = models.ImageField(upload_to="users/%Y/%m/%d/",  verbose_name="Фото")
    user = models.OneToOneField(User,on_delete=models.CASCADE,
verbose_name="Пользыватель")
    address = models.CharField(max_length=255, verbose_name="Адрес")

    def __str__(self):
        return self.user.username


class Book(models.Model):
    teacher = models.ForeignKey(User,on_delete=models.CASCADE,null=True)
    photo = models.ImageField(upload_to="photos/%Y/%m/%d/", verbose_name="Фото")
    author = models.ForeignKey("Author", on_delete=models.CASCADE,
verbose_name="Автор",blank=True,null=True)
    category = models.ForeignKey("Category", on_delete=models.CASCADE,
verbose_name="Категория")
    name = models.CharField(max_length=100, verbose_name="Название")
    file = models.FileField(upload_to="files/%Y/%m/%d/",verbose_name="Книга")
    created_at = models.DateTimeField(auto_now_add=True,verbose_name="Создано")
    updated_at = models.DateTimeField(auto_now=True,verbose_name="Последний изм.")
    desc = models.TextField(verbose_name="Описание")
    type =
models.CharField(max_length=255,verbose_name="Тип",choices=TYPE_CHOICES,default="Кни
га")
    def __str__(self):
        return self.name
```

```
    class Meta:
        verbose_name = "Книга"
        verbose_name_plural = "Книги"



class TheSisCategory(models.Model):
    name = models.CharField(max_length=100, verbose_name="Категория")

    def __str__(self):
        return self.name


class TheSis(models.Model):
    student = models.ForeignKey(User,on_delete=models.CASCADE)
    name = models.CharField(max_length=100, verbose_name="Название")
    photo = models.ImageField(upload_to="photos/%Y/%m/%d/", verbose_name="Фото")
    category = models.ForeignKey(TheSisCategory, on_delete=models.CASCADE,
verbose_name="Категория")
    file = models.FileField(upload_to="files/%Y/%m/%d/",verbose_name="Дипломная")
    created_at = models.DateTimeField(auto_now_add=True,verbose_name="Создано")
    updated_at = models.DateTimeField(auto_now=True,verbose_name="Последний изм.")
    desc = models.TextField(verbose_name="Описание")
    type =
models.CharField(max_length=255,verbose_name="Тип",choices=TYPE_CHOICES,default="Ди
пломная работа")

    def __str__(self):
        return self.name

    class Meta:
        verbose_name = "Дипломная"
        verbose_name_plural = "Дипломные"


class Category(models.Model):
    name = models.CharField(max_length=100, verbose_name="Название")

    def __str__(self):
        return self.name

    class Meta:
        verbose_name = "Категория"
        verbose_name_plural = "Категории"

class Author(models.Model):
    name = models.CharField(max_length=100, verbose_name="Название")
    desc = models.TextField(verbose_name="Описание")
```

```
    photo = models.ImageField(upload_to="author_photos/%Y/%m/%d/", verbose_name="Фото")

    def __str__(self):
        return self.name


    class Meta:
        verbose_name = "Автор"
        verbose_name_plural = "Авторы"




class Order(models.Model):
    customer = models.ForeignKey(User, on_delete = models.CASCADE)
    name = models.CharField(max_length=30)
    email = models.EmailField()
    phone = models.CharField(max_length=16)
    address = models.CharField(max_length=150)
    account_no = models.CharField(max_length = 20)
    transaction_id = models.IntegerField()
    totalbook = models.IntegerField()
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(auto_now=True)
    status = models.BooleanField("Статус",default=False)

    class Meta:
        ordering = ('-created', )

    def __str__(self):
        return 'Order {}'.format(self.id)

    def get_total_cost(self):
        return sum(item.get_cost() for item in self.items.all())


class OrderItem(models.Model):
    order = models.ForeignKey(Order, on_delete=models.CASCADE)
    book = models.ForeignKey(Book, on_delete=models.CASCADE)
    quantity = models.PositiveIntegerField(default=1)

    def __str__(self):
        return '{}'.format(self.id)

    def get_cost(self):
        return self.price * self.quantity
class ArticleCategory(models.Model):
    name = models.CharField(max_length=255,verbose_name="Категория")
```

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

class Article(models.Model):
    teacher = models.ForeignKey(User,on_delete=models.CASCADE,null=True)
    photo = models.ImageField(upload_to="photos/%Y/%m/%d/", verbose_name="Фото")
    name = models.CharField(max_length=100, verbose_name="Название")
    category =
models.ForeignKey(ArticleCategory,verbose_name="Категория",on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True,verbose_name="Создано")
    updated_at = models.DateTimeField(auto_now=True,verbose_name="Последний изм.")
    desc = models.TextField(verbose_name="Описание")
    type =
models.CharField(max_length=255,verbose_name="Тип",choices=TYPE_CHOICES,default="Ста
тья")

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

Forms.py

```
from django import forms
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm
from django.forms import Textarea

from .models import Order, StudentProfile, TeacherProfile, TheSis, TheSisCategory, Book,
Category, Author, Article, \
    ArticleCategory, AdminProfile, OrderItem


class LoginForm(forms.Form):
    username = forms.CharField(max_length=50,
widget=forms.TextInput(attrs={"placeholder":"Логин"}))
    password = forms.CharField(max_length=50,
widget=forms.PasswordInput(attrs={"placeholder":"Пароль"}))

class TeacherExtraForm(forms.ModelForm):
    photo = forms.ImageField(max_length=50,
widget=forms.FileInput(attrs={"placeholder":"Фото"}))
    address = forms.CharField(max_length=50,
widget=forms.TextInput(attrs={"placeholder":"Адрес"}))

    class Meta:
        model = TeacherProfile
        fields = ["photo","address"]

class StudentExtraForm(forms.ModelForm):
photo = forms.ImageField(max_length=50,
widget=forms.FileInput(attrs={"placeholder":"Фото"}))
    address = forms.CharField(max_length=50,
```

```
 widget=forms.TextInput(attrs={"placeholder":"Адрес"}))

    class Meta:
        model = StudentProfile
        fields = ["photo","address"]

TYPE_CHOICES = (
    ("Книга","Книга"),
    ("Дипломная работа","Дипломная работа"),
    ("Статья","Статья")
)
class BookForm(forms.ModelForm):
    type = forms.ChoiceField(choices=TYPE_CHOICES,widget=forms.RadioSelect())
    class Meta:
        model = Book
        fields = ["name","photo","author","category","file","desc","type"]

class ThesisForm(forms.ModelForm):
    class Meta:
        model = TheSis
        fields = ["name","photo","desc","file","category"]

    # def __init__(self, road_choices=None, *args, **kwargs):
    #     super(ThesisForm, self).__init__(*args, **kwargs)
    #     if road_choices:
    #         self.fields['category'].choices = road_choices

class TeacherUserForm(forms.ModelForm):
    first_name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"placeholder":
"Имя"}))
    last_name = forms.CharField(label="Фамилия", widget=forms.TextInput(attrs={"placeholder":
"Фамилия"}))
    email = forms.EmailField(label="Email", widget=forms.EmailInput(attrs={"placeholder":
"Почта"}))
    username = forms.CharField(label="Логин", widget=forms.TextInput(attrs={"placeholder":
"Логин"}))
    password = forms.CharField(label="Пароль",
widget=forms.PasswordInput(attrs={"placeholder": "Пароль"}))

    class Meta:
        model = User
        fields=['first_name','last_name',"email",'username','password']


class StudentUserForm(forms.ModelForm):
    first_name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"placeholder":
"Имя"}))
  last_name = forms.CharField(label="Фамилия", widget=forms.TextInput(attrs={"placeholder":
"Фамилия"}))
```

```
  email = forms.EmailField(label="Email", widget=forms.EmailInput(attrs={"placeholder":
"Почта"}))
    username = forms.CharField(label="Логин", widget=forms.TextInput(attrs={"placeholder":
"Логин"}))
    password = forms.CharField(label="Пароль",
widget=forms.PasswordInput(attrs={"placeholder": "Пароль"}))

    class Meta:
        model = User
        fields=['first_name','last_name',"email",'username','password']


class RegistrationForm2(UserCreationForm):
    first_name = forms.CharField(label="Имя",
widget=forms.TextInput(attrs={"placeholder":"Имя"}))
    last_name = forms.CharField(label="Фамилия",
widget=forms.TextInput(attrs={"placeholder":"Фамилия"}))
    email = forms.EmailField(label="Email",
widget=forms.EmailInput(attrs={"placeholder":"Почта"}))
    username = forms.CharField(label="Логин",
widget=forms.TextInput(attrs={"placeholder":"Логин"}))
    password1 = forms.CharField(label="Пароль",
widget=forms.PasswordInput(attrs={"placeholder":"Пароль"}))
    password2 = forms.CharField(label="Подтверждение Пароля",
widget=forms.PasswordInput(attrs={"placeholder":"Подтверждение Пароля"}))

    class Meta:
        model = User
        fields = [
            'first_name',
            'last_name',
            'email',
            'username',
            'password1',
            'password2',
        ]

    def save(self, commit=True):
        user = super(RegistrationForm2, self).save(commit=False)
        user.first_name = self.cleaned_data['first_name']
        user.last_name = self.cleaned_data['last_name']
        user.email = self.cleaned_data['email']

        if commit:
            user.save()
        return user
class RegistrationForm(UserCreationForm):
    name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"required
class":"registration_inputs","id":"name"})
```

```
    name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"required
class":"registration_inputs","id":"name"}))
    email = forms.EmailField(label="Email", widget=forms.EmailInput(attrs={"required
class":"registration_inputs","id":"email"}))
    username = forms.CharField(label="Логин", widget=forms.TextInput(attrs={"required
class":"registration_inputs","id":"surname"}))
    password1 = forms.CharField(label="Пароль 1", widget=forms.PasswordInput(attrs={"required
class":"registration_inputs","id":"pass"}))
    password2 = forms.CharField(label="Пароль 2", widget=forms.PasswordInput(attrs={"required
class":"registration_inputs","id":"conf_pass"}))

    class Meta:
        model = User
        fields = [
            'name',
            'email',
            'username',
            'password1',
            'password2',
        ]

    def save(self, commit=True):
        user = super(RegistrationForm, self).save(commit=False)
        user.first_name = self.cleaned_data['name']
        user.email = self.cleaned_data['email']

        if commit:
            user.save()

        return user

class UserForm(forms.ModelForm):
    class Meta:
        model = User
        fields = ("username","first_name","last_name","email")


class OrderItemForm(forms.ModelForm):
    class Meta:
        model = OrderItem
        fields = "__all__"

class OrderCreateForm(forms.ModelForm):
    class Meta:
        model = Order
        fields = ['name', 'email', 'phone', 'address', 'account_no', 'transaction_id','status']
class OrderCreateForm2(forms.ModelForm):
    class Meta:
        model = Order
```

```
        fields = ['name', 'email', 'phone', 'address', 'account_no', 'transaction_id',]

class ThesisCategoryForm(forms.ModelForm):
    class Meta:
        model = TheSisCategory
        fields = "__all__"



class CategoryForm(forms.ModelForm):
    class Meta:
        model = Category
        fields = "__all__"

class AuthorForm(forms.ModelForm):
    class Meta:
        model = Author
        fields = "__all__"


class ArticleForm(forms.ModelForm):
    class Meta:
        model = Article
        fields = ["photo","name","category","desc","type"]

class ArticleCategoryForm(forms.ModelForm):
    class Meta:
        model = ArticleCategory
        fields ="__all__"

class AdminUserForm(forms.ModelForm):
    first_name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"placeholder":
"Имя"}))
    last_name = forms.CharField(label="Фамилия", widget=forms.TextInput(attrs={"placeholder":
"Фамилия"}))
    email = forms.EmailField(label="Email", widget=forms.EmailInput(attrs={"placeholder":
"Почта"}))
    username = forms.CharField(label="Логин", widget=forms.TextInput(attrs={"placeholder":
"Логин"}))
    password = forms.CharField(label="Пароль",
widget=forms.PasswordInput(attrs={"placeholder": "Пароль"}))
    class Meta:
        model = User
        fields=['first_name','last_name',"email",'username','password']
class AdminUserForm2(forms.ModelForm):
    first_name = forms.CharField(label="Имя", widget=forms.TextInput(attrs={"placeholder":
"Имя"}))
    last_name = forms.CharField(label="Фамилия", widget=forms.TextInput(attrs={"placeholder":
"Фамилия"}))
```

```
    email = forms.EmailField(label="Email", widget=forms.EmailInput(attrs={"placeholder":
"Почта"}))
    class Meta:
        model = User
        fields=['first_name','last_name',"email"]

class AdminExtraForm(forms.ModelForm):
    photo = forms.ImageField(max_length=50,
widget=forms.FileInput(attrs={"placeholder":"Фото"}))

    class Meta:
        model = AdminProfile
        fields = ["photo"]
```