

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Коммерциялық емес акционерлік қоғамы  
АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТІ

Ақпараттық жүйелер кафедрасы

«Қорғауға жіберілді»

Кафедра меңгерушісі

Иманташев Ш.Н., т.ғ.к., доцент  
(аты-жөні, ғылыми дәрежесі, атағы)

«    » 20 ж.  
(қолы)

ДИПЛОМДЫҚ ЖОБА

Тақырыбы: АЭБЗ кітапханасынан жиналған автоматтандырылған ақпараттық жүйесінің бағалау құрамы

5В070300-Ақпараттық жүйелер  
мамандығы бойынша

Орындаған Смазиев Н.Т. Ис-10-1  
(аты-жөні) (тобы)

Жетекші Ибраев М.С., т.ғ.к., аға оқытушы  
(аты-жөні, ғылыми дәрежесі, атағы)

Кеңесшілер :

Экономикалық бөлім бойынша :

Ж.К. ПРОКТОРОВ В. Базилев  
(ғылыми дәрежесі, атағы, аты-жөні)  
«05» 05 2014 ж.  
(қолы)

Өмір тіршілігі қауіпсіздігі бойынша:

ата оқытушы Тордаев Д.Д.  
(ғылыми дәрежесі, атағы, аты-жөні)  
«20» 05 2014 ж.  
(қолы)

Есептеу техникасын қолдану бойынша :

Т.ғ.к., аға оқытушы Қасымбаева Б.К.  
(ғылыми дәрежесі, атағы, аты-жөні)  
«05» 06 2014 ж.  
(қолы)

Молшер бақылаушы:

к.ғ. ғыл. кандидаты, аға оқытушы Аманжолдасов К.Б.  
(ғылыми дәрежесі, атағы, аты-жөні)  
«06» 06 2014 ж.  
(қолы)

Пікір жазушы :

(ғылыми дәрежесі, атағы, аты-жөні)  
«    » 20 ж.  
(қолы)

Алматы 2014 ж.

Коммерциялық емес акционерлік қоғамы  
АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТИ

1. Аналитикалық бөлім
2. Жобалау бөлім
3. Инженерлік бөлім
4. Эксперименталь бөлім
5. Тіршілік қауіпсіздігі

ДИПЛОМ      жобасын      дайындау

## KECTECI

[illegible]

Тапсырманың берілген уақыты « 22 » қапан 20 14 ж.

Кафедра меңгерушісі \_\_\_\_\_ Иманалиев Ш. И., Т.ғ.к., доцент  
(қолы) (аты-жөні, ғылыми дәрежесі, атағы)

Жоба жетекшісі \_\_\_\_\_ Ибраев М.С., т.е.к., өаа өксетунис  
(қолы) (аты-жөні, ғылыми дәрежесі, атағы)

Орындалатын тапсырманы  
кабылдаған студент

Сызба материалдарының (міндетті түрде дайындалатын сызуларды көрсету) тізімі

- 2.1 Сурет - Нүктені қолданудың абстракциялық моделі
- 2.2 Сурет - Web-нүктенің компоненттерінің орналасу ұяғрафиясы
- 2.3 Сурет - Паралельді және қосымша өлшемдерді бар күйдегі орналасу
- 2.4 Сурет - Web-нүктенің негізгі компоненттерінің жинақталу көрінісі
- 2.5 Сурет - Web-нүктенің құрамының объектілердің жинақталу көрінісі
- 2.6 Сурет - Веб-суреттің өлшемдері мен жинақталу ұяғрафиясы
- 2.7 Сурет - Ерекшелік моделі
- 3.1 Сурет - Kivy-тің GTK-мен ұяғрафиялық EER
- 3.2 Сурет - Келесі кезеңіс-әрекеттің жинақталу ұяғрафиясы
- 3.3 Сурет - Kivy-тің GTK-мен ұяғрафиялық EER
- 3.4 Сурет - Kivy-тің GTK-мен ұяғрафиялық EER
- 3.5 Сурет - Kivy-тің GTK-мен ұяғрафиялық EER
- 3.6 Сурет - Kivy-тің GTK-мен ұяғрафиялық EER

Негізгі ұсынылатын әдебиеттер

1. Бут, Прад, Максимчук, Роберт А., Эжел, Майкл Ч., Энз, Робби Дие, Кокамен, Рини, Хюстон, Келли А. Объектно-ориентированный анализ и проектирование с примерами приложений, 3-е изд.: Пер. с англ. - М.: ООО "И.Д. Вильямс", 2008. - 710 стр.
2. Хорстманн, Кей С., Коррелл, Гарри. Java 2 Библиотека профессионала, том 1. Вектор. 7-е издание, 2012 - 716 стр.
3. Specification: JSR-342 Java Platform, Enterprise Edition. 7
4. Bloxwich Sergei Open Source Edition Administration Guide, Release 4.0
5. My360 сайт <http://dev.mysql.com/doc/>

Жоба бойынша бөлімшелерге қатысты белгіленген кеңесшілер

бөлімшелер	кеңесші	мерзімі	қолы
Экономика	Базылев К.В.	20.04-22.05	Базылев К.В.
Технологиялық	Торбаев В.В.	24.04-20.05	Торбаев В.В.
Негізгі білім	Торбаев М.С.	27.11-11.05	Торбаев М.С.

Аннотация

В данной работе представлен отчет дипломного проекта по теме “Разработка информационной подсистемы для автоматизации работы библиотеки АУЭС”. Здесь полностью описывается как это можно сделать, какими средствами, и что это даст в результате. Показаны преимущество использование Web технологии для автоматизации работы библиотеки.

Данный отчет состоит из пяти глав и четырнадцати подглав. Количество страниц 82, использовано 28 рисунков и 16 таблиц.

### **Андатпа**

Берілген жұмыста “АЭЖБУ кітапханасының жұмысын автоматтандыруға арналған ақпараттық жүйе бөлігін құру” атты дипломдық жобаның жасалу қорытындысы берілген. Мұнда кітапхана жұмысын автоматтандыруға Web технологияларды қолданудың маңыздылығы, беретін артықшылықтары көрсетілген.

Жұмыс 5 тараудан және он төрт тарау бөлшегінен ырады. Есеп беруде 82 бет, 28 суреттен және 16 кесте бар.

### **Annotation**

This paper presents a report of the graduation project on “Development of an information subsystem for automation of AUPET library”. It is fully described how to do it, by what tools, and that this will result. The advantages of using Web technology to automate the library.

This report consists of five chapters and subchapters fourteen. Number of pages 82, used 28 figures and 16 tables.

## Мазмұны

Кіріспе	8
1 Аналитикалық бөлім	11
1.1 Жоғары білім беру орындарында ақпараттық технологияларды енгізу маңыздылығы	11
1.2 АЭЖБУ кітапханасы	15
1.3 Қолданылған құрал саймандар	17
2 Жобалық бөлім	29
2.1 АЖ бөлігіне қойлатын талаптар	29
2.2 АЖ бөлігінің құрылуы	31
2.3 АЖ бөлігін құруға қолданылған бағдарламалық қамтамалар	37
3 Эксперименталды бөлім	47
3.1 Жүйені қамтамасыз ететін ДҚ-н құру	47
3.2 АЖ бөлігін құрастырылуы	49
3.3 АЖ бөлігін қолдану жайлы нұсқаулар	57
4 Экономикалық бөлім	64
4.1 Жұмыстың сипаттамасы мен қажеттілігінің негіздемесі	64
4.2 АЖ бөлшегін жобалауға және құруға кететін шығынды есептеу	64
4.3 Берілген БҚ – ның экономикалық тиімділігі	69
5 Тіршілік қауіпсіздігі	71
5.1 Кітапхана АЖ-ның администраторының жұмыс орнындағы негізгі қауіпті факторларды талдау	71
5.2 Есептік бөлім	77
Қорытынды	83
Қолданылған әдебиеттер тізімі	84
Қосымша А	85
Қосымша Ә	110

## Кіріспе

Адамзат баласының даму тарихының кез келген уақытын алсақ та ең маңызды мақсаттарының бірі ақпаратты уақыт аралығында сақтап келесі болашақ ұрпаққа жеткізу болды. Сол ақпараттың ұрпақ аралығында жетіп отырғанның арқасы бола қазіргі күнде ғылым, өнер, мәдениеттер осындай жоғары деңгейге жеткетті.

Кітап шығару ісі шыққанан бастап ақпаратты сақтау, қол жеткізу басты әдісі ретінде - басылымдық әдіс, ал олардың сақталу жері ретінде кітапхана болып қалыптасты. Осы күнде жазба және баспа құжаттарды сақтау және қолдану әжептәуір жақсы игерілген және бұл салада көптеген жылдар бойы келе жатқан мамандар ұрпақтарының және зертеушілер жұмысының мол тәжірибесі бар. Алайда бұндай түрде сақталған ақпарат көбейген сайын олармен сақтау, іздеу, сұрыптау, жеткізу жұмыстарын жүргізу өте қиынға соғады.

XX ғасырдың 70-80 жылдарынан бастап есептеуіш техникасының дамуымен ақпаратты электронды түрде сақтау және тарату мүмкіндіктері пайда болды, бұл кезіндегі кітап басылым технологиясы шыққан тәрізді адам баласының даму сатысындағы үлкен бір революциялық жетістігі.

Электронды форма бүгінгі таңда бізге ақпаратты мейлінше берік және тұтас сақтауға, шапшаң және кең таратуға және де алдыңғы баспа формасында болмаған, ақпаратпен тікелей манипуляция жұмыстарын жасауға мүмкіндік берді. Осыған байланысты соңғы жылдары бүкіл әлемде электронды ақпарат саны өте жылдам өсуде. 1998 жылы жүргізілген бірнеше зерттеулерге қарағанда ғаламторда электронды ақпарат саны 300 млн.-ға жетті және әлі күнде қарқынды өсуде.

Сонымен қатар жазба ақпараттын электронды формаға ауыстыру процессі бүкіл әлемде қарқынды жүріп жатыр. Оған дәлел ретінде АҚШ-тағы 20 млн. құжатты электронды түрге ауыстыруға арналған Пентогонның жобасы, Жапониядағы Ұлттық Электронды Кітапхана жасау жобасының барысында 10 млн. – ға жуық бет мәтіндік ақпарат электронды формаға ауыстырылуын айтуға болады.

Тек басқару саласында ғана емес бұндай қарқын білім саласындағы ақпараттарды ауыстыруда да байқауға болады. Ондай жобалардың алғашқыларының бірі Гутенберг жобасы.

Гутенберг жобасы - әмбебап электронды кітапхананы жасауды және таратуды мақсат ететін қоғамдық бастама. 1971 жылы құрылған жоба әлемдік әдебиеттің әртүрлі шығармаларын электронды қалыпқа өткізу мен мәтіндік форматта сақтауды қарастырады. Бұл негізінен еркін қолдануға болатын мәтіндер.

Және де XX ғасырдың ең үлкен жобаларының бірі ол Google Books жобасы.

Google Books (Google Кітаптар) - Google компаниясымен сандық үлгіге өткізілген кітаптардың мәтіні бойынша іздеу қызметі (АҚШ-тың ең үлкен кітапханаларының 10 миллионнан астам кітабы).

Бұл жобаны жасау барысында 2005 жылы желтоқсан айында Google компаниясы АҚШ пен Ұлыбританияның 5 ең үлкен кітапханаларымен олардың кітапхана қорын сандық үлгіге өткізу турасында келісімге отырды.

- Стэнфорд университетінің кітапханасы - толығымен (8 000 000 кітап);
- Мичиган университетінің кітапханасы - толығымен (7 000 000 кітап);
- Гарвард университетінің кітапханасы - 40 000 кітап;
- Оксфорд университетінің кітапханасы - 1900 ж. дейін басылған барлық кітап;

- Нью-Йорк қоғамдық кітапханасы - копирайтпен қорғалмаған студенттер мен мектеп оқушыларына арналған материалдар.

2010 жылға қарағанда проект барысында 15 миллион кітапқа қол жеткізуге мүмкіндік болды, оның 1 миллионға жуығы - қоғамдық қазына. Проект қызметкерлері әлемдегі кітаптар санын (қайта басылуларды санамағанда) 130 миллионға бағалап, осыншама кітапты онжылдықтың соңына дейін сандық үлгіге өткізбекші.

Өзіміздің отанымыз Қазақстанда да бұндай жобалар көптеп жүзеге асырылып жатыр. Олардың бірнешесіне тоқталып кетсек.

“Қазақша Уикипедия” жобасы - қазақ тіліндегі Уикипедияны дамыту мақсатында 2011 жылы WikiBilim қоғамдық қоры бастаған қоғамдық жоба. Жобаның негізгі мақсаты интернеттегі қазақ тілді мазмұнды жазбаша мәліметтерді арттыру, қазақ тіліндегі интеллектуалды ортаны, уики қауымдастықты қалыптастыру.

2011 жылдың қараша айының басында Қазақ Уикипедиясы 100 000 мақала межесіне жетті. Жоба ұйымдастырушылары ендігі мақсат ретінде мақала сапасын арттыру көзделіп отыр.

Және де Қазақстандық Ұлттық Электронды Кітапхана жобасының жасалуы барысында көптеген бағдарламалар іске асырылды.

«Мәдени мұра» бағдарламасы ұлттың бағалы құндылықтарының электрондық қорын құру мақсатымен жасалған. Осы қор ҚР Президенті Н.Ә. Назарбаевтың бастамасымен қабылданған. Қордың көлемін тарих, археология, этнография, халық әдебиетінің озық үлгілері мен мемлекеттік тілдегі жазбалар, сондай-ақ заманауи анықтамалық басылымдардан тұратын 500 сандық кітаптар құрайды.

«Қазақстан кітапханаларындағы сирек қолжазбалар» өткен жылдардағы (XIX ғасырдан 1940 жылға дейінгі) жалпы мәдениет пен ғылыми маңыздылыққа ие араб және орыс тілдеріндегі 1217 электронды қолжазбадан тұрады.

Және де басқа «Қазақстандық», «Қазақстан Халқы Ассамблеясы», «Ауылшаруашылық әдебиеттері», «Авторефераттар мен диссертациялар», «Университет хабаршылары» бағдарламаларының жасалуы барысында



Қазақстан Мемлекетінің ұлттық қазынасы болып табылатын 20 мыңға жуық кітаптың электронды формасы жасалды.

Электронды ақпараттың бұлай қарқынды өсуі оларды басқаратын ақпараттық жүйе құру қажеттілігіне алып келді. Соған байланысты XX ғасырдың 90 – шы жылдарынан бастап электронда формадағы баспа ақпаратын басқаруға арналған ақпараттық жүйе құру жұмысы кең ауқым алып, соның барысында «электронды кітапхана» деген жаңа термин пайда болды.

Электрондық кітапхана – электронды формадағы құжаттардың түрлі топтамаларын (мәтіндік, бейнелік, кескінді дыбыстық және т.б) берік сақтауға, тиімді пайдалануға және оларға телекоммуникациялық желілер арқылы қол жеткізуге мүмкіндік беретін ақпараттық жүйе. Электронды кітапхананың негізгі міндеті – ақпараттық ресурстарды интеграциялап, онда тиімді және әсерлі түрде навигация жүргізуге рұқсат беру.

Ақпараттық ресурстарды интеграциялау дегенімізде әр түрлі ақпараттың өзіне тән қасиеттерін, ұсыну ерекшеліктерін сақтап оларға қолайлы және бірегейлендірілген қолданбалы интерфейстер жәрдеммен қол жеткізуге, манипуляция жұмысын жүргізуге мүмкіндік беру мақсатымен ақпаратты біріктіру түсіндіріледі. Сонымен қатар ақпараттық ресурстарды біріктіру физикалық тұрғыда жасалуы міндетті емес, оны виртуалды түрде жасауға болат, ең бастысы ол қол жетімді ақпараттың қолданушыға біртұтас ақпараттық кеңістік ретінде сезілуін қамтамасыз ету керек.

Ал электронды кітапханадағы тиімді навигация дегенімізде, қолданушының өзіне қызықты және керекті, мейлінше толық және дәл ақпаратты барлық қол жетімді ақпараттық кеңістікте, аз күш шығынын жұмсау арқылы табу мүмкіндігі түсіндіріледі.

Бұл дипломдық жұмыстағы менің мақсатым осы берілген анықтамаларға жауап беретін Алматы Энергетика және Байланыс Университетіндегі студенттердің шығармашылық электронды кітапқа оңай қол жеткізуіне арналған, жүйелендірілген электронды кітапхананы құру, бүгінгі таңдағы оның маңыздылығын және артықшылықтарын көрсету болып табылады.

## **1 Аналитикалық бөлім**

### **1.1 Жоғары білім беру орындарында ақпараттық технологияларды енгізу маңыздылығы**

XXI ғасырдағы білім беру жүйесіне жаңаша ақпараттық технологияларды ендірудегі басты ұстаным – сапалы түрде іске кірісіп, тиімді әрекеттену, құндылықтарды тұтынып, нәтижелі табысқа жету. Бұл қағиданы ұстанып, оң әрекетке бағытталған тұлға сөзсіз табысқа жетеді, көздеген мақсаты орындалады. Дәлірек айтқанда, бұл – нәтижеге ие болудың кілті. Ал білімдендірудің арнасын кеңейткен интернетке қосылу, педагогтердің кәсіби мамандығын жетілдіру, ақпараттық тасқын, дамудағы жылдамдығына көз ілеспейтін бүгінгі шапшаңдық, білім жүйесіндегі жаңалықтар мен өзгерістерді игеру осыған дейінгі қолданылып келген әдіс-тәсіл, амалдарға бой бермей отыр. Олай болса, оқу-тәрбие үдерісінде болашақ мамандарды даярлауда ақпараттық технологиялар жүйесін пайдалану кәсіби міндеттердің біріне жатады.

Қазіргі орта кәсіптік және жоғары оқу орындарында электрондық байланыс жүйелерінде ақпарат алмасу интернет, электрондық пошта, телеконференция, бейне-конференция, телеқатынастық жүйелер арқылы іске асырылуда. Заман ағымына қарай күнделікті сабаққа бейне, аудио қондырғылары мен теледидарды, компьютерді қолдану айтарлықтай нәтижелер беруде. Кез келген сабақта электрондық оқулықты пайдалану оқушылардың танымдық белсенділігін арттырып қана қоймай, логикалық ойлау жүйесін қалыптастыруға, шығармашылықпен еңбек етуіне жағдай жасайды. Дәстүрлі оқулықты оның электрондық нұсқасына оңай айналдыруға болады. Бұл нұсқаның жетістігі – оны компьютер жадында сақтау мүмкіндігі, оның компьютерлік желілер арқылы таратылуы болып табылады. Ақпараттық технологияларды пайдаланудың ғылыми-педагогикалық, әдістемелік негіздері әлі қалыптасу үстінде.

Ақпараттық технологияларды дамыту Қазақстан экономикасының бәсекеге қабілеттілігін арттырудағы маңызды факторлардың бірі болып табылады. Соңғы жылдары елдегі ақпараттық технологияларды қолдануды дамыту қарқыны үлкен серпін алуда.

Біз тәуелсіз, мемлекеттік білімді жетілдіруге аса мән берген елде тұрамыз. Жалпы білім берудің мақсаты – терең білімнің, кәсіби дағдылары негізінде еркін бағдарлай білуге, өзін-өзі дамытуға, адамгершілік тұрғысынан жауапты шешімдерді қабылдауға қабілетті жеке тұлғаны қалыптастыру. Яғни жеке тұлғаны қалыптастыруға негізделген, ақпараттық технологияларды терең меңгерген, жылдам өзгеріп жататын бүгінгі заманға лайықты, жаңашыл тұлғаны қалыптастыру.

Ақпараттық технологияны бәсекеге қабілетті ұлттық білім беру жүйесін дамытуға және оның мүмкіндіктерін әлемдік-білімдік ортаға енудегі сабақтастыққа қолдану негізгі мәнге ие болып отыр.

Осыған сәйкес, жоғары білім беру саласындағы бүгінгі күнгі алға қойылған мақсаттардың бірі – жаңа ақпараттық технологияны білім беру жүйесіне пайдалана алатын, ақпараттық мәдениеті жоғары, білімді тұлғаны қалыптастыру. Білім беруді ақпараттандыру нәтижесінде әрбір студент ақпаратты еркін пайдалануға, оны қажетіне қарай талдай білу мүмкіндігіне ие болуы қажет. Өйткені бұл білім беру үдерісіне қатысушылардың интеллектуалдық белсенділігінің артуына, сол арқылы білім беру жүйесінің дамуын жетілдіруге негіз болады.

Білім беру жүйесіне оқытудың жаңа технологияларын енгізу және ақпараттық технологиямен қамтамасыз етуді жақсарту, студенттің ақпараттық мәдениетін қалыптастыру міндеттері бір-бірімен өзара тығыз байланысты, яғни оқу үдерісін қазіргі заманғы технологиялар мен оқытудың техникалық құралдарын енгізуді талап етеді. Мұндай талаптарды қанағаттандыру – білім беру жүйелеріне жоғары білікті мамандарды дайындау мәселелерін ғылыми негізде шешу қажеттілігімен тікелей байланысты.

Ақпараттық технологияны пайдалану жөніндегі қызметтің мақсаты: үйренушінің шығармашылық әлеуетін дамыту; коммуникативтік әрекеттерге қабілетті болуды дамыту; сараптамалық зерттеу қызметі дағдыларын дамыту; оқу қызметі мәдениетін дамыту; оқу-тәрбие үрдісінің барлық деңгейлерін қарқындандыру, оның тиімділігі мен сапасын арттыру; қазіргі қоғамның ақпараттануымен байланысты пайда болатын әлеуметтік тапсырысты жүзеге асыру.

Сондықтан оқытудың ақпараттық технологиясы-білімді жаңаша беру мүмкіндіктерін жасау, білімді қабылдау, білім сапасын бағалауы, оқу-тәрбиесі үрдісінде студенттің жеке тұлғасын жан-жақты қалыптастыру үшін, бойында ақпараттық мәдениетті, сауатты адам-ақпараттың қажет кезін сезіну, оны табу, бағалау және тиімді қолдану қабілетін арттырады деп санаймыз.

Ақпараттық технологияларды пайдалану арқылы шешілетін тағы да бір міндет – оқушылардың оқу материалдарын толық меңгеруі үшін оқу материалдарының практикалық жағынан тиімді ұсынылуына мүмкіндіктің болуы. Бұл мақсаттарға жету жолында электрондық оқулықтар, тексеру программалары, оқыту программалары сияқты программалық өнімдер қызмет етеді. Білім саласында компьютер – оқу үшін оқу құралы, ал оқытушы үшін жұмысшы болып табылады.

Оқытудың ақпараттық технологиялары осы ақпараттық білім жүйесінің шегінде іске асырылатын болғандықтан, осы білім технологиясына ақпараттық және бағдарламалық қолданумен көрсететін құралдар бір ғана компьютермен, оған енгізілген бағдарламамен шектеліп қалмауы керек. Шын мәнінде бәрі керісінше оқытудың ақпараттық-технологияларының бағдарламалық құралдары және білім технологияларының өздері ақпараттық білім ортасына ақпараттық білім жүйесінен бөлінген жүйелерге қосылады.

Ақпараттық технология электрондық есептеуіш техникасымен жұмыс істеуге, оқу барысында компьютерді пайдалануға, модельдеуге, электрондық оқулықтарды, интернетте жұмыс істеуге, компьютерлік оқыту

бағдарламаларына негізделеді. Ақпараттық әдістемелік материалдар коммуникациялық байланыс құралдарын пайдалану арқылы білім беруді жетілдіруді көздейді. Заман ағымына қарай ақпараттық технологияны қолдану айтарлықтай нәтиже беруде. Кез келген сабақта электрондық оқулықты пайдаланып, студенттердің танымдық белсенділігін арттырып қана қоймай, логикалық ойлау жүйесін қалыптастыруға, шығармашылықпен еңбек етуіне жағдай жасайды.

Ақпараттық технологияларды оқу-тәрбие үрдісінде қолдану студенттің қызығушылығын арттырып, шығармашылық шабытын шыңдап, ғылыми көзқарасын қалыптастырады.

«Қазіргі заманда жастарға ақпараттық технологиямен байланысты әлемдік стандартқа сай мүдделі жаңа білім беру өте қажет» деп, Елбасы атап көрсеткендей, жас ұрпаққа білім беру жолында ақпараттық технологияны оқу үрдісінде оңтайландыру мен тиімділігін арттырудың маңызы зор. Білім беру саласында АТ-ны дамытудың барлық жаңа үрдістерін ескеру және оларды білім беру процесінде пайдалану елдегі білім беру деңгейін жаңа деңгейге шығаруға көмек береді.

Бүгінгі күні білім беру ісіне құзыреттілік тұрғыдан келу арта түсуде. Бұл ретте басты орында – студенттердің, сондай-ақ, оқытушылардың бойында ақпараттық-байланыс құзыретін қалыптастыру қажеттілігі артады. Ақпараттық технологиялары (АТ) адамның іс-әрекетінің көптеген өрістерін қамтып, сапалы түрде іс-әрекет үрдісін ғана емес, сонымен қатар, оған деген қатынасты да өзгертеді. Бүгінде АТ білім беру үрдісінде де жан-жақты қолданыс тауып отыр, бірақ оның білім беру әдістемесіне қатысы жоқ, көбінесе тек ұйымдастырушылық, ақпараттық, т.б. мақсатта ғана, мысалы, оқытушы студенттерге белгілі бір ақпарат көздерін Ғаламтордан іздестіруді ұсынады. Білім беру үрдісінде АТ-ны пайдалануға қатысты мүмкіндіктердің кең ауқымы әлі де өз қолданысын тапқан жоқ.

АТ-ны белсенді түрде енгізіп жатқан оқытушылар дәл осы ақпараттық технологиялар оқытушыға өзінің іс-әрекетіне талдау жасап, шеберлігін ұштауға мүмкіндік береді дейді. Жаңа технологиялардың қай кезде студенттердің тақырыпты меңгеруіне көмектесе алатынын, ал қай кезде тікелей бетпе-бет қатынастың қажеттігін түсінуге болады. АТ қолдану барысында қандай да бір «із» қалып отырады, сол ізді талдай келе, өзіндік даму жолдарын айқындауға болады.

Заманауи әлемнің дамуы ақпараттық қоғамның қалыптасуына ықпал еткен технологиялық жаңалықтарға тікелей байланысты. Мұндай қоғамда жаңа ойлар тудыру, ақпаратты өңдеу – басты орында, заманауи байланыс технологиялары адамзаттың іс-әрекетінің барлық өрістеріне ықпал етеді, бұл жеке тұлғаның өзін-өзі дамыту, өзін-өзі жетілдіру қажеттілігін арттырады. Қазіргі уақытта білімі мол қоғам кез келген елдің технологиялық және әлеуметтік-экономикалық дамуының ең басты факторы болып отыр. Гуманистік бағдарлы заманауи білім беруде сан алуан түрлер, әдістер, технологиялар басты мақсат емес, олар жеке тұлғаның өзін-өзі дамытуы үшін

аса қолайлы жағдайды қамтамасыз ету: білім беру жүйесіне ақпараттық-байланыс технологияларын белсенді түрде енгізу; адамға өз болмысын түсініп, байыту үшін, қоршаған шындықпен қатынасындағы өзінің әлеуметтік орнын айқындау үшін жәрдем етуге бағытталған жаңа білім беру парадигмасын іске қосу сияқты білім берудің ең басты міндеттері аясында қарастырылады.

Ағарту саласында еңбек етіп жүрген әр оқытушы студенттерге жүйелі, сапалы білім беру үрдісінде бүгінгі заман талап етіп отырған ақпараттық технологиялардың мүмкіндіктерін игертуі қажет.

Ақпараттық білімді игертуде: таным, қабылдау кезінде бейнелерді құру, ең алдымен, бейнелі ес, көрнекі- бейнелі ойлау, шығармашылық ойлау елесі, егжей- тегжейлі көріністер қабылдау, сөздік- логикалық ойлау, сөздік- ойлау, ес, сезім қатынастары қамтылды.

Ақпараттық технологиямен жұмыс істеу барысында студент материалды толық меңгереді, теориялық және эмпирикалық ойлауы біркелкі дамиды, білім - білік дағдысы, қызығушылығы, танымдық белсенділігі артады, ал оқытушы студенттердің зейінін өзіне тарту жетістігіне ие болады. Оларды ойландыра, толғандыра отырып, зейінін оқу үрдісіне тарту әрекеттері оқытушының тиімді құралы болып табылады.

Білімді игеру дегеніміз – оны ұғу, есте сақтау, қолдана білу. Білімді игеру – алғашқы оқу материалын меңгеретін, білімді бекітетін, бақылау және оқу жұмыстарының нәтижесін бағалайтын, сондай- ақ, дидактикалық міндеттерді шешетін күрделі жүйе.

Ақпараттық технологиялардың мүмкіндіктерін қолданудың оқушыға берер мынандай тиімділіктері аңғарылды:

- түрлі ақпараттық, бейнелік, дыбыстық анықтамалар арқылы білімін жан- жақты жетілдіреді, дамытады ;
- өз бетінше сарамандық тапсырмаларды орындайды;
- тақырыптан қалып кеткен немесе дұрыс түсінбеген тақырыпты қосымша қайталауға мүмкіндік береді;
- пәнге қызығушылығы, үздіксіз ізденісі артады;
- ойлау, есте сақтау, пікірсайыстық қабілеті дамиды;
- өз ойын сызба, сурет, кескіндеме, кесте, графиктік модельдер түрінде жеткізеді;
- түрлі бейнелік, сілтемелік, нұсқаулық тапсырмаларды орындайды;
- түрлі деңгейдегі тест тапсырмаларын орындап, өзінің алған білімін тексереді.

Компьютерлік технологияның қиындықтарын жеңе отырып, болашақ ұстаздың өзіне деген, өз мүмкіндіктеріне деген сенімі ұлғайып, әрбір жаңа бағдарламаны «жеңген» сайын табысқа жету үміті арта түседі. Ол өзін білім берудегі «өркениетті» бағыттан хабарсыз «қараңғы адам» деп сезінбейді, керісінше, ол өзін оқу үрдісін ақпараттандыруға өз үлесін қосқан тұлға деп есептейді. Осы ішкі толғаныс адамды сомдау үрдісі ретінде жеке тұлғаның өзін-өзі дамытумен байланысты.

Оның үстіне, жеке тұлғаның өзін-өзі дамытуы мультимедиялық технологияның интербелсенділігі қасиетіне негізделген қарым-қатынас себепші болған жасанды орта жағдайында жандана түседі.

Мұндай қарым-қатынастың болуы адам мен машинаның өзара әрекеттестігі кезінде бейімделудің негізгі ықпалы болып табылады, оны адамның тіршілік әрекетінің жаңа өрісі деп қарастыруға болады.

Қазіргі қоғамды ақпараттандыру үдерісінің басым бағыттарының бірі – білімді ақпараттандыру, білім беру аясын әдістемелік және техникалық жабдықтармен қамтамасыз ету, оқыту мен тәрбие берудің психологиялық-педагогикалық мақсаттарына негізделген заманауи технологияларды оңтайлы пайдалану болып табылады. Бұл үдеріс төмендегі әрекеттерді іске асырады:

- білім беру жүйесін басқару тетіктерін автоматтандырылған ғылыми-педагогикалық қорларды, коммуникациялық желілерді пайдалану негізінде жетілдіру;

- қазіргі заманғы қоғамды ақпараттандыру шартында тұлғаны дамыту міндеттеріне сәйкес оқыту, тәрбиелеудің ұйымдастырылған формаларын, әдістерін сұрыптау, стратегиялары мен әдістемесін жетілдіру;

- студенттердің интеллектуалдық дамуына, өз бетінше білім алу дағдыларын қалыптастыруға, ақпараттық-оқу, тәжірибелік-зерттеу әрекеттерін іске асыруға, ақпараттарды өңдеу бойынша өзіндік жұмыстардың әртүрлілігіне бағдарланған оқытудың әдістемелік жүйелерін құру;

- студенттердің білім деңгейін бағалау мен бақылауды айқындаушы компьютерлік тестілік бағдарлама жасау әдістемесін құру және қолдану.

Жүйелей келе, ақпараттық технологиялардың, біріншіден, білім сапасының артуына, екіншіден, оқытушы мен студенттің үздіксіз ізденіс арқылы дамуына ықпалы зор екендігімен түйіндеймін.[12]

## 1.2 АЭЖБУ кітапханасы

XXI – ғасыр мәдениет, ақпарат ғасыры екені қазір әлемдегі адамзатқа белгілі және оны мойындауда. Қай қоғамның да тұрақты дамуында мәдениеттің алар орны ерекше, сол рухани мәдени тынысымыздың ордасы кітапхана. Кітапханада ғылым мен білім жинақталған зор рухани әлеуметтік күш бар. «Кітап білім бұлағы – білім - өмір шырағы» деген сөз тегіннен тегін айтылмаған. Бүгінгі күні көңілі ояу, көзі ашық сауатты оқырман кітаптан өзіне қажетті рухани нәр алады. Көптің ойын байытып, сауатын молайтып білім нәрінен сусындатар орда – кітапхана.

АЭЖБУ кітапханасы 1997ж. В.И.Ленин атындағы Қазақ Политехникалық институты және Алматы Энергетика институттарының кітап қорларының негізінде құрылды.

Алматы энергетика және байланыс университеті кітапханасы оқу процесін, студенттердің ғылыми-педагогикалық қызметін, оқытушылар мен ЖОО-ның ғылыми қызметкерлерін ақпараттық деректермен және әдебиеттермен қамтамасыз етеді. Кітапхана «А» ғимаратының 1 және 2-

қабатында, «С» ғимаратының 1-қабатында және №1, №2, №2А, №3 жатақаналарында орналасқан. 1614 м2 ауданды алып жатыр. 446 орынды 11 оқырман залы, 3 абонементі бар. Кітапхана жыл сайын 122485-тен астам оқырмандар қабылдайды. Кітапхана құрамында 38 адам жұмыс істейді, оның 34-і жоғары білімді, 4-і арнайы орта білімді.

Кітапхана қоры 832266 баспа бірлігін құраса, оның ішінде қазақ тілінде - 191097. Кітапхана қорында әр алуан басылымдар: қазақ, орыс, ағылшын тілдерінде кітаптар мен журналдар, ақпарат деректері, авторефераттар, диссертациялар және т.б. бар. Тақырыбы жағынан оқулықтар, ғылыми, ғылыми-техникалық, әлеуметтік-экономикалық, қоғамдық-саяси, көркем әдебиеттер және өнер, спорт бойынша әдебиеттер жинақталған. Кітапхана биыл 34538 дана (2593 атаумен) жаңа түсірілім қабылдады. 2014 жылы 300 мерзімді басылымдарға жазылды.

1998 ж. бері кітапхана процесін ЭВМ негізінде автоматтандырумен айналысады. 2005 ж. қарашасынан бастап «RABIS-КАТАЛОГИЗАТОР», «RABIS-ІЗДЕУШІ» и «RABIS-ОҚЫРМАНДАРДЫ ТІРКЕУ» электрондық каталогы жүргізілген. Каталог әмбебапты, библиографиялық сипаттамасымен орыс, қазақ және ағылшын тілдерінде мазмұндалған. 36800 құжаттар электрондық каталогта көрсетілген. «Кітапхана жұмысын ұйымдастыру», реферативтік деректер қорында 1310 атау, оның ішінде 130 қазақ тілінде бар.

Кітапхана EBSCO жобасы бойынша шетел деректер қорына, сонымен қатар «Polpred.com», «Elsevier», «Tomson», «Springer», және «РМЭБ» ресурстарына еркін кіру мүмкіндігіне ие.

1997 ж. бастап компакт-диск коллекциясы жинақталған. Кітапханада электрондық ресурстар 1886, оның ішінде 816-сы қазақ тілінде.

Кітапханада 79 компьютер, №2(А-127), №1, №2, №2А, №3 жатақаналарда 60 компьютер Интернет пен АЭЖБУ электрондық кітапханасына қосылған. 5 компьютер электрондық каталогқа және 19 компьютер кітапхана қызметкерлеріне арналған.[9]

Кітапхана – білімнің, мәдениеттің және руханияттың орталығы. Сонымен қатар өзіне қосымша міндеттемелер алды: «Парасат» клубымен бірлесіп танымал адамдармен ашық, есте қалатын кездесулер, «Ұлағат» көркем-әдебиеттер студенттік клубымен мәдениеттік-тәрбиелік кештер ұйымдастырады. Ай сайын кітапхананың оқырман залында кітап-иллюстрациялық көрмелер ұйымдастырылады. Барлық университеттік конференцияларда және семинарларда тақырыптық көрмелер ұйымдастырылады. Кітапхананың басты әлеуметтік мақсаты ол қолданушыларға ақпаратын ашық және тең қол жеткізуіне мүмкіндік беру. Алайда интернет-технологиялардың екпінді дамуы, кітапханалар үшін қызметтің қосымша әрекет өрісін ашты.

АЭЖБУ кітапханасы үшін, басқа жоғары оқу орындарының кітапханалары секілді, басты қолданушылар көзі студенттер болып табылады.

Жоғары оқу орнында толық білім алу мерзімі барысында студенттер қай уақытта да болмасын кітапхана көмегіне мұқтаждық танытады. Бүгінгі таңда

жоғары профессионалды білім беру мемлекеттік стандартынна сәйкес студенттердің өзіндік жұмысы жалпы сабақ мөлшерінің 50%-зын құрайды.

Сондықтан, студенттердің өзіндік жұмысын қалыптастыру үшін кітапханадан, өзіндік білім алу жағдайын жасау және кітапхананың ақпараттық-білім беру ортасы арқылы студенттің тұлға ретінде өзіндік дамуын және сол ақпараттық ортаның тоқтамастан жетілуі талап етіледі.

Осының барлығын ескере отырып біз АЭЖБУ студенттерінің кітапқай діген құштарлығын ояту және студенттер өздерінің ақпараттық мәдениетін қалыптастыруда өзіндік жұмысын өсіру үшін, кітапқа қол жетімділік барынша оңайлатуға арналған ақпараттық жүйе бөлігін құрсытыру мысалы көрсетелік.

### 1.3 Қолданылған құрал саймандар

Осы дипломдық жобаны жасау барысында мен көптеген заманауи құрал саймандар көмегіне жүктендім. Солардың тізімін келтірсе:

- HTML;
- CSS;
- JavaScript;
- Java Технологиясы.

Осы тізімде берілген технологиялардың әр-қайсына жеке тоқталып, олардың ерекшеліктерін, артықшылықтарын және кемшіліктерін айта кетерлік.

#### 1.3.1 HTML тілінің негіздері

HTML бағдарламалау тілі болып табылмайды, бұл мәтіндік құжаттарды белгілеу тілі. Онда “кәдімгі” бағдарламалау тілдерінде сияқты айнымалыларды, процедураларды, класстарды және басқа да атрибуттарды баяндалуы синтаксиске нақты сәйкес келу маңызды емес. Егер де сіз TITLE және BODY сияқты маңызды элементтерді көрсетпесеңіз, онда браузер сәйкесті мәтіндерді үндеусіз түрде қолданады. Егер де сіз синтаксистік қате істесеңіз, оны браузер терезесінен көріп, оңай табуға болады.

HTML – ді меңгеру үшін екі нәрсе қажет:

- кез-келген браузер, яғни HTML – файлдарды көру үшін қолданылатын бағдарлама – Internet Explorer немесе Google Chrome;
- мәтіндік файлдардың кез-келген редакторы. Windows үшін Notepad толығымен жарайды;
- мәтіндік редакторды HTML – файлдарды дайындау үшін, ал браузерді істелгенді бақылау құралы ретінде пайдаланады.

*& - тізбектері*

“<” және “>” символдарды браузермен HTML – белгілемесі ретінде қабылдағандықтан, мынандай сұрақтар туады: ал бұл символдарды экранда қалай көрсетуге болады? HTML да бұл & тәзбегі көмегімен жасалынады (оларды тағыда символдық объектілер немесе эскейп – тізбегі деп атайды). Егер мәтінде &lt; (ағылшын сөзінің бірінші әріптерінен less than азырақ)



тізбегі кездессе, браузер экранда “<” символын көрсетеді. “>” белгісі &gt; (ағылшан сөзінің бірінші әріптерінен greater than - көбірек). “&” (амперсant) символы &amp; тізбегімен кодталады.(“) екі тырнақшалар &quot; тізбегімен кодталады. Есте сақтаңыз нүктелі үтір & - тізбегінің міндетті элементі. Сонымен қатар, тізбекті құрайтын барлық әріптер төменгі регистрде болуы керек. &QUOT; немесе &AMP; типті белгілерін қолдануға рұқсат берілмейді. Жалпы айтқанда, & - тізбектемелері ASCII – кестесінің жарты бөлігіндегі (онда әрине орыс әріптері кіреді) барлық символдары үшін анықталған. Кейбір символдар сегіз биттік мәлеметті беруді қамтамасыз етпейді, сондықтан символдарды ASCII – кодтармен 127 ден жоғары & - тізбектері түрінде беруге болады.

#### Қаріптерді форматтау

HTML мәтін бөлігін белгілеудің екі әдісін рұқсат етеді. Бір жағынан, мәтіннің кей бір бөлігіндегі қаріп қою немесе иілген болу керектігін тікелей көрсетуге болады, яғни мәтіннің физикалық стилін өзгерту. Бір жағынан, мәтіннің кейбір бөлігін қалыпты логикалық стильден өзгешелеп белгілеуге болады, осы стиль интерпретациясын браузерге тастау керек.

#### Физикалық стильдер

Физикалық стиль деп ағымдағы қаріп модификациясын браузерге тікелей көрсету деп түсінуге болады. Мысалы, <B> және </B> белгілемелері арасындағы, қою қаріппен жазылады.<I> және </I> арасындағылар иілген қаріппен жазылады. <TT> және </TT> белгілемелері ерекше, себебі бұлардың арасындағы мәтін жазба машинасымен жазылған қаріптерге ұқсайды.

#### Логикалық стильдер

Логикалық стиль қолданғанда құжат авторы алдын ала, оқушы экранда нені көретіндігін білмейді. Әртүрлі браузерлер бірдей логикалық стильдерді әртүрлі көрсетеді. Кейбір браузерлер кейбір белгілемелерді мүлдем елемей, логикалық стильмен белгіленгеннің орнына қалыпты мәтінді көрсетеді.

Қарапайым HTML құжатының құрамы: Ең біріншіден түсіндіретінім HTML құжат тэгтерден тұрады, одан барлық интернет беттері құралады. Тэг – ол HTML кодтың бірлігі. Мысалы, <HTML>, <HEAD>, <TITLE> және тағы басқа осының бәрі тэг болып келеді. Тэгтер ашылатын және жабылатын болады. Ашылатын тэгтар жоғарыда көрсетілген. Оларға жабылатын мысалы: </HTML>, </HEAD>, </TITLE> тэгтері сәйкес келеді. HTML құжатының ең қарапайым құрылымы келесідей: <HTML> <HEAD> <TITLE> Құжаттың тақырыбы </TITLE> </HEAD> <BODY> Беттегі бірінші мәтін </BODY> </HTML> .

Кез келген HTML құжат <HTML> тәгінен басталып және </HTML> тәгімен бітуі керек. Ол MP3 және GIF емес, HTML құжат екенін интернет құраушының түсінуіне мүмкіндік береді.

#### HTML-құжатының құрылымы

HTML-дің конструкциясын тэгтер деп атайды. Браузерді кәдімгі мәтіннен ажырату үшін олар бұрыштық тырнақшаларға алынады.Тэг ол кез-келген көрсетілімнің құрылымының бастапқы қозғалысын көрсетеді. Егер де

бұл құрылым барлық құжатта пайдаланылса, онда бұл тэг жабылатын сыңарын қолданбайды. Бірақ көптеген тэгтердің сыңарлары болады. Мысалы кез келген Web - беттері `<html>`, тәгінен басталып, оның `</html>` жабылатын сыңарымен аяқталады. Жабылатын тэг ашылатын тэгтен бұрыштық тырнақшадан кейін қисық сызықпен ерекшеленетініне назар аударыңыз. Барлық сөздік параметрлердің мағынасы тырнақшаға алынады.

Түстер қалай берілетінін келесі тақырыпшадан білеміз, ал қазір `<body>` тәгінің параметрлеріне қайта ораламыз.

`lang` параметрі Web - беттердің мәтіндік құрылымы қай тілде жазылғанын көрсетеді. RFC 1766 құжатында келтірілген екі әріпті кодтық тілдің мағынасы мән ретінде қолданылады. Ал, шын мәнінде, бізге бұл белгілеулердің барлығын білу қажет емес. Көн жағдайларда біз орыс немесе ағылшын тілін қолданамыз. Олардың кодтары: "ru" және "en" сәйкесінше.

`<body>` тәгінің жоғарыда келтірген параметрлері екі параметрлерді `id` және `class` иелене алады, бірақ іс жүзінде олар бұл тэг үшін почталар ешқашан қолданылмайды. Көріп отырғанымыздай, барлығы қиын емес, қарапайым. Енді метамәліметтердің не екенін білетін уақыт жетті. Метамәліметтерді құжаттағы көрінбейтін ақпарат түрінде анықтауға болады. Олар құжаттың идентификациясы ретінде және Web - парақтардың бейнелену режимінің бағыты ретінде қолданылады. Web - параққа метамәліметтерді енгізу үшін `<meta>` тәгі қолданылады. Ол көп жағдайда келесідей түрде болады: `<meta name="айнымалы аты" content="айнымалы мәні">`.

Осындай тәртіппен, егер біз кез келген Web - парақтың авторын көрсеткіміз келсе, блокқа оның тақырыбын қоятын келесідей конструкцияны пайдалансақ жеткілікті: `<meta name="Author" content="It's me!!!">`.

Кілттік сөздерді көрсеткенде біз `<meta>` тәгіне `lang` қосымша параметрін қостық соған аса назар аударуымыз керек. Бұл параметр сол немесе басқа мәтінде жазылған тілдің көрсетілімі үшін арналған екені жайлы айтқанбыз. Біздің мысалымызда кілттік сөздеріміз орыс тілінде жазылғанын көрсеттік, алайда біз кілттік сөздерді әр түрлі тілде `<meta>` тәгінің бірнешеуін пайдалана отырып жаза аламыз. Сондайақ мәліметтер `http` тақырыбын беруге көмектеседі. Бұл жерде кішкене техникалық шегерім жасауымыз керек. Барлық HTML-құжаттар мамандандырылған программалар көмегімен беріледі. Бұл қабылдау ережесінің жиынын және ақпаратты жіберу компьютерлік өндірісті протокол д.а. Web - беттерде жіберу және пайдаланушы деректерді өшіргеннің жиынтығы HTTP (HyperText Transfer Protocol), протоколы деп аталады. Бұл протокол директивалардың жиынтығымен және айнымалымен HTTP - тақырыбы деп атайды.

Сондай-ақ HTML - құжаттың құрылымының сұрақтарына әртүрлі мәтіннің тақырыбын қолдану жатады. HTML-дегі тақырыпқа өзіндік тэгтер берілген. HTML құжатта барлық мәтіндік тақырыптың алты деңгейі қолданылады. Ең үлкен деңгей - бірінші және әр тақырыптың өзіндік тәгі және өзінің көрсетілімінің ережелері болады. Тақырыптарды белгілеудің тәгі өте қарапайым. Бірінші деңгейдің тақырыбына `<h1>` тәгі пайдаланылады, ал

жабылу сыңарына `</h1>`, екінші деңгейдің тақырыбына `<h2>` - `</h2>` сыңарлары пайдаланылады және солай алтыншы деңгейге дейін қайталана береді.

Қарапайым HTML кестелерін құру

Кестенің сипатталуы `<BODY>` құжатының ішінде орналасуы керек. Құжатта кестелер саны әр түрлі бола алады, сонымен қатар бір біріне енгізілген кестелер де болуы мүмкін. Әр кесте `<TABLE>` тәгімен басталып, `</TABLE>` тегімен аяқталуы керек. Бұл жұп тәгтердің ішінде кестенің құрамы сипатталады. Кез келген кесте бір немесе бірнеше жолдардан тұрады, олардың әрқайсысында жеке ұяшықтарға мәліметтер беріледі.

Әр жол `<TR>` (Table Row) тәгімен басталады және `</TR>` тәгімен аяқталады. Жолдарда орналасқан жеке ұяшық `<TD>` және `</TD>` (Table Data) жұп тәгімен немесе `<TH>` және `</TH>` (Table Header) тәгімен қоршалады. `<TH>` тәгі әдетте кестенің тақырып-ұяшықтары үшін, ал `<TD>` мәлімет ұяшықтары үшін қолданылады. Бұларды қолданудағы айырмашылық ұяшықтардың, құрамын көрсетуге арналған қаріптің типінде және ұяшықтардағы мәліметтердің орналасуында ғана. `<TH>` типті ұяшықтардың құрамы жартылай қалың (Bold) қаріппен бейнеленеді және ортада (`ALIGN=CENTER`, `VALIGN=MIDDLE`) орналасады. `<TD>` тәгімен анықталған ұяшықтар үндеместен ортада тік бағытта және сол жаққа (`ALIGN=LEFT`) қарай түзетілген мәліметтерді көрсетеді.

`<TD>` және `<TH>` тәгтері `<TR>` кестесінің жолының сипаттамасында ғана орналасады. `</TR>`, `</TD>` және `</TH>` аяқтайтын кодтарды жазбай-ақ қойса болады. Бұл жағдайда жолдың немесе ұяшықтың сипатталуының соңы келесі жолдың немесе ұяшықтың басы немесе кестенің соңы болып табылады. Кестенің аяқталуының тәгі `</TABLE>` міндетті түрде жазылуы керек.

Кестедегі жолдар саны ашылған `<TR>` тәгінің санымен анықталады, ал бағандардың саны барлық жолдар арасындағы `<TD>` немесе `<TH>` максималды мәнімен анықталады. Ұяшықтардың кейбіреулерінде ешқандай ақпарат болмауы мүмкін, бұндай ұяшықтар бірінен соң бірі орналасқан - `<TD>`, `</TD>` жұп тәгтерімен анықталады. Егер кез келген жолдың соңында орналасқан бір немесе бірнеше ұяшық ешқандай мәлімет сақтамаса, онда олардың сипатталуын жазбаса да болады, браузер керекті бос ұяшықтар санын автоматты түрде өзі қосады.

Кестенің `<CAPTION>` және `</CAPTION>` жұп тәгтерімен қоршалған тақырыбы болуы мүмкін. Кестенің тақырыбының сипатталуы кез келген орында `<TABLE>` және `</TABLE>` тәгтерінің ішінде орналасуы керек. HTML тілінің ерекшелігіне байланысты тақырыптың сипатталуының орналасуы қатаң бекітілген: ол `<TABLE>` тәгінен кейін және бірінші `<TR>` тәгіне дейін орналасуы керек.

Үндеместен кестенің тақырыбының мәтіні кестенің үстінде (`ALIGN=TOP`) және көлденең бағытта ортада орналасады.

Кестенің тақырыбының `<CAPTION>` тәгі бір `ALIGN` параметрін ғана қабылдайды, ол параметр `TOP` (кесте үстіндегі тақырып) немесе `BOTTOM`

(кесте астындағы тақырып) мәндерін қабылдайды. ALIGN параметрін жазбаса да болады, ол ALIGN=TOP мәніне сәйкес. Көлденең бағытта кесте тақырыбы әрқашанда ортада орналасу керек. Кестенің тақырыбы да болмауы мүмкін. Көп жағдайларда кестенің тақырыбы ретінде қарапайым мәтін қолданылады, бірақ нақтысында <CAPTION> және </CAPTION> тэгтерінің арасында <BODY> бөлімінде қолданылатын кез келген HTML-элементтерді жазуға болады. Кестенің тақырыбының жазылуының мысалын келтірейік: <CAPTION ALIGN=BOTTOM> Кестенің астында орналасатын тақырып </CAPTION>.

Microsoft Internet Explorer браузері тақырыпты орналастырудың қосымша мүмкіндіктерін ұсынады. ALIGN параметрі жоғарыда көрсетілген мәндермен қатар көлденең түзету үшін LEFT, CENTER және RIGHT мәндерін рұқсат етеді. ALIGN=RIGHT жазбасы тақырыптың оң жаққа қарай ығысуын және кестенің үстінде орналасуын қамтамасыз етеді. Егер ALIGN=BOTTOM жазбасын қолдансақ жоғарыда келтірілген мысалдағыдай тақырып кестенің астында орналасады. Бірақ бір тақырыпта ALIGN параметрін екі рет қолдануға болмайды. Сол себептен тік түзету үшін қосымша арнайы параметр енгізілген VALIGN, ол TOP немесе BOTTOM мәндерін қабылдайды. Мысалы, сол жаққа қарай түзетілген және кестенің астында орналасқан тақырыптың сипатталуы келесі түрде болады: <CAPTION ALIGN=LEFT VALIGN=BOTTOM> Сол жаққа қарай түзетілген және кестенің астында орналасқан тақырып </CAPTION>.

<TABLE> тәгінің параметрі

Кестелерді құруда қолданылатын негізгі тэг <TABLE> болып есептеледі. Ол бірнеше параметрлермен қолданыла алады, олардың әр қайсысы жазылмауы мүмкін. Рұқсат етілген параметрлердің жиынтығы браузерге байланысты болады. HTML ерекшелігіне сәйкес <TABLE> тәгінде келесі параметрлер қолданылуы мүмкін: BORDER, CELLSPACING, CELLPADDING, WIDTH, ALIGN. NetScape және Microsoft Internet Explorer браузерлері жоғарыда айтылған бес параметрмен қатар HEIGHT және BGCOLOR параметрлерін қолдануға рұқсат етеді. Кейбір браузерлер басқа параметрлерді де қолдана алады. <TABLE> тәгінің жалпы қолданылатын параметрлерінің тағайындалуын қарастырайық.

CELLSPACING параметрі

Параметрдің жазылуының формасы мынандай: CELLSPACING = num, мұндағы num - параметрдің пикселдердегі сандық мәні, ол міндетті түрде жазылады. num-ның өлшемі көлденең және тігінен орналасқан көршілес ұяшықтардың арасындағы арақашықтықты анықтайды. Үндеместен оның мәні екі деп алынады. Байқағанамыздай, дәстүрлі баспалық жүйелерде кестенің көршілес ұяшықтарында ортақ шекара болады. HTML-кестелерде үндеместен орын қалдырылады. CELLSPACING=0 болған кезде көршілес ұяшықтардың рамкалары бірігіп, кестені бір торлы кесте ретінде көрсетеді.

CELLPADDING параметрі

Параметрдің жазылу формасы CELLSPACING-ге ұқсас. num-ның өлшемі ұяшық рамкасы мен ұяшықтың ішіндегі ақпараттың арасындағы бос аймақтың өлшемін анықтайды. Үндеместен оның мәні бірге тең деп алынады. CELLPADDING параметрін нөлге тең деп алу ұяшықтағы мәтіннің кейбір бөліктерінің рамкаға тиіп тұруына алып келуі мүмкін, бұл эстетика жағынан дұрыс емес. CELLPADDING және CELLSPACING параметрлерінің әрекеттері бір біріне өте ұқсас. Рамкасыз берілген кестелерде параметрлердің екеуінің біреуінің өзгеруі бір нәтижеге алып келеді.

CELLPADDING және CELLSPACING параметрлері бір бірінен тәуелсіз, егер олардың біреуі жазылмаса, онда оның мәш үндеместен алынады. Әдетте, бұл параметрлердің бәрі жазылмаса, онда көршілес ұяшықтардағы мәліметтер арасындағы минималды ара қашықтық алты пикселға (Netscape үшін) тең болады. Бұл мән CELLSPACING үшін екі пикселдан, CELLPADDING үшін бір пикселдан және әр ұяшықтың рамкасы үшін бір пикселдан құралады. Келесі сипатталуды жазу арқылы кішкентай кестені ала аламыз. <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>. Тек осындай нұсқада ғана ұяшықтар бір біріне жақын орналасады.

WIDTH және HEIGHT параметрлері

Кестелер бейнеленген кезде олардың ені және биіктігі автоматты түрде браузермен есептеледі және көптеген факторларға тәуелді болады: берілген кестенің бүкіл құжаттың сипатталуындағы параметрлердің мәні, қаріптің өлшемі, көру терезесінің өлшемі және тағы басқалары.

Кейде кестенің ені мен биіктігін көрсету керек болады. Бұл мақсатта <TABLE> тәгінің WIDTH (кестенің ені) және HEIGHT (кестенің биіктігі) параметрлері қолданылады. Жазылу формасы мынандай: WIDTH=num немесе WIDTH=num%, мұндағы num - бүкіл кесте енінің пикселдағы сандық мәні немесе бүкіл терезенің өлшемінің проценті. Сонымен қатар 100 пайыздан үлкен мәнді беруге болады, бірақ бұндай жағдайда елестету қиынға соғады. Мысал: <TABLE WIDTH=200>.

### 1.3.2 Cascading Style Sheets технологиясы

CSS (Cascading Style Sheets – Каскадтық стильді кестелер) – HTML, XHTML және т.б. белгілеу тілдерінде жазылған уеб беттердің сыртқы көрінісін суреттейтін, әрлейтін тіл болып есептеледі. Бұл тілде уеб беттердің сыртқы көрінісіндегі шрифттер, түстер, жайғасуы, блоктар сияқты элементтерді жасауға болады. CSS-тің пайда болуының негізгі себебі белгілеу тілдерінде (HTML, XHTML, XML және т.б.) уеб беттердің логикалық құрылымдарымен сыртқы көріністерін суреттеуді бір-бірімен бөліп қарастыру болды. Себебі әр қайсысы бөлек болса, сайт жасағанда да, сайттың кодын оқығанда да түсінуге ыңғайлы болады.

Тарихы

1990 жылдары Web стандарттау істері қарқынмен дамыды және уеб дизайнерлер үшін сайт жасауда ортақ бір стандарт керек болды. Сөйтіп HTML

4.01, XHTML және стандарт CSS пайда болды. “Каскадтық стильді кестелер” терминін 1994 жылы Хокон Виум Ли енгізді. Хокон Берт Боспен CSS-ті дамытып, оған Ғаламтор Консорциумы “Web стандарты” деген атау берді.

Артықшылықтары:

- Анық бақылау кодтың көлемін едәуір кемейтеді және оны оқуға түсінікті етеді;

- CSS тілі көмегімен HTML тілінде беруге болмайтын параметрлерді беруге болады. Мысалға алатын болсақ сілтемелердің астындағы сызықты алып тастауға болады;

- CSS арқылы уеб беттің сыртқы көрінісін оңай өзгертуге болады. Көп құжаттардың сыртқы көрінісін бір кесте арқылы көрсету. Мысалға, сіз 30 бет кодтағы шрифттарды жасыл түс қылдыңыз. Бірақ уақыт өткен соң, көк не қызыл түске өзгерткіңіз келіп, барлық 20 бетке кіріп керекті атрибуттағы шрифтты өзгертіп шығасыз. Ал CSS тілі арқылы сол 20 беттің барлығын бір ғана стильдердің кестесінде өзгерте салуға болады;

- Құрамалы және жинақталған дизайн техникасы. CSS тілінде сайт версткасы деген ұғым бар.

### 1.3.3 JavaScript тілінің негіздері

JavaScript – бұл объектіге және тілдік құралдарға негізделген, және орта мүмкіндіктері объектімен ұсынылатын, бағдарламалау тілі. JavaScript сценарилер (бағдарлама) - бұл өзара-әрекеттенуші объектілер жинағы. JavaScript объектісі – бұл реттелмеген қасиеттер жинағы, оның әрбіреуінің нөл немесе одан көп атрибуттары бар, олар бұл қасиет қалай қолданылуы мүмкіндігін анықтайды. Мысалы, егер ReadOnly қасиетінің атрибутына true (шын) мәні меншіктелсе, онда осы қасиеттерді бағдарламалау арқылы өзгерту мүмкіндіктері нәтижесіз болады. Қасиет - бұл басқа объектілерден, көп қолданылатын мәндер мен әдістерден құралатын контейнерлер. Көп қолданылатын мәндер - бұл кез келген енгізілген Undefined, Null, Boolean, Number және String типтерінің элементі; объект - бұл енгізілген Object типінің тағы бір элементі; әдіс-қасиет арқылы объектімен ассоциативті функция.

JavaScript, Global, Object, Error, Function, Array, String, Boolean, Number, Math, Date, RegExp сыйақты бірнеше енгізілген объектілерден құралады. Бұдан басқа, JavaScript қатан түрде айтқанда, міндетті түрде функция мен әдіс болып табылмайтын енгізілген операциялар жинағынан, сонымен қатар, бағдарламаның орындалу логикасын басқаратын енгізілген операторлардың жинағынан құралады.

JavaScript синтаксисі негізінен Java тілінің синтаксисіне сәйкес келеді, бірақ сценарилер тілін үйренуге жеңілдету үшін онымен салыстырғанда жеңілдетілген. Мысалы, айнымалылар декларациясы оның типінен құралмайды, қасиетте типсіз болады, ал функция декларациясы бағдарлама текстінде оны шақырғаннан кейін тұруына болады.

Объектілер жайында жалпы мағлұматтар

JavaScript тілі Java және C# тілдерімен салыстырғанда қатаң түрде объектілер классынан тұрмайды. Оның орнына олар үшін объектілерді құруды жады бөлу жолымен жадыны конструкторларды қолдайды, ол оның орнына ол объектілерді олар үшін жады бөлу және олардың барлық немесе кейбір қасиеттерін инициализациялау жолымен құратын конструкторларды қолдайды. Барлық конструкторлар объектілер болып табылады, бірақ бар объектілер конструктор болмайды. Әрбір конструктор, прототипті және бөлінетін қасиеттерге негізделген ізін басуды іске асыру үшін қолданылатын `prototype` қасиетін иеленеді. Объектілер `new` операциясында конструкторларды шақыру жолымен құрылады; мысалы `new String` (жаңа жол) `String` жаңа объектісін құрады. `New` - сыз конструкторды шақыру нәтижесі конструкторға тәуелді. Сонымен, `String("жаңа жол")` , объект емес көп қолданылатын жолды құрады.

JavaScript прототипке негізделген, із басуды қолдайды. Әрбір конструктормен сәйкес прототип байланысқан, және конструктормен құрылған әрбір объект осы прототипке сәлтемені құрайды (объектінің прототипі деп аталынатын). Прототип өз кезегінде, өз прототипіне және ары қарай сәлтеме құруы мүмкін. Осылайша прототиптер тізбектері құрылады. Объектінің қасиетіне сілтеме - бұл берілген атпен қасиетті құрайтын объектілер прототипінің тізбегінің бірінші прототипіне сілтеме. Басқа сөзбен айтқанда, егер берілген объектінің берілген атпен қасиеті болса, онда осы қасиетке сілтеме қолданылады, егер жоқ болса, онда осы объектінің қасиеті зерттеледі және т.б..

Объектілер классына негізделген, объектке бағытталған тілдерде, ағымдағы қалып класстар экзеплярларымен іске асырылады. Әдістер класстармен, ал із басушылық құраммен және мінез құлқымен . JavaScript та ағымдағы қалып және әдістер объектімен іске асырылады, ал құрамы мен мінез құлқы із басады. Барлық объектілер, олардың прототипінен құралатын, қасиеттерден құралады, осы қасиеттерді және оның тағайындалуын бөледі. Класстарға негізделген тілдерден айырмашылығы, қасиеттер объектіге оларға мәндерді меншіктеу жолымен динамикалық түрде қосылуы мүмкін. Жеке жағдайда, конструкторлар мәндерді барлығына немесе құрылатын объектінің кейбір қасиеттеріне меншіктеуге міндетті емес.

#### 1.3.4 Java технологиялары

##### Java тілі

Sun Microsystems компаниясының жасап шығарған объектіге-бағытталған бағдарламалау тілі. Java қосымшалары әдетте арнайы байт-кодта компиляцияланады, сол себепті олар кез келген виртуалды Java-машинасында (JVM) компьютерлік архитектурасына тәуелсіз орындалады. Ресми шығу күні - 23 мамыр 1995 жыл.

Айтылуы, атаудың қолдануы:

Айтылуы жөнінде қазақ тілінде басқа тілдердегі сияқты екі бір-бірінен ажыратылған қағида пайда болды: ағылшын тілінен алынған /'dʒɑ:və/ («джава») және Ява аралының айтылуына сәйкес келетін дәстүрлі-халықтық «ява». Sun компаниясы ағылшындық айтылуды бүкіл әлемде ұстанады. Java - деп тек қана тілді ғана емес, сонымен қатар осы тіл негізінде жасайтын және орындайтын платформаны атайды. Бастапқыда тіл Oak («емен») болып аталған және оны Джеймс Гослинг тұрмыстық электрондық құрылғыларды бағдарламалау үшін жасалынатын. Біраз уақыт өткеннен кейін тілдіі атын JAVA деп ауыстырады және оны клиенттік қосымшаларды және серверлік бағдарламалық қамтаманы жасау үшін қолданыла бастады. Кейбір бағдарламалаушылардың сүйікті кофе маркасы Java құрметіне аталды. Сол себепті тілдің эмлемасында түтіндеп тұрған кофесі бар шынаяқ бейнеленген. Сонымен қатар осы java атауының шығу тегі туралы басқа да нұсқалар бар. Тілдің негізгі ерекшеліктері:

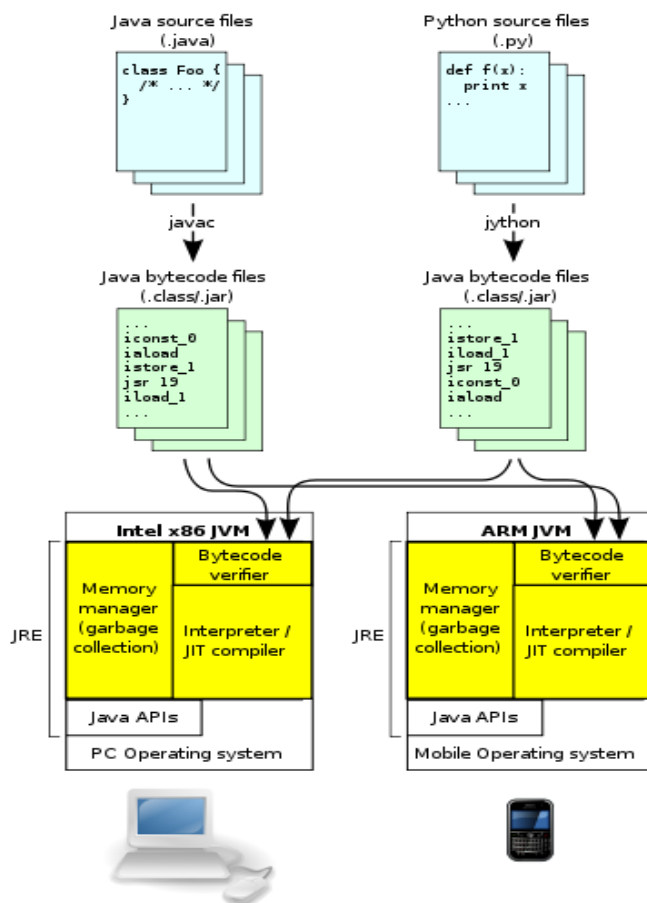
Java - дағы бағдарламалар, байттық кодты өндейтін және жабдықтамаға нұсқамаларды беретін интерпретатор болып табылатын (JVM) Java виртуалды машинасында орындалатын байт-кодқа трансляцияланады. Бағдарламалау ерекшеліктері:

Бағдарламалау - қандай да бір тілде код құру процесі, бұл код арнайы бағдарлама - транслятор арқылы файлға өзгертіледі. Транслятордың екі типі бар: компилятор және интерпретатор. Компилятор код тілінде жазылған мәтінді файлға аударады. Ал интерпретатор код тілінде жазылған мәтінді машиналық кодқа ауыстырады да оның орындалуын қамтамасыз етеді. Java - компиляциялайтын тіл болып табылады. Javaда бағдарламалаудың жақсы жақтарының негізгісін айта кетсек, кодың ауысу ерекшелігі:

Java тіліндегі КОД → (арнайы бағдарлама - транслятор) → БАЙТТЫҚ КОД → (JVM – Java Virtual Mashine) → МАШИНАЛЫҚ КОД , сондықтан бұл тілде жазылған бағдарлама кез келген жүйелі кез келген компьютерда жұмыс істей алады, тек ол компьютерда JVM орнатылған болу керек. Осыдан Javaда жазылған бағдарламалардың жақсы қасиеті - әр түрлі компьютерларда жұмыс істей алу қабілеттілігі туады.

Java virtual machine (Джава виртуалды машинасы, JVM) - Джава байткодта (Java bytecode) жұмыс істейтін виртуалды машина (virtual machine). Java бағдарламалық жасақтама платформасының кодын орындау компоненті. Sun Microsystems мәліметі бойынша әлемде 5.5 миллиард JVM-жұмыс істейді құралдар бар.[5]





1.1 Сурет - Java виртуалды машинасы (JVM) архитектурасы. Java Runtime Environment (JRE) Java API және JVMнен құралады

### Java (платформасы)

Java бірнеше компьютер бағдарламалық жасақтама өнімдерінен құралған өнім және Sun Microsystems мекемесі жазған спецификация (қазір Oracle Corporation мекемесімен бірге), барлығы қосыла келіп қосымшалар бағдарламалық жасақтарын (application software) дайындауға қолданысын табатын жүйені құрайды, бұл жүйе кросс-платформалы есептеу орталықтарында (cross-platform computing environment) қолданысын таба береді.

### Java платформаларының классификациясы

Java ішінде технологиялардың бірнеше негізгі топтары бар:

- Java SE - Java Standard Edition, Java негізгі басылымы, API, Java Runtime Environment компиляторлары бар; қолданушылық қосымшалар жасауға келтірілген, ең алдымен - үстелдік жүйелер;
- Java EE - Java Enterprise Edition, кәсіпорын деңгейіндегі бағдарламалық жабдықтамалар жасауға арналған арнайы құралдар жиыны болып табылады;
- Java ME - Java Micro Edition, есептеуіш қабілеттері шектелген құрылғыларда қолдануға арналған, мысалы ұялы телефондар, қалталық дербес компьютерлар, кіріктірілме жүйелер;

- JavaFX - Java эволюциясының Rich Client Platform ретіндегі келесі қадам технологиясы; бірлестіктің қосымшалары мен бизнестің графикалық интерфейстерін жасауға арналған;

- Java Card - смарт-карталар мен жад көлемі мен өңдеу қабілеті өте шектеулі басқа да құрылғыларда жұмыс істейтін қосымшалардың қауіпсіз ортасын құратын технология.

Негізгі мүмкіндіктері

- жадты автоматты түрде басқару;
- ерекше жағдайларды өңдеудің кеңейтілген мүмкіндіктері;
- енгізу/шығару фильтрация құралдарының бай жиыны;
- стандартты топтамалар жиыны: массив, тізім, стек және т.б.;
- желілік қосымшалар (оның ішінде RMI протоколын қолдану) жасау қарапайым құралдар болуы;

- HTTP-сұраныстар орындау мен жауаптарды өңдеуге мүмкіндік беретін кластар болуы;

- тілге кіріктірілген көпбағымды қосымшалар жасау құралдары;
- дерекқорға унификацияланған рұқсат;
- жекелеген SQL-сұраныстардың деңгейінде - JDBC, SQLJ негізінде;
- дерекқорды сақтау қабілеті бар объектілер концепциясы деңгейінде - Java Data Objects (ағл.) және Java Persistence API негізінде;
- жалпыламаларды қолдау(1.5 нұсқасынан бастап);
- бағдарламалардың параллельді орындалуы.

Java Platform, Enterprise Edition

Java Platform, Enterprise Edition, қысқаша Java EE (5.0-ші нұсқасына дейін - Java 2 Enterprise Edition немесе J2EE) - орта және ірі кәсіпорындық мәселелерді шешуге арналған, серверлік архитектураны суреттейтін, Java тілі үшін жазылған спецификациялармен тиісті құжаттар жиынтығы.

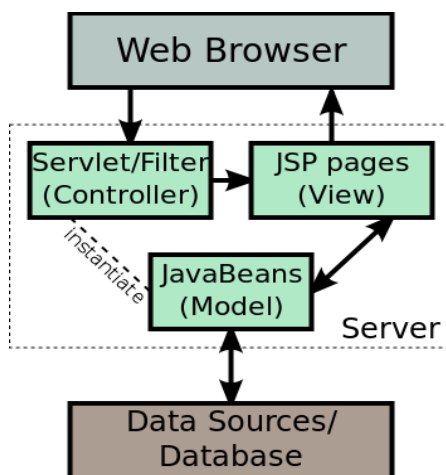
Спецификациялардың талданғаны соншалық, бір платформада жүзеге асырылған бағдарламаны еш қиындықсыз басқа платформаға тасымалдауды қамтамасыз ететді. Спецификацияның негізгі міндеті - бағдарламаның ауқымдалуын және жүйенің жұмыс істеу барысында деректердің тұтастығын қамтамасыз ету. JEE негізінен интернетте немесе локалды желіде, web арқылы жұмыс істеуге негізделген. Барлық спецификациялар Sun Microsystems Inc. компаниясының қолдауымен құрылған JCP (Java Community Process) бірлестегімен құрылып бекітіледі.

JEE өнеркәсіптік технология болып табылады және сондықтан көбінде сенімділікті, оралымдылықты, масштабталулықты қажет ететін жоғары өндірімелі жобаларда қолданылады.

JEE-нің танымал болуының бірден бір себебі ол SUN компаниясының кәсіпорындарға көп шығын жұмсамай, өзінің бағдарламалық қамтамасыз ету жүйелерін құруға арналған саймандар жиынтығын SDK-ны тегін, ашық таратуы. Бұл жиынтыққа бағдарлама құру лицензиясымен GlassFish қосымшалық сервері кіреді.[8]

## Java Servlet

Servlet сервлет - компьютер сервері мүмкіншіліктерін арттыруға арналған Java бағдарламалық тілінде жазылған класс. Java Servlet-тің Applet-тен айырмашылығы Java Servlet сервер құрылғысында жұмыс істейді, ал Applet клиент жақта жұмыс істейді. Әдетте веб-браузер рөлін атқарады, веб-браузер серверден сервлет арқылы сұрау жібереді, жауабын сервлет HTML-беті ретінде қайтарады.

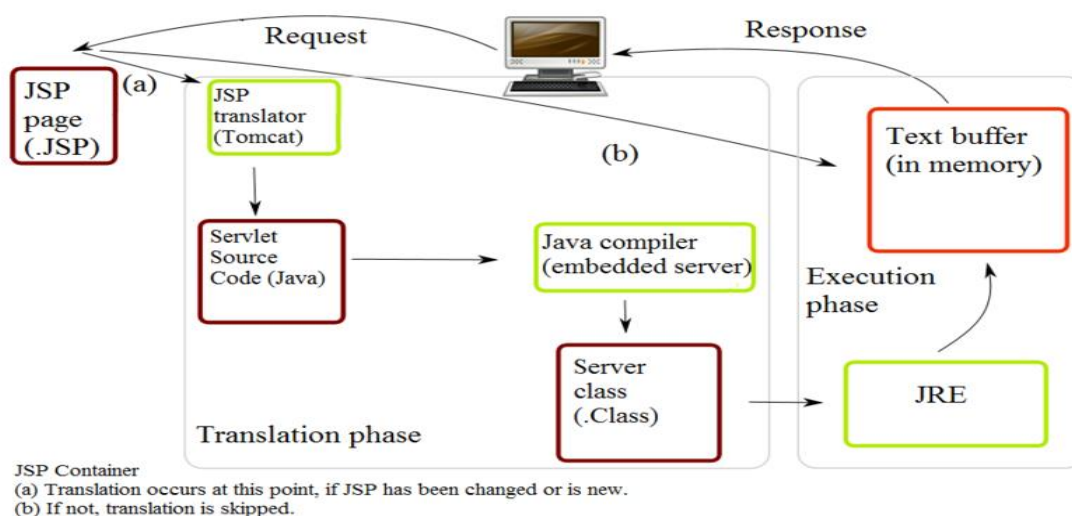


1.2 Сурет - Servlet және JSP негізіндегі MVC моделі

## JSP парақшасы

Веб-браузерге қайтарылатын HTML жауап дұрыс форматта сақталуы үшін сервлетте Java EE-нұсқасынан бастап JSP беттері технологиясы қолданыла бастады.

JavaServer Pages (JSP) - HTML, XML, немесе өзге түрлі құжаттар 1999 жылы Sun Microsystems мекемесі өңдеген, JSP бағдарламасы PHP тіліне ұқсас жұмыс істейді, бірақ Java бағдарламалық тілін қолданады.



1.3 Сурет - JSP файлының бағдарламалық жасақтамадағы қызметі

## 2 Жобалық бөлім

Қазіргі таңда көптеген жоғарғы оқу орындарының кітапханаларында жұмысты автоматтандырыға арналған ақпараттық жүйе қолдану кең ауқым алды. Сол АЖ – лер көптеген жұмыс түрімен айналысады. Ол кітапты тіркеу, оқырманды тіркеу, кітапхана базасынан кітапты іздеу немесе сол кітаптарға қол жеткізу жұмыстары. Бол дипломдық жобадығы менің мақсатым осындай АЖ – бөлігін құрастыру болып табылады. Бұл бағдарлама ашық қолданысқа берілген және авторлық құқықпен шектелмеген кітаптарға студенттердің қол жеткізуін оғайатуға арналған шешім болуы тиіс.

Кез-келген АЖ-ні жасау үшін оны алдымен жобалайды. Қазіргі таңда жобалаудың көптеген түрі бар, алайда солардың ішінде ең жақсы дамыған және бүгінгі күнде көп қолданысқа ие ол объектіге-бағытталған талдау мен жобалаудың методологиясы. Мұндай методологиның құрайтын маңызды екі бөлшектен тұрады. Олар: 1) объектіге-бағытталған декомпозиция 2) логикалық (класстар және объектілер) және физикалық (модульдер және процесстер) структурасын көрсетуге арналған көптеген әдістер. Осы бөлімде берілген жобаның ең маңызды архитектуралық аспектілері көрсетілген.

### 2.1 АЖ бөлігіне қойлатын талаптар

Жүйені құрастырудағы ең маңызды мақсатары – студенттің ақпараттық мәдениетін қалыптастыру жұмысында, студенттің өзі жауапкершілікті сезінетіндей әсер қалыптастыру. Бұл процессте кітапхана жұмысшыларының жұмысын минималдылау және студент өз бетімен жұмыс істейтіндей әсер қалдыру. Бұл мақсаттарға жету үшін жүйе қолдануға ынғайлы, интуитивті және интеллектуалды түсінікті болуы қажет. Осыдан ең маңызды екі мақсат түсінікті.

Жүйе қолдануға ынғайлы болуы тиіс.

Жай оқырманға мұндай талап өте қарапайым болып көрінуі мүмкін. Алайда осындай қарапайым мақсат болмауы жүйенің құлауына алып келуі мүмкін. Осындай абстракты мақсатты алдымызға қоя отырып біз осыдан түсінікті және нақты көптеген талаптар құраймыз. Бұл автоматтандырылған жүйе студентке кітапты оңай және тез табуға мүмкіндік беруі тиіс. Және де жүйені қолдану студенттің уақытын үнемдейді және кітапхана жұмыскерлерінің жұмысын азайтады.

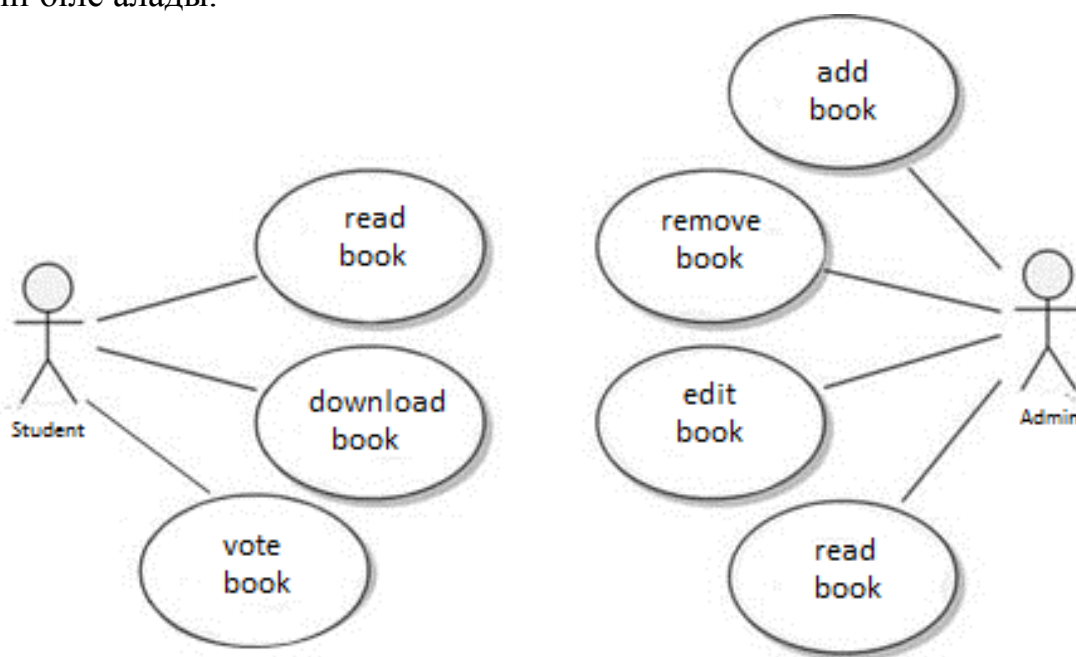
Жүйе келесідей ерекшеліктерге ие болуы тиіс:

- Студенттің кітапқа деген сұранысын арттыру;
- Жай оқырманмен администратордың мүмкіндіктерін бөлу;
- Ең соңғы ізделген кітапты көрсету;
- Жай оқырман администратормен кері байланыс жасауы мүмкіндік беру;
- Бар апараттық және бағдарламалық қамтаманы қолдану;
- Аутендификация процессін жасауға мүмкіндік беру;

- Қолданушының жеке мәліметтерін сақтау.

### 2.1.1 Жүйені қолдану моделі

Жүйені қолданудың мейлінше абстракциялық моделі екі қатынасушы кісіден және алты қолдану сценариінен тұрады (2.1 Сурет). Қолдану сценариі бұл - тек функционалдық мүмкіндікті сипаттау ғана емес, ол қолданушыға бағалы әрекеттің сипатталуы болып табылады. Мысалы белгілі бір кітапқа дауыс беру жай қолданушыға қарапайым мүмкіндік қана, алайда кітапхана қызметкерлері үшін оның маңызы өте зор. Бұл арқылы олар оқырмандарға қай кітаптар көбірек ұнайды және қандай типаждағы кітаптарды салуы керек екендігін біле алады.



2.1 Сурет - Жүйені қолданудың абстракциялық моделі

Бір қызықтысы Web-жүйелерде қолдану сценарилерін тек сұраныс және жауап арқылы құруға болады. Басқаша айтқанда мұндай жүйелерде қолдану сценарилерін әрекет етуші кісілердің әрекетінің тізімі және сол әрекеттерге жүйенің бұлжытпай жауап беруі деп айтуға болады. Және де осы жүйелер қалыптастырылған Web-парақшаларымен немесе сайттармен тығыз байланысты. Көбіне басқа типті бағдарламаларда қолдану сценарилері мәліметке және қолданушы әрекетіне негізделген, ал web-жүйелерде қолданушы интерфейстерінің (парақшалар немесе сайттар) дискреттік бірлігін анықтауға негізделген.

Берілген жүйе келесі әрекет етуші кісілерден тұрады:

- (Student) Студент немесе оқырман;
- (Admin) Жүйе администраторы немесе кітапхана қызметкері.

Берілген жүйенің қолдану сценарилері:

- (read book) Таңдап алынған кітапты онлайн оқу;

- (download book) Берілген кітапты локалды машинаға жүктеп алу;
- (vote book) Ұнаған кітапқа АЖ қолданушысының дауыс беруі;
- (add book) Жүйеге жаңа кітап қосу;
- (remove book) Жүйеден берілген кітапты өшіріп тастау;
- (edit book) Берілген кітаптың мазмұндамасын өзгерту.

## 2.2 АЖ бөлігінің құрылуы

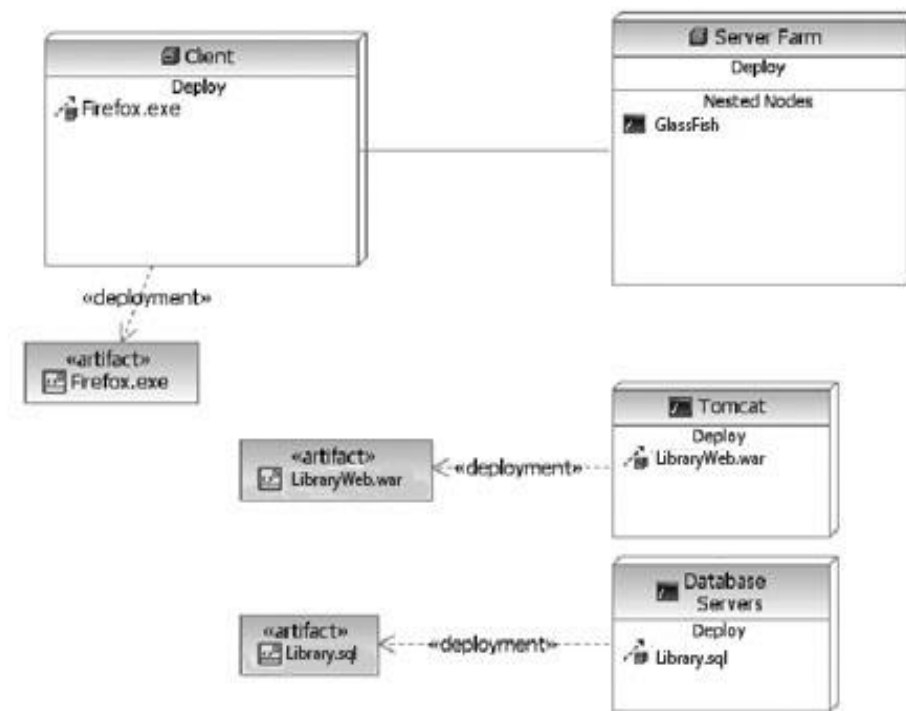
Кейде алғашқы сатыларда қай жерден талаптар жиынтығын жинау бітіп анализ басталатынын байқау қиынға соғады. Сондықтан да құрастырудың интеративті процедурасын қолдану өте танымал, ал құрастыруда “сарқырама” моделін қолдану көптеген күдік туғызады. Сонда да алдымен маңызды деген, кейіннен архитектуралық ерекшеліктерге әсер ететін жүйенің қолдану сценарилерін құрсатыру керек. Әрине сценарилерді детальді қарау бұл этапта қажет емес, бірақ маңызды архитектуралық аспектілерді анықтап алу керек.

### 2.2.1 АЖ – нің орналастырылу көрінісі

Web-жүйе клиент/серверлік жүйелердің бір түрі болып табылады. Сондықтан ол кемінде екі узелді иемденеді – сервер және клиенттің браузері. Сервер – бұл желіде анықталған адресі бар және тыңдаушы ретінде белгілі бір порт жіберетін, http протоколы үшін көбінесе 80 немесе 8080, машинамен бағдарламалық қамтаманың қосындысы. Клиент сервердің ресурстарына HTTP протоколы бойынша бекітілген сұраныс жасайды. Сервер көбінесе өзінде web-бағдарламаларды, web-қызметтерді және деректер қоры қызбеттерін біріктіреді. Біздің жобамыздағы осындай узелдар 2.2 суретінде көрсетілген.

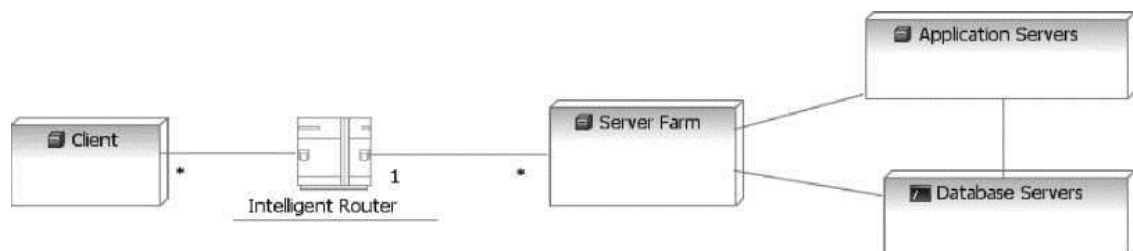
Web-жүйенің компоненттерінің орналасу диаграммасында (2.2 Сурет) АЖ – нің орындалу ортасы ретінде GlassFish қосымшалық сервері алынған. GlassFish узелында LibraryWeb.war архивті файл артивакты орналасқан. Database Server бұл MySQL ДҚБЖ орындалу ортасы және онда library.sql атты файл орналасқан.

Client узелында FireFox.exe артевакты орналасқан, ол орындамалы бағдарлама HTML браузер. Client және Server узелдары өзара каналды желі арқылы байланысқан, ол телефонды сызықтан бастап кең жолақты арна немесе радиоканал, кез келген желі болуы мүмкін. Және де осы канал арқылы клиент бірге қосылған web-сервер және деректер қорымен қарым-қатынас жасайды.



2.2 Сурет - Web-жүйенің компоненттерінің орналасу диаграммасы

2.3 Суретінде интеллектуалды маршрутизатор және серверлар тобын қалай қолдануға болатындығын және де қосымшалық сервердан келетін барлық бағдарламалар тобының жиынтығын қалай орындауға болатындығын көрсететін абстракты орналастыру диаграммасы көрсетілген. Мұны деталды түрде көрсету мүмкін емес алайда жалпы түрде осылай көрініс табу қажет.

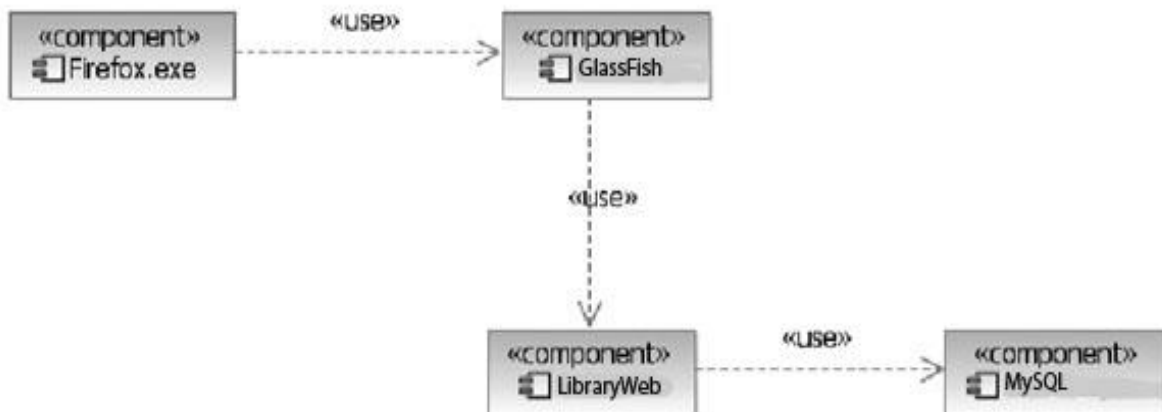


2.3 Сурет - Паралельді және қосымша өңдеу узелдары бар сервердің орналасу диаграммасы

### 2.2.2 Логикалық көрінісі

Негізінен web-жүйе кемінде 4 компоненттан тұрады: браузер, клиентпен қосылатын, web-контейнер немесе жай контейнер, бағдарламаның логикасына арналған және деректер қоры серверінің компоненті. 2.4 Суретінде Firefox.exe компоненті көрсетілген, ол атақты көплатформалы браузер, және де GlassFish – Java Server Pages (JSP) спецификациясының негізінде құрылған атақты web-контейнер. LibraryWeb компоненті бизнес-бағдарлама болып табылады, ал MySQL компоненті қарапайым мобильді деректер қоры сервері. Бұл диаграммадағы ең маңызды логикалық ерекшелігі, клиенттің браузері

және де бизнес-бағдарлама тікелей ешқашан деректер қорымен қарым-қатынас жасмайтыны болып табылады. Сервердің ресурстарына қол жеткізуді web-контейнер басқарады. Сондақтан оның маңызы өте зор және ол өте берік болу қажет.

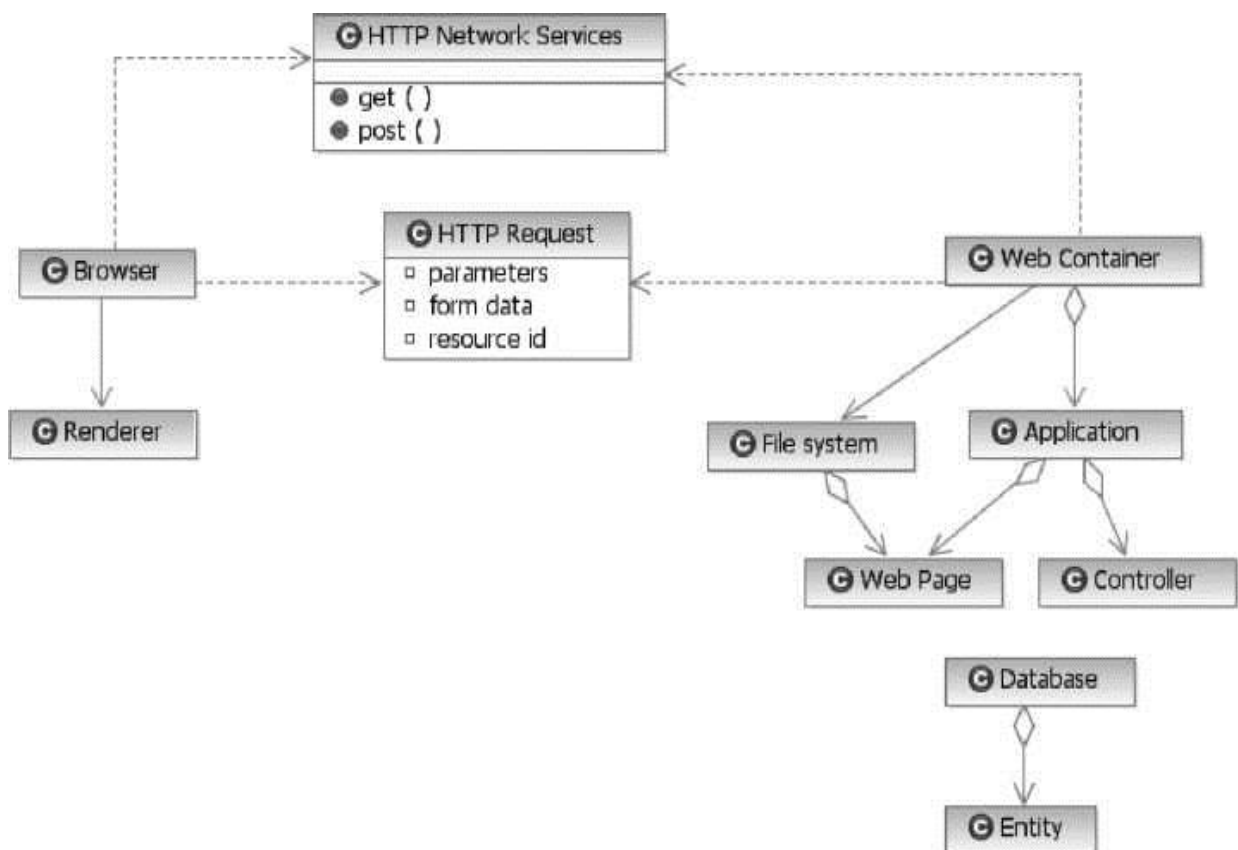


2.4 Сурет - Web-жүйенің негізгі компоненттерінің логикалық көрінісі

Web-қызметтер белгілі бір порттарды сұрайды, - қағида бойынша 80 немесе 8080, - осы порттармен келетін GET және POST сұраныстарды күтеді. Бұл сұраныстар HTTP протоколымен бекітілген командалар болып табылады. Олар web-сервердан мәліметті оқудың екі қарапайым тәсілін қамтамасыз етеді. POST әдісі көп мүмкіндіктерге ие, және сондықтан клиенттен серверге мәлімет жібергенде көбірек қолданылады. Қай әдісті таңдау керектігі алдын ала HTML файлында жазылады, және браузермен анықталады. Алайда сервер мәліметті кез-келген түрде қайтаруы мүмкін, және келген мәліметті клиент басқа орнатылған бағдарламаға жіберме әлде браузерда қалдырама өзі шешеді.

2.5 Серетінде сервермен клиент арасын қосатын элементтердің көрінісін беретін логикалық класстар көрсетілген. Клиент жағынан браузер web-парақтарды (дәлірек айтқанда, ресурстарды) қарапайым GET немесе POST сұранысы арқылы сұрауы тиіс. Ол қызметтер операциялық жүйелер арқы жасалуы мүмкін, әлде браузердің өзімен жасалады (бұл жағдайда браузер төменгі сатылы желі бағдарламаларының интерфейстерін шақырады). Web-контейнер әрқашанда браузермен оратылып жіберілетін HTTP-сұраныстарды, белгілі порттар арқылы сұрайды (мысалығы 8080). Web-контейнерде арнайы идентификаторлар болады, сол идентификаторлар арқылы белгілі web-парақты немесе web-бағдарламаны қосады. Сұраныстарды мәтіндік жолмен берілген, кілттік мәндер жұбымен (параметрлермен) қолдауға болады. Кейбір HTTP-сұраныстар күрделі түрде оратылуы мүмкін, ондай жағдайда қолданушы парақты ашардан алдын өзі анықтайды.





2.5 Сурет - Web-жүйені құрайтын объектілердің абстракты логикалық көрінісі

Web-контейнерде ресурсқа сұраныс келіп түскенде ол алдымен қандай файл немесе бағдарламаны шақыру керектігін анықтайды. Сосын контейнер сол анықталған ресурсты шақырады. Егер файл немесе бағдарлама динамикалық түрде анықталу керек болса, соны өңдейді. Өңдеу барысында, HTTP-сұраныста анықталған ақпарат (параметрлер және мәліметтер формасы) тексеріледі, сосын барып керекті бизнес-логика орындалады. Көбінесе ондай логика басқада деректер қорына рұқсаты бар бөлек қосымшалы серверде орналасыды.

Логикалық көрініс тұрғысында, web-жүйе web-парақтардан, контролерлерден және болмыстардан тұрады (аналетикалық моделдерде болатын, негізгі үш стереотип). Статикалық web-парақтар (өңдеуді қажет етпейтін) файлдық жүйеде сақтауға болады, ал бизнес-функцияны атқаратын web-парақтар, контейнер контексіне жүктелуі және орындалуы қажет. Көбінесе контролерлер компоненттерге біріктіріледі, ал персистенті болмыстар деректер қорымен басқарылады.

### 2.2.3 Процесстер көрінісі

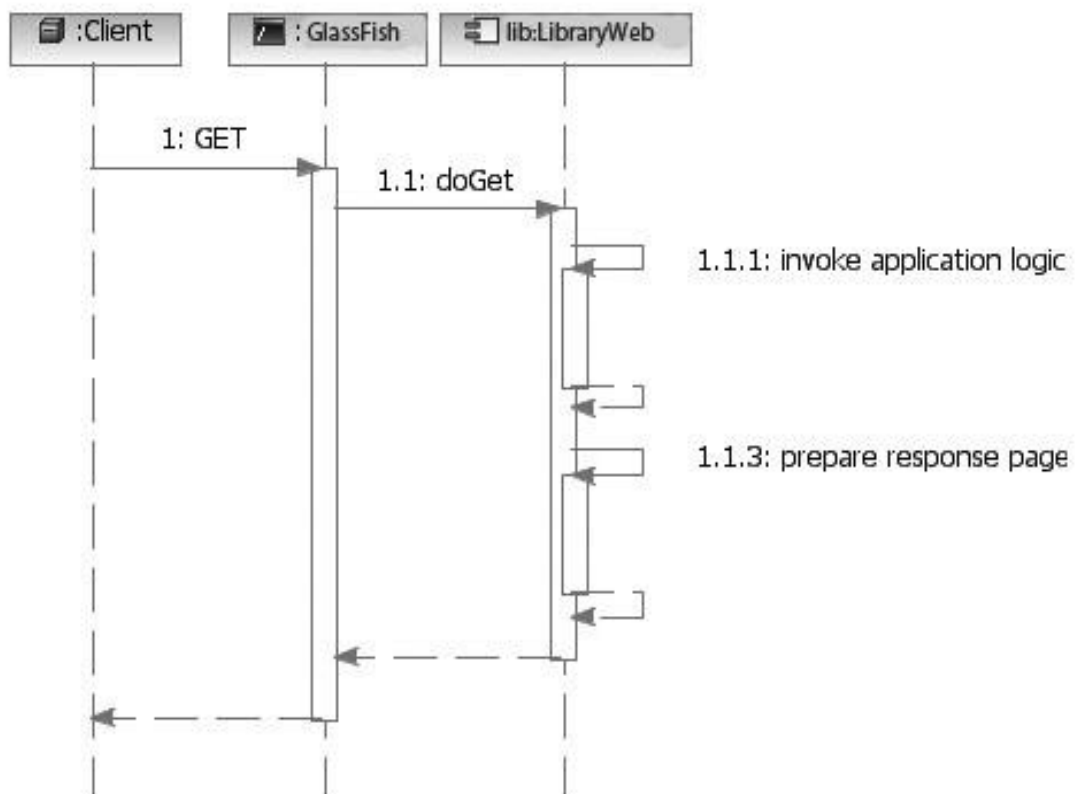
Қарапайым web-жүйелерде минмалды жағдайда екі процесс (негізінде оданда көп) болады. Клиент және сервер асинхронды фунциоланады, HTTP-протоколы бойынша сұраныстарды өңдеуді қоспағанда. Көбінде web-бағдарламаларға қосымша деректер қоры, және де аутентификация және

хаттарды өңдеу серверлері кіреді. Олар бір сервер узелында болуы мүмкін немесе бірнеше узелға бөлініп орналасуы мүмкін. Web-бағдарламаның процесстер схемасы кез-келген клиент серверлік бағдарламалардың архитектурасына тән икемділіктерге ие. Одан талап етілетні тек, клиент узелдарында сервер жақта инициализация жасайтын, белгілі бір программаны қосуды мұқтаж етеді.

Web-жүйенің архитектірасын түсіну үшін олардың автономды жұмыс жасайтынын ұмытпаған жөн. Басқаша айтқанда клиентпен сервер GET және POST сұраныстарына керекті уақыттан артық байланысқа түспейді. Сервер клиенттен белгілі бір ресурсқа (мысалы web-параққа) сұраныс алса, сол сұранысты (HTML файыл құрып жіберіп) қанағаттандырғаннан соң екеуінің арасындағы байланыс үзіледі. 2.6 суретінде клиенттің GlassFish серверіне қарапайым GET сұранысын жіберу процесс диграммасы көрсетілген. Сервер қандай web-бағдарламаны немесе web-парақшаны шақыру керектігін анықтайды. Дәл осы уақиға GET және POST сұраныстарын өңдеу кезінде кілттік қызмет атқарады. Процесс аяқталғанда бағдарлама жауап ретінде қайтарылған парақшаны алады.

Мұндағы қиындық, қалай серверлік бағдарлама әрқашан анық емес клиент сұраныстарының тарихын бақылайды және сол клиенттің ішкі жағдайын басқарады. Мысалыға, HTTP протоколының кейбір ерекшелдіктерін қолданбасақ немесе серверлік бағдарламаның белгілі бір архитектуралық механизмдерді жасамасақ, ең қарапайым деген карталық заказ немесе клиенттің көп парақты web-бағдарламасында қай беттерде жүргенін бақылау мүмкін бопайтын. Бақытқа орай қазіргі таңда клиенттің ішкі жағдайын басқаруға мүмкіндік беретін орталар немесе утилиттар көп.

Бұл архитектураның екінші кемшілігі, сервер клиенттің бір сценариді бітірмей сервер бағдарламасын қолдануды тоқтатыма әлде қолданудама, соны біле алмаймыз. Бұл алыстаға қолданушы желіден шығып, басталып қойған транзакцияны бітіре алматынды еске салады. Көбінесе қарапайым клиент серверлік бағдарламаларды, егер клиент желіден шықса, сервер одан алдын ала ескерту алуы қарастырылған, ал web-бағдарламаларда ондай ескерту қарастырылмаған.

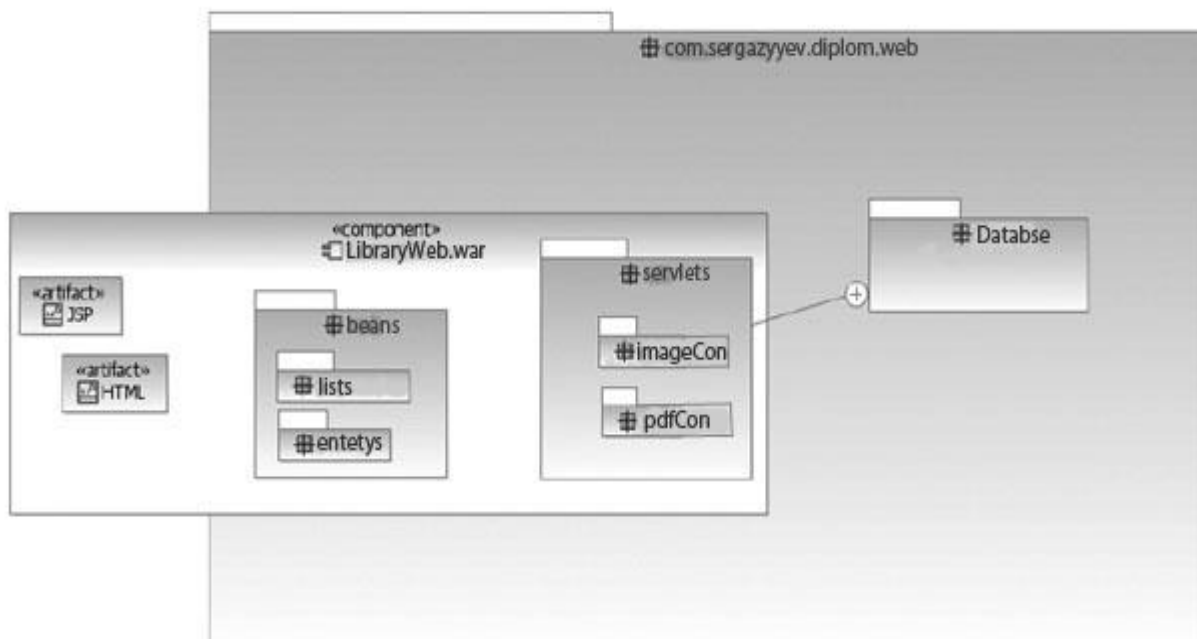


2.6 Сурет - GET сұранысын өңдеудегі реттілік диаграммасы

Сондықтан web-жүйе әрқашанда қай ресурстар ашылды және бір парақшадан екінші web-парақшаға өткенде қандай ресурстар берілгені туралы мәліметті білуі қажет. Мысалыға, web-жүй құрудағы ең басты ержелердің бірі бір транзакцияны бір бетте ашып сосын оны басқа бетте жабуға болмайды. Бір web-парақшадан екінші парақшаға өту уақыты көбінесе секундтармен өлшенеді және кез-келген уақытта байланыс үзіліп кетуі мүмкін. Көп транзакция және блокировка саны web-жүйенің жұмысын тоқтатуға алып келуі мүмкін.

#### 2.2.4 АЖ-ны жасау көрінісі

2.7 суретінде АЭЖБУ кітапханасының жұмысын автоматтандыруға арналған ақпараттық жүйе бөлігін жасау диаграммасының көрінісі берілген, жәнеде жүйенің пакеттерімен деңгейлері көрсетілген. Диаграммадан аңғаратынымыз негізгі байланыс web-компоненті LibraryWeb.war web-парақ артефактарына (HTML және JSP файылдар) ие, beans, db пакеттерінде Java класстары бар. Бұл класстар деректер қрымен, EJB модульмен және бизнес логикамен жұмыс жасауға негізделген. Барлық Java компоненттерінің пакеттеріне ортақ ат ретінде артқы плтанда көрініп тұрған com.sergazyuev.diplom.web аты алынды. Компоненттердің орналасуы деталды түрде орналасу көрінісінде (Deployment View) көрсетілген, ал олардың болмысы және компоненттердің қызметі – олардың логикалық көрінісі (Logical View) көрсетілген.



2.7 Сурет - Орындалу моделі

Орналасу көрінісі, логикалық көрінісі, және де процесстер көрінісі және жасалу көрінісі қодану сценарилерін “сұраныс-жауап” принципі бойынша орындауға біріктіріледі. Клиент ашық web-парақтарға HTTP протоколы бойынша сұраныс жасайды. Сервер сұранысты тексеріп, қандай ресурстарды ашу керек, қандай мәліметтерді өңдеу керектігін анықтайды. Кейбір ресурстар бизнес-логиканы орындайды, ал кейбіреулері жай статикалық мәліметтерді экранға шығарады. Барлығын өңдеп болған соң сервер жауап құрады, негізінен ол HTML тілінде жазылған web-парақша, онда керекті суреттерді, тексті мәліметтерді қойып қайтарады. Ол web-парақша жаңа көрініс табады және қолданушыны жаңа парақша шақыруға немесе функцияны орындауға итермелейді.

## 2.3 АЖ бөлігін құруға қолданылған бағдарламалық қамтамалар

### 2.3.1 MySQL деректер қорының ерекшеліктері

Деректер қоры - бұл құрамында белгілі бір ақпараты бар екі өлшемді, өзара байланысқан кестелер жиыны. Деректер қорын басқаратын (жаңа деректер қорын, кестелерді құрады және де құрылған объектілерді қолдануға мүмкіндік береді) программалық қамтама деректер қорын басқару жүйесі (ДҚБЖ) деп аталады.

Клиенттердің сұрауларын сипаттау үшін тұтас тіл құрылған - ол SQL (Structured Query Language - құрылымдық сұраулар тілі). SQL сұраулары арқылы сіз келесі әрекеттерді орындай аласыз:

- деректер қоры мен кестелерді құруға;
- кестелерге ақпаратты қосуға;
- ақпаратты өшіруге;

- ақпаратты модификациялауға;
- қажет ақпаратты алуға.

Әрине, `admin` қолданушысы бар болудан басқа сәйкес құқықтарға ие болу керек. MySQL әрбір сұрауы үтірлі нүктемен аяқталуы тиіс.

MySQL - реляциялық мәліметтер базасын басқарудың өте жылдам әрі сенімді жүйесі. Мәліметтер базасы мәліметтерді іздеуді, алуды, сорттауды және сақтауды қамтамасыз етеді. MySQL - сервері бірнеше қолданушыларға бір уақытта мәліметтермен жұмыс істеуді қамтамасыз етілуін басқарып отырады. Және де мәліметтермен жұмыс істеуге мүмкіндігі бар қолданушының кіруін қамтамасыз етеді. Ол мәліметтер базасына сұраныс құру үшін дүние жүзінде қолданылатын стандартты SQL (Structured Query Language) тілі пайдаланады.

MySQL - бұл ылғи да компьютерде жұмыс істеп тұратын сервер-программа. Клиенттік программалар (мысалыға, сценари) оған арнайы сұрауларды сокет (демек, желілік құралдар арқылы) механизмі арқылы жіберіп отырады, ол оларды өңдеп нәтижесін сақтайды. Осыдан кейін, қайтадан клиенттің арнайы сұрауы бойынша бүкіл нәтиже немесе оның бір бөлігі қайта жіберілед. Неге ылғи тұтас нәтиже жіберілмейді? Өте қарапайым: мәселе мәліметтер жиынының нәтижесі өте үлкен болып, оны желі бойынша тасымалдауға көп уақыт болуы мүмкін.

Сокеттерді қолдану механизмі клиент-серверлік технологиясына ойластырылған, бұл жүйеде программалардан арнайы сұрауларды қабылдап және өңдейтін арнайы программа MySQL - сервері қосылуы керек деген сөз.

Деректер қорымен байланыс құру

Бірақ деректер қорымен жұмыс бастамас бұрын онымен желілік байланыс құру, және де қолданушы авторизациясын орындау қажет. Ол үшін арнайы функция бар `mysql_connect()`.

```
int mysql_connect([string $hostname] [,string $username] [,string $password]).
```

`mysql_connect()` функциясы MySQL деректер қорымен желілік байланысты құрады, ол `$hostname` (үндемес бойынша `localhost`, яғни осы компьютер) атты хостта орналасқан, және ашық байланыстың идентификаторын қайтарады. Келесі жұмыстардың барлығы дәл осы идентификатормен жүргізіледі. Тіркелген қолданушы `$username` аты және `$password` құпиясөзі (үндемес бойынша берілген үрдісті қосқан қолданушы аты мен бос құпиясөз) көрсетіледі. Сонымен қатар `$hostname` жолында `хост_аты: порт` (егер MySQL стандартты емес басқа портқа қосылған жағдайда) форматты порт аты болуы мүмкін. MySQL-серверімен байланыс сценари жұмысының бітісімен не болмаса `mysql_close()` функцисы шақырылғанда автоматты түрде жабылады. Егер сіз сценаридің бүкіл жұмысы барысында тек бір ғана деректер қорымен байланыс құруды жоспарласаңыз, онда сіз қайтарылған мәндерді сақтамасаңыз да болады, және қалған функцияларды шақырғанда байланыс идентификаторын көрсетпесеңіз болады.

```
int mysql_select_db(string $dbname [,int $link_identifier])
```

MySQL серверіне алғашқы сұрауды жібермес бұрын қандай деректер қорымен жұмыс істейтінімізді көрсетуіміз керек. Осы үшін сипатталған функция қажет. Ол JDBC-ді \$link\_identifier байланысының келесі операцияларында \$dbname деректер қоры қолданылатынын ескертеді.

Қателерді өңдеу

Егер MySQL - мен жұмыс барысында қателер туса (мысалы, сұрауда жақшалар дұрыс қойылмаса немесе параметрлер жетіспесе), онда қате туралы хабарлама және оның номерін келесі екі функция арқылы алуға болады: `int mysql_errno([int $link_identifier])`.

Функция соңғы тіркелген қатенің номерін қайтарады. \$link\_identifier байланыс идентификаторын көрсетпесе болады, егер де программа жұмысы барысында тек бір байланыс құрылған жағдайда. `string mysql_error([int $link_identifier])`.

Деректер қорына сұрауларды орындау

Ол үшін арнайы жалғыз `mysql_query()` - функциясы деректер жиынын нәтижелейтін идентификаторды қайтарады.

Есте болсын, біз нәтиже бірден клиентке жіберілмейтіндігін айтқан болатынбыз. Міне, сондықтан оған жету үшін идентификатор керек. Оны параметр ретінде қабылдайтын және сол не басқа деректерді қайтаратын өте көп функциялар бар. Олар туралы кешірек сөз қозғаймыз.

```
Int mysql_query(string $query [,int $link_identifier])
```

Бұл функция өзінің қолданысы әмбебап: ол MySQL - серверге \$query сұрауын жібереді және жауаптың немесе нәтиженің идентификаторын қайтарады. \$query параметрі SQL тілі ережесі бойынша құрылған жол. Алдындағы орнатылған байланыс қолданылады \$link\_identifier, ал ол болмаған жағдайда - соңғы ашық байланыс.

Олар сәтті орындалған не орындалмағаны туралы тек белгіні қайтаратын бірнеше SQL командалары (мысалы, ол UPDATE, INSERT және т.б. командалар) бар. Бұл жағдайда белгі функциямен қайтарылады. Керісінше, SELECT сұрауы үшін шығарылым идентификаторы қайтарылады, егер нөл болса, қате болғандығын білдіреді.

Шын мәнінде сұрауды орындауға арналған тағы бір функция бар, бірақ ол ондай қолдануға ыңғайлы емес. Себебі әр сұрау сайын қатынауға қажет деректер қорының атын көрсету керек.

```
Int mysql(string $dbname, string $query [,int $link_identifier])
```

Қызметі `mysql_query()` сияқты, тек қатынау таңдалған деректер қорына емес, \$dbname параметріне көрсетілгенге орындалады. Егер сізде бірнеше деректер қоры бар бола, бір уақытта қатынасаңыз, бұл функцияны пайдалану сізге тиімді болар. Әдеттігідей, \$link\_identifier параметрін түсіруге болады, онда соңғы ашық байланыс қолданылатын болады.

Кесте құру `create_table` КестеАты (\_ӨрісАты тип, ӨрісАты тип, ...)

Бұл командамен деректер қорында өздерінің аттарымен (ӨрісАты) және көрсетілген типтерімен анықталатын бағандары (өрістермен) бар жаңа кесте құрылады.

Осы өріс типтерінің көптеген түрлері бар, осылар ға тоқтола кетсек кетсек.

Бүтін сандар типтері

Бүтін сандардың бірнеше типтері болады, олар сақтауға арналған деректер қорында бөлінетін деректердің байт сандарымен ерекшеленеді. Бұл типтердің барлығында айырмашылық тек аттарында (кейбір қысқартулармен), жазылуы келесідегідей:

префикс INT [UNSIGNED]

Міндетті емес UNSIGNED жалауы таңбасыз (0-ден үлкен не тең) сандарды сақтайтын өрісті құратынын білдіреді. Типтер аттары жалпы түрде префикс INT сияқты 1 кестеде келтірілген.

## 2.1 кесте - Бүтін санды мәндер типтері

Тип	Сипаттамасы
TINYINT	-128 ден +127 дейінгі сандарды сақтай алады
SMALLINT	-32 768 ден 32767 дейінгі диапазондағы сандар
MEDIUMINT	-8 388 608 ден 8 388 607 дейінгі диапазон
INT	-2 147 483 648 ден 2 147 483 647 дейінгі диапазон
BIGINT	-9 223 372 036 854 755 808 ден 9 223 372 036 854 775 807 дейінгі диапазон

Бөлшек сандар типтері

Дәл бүтін сандар сияқты MySQL-де бірнеше түрге бөлінеді. Жалпы түрде былай жазылады ТипАты[(length,decimals)] [UNSIGNED]

Мұнда length - Java-да берілгенде бөлшек сандар орналасатын белгі орын саны (өріс ені), ал decimals - ескерілетін ондық үтірінен кейінгі белгілер (сандар). Әдеттегідей, UNSIGNED таңбасыз сандарды белгілейді.

## 2.2 кесте - Рационалды мәндер типтері

Тип	Сипаттамасы
FLOAT	Үлкен емес дәлдікті жылжымалы нүктелі сан
DOUBLE	Екілік дәлдікті жылжымалы нүктелі сан
REAL	DOUBLE үшін синоним
DECIMAL	Жол түрінде сақталатын бөлшек сан
NUMERIC	DECIMAL үшін синоним

Жолдар типтері

Жолдар дегеніміз символдар массиві. Әдетте SELECT сұрауы бойынша мәтіндік өрістен іздегенде символдар регистрі қарастырылмайды, демек «қарындаш» пен «ҚАРЫНДАШ» бір болып саналады. Бұдан басқа, егер

мәтінді енгізгенде немесе өзгерткенде дерек қор оны автоматты түрде қайтакодтауға бапталған болса, онда бұл өрістер сіз көрсеткен кодировкада сақталады.

Ең алдымен length символдарынан аспайтын жол типімен танысайық, мұндағы length 1 ден 255 дейінгі диапазонда жатады.

**VARCHAR(length) [BINARY]**

Осындай типті жолға өзгеріс енгізгенде одан ақырғы пробелдер қиып алынады. (rtrim() функциясының шақырылуындай). Егер BINARY жалауы көрсетілген болса, онда SELECT сұрауында жол регистр есебімен салыстырылады. VARCHAR типі 255 символдан аспайтындығымен ыңғайсыз. Оның орнына 3 кестеде келтірілген басқа мәтіндік типтерді қолдануға болады.

### 2.3 кесте - Жолдық мәндер типтері

Тип	Сипаттамасы
TINYTEXT	Максимум 255 символ сақтай алады
TEXT	Максимум 65 535 символ сақтай алады
MEDIUMTEXT	Максимум 16 777 275 символ сақтай алады
LONGTEXT	Максимум 4 294 967 295 символ сақтай алады

### Бинарлы мәліметтер типтері

Бинарлы мәліметтер. Бинарлы мәліметтер - бұл TEXT форматындағыға ұқсас мәліметтер, бірақ іздеу жүргізгенде символдар регистрі ("abc" мен "ABC" - әртүрлі жолдар). Барлығы бинарлы мәліметтердің 4 типі болады (4 кесте).

### 2.4 кесте - Бинарлы мәндер типтері

Тип	Сипаттамасы
TINYBLOB	Максимум 255 символ сақтай алады
BLOB	Максимум 65 535 символ сақтай алады
MEDIUMBLOB	Максимум 16 777 275 символ сақтай алады
LOBLOB	Максимум 4 294 967 295 символ сақтай алады

### Уақыт пен дата

MySQL уақыт пен датаны әртүрлі форматта сақтауға арнайы бейімделген бірнеше жол типін қолдайды (5 кесте).

### 2.5 кесте - Уақыт пен датаның мәндік типтері

Тип	Сипаттамасы
DATE	YY-MM-DD форматындағы дата
TIME	HH:MM:SS форматындағы уақыт
DATETIME	YY-MM-DD HH:MM:SS форматындағы уақыт пен дата
TIMESTAMP	timestamp форматындағы уақыт пен дата



## Модификаторлар мен типтер жалаулары

Типке модификаторды қосуға болады, олар «тәртіп» пен сәйкес бағандармен орындауға қажет операцияларды тағайындайды. Олардың ең көп тарағандары 6 кестеде келтірілген.

### 2.6кесте - Негізгі модификаторлар

Модификатор	Сипаттамасы
not null	өріс анықталмаған мәнді қабылдамайтынын білдіреді
primary_key	өріс басты кілті екенін білдіреді, яғни жүгінуге болатын жазба идентификаторы екендігі
auto increment	жаңа жазба қойғанда өріс уникалды мәнге иеленеді, сол себепті кестеде екі бірдей нөмірлі өріс болмайды
DEFAULT	қолданылатын өріс үшін мәнді үнсіздік бойынша орнатады

Кестені өшіру `drop table КестеАты`. КестеАты кестесін өшіреді. Кесте міндетті түрде бос болудың қажеті жоқ, сондықтан байқаусыздан құрамында ақпараты бар кестені өшіріп қоймаңыз.

Жазбаны қою

`insert into КестеАты(ӨрісАты1 ӨрісАты2 ...) values('мән1','мән2',...)`

КестеАты кестесіне ӨрісАты N деп белгіленген өрісті қосады, мұнда мән N мәндеріне сәйкес келеді. Бұл командада аталып өтпеген өрістер «анықталмаған» (анықталмаған мән - бұл бос жол емес, берілген жолда мәндер жоқ екендігі) мәндеріне ие болады. Қайта көрсетілмеген өрісте кестені құрғанда `not null` берілген жағдайда бұл команда сәтсіз аяқталады. Өрістер мәндерін кәдімгі жақшаға алуға болады, бірақ мұнда апострофтарды қолданған ыңғайлырақ сияқты. Кестеге бинарлы деректері енгізгенде (немесе құрамында слэш пен апострофтары бар) кейбір символдар кері слэштермен «қорғалуы» тиіс, демек `()`, `(,)`, `(')` символдары мен және нөлдік кодты символдармен.

Жазбаны өшіру `delete from КестеАты where` өрнек.

КестеАты кестесінен өрнек орындалған барлық жазбаларды өшіреді. Өрнек параметрі - бұл Java ережелері бойынша құрастырылған жай ғана логикалық өрнек. Мысалға:

`(id < 10) and (name regexp 'a*b') and (age=25)`

Өрнекте өріс аттары, константалар және операторлардан басқа қарапайым «есептелетін» бөліктер де кездесуі мүмкін, мысалы: `(id<10+1*234)`.

Жалпы айтқанда, өрнек форматы келешекте кездесетін сұраулардың барлық командалары үшін жалпыға бірдей. Мысалы, ол `select` операциясында да, `update` операциясында да қолданылады.

Жазбаларды іздеу

`select * from Кесте where Өрнек [order by ӨрісАты [desc]]`

Бұл команда - Негізгі және өте мықты. Ол “Өрнек” өрнегін қанағаттандыратын барлық жазбаларды іздейді. Оның мүмкіндіктері әлдеқайда молырақ бірақ оған біз тоқталмаймыз. Егер жазбалар бірнешеу

болса, онда order by ұсынысы бойынша олар бұл кілттік сөздің оң жағында жазылатын аты бойынша іріктеледі (егер desc сипаттаушысы жазылса, онда іріктеу кері ретпен орындалады). Сонымен қатар order by ұсынысында бірнеше өріс берілуі мүмкін. \* символының рөлі маңызды. Ол тандалынған жазбалардан барлық өрістерді көрсету керектігін білдіреді. Басқа жағынан, жұлдызша орнына үтір қойып, көрсетуге қажетті өрістің аттарын жазып шығуға. Бірақ \* өзі жиірек қолданылады.

Жазбаны жаңарту

```
update Кесте set (ӨрісАты1='мән1', ӨрісАты1='мән2') where Өрнек
```

Өрнек өрнектерін қанағаттандыратын Кесте кестесінде барлық жазбалар үшін көрсетілген өрістер сәйкес мәндерде орнатылады. Бұл команда егер бір жазбаның барлық өрістерін емес, тек кейбіреулерін ғана жаңарту қажет болған жағдайда қолданылады.

Өрнекті қанағаттандыратын жазбалар санын алу

MySQL-дің тағы бір жиі қолданылатын мүмкіндігі - Өрнекті қанағаттандыратын жазбалар санын алу қарастырған жөн. Жалпы айтқанда, бұны жасаудың бірнеше тәсілі бар. Соның бірі мынау:

```
select count(if(Өрнек,1,NULL)) from Кесте
```

Бұл мысалдың өзі MySQL-дің тілі сипатталғанға қарағанда қаншалықты бай екенін көрсетеді...

Бағанның бірігей мәндерін алу

Деректерқорды қолданғанда, берілген кестеден бағанда қандай бірігей мәндері бар екендігін білген ыңғайлы болады. Мысалы, егер кез-келген бір статистикалық кестеде әрбір жазбада, оның құрамында адамдар жайында мәлімет болса, және онда Country (ел) өрісі бар болса, онда кестеде 30 жасқа дейін өмір сүріп келген адамдардың барлығы қай елде тұратынын табу үшін келесі сұрауды орындаймыз.

```
select distinct ӨрісАты from Кесте where Өрнек
```

Біздің жағдайда ӨрісАты = Country, ал Өрнек - age>=30. Бұл сұрау ізделіп отырған елдерден тұратын бір бағаннан тұратын нәтижені тудырып береді.

Нәтиже алу

Нәтиже - бұл жай ғана деректер жиыны, және оған кіретін жазбалар санын mysql\_num\_rows() арқылы білуге болады. Мысалы, егер алдыңғы мысалда кестеден таңдау кезінде 30 жастан 10 адам бар болса, онда біздің нәтиже идентификаторында 10 «жолға» «сілтеме» аламыз.

Әрбір жазба – бұл өрістер мәндерінің тізімі, яғни, select ... from Кесте сұрауында көпнүкте орнына (жұлдызша болса, барлық өрістер) көрсетілген реті бойынша. Осылайша, нәтиже - бұл өзіндік екі өлшемді массив: бірінші индекс - жазба номері және екінші - өріс аты.

Нәтиже параметрлері

```
int mysql_num_rows(int $result)
```

`mysql_num_rows()` функциясы сұрау нәтижесінде жазбалар санын қайтарады. Осылайша, функция «екіөлшемді массив нәтижесінің» вертикальды өлшемін анықтауға мүмкіндік береді

```
int mysql_num_fields(int $result)
```

Бұл функция нәтиженің бір жолына өрістер санын қайтарады, яғни, саны `$result` нәтижесінде бағандарды. Сонымен қатар «екіөлшемді массив нәтижесінің» горизонтальды өлшемін анықтауға мүмкіндік береді.

Нәтиже өрісін алу

```
int mysql_result(int $result, int $row, mixed $field)
```

Функция `$field` өрісінің нәтижесін `$row` номерлі жолға қайтарады. `$field` параметрі тек өріс атын беріп қоймай, сонымен қатар оның номерін - позициясын, кесте құрғандағы «тұрған» бағанда. Дегенмен, мүмкіндігінше барлық жерде өрістің атының өзін қолдануға тырысу керек.[10]

Функция әмбебапты: оның көмегімен бүкіл нәтижені бір-бір ұяшық бойынша «өтіп» шығуға болады. Мұны істеуге тиым салынбағамен, қолданбауға тырысу керек, өйткені `mysql_result()` біршама баяу жұмыс жасайды.

### 2.3.2 GlassFish Қосымшалық сервер ерекшеліктері

Қосымшалық сервер (англ. application server) - бұл бағдарламаның құрылысын қамтамасыз ететін процедураларды(бағдарламалар, механикалық оперциялар, скриптар) тиімді орындауға арналған бағдарламалық платформа (software framework). Қосымшалық сервер сол платформаның өзімен анықталған, бағдарламалар құраушысына API(қолданбалы бағдарламалау интерфейсі) арқылы қол жетімді, компоненттер жиынтығы ретінде әрекет етеді.

Web-бағдарламасы үшін ол компоненттер әдетте, web-сервер қосылған машинада жұмыс жасайды. Оның негізгі жұмысы - динамикалық web-беттердің құрылуын қамтамасыз ету. Алайда қазіргі таңдағы қосымшалық серверлер тек web-беттерді құруға ғана емес, оданда көп, кластерезация, тоқтап қалуға тұрақтылығын сақтау және жұмыс салмағын балансировка жасау сервистарын ұсынып, бағдарлмаушыға тек бизнес-логикаға көңіл бөлуінне мүмкіндік береді.

Әдетте бұл термин Java-сервер қосымшаларына қатысты. Бұл жағдайда қосымшалық сервер, бағдарлама қосуға арнлған кеңейтілген виртуладық мишина ретінде жұмыс імтейді, және сонысымен деректер қорымен өз алдына және қолданушылық машиналармен өз алдына қосылу мүмкіндігін береді.

Қосымшалық сервер артықшылдықтары:

- Код және мәлімет тұтастығы.

Бизнес-локиканы бір бөлек серверге немесе бірнеше бір-бірімен қосылған серверге белгілеу арқылы, бағдарламының жаңартылуы немесе жақсартылуы барлық қолданушыларға баратындғына кепілдік беруге болады.

Ескі нұсқадағы бағдарлама ақпаратқа қол жеткізу немесе оны ескіше қайта келместей өзгерту мүмкіндігінің қатерін жоғалтады:

- Орталықтандырылған басқару және жөн күйіне келтіру.

Деректер қорының серверін өзгерту немесе жүйлік күйді өзгерту секілді бағдарламаның жөн күйін өзгерту, орталықтандру арқылы жүзеге асырыла алады.

- Қауыпсіздік

Қосымшалық сервер орталықтандырылған нүкте ретінде әрекет етеді, және соны қолдана отырып, сервис құраушылар мәліметке қол жеткізуді және сол бағдарламаның бөлшектерін басқара алады, бұл қауыпсіздіктің белгісі болып сналады. Бұл бізге қауыпсіз емес қолданушы деңгейдегі аутентификация процессін серверлік деңгей жауаптығына ауыстыруға, қосымша сонымен қатар деректер қоры өз алдына бөлек болуына мүмкіндік береді.

Транзакцияға рұқсат беру

Транзакция бірігей белсенділік бірлігін ұсынады, оның орындалуы барысында көтеген ресурстардың өзгерістері (бір немесе бірнеше деректер көзінен) атомарлы (бөлінбей жұмыс жасайтын бірлік ретінде) жүзеге асырылуы мүмкін. Әрине бұдан қолданушылар жүйенің стандартты жұмыс жасауынан, оны жасау ауқытының төмендеуінен және бағаның түсуінен ұтыс табады. Қосымшалық сервер көптеген жұмыстардың орнына өзі код өңдеп автоматты жұмыс жасауы, бағдарламаушыларға тек бизнес-логика мәселелеріне көңіл бөлуге мүмкіндік береді.

Жасалу мысалдары:

- JavaEE қосымшалық сервері дегенде JavaEE концепцияларын орындай отырып, өзінде сол бағдарламаларды қосуға мүмкіндік беретін бағдарламалық комплекс түсіндіріледі. Қазіргі таңда танымал бұндай қосымшалы серверге мына бағдарламалық құралдарды Sun GlassFish, IBM WebSphere, RedHat JBoss Application Server, Apple WebObjects және т.б жатқызуға болады.;

- Zope, дамыған web-бағдарламалық сервері;

- Терминалдык серверлер, мысалы Citrix компаниясымен жеткізіліп отырған серверлер.

GlassFish - ашық бастапқы кодпен таратылатын, JavaEE спецификациялары толығымен жүзеге асырылған, алғашында Sun Microsystems құрылған қосымшалық сервер. Бүгінгі таңда Oracle компаниясының демеуінде. Өзекті платформаның нұсқасы Oracle GlassFish Server 4 деп аталады.

GlassFish қосымшалық серверінің түп тамырына Sun компаниясының Java System Application Server және ORM TopLink (Oracle компаниясымен ұсынылған, Java объектілерді реляциялық ДҚ сақтауға арналған шешім) бағдарламалар кодтарының бөлігі енді. Онда сервле-контейнер ретінде өзгертілген Apache Tomcat қолданылған, және де Java NIO технологиясын қолданатын Grizzly қосымша компоненті бар.

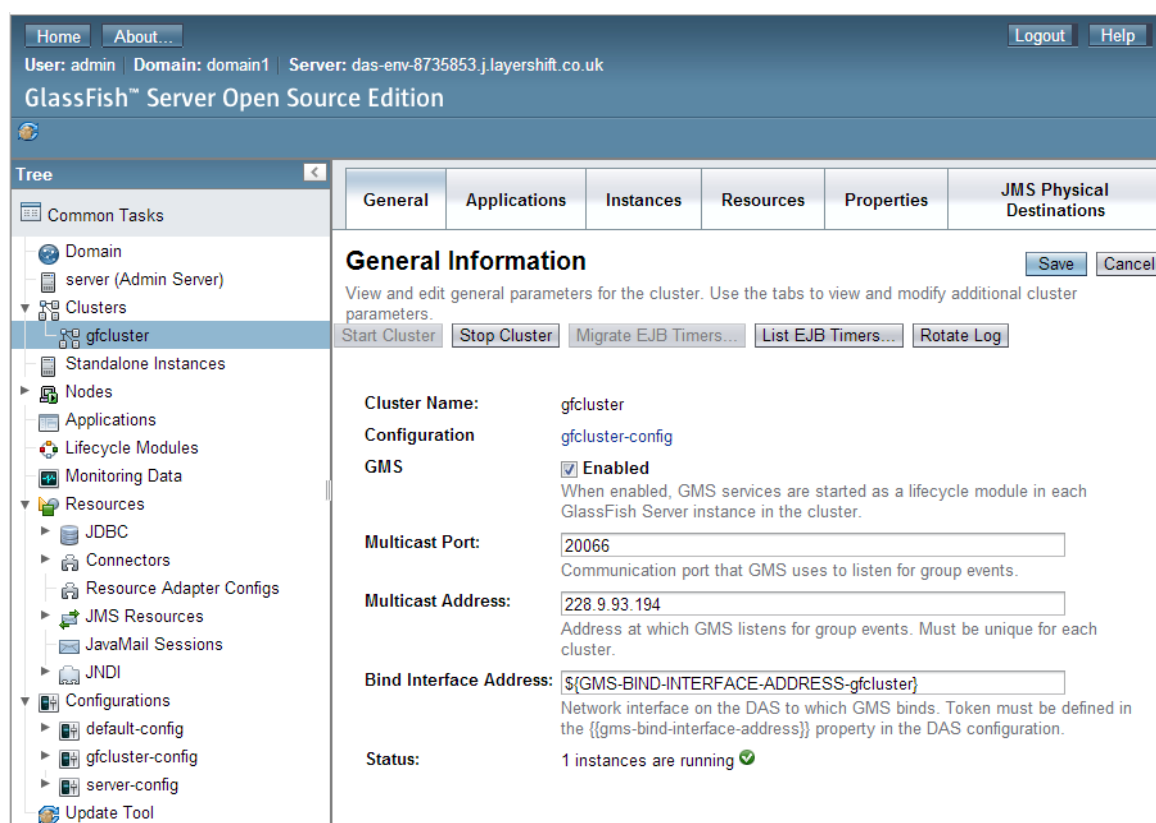
Тарихы:

Бұл жоба 2005 жылдың 5-ші шілдесінде бастау алды. 2006 жылдың 4-ші мамырында, GlassFish жобасы, Java EE5 спецификациясын толығымен қанағаттандыратын алғашқы нұсқасын шығарды.

2007 жылдың 8-ші мамырында JavaOne конференциясында GlassFish жобасының тармағы ретінде SailFin жаба бөлшегі шығатындығы туралы анонсталды. SailFin жобасында Session Initiation Protocol (SIP) протоколының функционалдықтарын GlassFish сервлеттеріне қосу көзделді.

2007 жылдың 17-ші қыркүйекінде GlassFish қауымдастығы 2-ші нұсқасын (Sun Java System Application Server 9.1 деген атпен танымал) жариялады. GlassFish - тің 2-ші нұсқасы өндірістік кластерицияны, Microsoft-әрекеттес веб-сервистерді қолдады және де SPECjAppServer өнімділік тестерінде жоғары орын алды (BEA Weblogic және IBM Websphere озып кетті). [4]

Бүгінгі күнде GlassFish қосымшалық серверінің өзекті нұсқасы GlassFish 4 және ол Java EE 7 спецификациясын толығымен қолдайды.



2.8 Сурет - GlassFish қосымшалық серверінің жөн күйге келтіру веб-интерфейсі

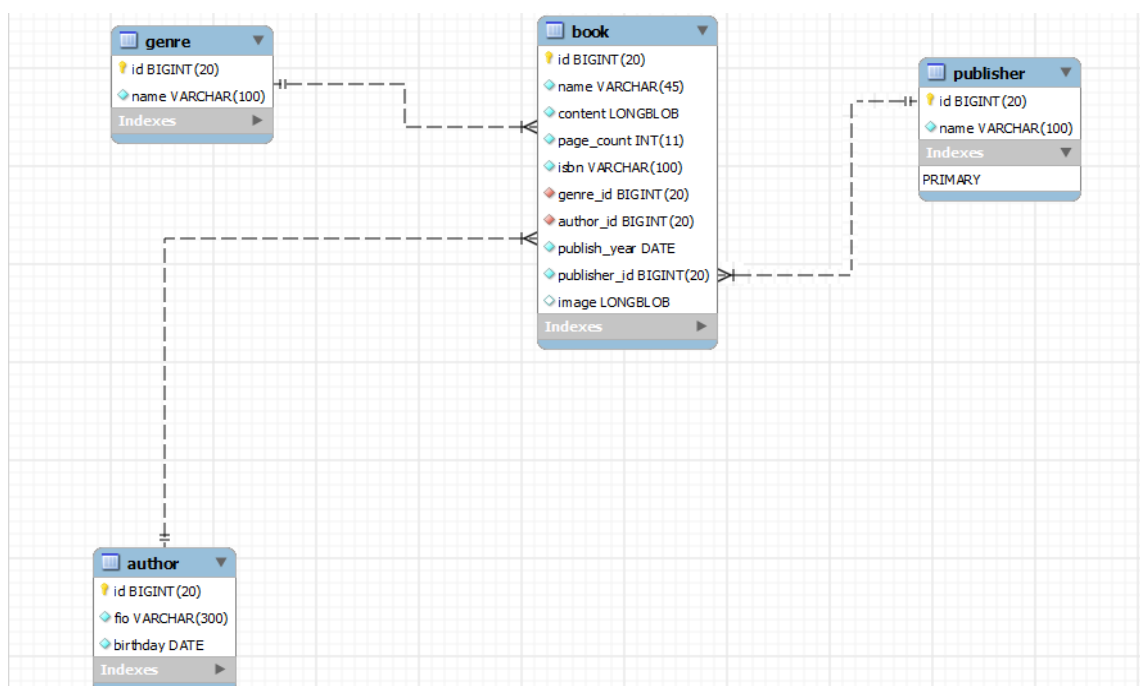
### 3 Эксперименталды бөлім

#### 3.1 Жүйені қамтамасыз ететін ДҚ-н құру

Кез-келген жүйеде деректер қоры (ДҚ) пайдаланылады. Деректер қоры негізінен жүйедегі көп өлшемді статикалық мәліметтерді сақтауға керек. Біздің жағдайда дерек қор кітаптар жайлы барлық қажетті деген мәліметті, кітаптардың жанырлары туралы мәліметті, кітаптың авторлары жайлы мәліметті және баспа мекемелері жайлы мәліметті сақтауы тиіс.

Көріп отырғанымыздай біздің сақтайтын мәліметтерді 4 болмыс жайлы, және сәйкесінше біздің дерек қорымызда 4 кесте (әр кесте 1 болмыс жайлы мәліметті сақтайды) болуы тиіс. Және осы 4 болмыс бір-біріне тәуелді болғандықтан, біздің 4 кестеде өзара байланысты болуы керек.

Біз деректер қорын басқару жүйесі ретінде MySQL 5 бағдарламасын қолдандық. MySQL ДҚБЖ – сі деректердің реляциялық моделін қолданады. Ал деректердің реляциялық моделінде мәліметтер кестелерде сақталады және кестелер өзара ішкі кілт (primary key) және сыртқы кілт (foreign key) арқылы байланыс құрады.



3.1 Сурет - library ДҚ - ның EER диаграммасы

3.1 суреттегі диаграммада біздің жүйге қажетті library ДҚ-ның кестелерінің өзара баланысы көрсетілген. Көріп отырғанымыздай бізде 4 кесте болады олар: book, author, genre, publisher. Осы 4 кестенің әрқайсының құрылымына тоқталып кетелік:

### 3.1 Кесте - author кестесі

Column name	Datatype	PK	NN	UQ	AI	Default
Id	BIGINT(20)	yes	yes	yes	yes	
Fio	VARCHAR (100)		yes			
Birthday	DATE		yes			

### 3.2 Кесте - genre кестесі

Column name	Datatype	PK	NN	UQ	AI	Default
Id	BIGINT(20)	yes	yes	yes	yes	
Name	VARCHAR (100)		yes			

### 3.3 Кесте - publisher кестесі

Column name	Datatype	PK	NN	UQ	AI	Default
Id	BIGINT(20)	yes	yes	yes	yes	
Name	VARCHAR (100)		yes			

### 3.4 Кесте - book кестесі

Column name	Datatype	PK	NN	UQ	AI	Default
Id	BIGINT(20)	yes	yes	yes	yes	
Name	VARCHAR (45)		yes			
Content	LONGBLOB		yes			
page_count	INT(11)		yes			
Isbn	VARCHAR(100)		yes	yes		
genre_id	BIGINT(20)		yes			
author_id	BIGINT(20)		yes			
publish_year	INT(11)		yes			
Publisher_id	BIGINT(20)		yes			
Image	LONGBLOB					

Бұл кестелерде 4 кестенің ішкі құрылысы көрсетілген. Мұндағы:

- Column name – кестедегі баған атауы;
- Datatype – бағандағы сақтатйтын мәлімет типі;
- PK – (primary key) ішкі кілт;
- NN – (not null) мәліметтің null болу немесе болмауы;
- UQ – (unique index) мәліметтің уникалды болуы немесе болмауы;
- AI – (auto incremental) санның автоматты түрде өсіп отырылуы;
- Default – Үнсіздік бойынша қойылатын мән.

Осы кестелерге өзінің болмысы бойынша барлық бағандарға мәлімет енгізіді. 2.3 Суретінде MySQL Workbench бағдарламасы арқылы мәліметті енгізу мысалы көрсетілген.

id	name	content	page_count	isbn	genre_id	author_id	publish_year	publisher_id	image
1	Черный город	BLOB	256	978-5-8159-1145-1	1	1	2012	1	BLOB
2	На Западном фронте без перемен	BLOB	95	978-84-350-1603-2	4	6	1929	2	BLOB
3	Бородино	BLOB	16	978-5-9287-2385-9	5	7	2004	2	BLOB
4	Полита	BLOB	159	256-45-687	5	8	1967	1	BLOB
5	Бременские музыканты	BLOB	20	1254-2563	6	9	1953	3	BLOB
6	Мэри Поппинс	BLOB	121	978-5-237-05630-3	6	10	1972	1	BLOB
7	iКона: Стив Джобс	BLOB	221	978-5-91657-374-9	7	11	2012	2	BLOB
8	Стив Джобс. Уроки лидерства	BLOB	241	978-5-91657-441-8	7	12	2012	2	BLOB
9	Посмертные записки Пиквикского клуба	BLOB	1582	978-5-17-068025-2	14	13	1837	5	BLOB
10	Соборяне	BLOB	176	978-5-17-070886-4	12	14	1872	4	BLOB
11	Граф Монте-Кристо	BLOB	748	978-08-952-634-69	10	15	1845	2	BLOB
12	Капитан Фракасс	BLOB	488	5-85844-023-1	10	16	1992	1	BLOB
13	Три мушкетёра	BLOB	369	9788-17-1564-378	10	15	1844	2	BLOB
14	Основы Java	BLOB	813	978-5-8459-1378-4	9	17	2010	7	BLOB
* NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

### 3.2 Сурет - MySQL Workbench бағдарламасында кестенің толтырылуы

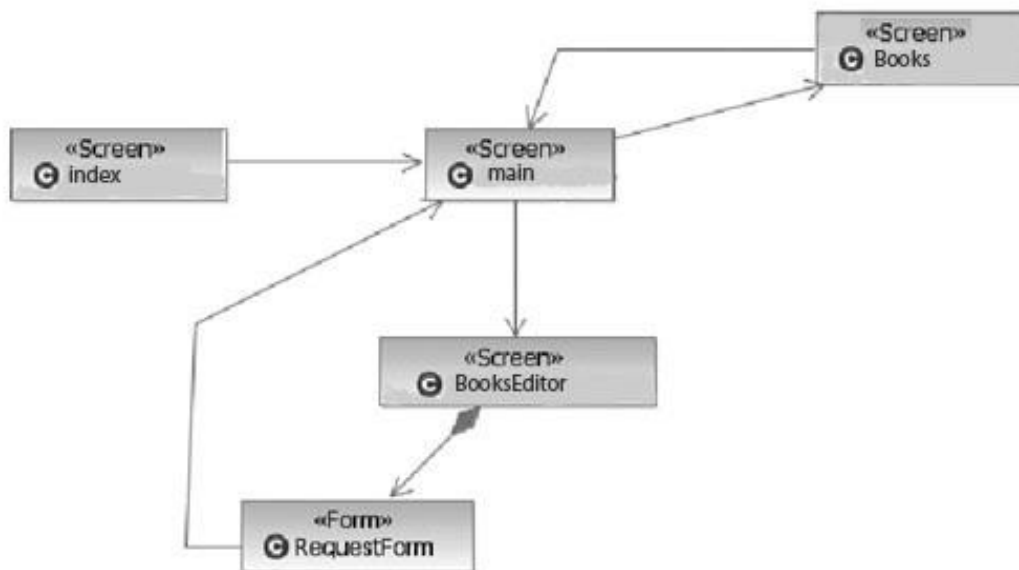
#### 3.2 АЖ бөлігін құрастырылуы

Web-жүйені анализдеу кезінде кез-келген бағдарламаны анализдеуде сияқты ең маңызды рөл оларды құрайтын болмаыстарды және жүйе процесстерін анықтап алу болып табылады. Болмыстар және жүйенің процесстері тақырып аумағын толық, деталды түрде суреттейді, архитектураға байланысты емес. Web бағдарламаларда ең маңызды артефакт көпшілігі навигация картасы және web-парақшаларды анықтау болып табылады. Дөрекі айтқанда, бұл элементтер болмыстардың стереотиптеріне тура келеді, контроллерлер және шекаралар, классикалық анализде қабылданған. Басын web-жүйенің моделдерін түсіндіруден басталық: қолданушының іс-әрекет моделі (User Experience - UX).[10]

#### 3.2.1 Қолданушының іс-әрекет моделі

Қолданушының іс-әрекет моделі – бұл қолданушының интерфейсін абстракцияның жоғарғы сатысында суреттейтін мысалы, web-парақшалармен стилдерді (шрифтер, өлшемдерді, түстерді және т.б), ынғайлы болу үшін құрастырудың кейінгі сатыларында жасайтын, аңғармай навигация жүргізуге мүмкіндік береді. 3.2 суретінде қолданушы интерфейстің жағары сатылы фрагментті моделін суреттейтін, маңызды сценарилерді қолдану үшін жасалған қолданушы іс әрекет моделі көрсетілген. Диаграммада екі стереотип көрсетілген олап: <<Screen>> және <<Form>>. <<Screen>> стереотипі классты ұсынады, қолданбалы интерфейсті суреттейтін, экрандағы суреттерді, яғни дөрекі айтқанда, - web-парақшаны. Кейбір экрандар HTML-формасы элементтерін иеленеді, олар серверге мәлімет жіберу үшін қолданылады. Бұл стереотипті класстар әр-қашан экранмен және басқа web-парақшаға өтуді басқарумен байланысты. Біздің ассоциядан шыққан бағыт экрандармен ауысу бағытын көрсетеді.

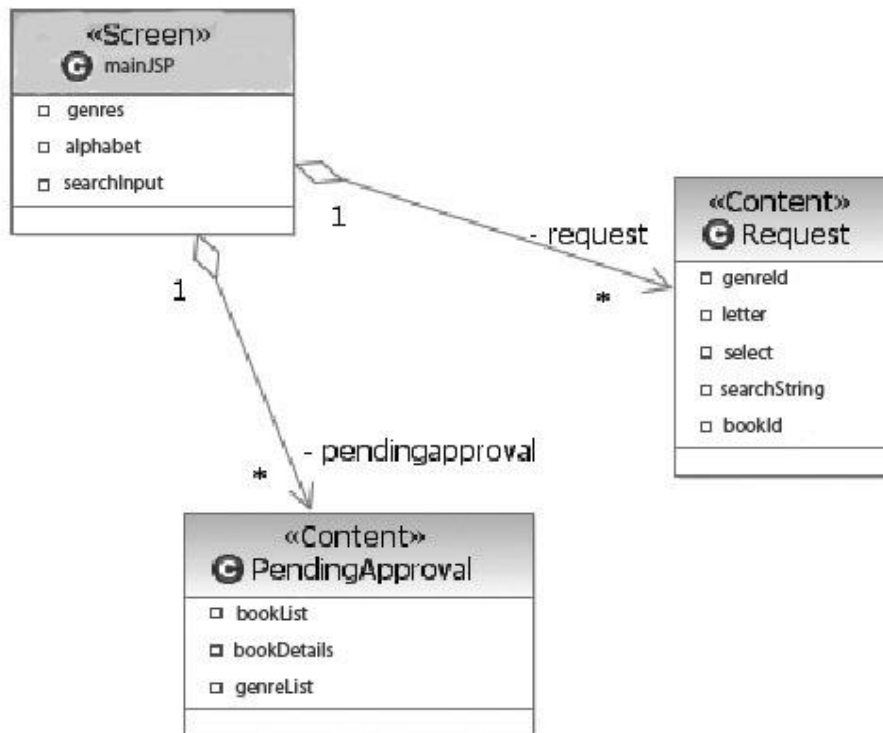




3.3 Сурет - Қолданушы іс-әрекетінің жоғары деңгейлі моделі

3.3 суреті экраннан экранға ауысу туралы өте көп мәлімет беретініне қарамастан, ол сол беттің құрамы жайлы өте аз мәлімет береді. Осы сатыда әр парақшаның құрамы жайлы мәлімет беру жақсы болар еді. Мысалыға 3.4 суретінде mainJSP экранының құрамы толығымен көрсетілген. Осы стереотипті класс <<Screen>> атрибуттары бөлек мәндерді сипаттайды, көбінесе олар жай мәтін немесе қарапайым типті мәліметтер болады, HTML оларды оңай оқиды. Genres (жанрлар тізімі) және alphabet (алфавит бойынша әріптер тізімі) атрибуттары барлық парақшада кездесетін, жоғары қалпашықта және сол жақта менюда орналасатын тұрақты мәліметтер. SearchInput атрибуты қолданушы жазатын іздеу өрісінің мәні болып табылады. Мысалыға, қолданушы белгілі бір сөз жазып сонымен кітап іздесе оған сәкес табылған кітаптар саны және кітаптар тізімі шығуы мүмкін, әлде ол мәтін бойынша кітап табылмаса кітаптар соны 0-ге тең деп шығарылады.

Жиі экранның құрамы әр-түрлі және қиын болады. Оның құрамын дәл табу үшін жүйеде <<Content>> стереотипті класы қолданылады. Ол лагикалық функционалды шынайы деректермен қосу үшін жасалады. Экранның құрамында құрамы көбінесе пункттер тізімі болады. 3.4 суретінде Request (сұраныс) және PendingApproval (күтіп тұрған сұраныстар) класстары бейнеленген, олар басты беттің құрымын құрады. Экранда осы класстардың көп объектілері көрінуі мүмкін. Құрылымдық класстар атрибуттарға ие, ережеге сәйкес, экранның атрибуттары болып табылады, және HTML-мен оңай оқып танылады. Бұндай абстракцияның жоғары деңгейінде шынайы типтерді қоспасақ болады. Олардың атын және солар туралы кішігірім мәлімет берсек жетеді.



3.4 Сурет - mainJSP класының деталды құрылымы

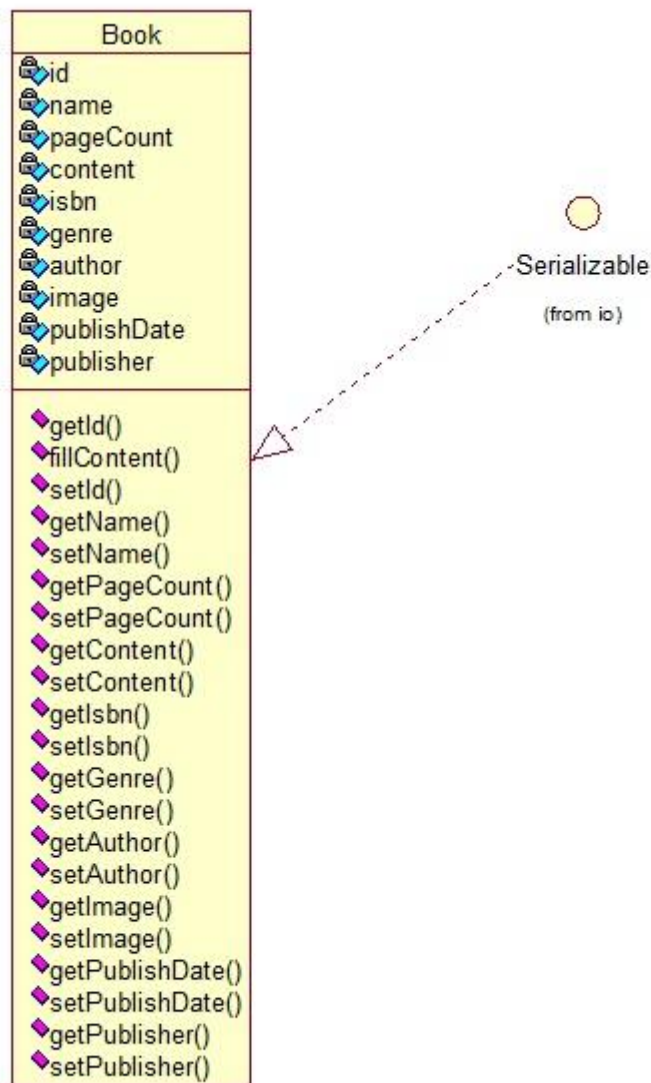
Айта кететін бір қызықты жай, осындай ерте кезеннің өзінде, өте маңызды шешімдер қабылдау қажет, және сол шешімдер сонынан алатын жүйге әжептеуір әсер етеді. Мысалыға, біз main парақшасының құрамын анықтау арқылы, ол тек қолданушыға ғана емес, администратордың өзіне де дәл осындай күйде болады. Қолданушының іс-әрекет моделі дәл осындай жайыптарды құжаттауға арналған. Сол үшін соңғы шешімдерді тек жүйенің бір тексеріп алған соң ұабылдаған жөн. Бірақ логикалық тұрғыда администратормен жай қолданушы бір басты бетті қолданады.

### 3.2.2 Болмыстар

Болмыстарды құру және жобалаудың басты қиын тапсырмасы оларды идентификациялау болып табылады. Кез-келген аналитикалық моделде барлық объектілерді, атрибуттары болсын болмасын оларды болмыс ретінде жобалауға болады. Мұндағы амал тек персистентті кластарды болмыс ретінде алу болып табылады. Мысалыға, кей бір класстар, атрибуттары болса да, арылқ класстар болуы мүмкін немесе олардың мәндерін суреттейді. Мысалыға Database классы тек жүйенің деректер қорымен байланысын құратын connection объектісін қайтарады. Оның қорытындысы жүйеде жазып сақталмайды, сол үшін оған персистеті болудың қажеті жоқ.

Бұл жобада болмыстар `com.java.sergazyuev.web.beans` пакетінде орналасқан. Олардың тізімі: `Book`, `Genre`, `Author`, `BookList`, `GenreList`, `LetterList`.

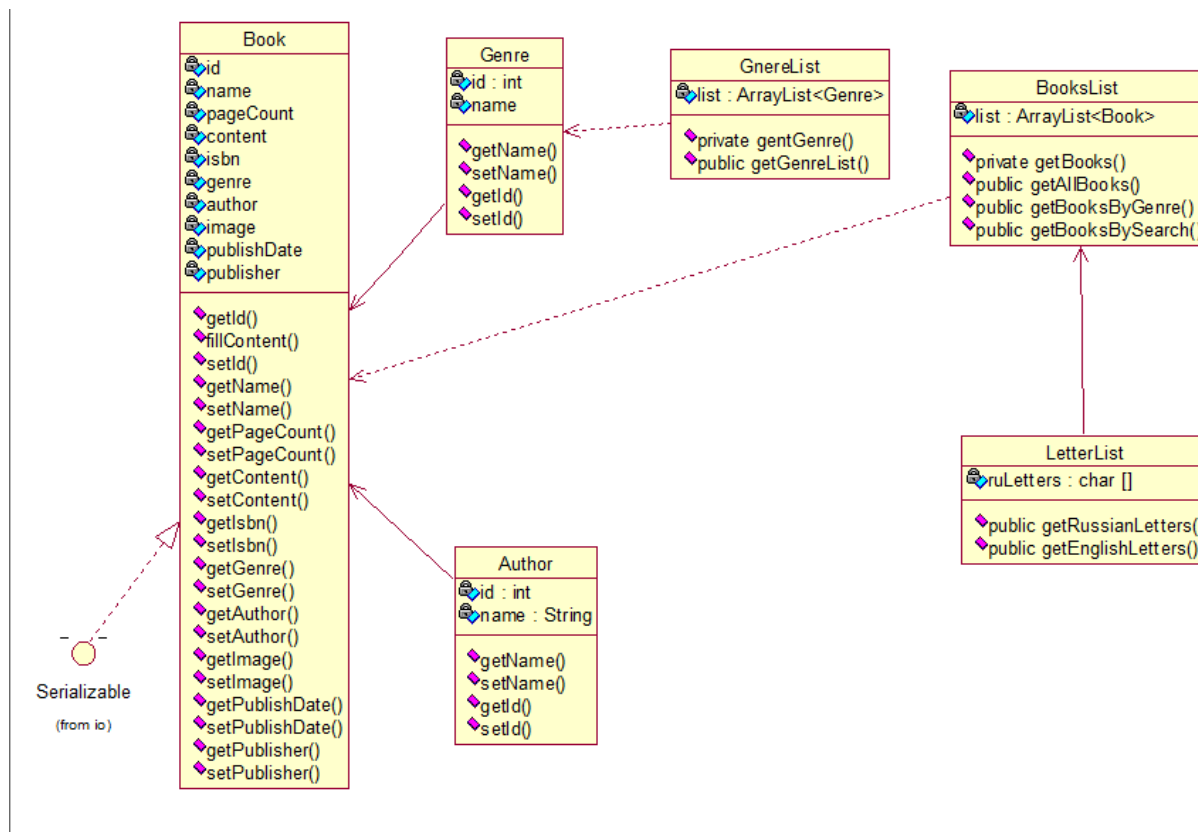
Book болмысының 10 атрибуты және 21 методы бар. Атрибуттардың әрқайсына деректер қорындағы кестенің өрістері сәкес келеді. Тек Autho, Genre, Publisher атрибуттары id бойынша сәйкесінше аталатын кестелерден алынады. 20 метод осы сәйкес болмыстардың get және set тен басталып олардың мәнін қайтарып алуға және оларды орнатуға қолданылады. Ал fillContent() методы қолданушы оқу деген батырманы басқанда деректерді PdfContent деген сервлет арқылы деректер қорынан шақыруға керек. Келесі суреттерде осы болмыстардың құрылымын бейнелейтін UML диаграммасындағы объектілік көріністері берілген.[6]



### 3.5 Сурет - Book класының құрылымы

Бұдан бөлек тағы бір-бірімен тығыз байланысты 5 болмыс класстар бар. Genre және Author класстары сәйкесінше деректер қорындағы осындай кестелердегі мәліметерді сақтайды. Сондықтан өзінің атрибуттарына set және get методтары бар. Және де GenreList, BookList және LetterList класстары бар. GenreList класы сол жақ менюда шығарылатын жанрлар тізімін сақтауға

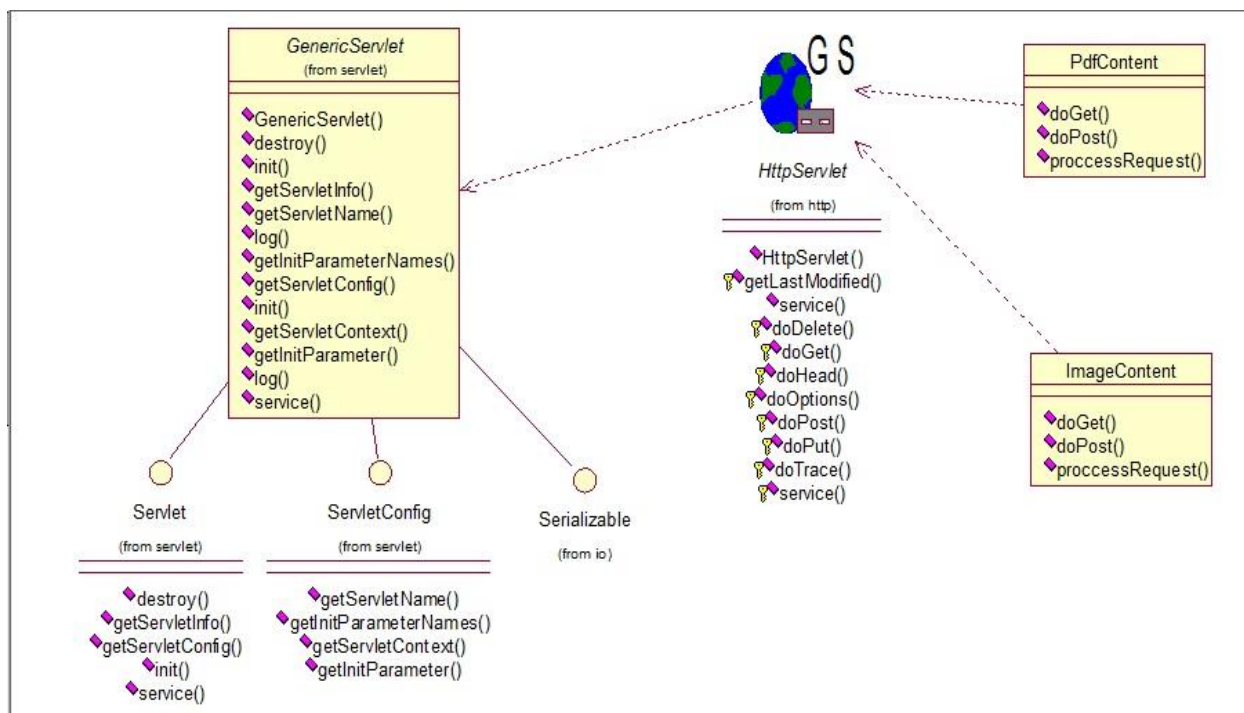
қажет. BookList классының қызметі Book болмыстарының сәкесінше тізімін жинап соларды қайтару. LetterList классының қызметі жағарғы қалпақшада алфавит тізімін шығарып сол бойынша іздеуге мүмкіндік беру болып табылады. 3.6 суретінде осы болмыстарды құрайтын барлық класстардың құрылымы және өзара байланысы көрсетілген.



3.6 Сурет - Болмыстарды құрайтын класстардың құрылымы

### 3.2.3 Класстар

Кез-келген java-да жазылған web жобалар сервлет негізінде құрастырылады. Сервлет қазіргі java үшін жазылған фрейворктардың барлығы осы сервлет негізіде жасалған. Бұл жобажа біз негізінен деректер қорына суреттерді және pdf форматтағы кітаптарды алып парақша бетіне орналастыру үшін қолданамыз. Сонымен бізде PdfContent және ImageContent деген екі сервлет жасадық. Ол сервлеттер негізінен HttpServlet класынан мұра алған, ал ол класс GenericServlet класынан мұрағат алған және 4 Serializable, Servlet, ServletConfig деген интерфейстерді орындаған. 3.7 суретінде осы сервлеттердің құрылысы толығымен көрсетілген.



3.7 Сурет - Қолданылған сервлеттердің класстық диаграммасы

Осы сервлеттерді және басқада барлық конфигурация жайлы мәліметтің Deployment Descriptor атты жай күй xml файлы жазылды. Олар жүйені орнатуға арнайы қосымшалық серверге орнату үшін қажет. Бізде web.xml және glassfish-web.xml атты файлдар құрла.

#### web.xml:

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
    <servlet>
        <servlet-name>ShowImage</servlet-name>
        <servlet-class>com.sergazyyev.diplom.web.servlets.ShowImage</servlet-
class>
    </servlet>
    <servlet>
        <servlet-name>PdfContent</servlet-name>
        <servlet-class>com.sergazyyev.diplom.web.servlets.PdfContent</servlet-
class>
    </servlet>
    <servlet-mapping>
        <servlet-name>ShowImage</servlet-name>
        <url-pattern>/ShowImage</url-pattern>
    </servlet-mapping>

```

```

<servlet-mapping>
  <servlet-name>PdfContent</servlet-name>
  <url-pattern>/PdfContent</url-pattern>
</servlet-mapping>
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>
<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
<jsp-config>
  <jsp-property-group>
    <display-name>header_and_footer</display-name>
    <url-pattern>/pages/*</url-pattern>
    <include-prelude>/WEB-INF/jspf/header.jspf</include-prelude>
    <include-coda>/WEB-INF/jspf/footer.jspf</include-coda>
  </jsp-property-group>
</jsp-config>
<resource-ref>
  <res-ref-name>jdbc/Library</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Shareable</res-sharing-scope>
</resource-ref>
</web-app>

```

### **glassfish-web.xml:**

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org//DTD GlassFish
Application Server 3.1 Servlet 3.0//EN" "http://glassfish.org/dtds/glassfish-web-
app_3_0-1.dtd">
<glassfish-web-app error-url="">
  <class-loader delegate="true"/>
  <jsp-config>
    <property name="keepgenerated" value="true">
      <description>Keep a copy of the generated servlet class' java
code.</description>
    </property>
  </jsp-config>
</glassfish-web-app>

```

Жобада бұдан басқада қосымша класстар қолданылды. Олар SearchType және Database.

**SearchType классы:**

```
package com.sergazyyev.diplom.web.enums;
public enum SearchType {
    AUTHOR,
    TITLE
}
```

Бұл ENUM класс болып табылады. Ондай класстардың басқа класстардан айырмашылығы тек белгі сан тұрақты объектілер құрауға ғана болады. Біздің жағдайда олар AUTHOR және TITLE. Олар қолданушы іздеу амалын жасауға қажет.[7]

**Database классы:**

```
package com.sergazyyev.diplom.web.db;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

public class Database {

    private static Connection conn;
    private static InitialContext ic;
    private static DataSource ds;

    public static Connection getConnection(){
        try{
            ic = new InitialContext();
            ds = (DataSource) ic.lookup("jdbc/Library");
            conn = ds.getConnection();
        }
        catch(SQLException ex){
            Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null,
ex);
        }
        catch(NamingException ex){
            Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

```

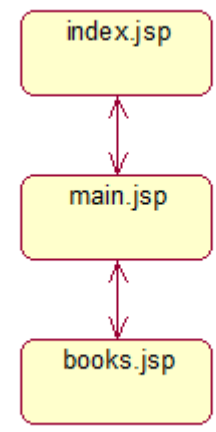
    return conn;
}
}

```

Datbase классының негізінен getConnection() методы басты логиканы құрайды. Оның міндеті jdbc/Library атты JNDI мен қосымшалық сервермен байланысып одан MySQL деректер қорымен байланыс құрайтын connection объектісін қайтарады. Сол объекті байынша болмыс класстар деректер қорымен тікелей жұмыс жасайды.

### 3.3 АЖ бөлігін қолдану жайлы нұсқаулар

Берілген АЖ негізінен 3 JSP парақшаларынан құралған. Олар қоданушыны қасы алу парақшасы index.jsp, қолданушы жүйеге кіргеннен соң кітаптарды іздеу амалдарын ұсынатын негізгі бет парақшасы main.jsp, және де табылған кітаптарды деталды түрде көрсетуге арналған books.jsp парақшасы. Бұл парақшаларда әрқашанда болатын header, footer, left\_menu, letters атты фрагменттер jspf (java server pages fragment) файылдары қолданылған. Бұл файылдар көп жерде котты қайталап жазбау үшін өте ынғайлы болып табылады. JSP технологиясы бізге көп мүмкіндіктер береді. Олар HTML кодпен java котты бірге жазу. Ол негізінен динамикалық web-парақшалар жасау үшін өте пайдалы. Енді осы 3 парақшаға толығымен тоқтала кетсек.



### 3.8 Сурет - Қолданушы итерфейсендегі web-парақшалар

#### 3.3.1 Қарсы алу беті

Қарсы алу парақшасы index.jsp деп аталады. Ол сәкесінше жоба жайлы қысқа мағлұмат береді және қолданушыға өзінің атын енгізіп жүйге кіруді сұрайды.





3.9 Сурет - Жүйенің қарсы алу беті

1-ші блокта жобаның логотипі, жоба жайлы қысқа мәлімет берілген, және кері байланыс жасау үшін жобаны жасаушы адамның электронды поштасы берілген.

2-ші блокта қолданушы өз атын жазып жүйге кіре алады. Ондай жай HTML тіліндегі форма берілген. Бұл форма:

```
<form class="login_form" name="username" action="pages/main.jsp"
method="POST">
    Имя: <input type="text" name="username" value="" size="20" />
    <input type="submit" value="Войти" />
</form>
```

3-ші блокта осы жобаны жасауша адамның аты жөні және жоба жасалған жыл көрсетілген.

### 3.3.2 Негізгі қолданушы беттер

Бізде негізінен екі басты қолданушы парақшалар бар, олар: main.jsp және books.jsp. main.jsp қолдашы қарсы алу бетінен өткеннен соң осы парақша түседі. Осы парақшадан кітапты іздеу амалын таңдаған соң, іздеу бойынша табылған кітаптар books.jsp парақшасын құрады. Бұл екі беттеде кездесетін 4 бөлшек. Осы 4 бөлшекті әр бетке қайталап жазбау үшін біз оларды арнайы Java Server Pages Fragmet атты файылда құрып, web-парақшаларды жай ғана соларды шақырамыз. Алғашында осы 4 фрагметке тоқтала кетсек.

Header.jspf web-парақша фрагменті



3.10 Сурет - web-парақшалар қалпақшасы

1-ші блокта жүйенің логотипі және қысқаша анықтамасы берілген.

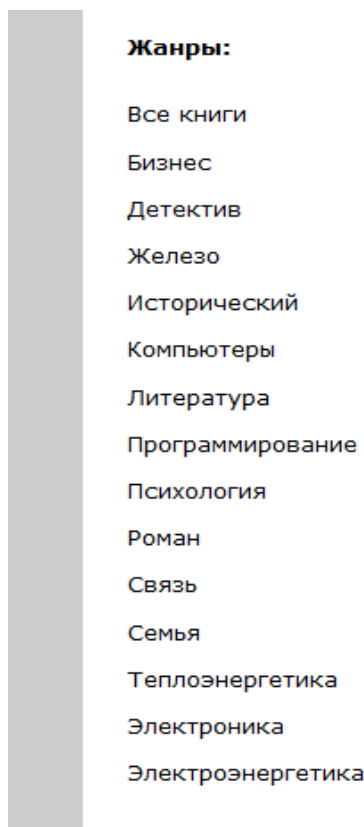
2-ші блокта жүйені қолданушының есімі мен оны қарсы алу сөздері жазылған. Және де онда артқа қайту сілтемесі бар. Қолданушы ол сілтемені басса артқа қарсы алу бетіне қайтады.

3-ші блокта қолданушыға кітаптың атын немесе авторын жазу арқылы іздеуге мүмкіндік беретін форма орналасқан.

4-ші блокта сол іздеу өрісінің қай тәсілмен іздеу керектігін бекітетін форма орналасқан. Онда қолданушы екі амалдың бірін, кітаптың аты бойынша немесе кітаптың авторы бойынша іздеуге мүмкіндік алады.

left\_menu.jspf web-парақша фрагменті

3.11 суретінде left\_menu.jspf атты web-парақшасының фрагменті берілгеню Оның негізгі қызметі, деректер қорында бар жанрлар тізімін алып, оны web-парақшада көрсету. Осы жанрлардың бірін басу арқылы қолданушы кітаптарды жанр бойынша іздей алады.



3.11 Сурет - Жанрлар тізімін шығару web-парақша фрагменті

letters.jspf web-парақша фрагменті

3.12 суретінде қолданушы алфавиттік әріп бойынша кітапты іздеуге мүмкіндік беретін web-парақшасының фрагменті берілген. Қолданушы белгілі бір әріпті басқанда books.jsp бетінде жүйеде сол әріптен басталатын барлық кітаптар шығады.

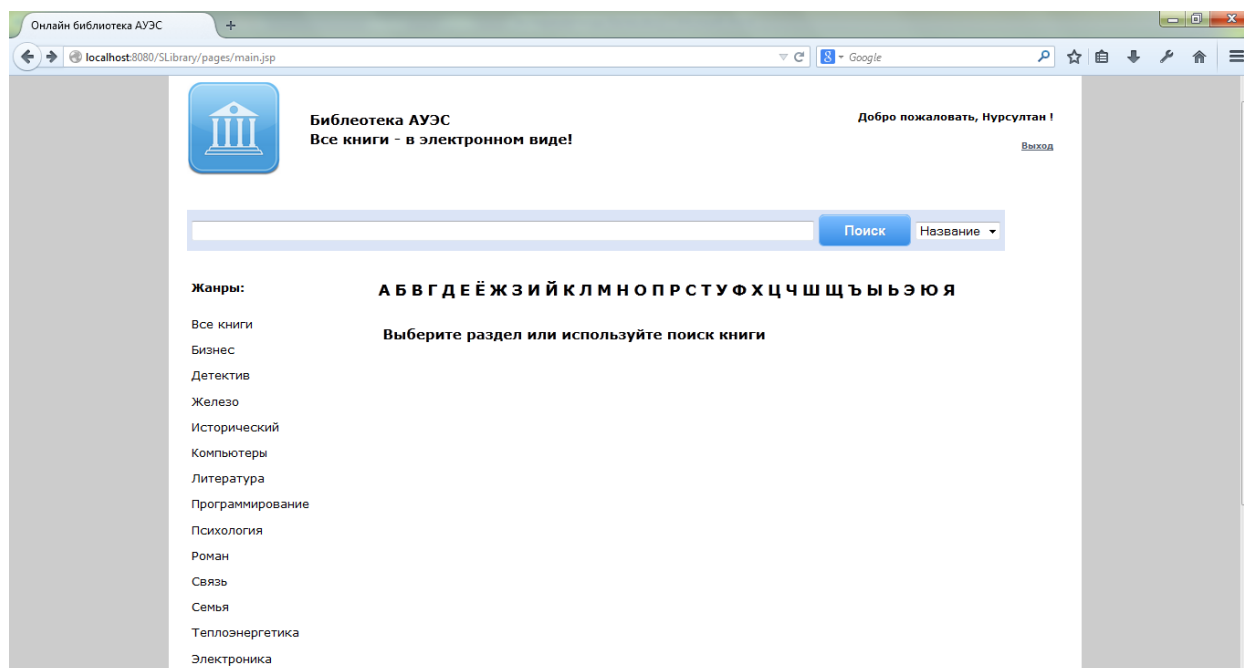
**А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я**

**Выберите раздел или используйте поиск книги**

3.12 Сурет - Алфавит тізімін шығарушы web-парақша фрагменті

Және соңғы 4-і3 web-парақша фрагменті ол жүйені құрушының аты жүнін және жүйенің құрылу уақытын шығарып тұратын footer.jspf web-парақша фрагменті.

3.13 суретінде қолданушы қарсы алу бетінен өткен соң шығатын main.jsp беті. Бұл бет жоғарыда айтылған барлық web-парақша фрагменттерін біріктіреді, және сол фргменттір беретін мүмкіндіктерді пайдаланады.



3.13 Сурет - Жүйенің негізгі бетінің көрінісі

Жағарыда index.jsp және main.jsp беттері толығымен түсіндірілді. Ендігі кезекте табылған кітаптарды шығарушы books.jsp бетіне тоқтала кетейік. Алдыңғы екі беттен бұл беттің үлкен айырмашылығы, бұл бет динамикалық түрде құрылады. Яғни бұл бетте статикалық түрде ендіне алдың ала жазылмаған. Қолданушы кітапты іздеу амалын таңдап сұраныс жасағаннан соң бұл бет бірақ құрылады. Бұл жағдайды шешу үшін жай ғана HTML тілін пайдалану жеткіліксіз болды, сондықтан біз JSP(Java Server Pages)

технологиясынның көмегіне жүгіндік. Бұл технологияның үлкен бір артықшылығы ол java кодпен HTML кодты арналстырып жаза беру болып табылады.

Яғни біз алдын ала, қолданушы кітапты іздеудің қай түрін таңдайтынын және қалай сұраныс жасайтынын білмейміз. Сондықтан біз мүмкін болатын барлық амалдардың бәрін тексеріп сонымен books.jsp бетін құруымыз керек болды. Бұл жағдайды техникалық тұрғыда шешкенімізді кішкентай код арқылы көрсете кетсек.

**books.jsp бетін жасау үшін қолданылған кодтың үзіндісі:**

```
ArrayList<Book> list = null;
    if(request.getParameter("genre_id") != null){
        long genreId = Long.valueOf(request.getParameter("genre_id"));
        list = bookList.getBooksByGenre(genreId);
    }
    else if(request.getParameter("letter") != null){
        String letter = request.getParameter("letter");
        list = bookList.getBooksByLetter(letter);
    }
    else if(request.getParameter("search_string") != null){
        String searchStr = request.getParameter("search_string");
        SearchType type = SearchType.TITLE;
        if(request.getParameter("search_option").equals("Автор")){
            type = SearchType.AUTHOR;
        }
        if(searchStr != null && !searchStr.trim().equals("")){
            list = bookList.getBooksBySearch(searchStr, type);
        }
    }
}
```

Кодтқа кішкене анықтама беріп кетелік. Мұнда біз алдымен list деген Generic коллекция құрып, оның алғашқы мәнін null деп алдық. Сосын барып әр параметр байынша шарттарды тексереміз. Бірінші шарт бізде қолданушы кітапты жанр арқылы іздеген жағдайда орындалады, яғни сол жағдайда list айнымалысы booklist объектісінің getBooksByGenre() методы қайтарған коллекцияға тең болады.[3]

Екінші шарт егер қолданушы алфавит тізіміндегі бір қаріпті басса орындалады. Сол жағдайда жағдайда list айнымалысы booklist объектісінің getBooksByLetter() методы қайтарған коллекцияға тең болады.

Үшінші шарт кішкене күрделірек. Бұл шарт қолданушы кітапты іздеу өрісініне мәтін жазу арқылы іздеген кезде орындалады. Алғашында қолданушы енгізген “search\_string” параметрі алынып searchStr айнымалысына жазылады. Сосын қолданушы іздеу амалы ретінде қандай батырманы таңдағанын білу үшін type айнымалысын алғашқы мәнін

SearchType.TITLE тең деп алып, осы шарт ішінде тағы шарт бойынша тексереміз. Егер ол шарт орындалса type айнымалысының мәні SearchType.AUTHOR болып өзгереді. type және searchStr айнымалыларын мәнін алған соң, екінші, соңғы ішкі шарт тексеріледі. Осы соңғы шарт орындалса жоғарыда көрсетілген list айнымалысы booklist объектісінің getBooksBySearch () методы қайтарған коллекцияға тең болады.

Осы шарттар біткен соң, біз list айнымалысының мәнін аламыз. Ол бізде белгілі бір кітаптар тізімін сақтайды. Сол тізімді цикл арқылы web-парақшаға жазып шығарамыз.

Найдено книг: 14

1

іКона: Стив Джобс



ISBN:978-5-91657-374-9  
Издательство:Эксмо  
Количество страниц:221  
Год издания:2012  
Автор:Джеффри Янг

[Читать](#)

Бородино

2



ISBN:978-5-9287-2385-9  
Издательство:Эксмо  
Количество страниц:16  
Год издания:2004  
Автор:Михаил Юрьевич Лермонтов

[Читать](#)

5

Бременские музыканты

3



ISBN:1254-2563  
Издательство:Литер  
Количество страниц:20  
Год издания:1953  
Автор:Братья Гримм

[Читать](#)

4

### 3.14 Суреті - Books.jsp web-парақшасының деталды түрдегі көрінісі

3.14 суретінде Books бетінің деталды түрі берілген, соны қарастыралық.

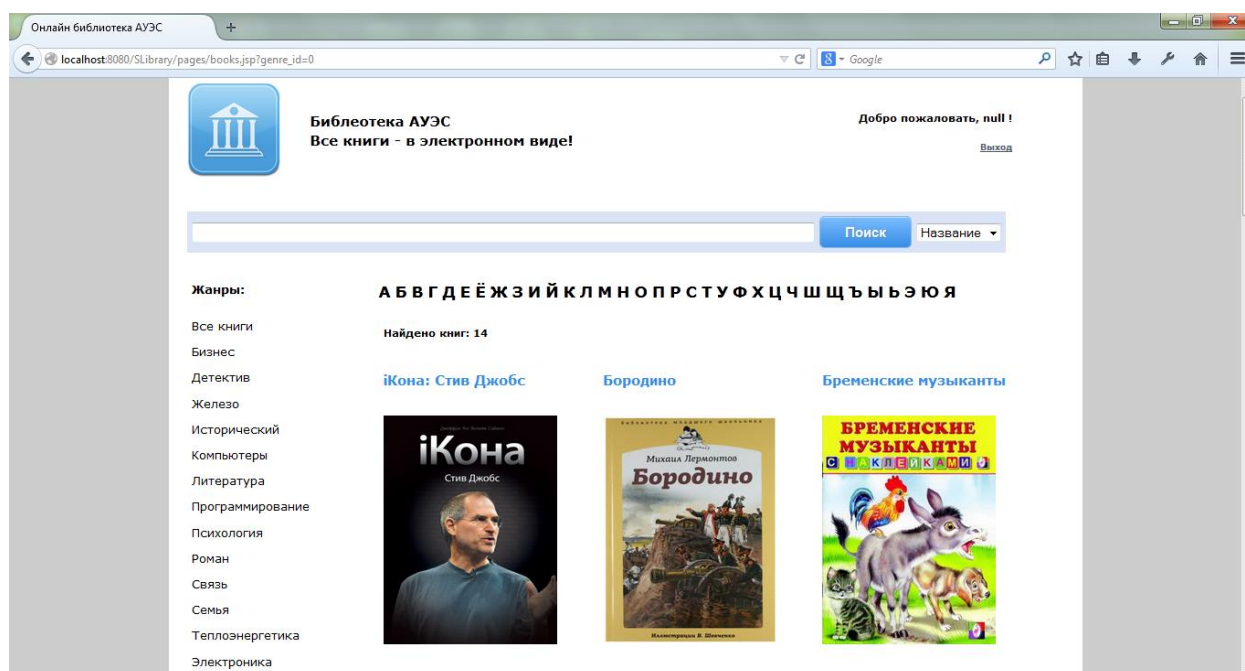
1-ші блокта белгіленген іздеу амалы бойынша қанша кітап табылғанын көрсетеді.

2-ші блокта кітаптың аты шығатын орын.

3-ші блокта кітаптың мұқабасы, сурет болып шығады.

4-ші блокта кітап жайлы толық мағлұмат берілген.

5-ші блокта кітапты оқуға сілтеме берілген. Қолданушы сол сілтемені басқанда кітап браузерде тікелей онлайн түрде ашылады.



3.15 Сурет - Books бетінің жалпы көрінісі

## 4 Экономикалық бөлім

### 4.1 Жұмыстың сипаттамасы мен қажеттілігінің негіздемесі

Елбасы Жолдауында «Ақпараттық Қазақстан-2030» бағдарламасын жасақтау туралы еліміздің Мәдениет және Ақпарат министрлігіне 2012-2014 жылдары Қазақ ұлттық электронды кітапханасының қорын ұлғайтуды қамтамасыз ету тапсырмасы берілген болатын. Бұл жоба негізінде мәдениет пен ғылым, білімнің ғасырында кітапхана ісін жандандырып, соңғы ғылыми технологиялар үрдісіне бейімделу қажеттілігіне басымдық беріліп отыр.

Қашанда нағыз білімнің қайнар көзі болып қала беретін кітапхана қызметін пайдаланушылардың сұраныстары күрделеніп, бұрынғыдан мазмұны да өзгерген.

Ғаламтор пайда болғаннан бастап қазіргі таңдағы көп қызмет түрлері осы ғаламтор арқылы жүргізілетін болды. Бұл жағдай кітапхана қызметінде айналып өтпеді. Бүгінгі күнде кезкелген студенттің (оқырманның) кітапханадағы кітаптарға қол жеткузуін тездетіп, оңайлатуға ғаламтор көмегі өте зор.

Кітапхананың ақпараттық жүйе бөлігі кітапхана оқырмандары мен студенттерге ақпараттық ресурстарды пайдалануда көмек көрсетеді және мәдени ақпаратты жоғарылатуға қызмет етеді.

Жобаның экономикалық тиімділігін анықтау бағдарламалық құралды құру процессінің ажырамас бөлігі болып табылады. Автоматтандыру құралдарының дамуы мекемені басқару облысындағы қажеттіліктерге сай жүзеге асырылуы тиіс. Сонымен қатар, аталған қажеттіліктерді тек технологиялық жаңартпаларды енгізу қажеттілігімен байланыстырып керек жоқ. Бағдарлама өңдеушісіне бизнес-бөлімшелердің нақты қажеттіліктерін түсіну керек, ал тапсырыс берушілер өңдеушілердің мүмкіндіктерін анық бағалауы тиіс. Жаңа жүйені құру жайында соңғы шешімге келу үшін мәліметтерді өңдеудің қазіргі әдістері мен енгізілуге тиіс әдістерді салыстыру қажет. Дегенмен, ақпараттық жүйені енгізудің экономикалық тиімділігі оның сипатына тәуелсіз, қойылған мақсатқа жету дәрежесімен анықталады.

### 4.2 АЖ бөлшегін жобалауға және құруға кететін шығынды есептеу

Бағдарламалық қамтаманың өзіндік құны келесі шығындардың қосындысымен есептелінеді:

- а) бағдарламалық қамтаманы құратын программистерге берілетін жалақы (яғни сол бағдарламаны жасауға тікелей қатысы бар адамдар) ( $З_{\Pi}$ );
- б) еңбекке ақы төлеу қоры ( $ЕАҚ = З_{\Pi} + З_{доп}$ );
- в) әлеуметтік салық ( $O_3$ );
- г) амортизациялық аударым ( $A$ );
- д) қосымша шығындық материалдар (қағаз, картридж, канцелярлық тауарлар және т.б.) ( $M$ );
- е) интернетке кететін шығын ( $P_{и}$ );

ж) жұмыс орнының жалға алу қаражаты, коммуналдық қызметтермен бірге ( $P_A$ );

з) және тағы басқа өндірістік шығындар ( $P_{\text{пр}}$ ).

Сонда барлық шығын мына формуламен анықталады:

$$C = EAK + O_o + A + M + P_{\text{и}} + P_A + P_{\text{л}} + P_{\text{р}} + P_{\text{пр}} \quad (4.1)$$

Алдымен бұл БҚ жасауға жұмсалатын еңбек көлемін анықтауымыз керек. БҚ жасау еңбек көлемі 4.1-ші кестеде берілген.

4.1 кесте - БҚ жасауға кететін еңбек көлемі

Кезеңнің аталуы	еңбек көлемі, адам/сағ
БҚ зерттеу	35
БҚ құруға тапсырмалар жасау	45
Код жазу	65
Сайтты құру	95
Дизайн жасау	95
Сайтты тестілеу және қолдану пікірін анықтау	35
Сайттың ережесін жасау	35
Сайтты қолдануды үйрету және тарату	10
Қосындысы	415

Берілген сайтты жасау үшін программист бір аптада 5 күн 6 сағаттан жұмыс жасайды. Сонымен 4.1 кестесінен сайтты жасауға 3,5 ай кететіндігін анықтадық.

Бағдарламалық және аппараттық құралдарға кететін шығын 4.2 кестеде көрсетілген.[13]

4.2 кесте - аппараттық және бағдарламалық шығыны

БҚ және аппараттық құрал	Құны, тенге
Lenovo G580	95000
Блокнот	Тегін таратылады
Барлығы	95000

Аппараттық және бағдарламалық құралдарға кететін қаражат ( $P_{\text{ПА}}$ ) 95000 тенге болады.



#### 4.3 кесте - амортизациялық аударым

Атауы	Алғашқы құны, тенге	Амортизация нормасы, %	Бір жылдық амортизация соммасы, тенге
Lenovo G580	95000	30%	
Барлығы			28500

Бір жылға кететін амортизациялық аударымды мына формуламен есептейміз:

$$A = \Phi_{\text{НАЧ}} * \text{НА} \quad (4.2)$$

Мұндағы, А – амортизация соммасы;

$\Phi_{\text{НАЧ}}$  – аппараттың бастапқы бағасы;

НА – амортизация нормасы.

Бір жылға кететін амортизация көлемін есептейміз:

$$A = \Phi_{\text{НАЧ}} * 0,3 = 95000 * 0,3 = 28500 \text{ тенге.}$$

1 айлық амортизация көлемі:

$$A = 28500 / 12 = 2375 \text{ тенге.}$$

3,5 айдағы амортизация көлемі

$$A = 2375 * 3,5 = 8313 \text{ тенге.}$$

4.4 кестесінде еңбек ақыны төлеуге кететін шығын көрсетілген.

#### 4.4 кесте - негізгі жалақыны есептеу

Кезеңнің аталуы	Еңбек көлемі, адам/сағ	Бір сағат құны, тенге	Жиынтығы, тенге
БҚ зерттеу	35	700	24500
БҚ құруға тапсырма жасау	45	700	31500
Код жазу	65	700	45500
Сайт құру	95	700	66500
Дизайн жасау	95	700	66500
Сайтты тестілеу және қолдану пікірін анықтау	35	400	14000
Разработка правил сайта	35	400	14000
Сайтты қолдануды үйрету және тарату	10	400	4000
Қосындысы	415	266500	

Зейнетақы Аударымы (ЗА) бүгінгі таңда еңбекке ақы төлеу қорының (ЕАҚ) 10% құрайды. Бұдан:

$$ЗА = ЕАҚ * 0,1 \quad (4.3)$$

Мұндағы, ЗА – зейнетақы аударым;  
ЕАҚ – еңбек ақы қоры.

$$ЗА = 266500 * 0,1 = 26650 \text{ тенге.}$$

Әлеуметтік салық ( $O_3$ ) ҚР-ның салық кодексына сәйкес ЗА алынып тасталынған еңбекке ақы төлеу қорының 11% құрайды.

$$O_3 = (ЕАҚ - ЗА) * 0,11 \quad (4.4)$$

Мұндағы,  $O_3$  – әлеуметтік салық;  
ЕАҚ – еңбек ақы қоры;  
ЗА – зейнетақы аударым.

$$O_3 = (266500 - 26650) * 0,11 = 26384 \text{ тенге.}$$

Қосымша шығындар 4.5 және 4.6 кестелерінде берілген.

#### 4.5 Кестесі - шығынды есептеу

Аталуы	Өлшем бірлігі	Бірлікке келетін баға, тенге	Барлығы	Ұзақтылық	Қосындысы, тенге
Энергоқолдану (ноутбук)	0,09 (кВатт*сағ)	19,59	37,35 (кВатт*сағ)	415 сағ	732
Орынды жалға алу	м <sup>2</sup>	4000	6	3,5 месеца	84000
Барлығы					84732

Интернетке кететін шығын ( $P_{и}$ ) айына 4000 тенге құрайды. Сайтты жасау үшін 3,5 ай кетеді, алайда интернетке жарты айға төлем жоқ болмағандықтан интернетке 4 айға төлейміз, сонда интернетке кететін шығын  $P_{и} = 16000$  тенге.[14]

#### 4.6 кесте - материалдық шығынды есептеу

Атауы	Саны, дана	Біреуінің бағасы, тенге.	Қосындысы
Altel 4G	1	2000	2000
USB-жинақтаушы	1	1000	1000
Қағаз	1	1500	1500
Қаламсап	3	50	150
Барлығы			4650

Барлық жалақыны дәл шығару мүмкін емес. Сондықтан қосымша шығындарды есептейміз – қосымша өндірістік шығындар ( $P_{\text{ПР}}$ ). Бұл шығындар барлық шығындардың 20% құрайды.

$$P_{\text{ПР}} = (EAK + O_a + A + M + P_{\text{и}} + P_a) * 0,2 \quad (4.5)$$

Мұндағы,  $P_{\text{ПР}}$  – қосымша өндірістік шығындар;  
 $O_a$  – әлеуметтік аударым;  
 $A$  – амортизация соммасы;  
 $M$  – қосымша материалдарға шығын;  
 $P_{\text{и}}$  – интернетке шығын;  
 $P_a$  – аренда төлемі.

$P_{\text{ПР}} = (274500 + 26384 + 8313 + 4650 + 16000 + 84732) * 0,2 = 414579 * 0,2 = 82916$  тенге.

Өзіндік құны мына формуламен анықталады:

$$C = EAK + O_a + A + M + P_{\text{и}} + P_a + P_{\text{л}} + P_{\text{р}} + P_{\text{ПР}} \quad (4.6)$$

Мұндағы,  $EAK$  – еңбек ақы қоры;  
 $O_a$  – әлеуметтік аударым;  
 $A$  – амортизация соммасы;  
 $M$  – қосымша материалдарға шығын;  
 $P_{\text{и}}$  – интернетке шығын;  
 $P_a$  – аренда төлемі;  
 $P_{\text{л}}$  – шығындар есебі;  
 $P_{\text{р}}$  – жұмысшыларға шығын;  
 $P_{\text{ПР}}$  – қосымша өндірістік шығындар.

$C = 266500 + 26384 + 8313 + 4650 + 16000 + 84732 + 82916 = 489495$  тенге.

4.1 диаграммасында БҚ – ның өзіндік құнын құрайтын барлық бөлшектерін көрсетейік және сол әр бөлшектің барлық өзіндік құндағы үлесін анықтайық.



Сурет 4.1 – Бөлшектік шығындардың БҚ-ның толық өзіндік құнына проценттік қатынасы

4.7 кестесінде БҚ – ның өзіндік құнын құрайтын бөлшектік шығындар көрсетілген.

4.7 кесте - БҚ өзіндік құнының структурасы

Бөлшектік шығын	Соммасы, тенге	Толық өзіндік құнда алатын бөлігі, %
Еңбек ақы төлеу қоры	266500	55
Әлеуметтік салық, 11 %	26384	5
Амортизациялық аударым, 30%	8313	2
Шығындық материалдар	4650	1
Интернетке кеткен шығын	16000	3
Орынды жалға алу құны, коммуналдық қызметтермен бірге	84732	17
Қосымша өндірістік шығындар, 20%	82916	17
Барлығы	489495	

#### 4.3 Берілген БҚ – ның экономикалық тиімділігі

БҚ - ның алғашқы бағасын тапқанда, қалаулы рентабельдік деңгейді берейк, біздің жағдайда ол 40%.

$$\Pi_{\Pi} = C(1+P/100) \quad (4.7)$$

Мұндағы, С – БҚ-ның өзіндік құны;  
Р – қалаулы рентабель деңгейі.

$$\Pi_{\Pi} = 491216 (1+40/100) = 687702 \text{ тенге.}$$

Жасалу құны қосымша құн салығымен (ҚҚС) бірге .

$$\Pi_P = \Pi_{\Pi} + \text{ҚҚС} \quad (4.8)$$

Мұндағы,  $\Pi_{\Pi}$  – бастапқы бағасы.

Қосымша құн салығы (ҚҚС) 2014 жылға қарағанда 12% құрайды.

$$\Pi_P = 687702 * 1,12 = 770226 \text{ тенге.}$$

Берілген ақпараттық жүйе бөлігі АЭЖБУ кітапханасындағы көркем әдебиет кітаптарына студенттердің, оқырмандардың қол жеткізуін ғаламтор желісі арқылы оңайлатады және көптеген жаңа қолайлы функциялар береді. Бұл жерде әр кітапты оқып болған соң оған баға беруге болады және сол баға басқа оқырмандарға көрінеді. Әр түрлі көркем әдебиет жанырларындағы кітаптарды тез және жылдам қол жеткізуге, және оны тікелей (онлайн) оқуға немесе жүктеп алуға болады. Бұл АЖ бөлігі АЭЖБУ студенттерінің ақпараттық мәдениетін қалыптастыруға көмектеседі.

## 5 Тіршілік қауіпсіздігі

5.1 Кітапхана АЖ-ның администраторының жұмыс орнындағы негізгі қауіпті факторларды талдау

5.1.1 Автоматтандырылған кітапхана ақпараттық жүйесінің жалпы сипаттамасы

Автоматтандырылған кітапхана ақпараттық жүйесімен жұмыс міндетті түрде компьютерде жүретіндіктен жүйе пайдаланушысы өзінің денсаулығына кері әсер ететін бірталай факторлармен бетпе-бет кездеседі. Бұл бөлімде ақпараттық жүйемен жұмыс барысында еңбек шартын қауіпсіз ету шаралары қарастырылады.



5.1 Сурет - Автоматтандырылған кітапхана ақпараттық жүйесімен жұмыс істеуге арналған дербес компьютер

Дербес компьютер (ДК) заманауи қоғамның өміріне пайдасы тиетін шексіз функцияларға толы. Дегенмен, электронды-есептеуіш техниканың негативті факторлары әлі де жойылған жоқ.

Дербес компьютерден және кеңселік перифериялық қондырғылардан келетін негативті факторларға келесілер жатады:

- электромагниттік өрістің жоғары деңгейі;
- акустикалық шу;
- ауадағы қауіпті бөлшектердің концентрациясы;
- жұмсақ рентгендік сәуле шығару;
- видеодисплейлердің қауіпті визуалды көрсеткіштері.

5.1.2 Кітапхана АЖ-ның администраторының жұмыс орнын ұйымдастыру

Қолданушының жұмыс орны орналасқан бөлменің (4.3-сурет) сипаттамасы келесідей: мекеме қабаты: 5; габариттік өлшемдері: ені – 4 м, ұзындығы – 7 м, биіктігі – 3,5 м; терезе саны: 2; ДК-мен жабдықталған жұмыс орны саны: 6.

Жұмыс үстелінің құрылымы қондырғылар мен оргтехниканың

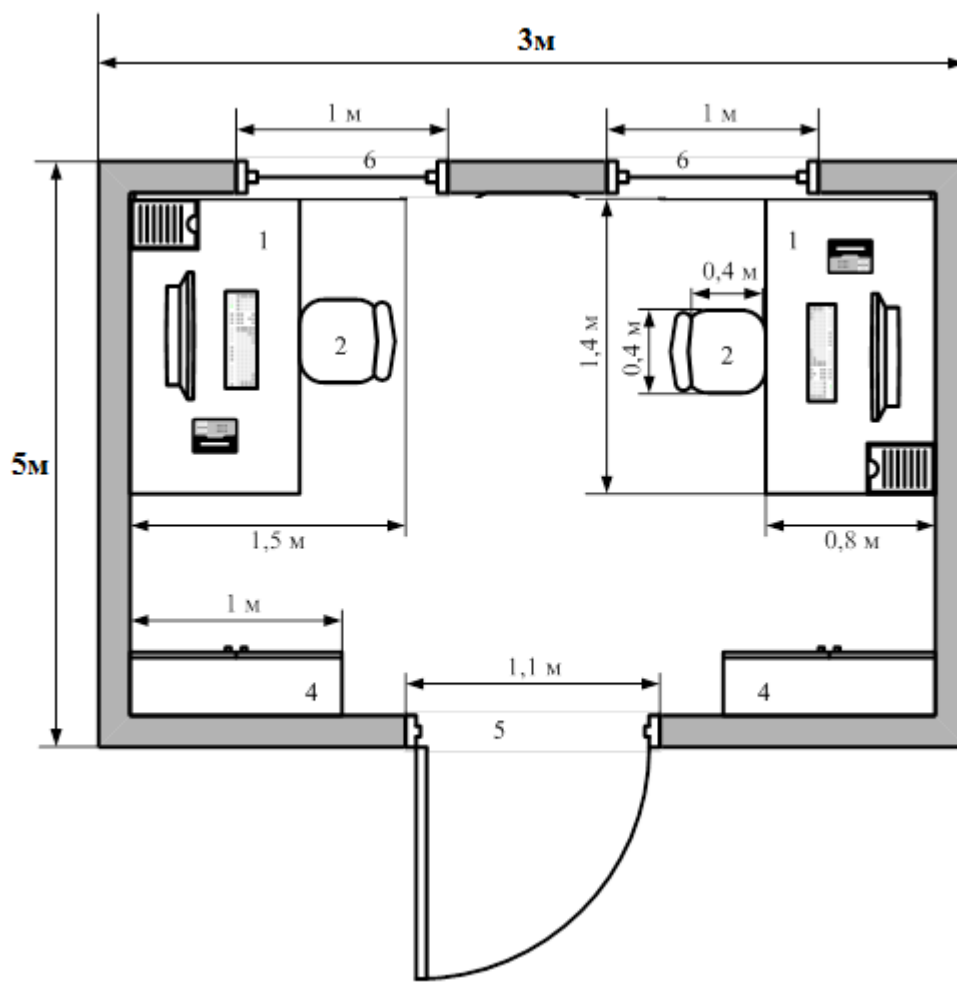
Technical drawing illustrating the ergonomic dimensions for a chair and desk setup. The drawing shows a side profile of a person sitting at a desk, with various dimensions and angles indicated for optimal posture and comfort.

Key dimensions and angles shown:

- Chair Backrest:**
  - Angle:  $15^\circ$
  - Height:  $40-70$
  - Angle:  $90^\circ$
- Chair Seat:**
  - Height:  $48-50$
- Chair Base:**
  - Height:  $42-50$
- Chair Dimensions:**
  - Width:  $68-84$
  - Depth:  $50-68$
  - Height:  $48$
  - Height:  $60$
  - Height:  $90$
- Desk:**
  - Height:  $80$
  - Width:  $40 \times 30 \times 15$

жұмыс үстелінің биіктігі 68-84 см аралығында; оптималды биіктігі Жүйемен жұмыс барысында қолданылатын үстелдер биіктігі нормаларына сәйкес келеді.

Жұмыс үстелінің модульді өлшемдері: ұзындығы – 140 см, ені – 80 см (4.3 сурет).



5.3 сурет. Автоматтандырылған кітапхана ақпараттық жүйесін пайдаланушыларының қызметтік бөлмесі

Жұмыс үстелінің конструкциясы оператордың рационалды жұмыс қалпын қамтамасыз етеді.

Жұмыс орындығы жұмыстың сипаты мен ұзақтығын ескере отырып таңдалады.

Орындық көтерілмелі-айналмалы болып табылады және биіктігі бойынша, орындық арқасының иілу бұрыштары бойынша басқарылады:

- орындықтың ені мен тереңдігі – 40 см, иілу бұрышын басқару мүмкіндігі бар: алдыға –  $15^{\circ}$ -қа дейін, артқа –  $5^{\circ}$ -қа дейін;

- орындық арқасының биіктігі – 30 см, ені – 38 см, қисықтық радиусы – 40 см;

- арқасының вертикалды кеңістікте иілу бұрышы  $0 \pm 30^{\circ}$  аралығында;

- орындық арқасынан оның алдыңғы шегіне дейінгі арақашықтық 26-40 см аралығында басқарылады.

Орындықтың стационарлы немесе алынып салынатын шынтак тірегіштері бар:

- ұзындығы – 25 см және ені – 5-7 см;



- биіктігі – 23 см;
- шынтақ тірегіштердің бір ірінен арақашықтығы – 35-50 см.

Клавиатура үстел шетінен 10-30 см қашықтықта орналасады.

Жүйе пайдаланушысының жұмыс орны ыңғайлы болып табылады, оны екі позицияда жұмыс істеуге келістіріп алуға болады. Орындықтың, монитор мен клавиатураның орналасуы әр жағдайда атқарылатын жұмысқа және пайдаланушы әдетіне сәйкес келеді. Операторға көбінесе ыңғайлы қалып – вертикалды, кішкене шалқайған қалып.

Үлкен көлемді жұмыстарды отыру қалпында атқару кезінде орындық сәл алдыға иіледі. Оператор денесінің қалпы көз бағытына сәйкес келеді [31].

### 5.1.3 Көз шаршауының алдын-алу

Психофизиологиялық зерттеулердің нәтижелері бойынша компьютермен жұмыс кезінде төрт сағаттан кейін көздің шаршауы басталады, ал физиологиялық көрсеткіштердің объективті өзгеруі екі сағаттан кейін басталады. Бұндай жағдайды болдырмау үшін екі сағат уақыттан кейін үзіліс жасалады. Үзіліс кезінде көз гимнастикасы, массаж және физикалық жаттығулар жасалады.

Көз шаршаудың алдын-алу үшін жұмысқа кірісу алдында және жұмыс барысында бейнелердің жарықтығы мен контрастылығы ең төмен оптималды деңгейге қойылады. Монитор экранына көп қадала қарауға болмайды, периодты түрде (әр 3-5 минут сайын) монитор экранынан бөлмедегі ең алыста тұрған затқа немесе терезеде көрінетін алыс объектіге назарды аударып тұру керек (3-5 секундқа).

Көзге қысым түспес үшін және көру қабілеттілігі нашарламас үшін келесідей жаттығулар кешені жасалады:

- көзді монитормдың бір бұрышынан келесісіне жиі ауыстыру;
- көзді периодты түрде алыс затқа 2-3 секунд ауыстырып отыру, кейіннен мұрынның ұшына 3-5 секундқа ауыстыру. Жаттығуды 6-8 рет қайталайды;
- көзді 1-2 минут барысында тез жыпылықтату;
- көзді жұмып, саусақ қимылымен айналмалы қимылды массаж жасау: көздің ішкі қабағынан сыртына, сыртқы қабағынан ішіне қарай. Ұзақтығы 1 минут;
- әр қолдың үш саусағымен үстіңгі қабақты әлсіз басып, 1-2 секунд өткеннен кейін жіберу. 3-4 рет қайталау керек;
- жоғары қарап, көзбен сағат тілі бойынша айналмалы қимылдар жасау, сағат тіліне қарсы айналмалы қимылдар жасау. 3-4 рет қайталау керек;
- Көзді жұмулы күйінде жоғары қаратып, төмен түсіру, оңға және солға қарату. 3-4 рет қайталау керек.

8 сағаттық жұмыс уақыты және ДК-де жұмыс кезінде регламенттелген үзілістер қойылған: жұмыс басталғаннан 2 сағат уақыттан кейін және түскі

үзіліс аяқталғаннан 2 сағат өткен кезде әрқайсысы 15 минут. Осы үзілістер кезінде жоғары жазылған жаттығулар орындалады.

ДК-дің белсенді пайдаланушылары жылына 2 рет окулист-дәрігерде тексеру өтеді [15].

#### 5.1.4 Жұмыс орнындағы электр қауіпсіздігі

Жалпы электр қауіпсіздігіне байланысты өндіріс 3 топқа бөлінеді:

1. электр қауіпсіздігі жоқ бөлмелер;
2. қауіпсіздігі жоғары бөлмелер;
3. қауіпсіздігі өте жоғары бөлмелер.

Солардың ішінде:

1- ші топқа жататын бөлмелер –  $15^{\circ}\text{C}$ -тан  $20^{\circ}\text{C}$ -қа дейін, едендері тоқ өткізбейтін, және тоқ өткізетін темір бұйымдары жоқ бөлмелер, бұған жататындар - әкімшілік бөлімшелері, жылы үйлер, тұрмыстық үйлер;

2- ші топқа жататын бөлмелер – құрғақ, салқын, шаң тозаңы көп бөлмелер. Бұл бөлмелерге жататындар, бұған жататындар - насосы станциялар, транспортты станциялар, электр станциялары;

3- ші топқа жататын бөлмелер – бөлмелерінің ішіш ылғалды, қабырғаларына, есік –терезелеріне шаң-тозаң көп тұратын бөлмелер. Бұған жататын бөлмелер – просеивательный отделение, бесторный хранение муки, бродительный, помещение пивзавода.

Өндірістегі электр қауіпсіздігін болдырмау шаралары мыналар:

- 1) изоляция;
- 2) тоқ жүретін тетіктерді қоршау;
- 3) тоқты жерге енгізу;
- 4) тоқты нөлге жалғау;
- 5) тізбекті ажырату.

Токты жерге енгізудің 2 түрі бар:

- 1) жабдықты контурлы жерге енгізу;
- 2) жерге енгізуді сыртқа ұйымдастыру (выносное заземление).

Қорғаныс бұйымдары

Электр қауіпсіздіктерінде қорғаныс бұйымдары 2 түрге бөлінеді:

- 1) жалпы қорғаныс бұйымдары (коллективное вещество);
- 2) жеке қорғаныс бұйымдары (индивидуальное вещество).

1) жалпы қорғаныс бұйымдарына жататындар: қорғаныс штанглері; электр тогын, кернеуін өлшейтін приборлар, қолғаптар, төсеніштер және қоршау бұйымдары;

2) жеке қорғаныс бұйымдарына жататындар: очки, каска, противогаз, белбеу.

Жарақат алған адамға жедел жәрдем көрсету жолдары – тоқтан жарақат алған адамды ажырату.

Оқшауланған қорғаныс құралдары адамды кернеу астындағы электр қондырғыларынан және жерден оқшаулайды.

Барлық оқшауланған қорғаныс құралдары негізгі және қосымша болып бөлінеді. Негізгі оқшауланған құралдарын қолданып адам кернеуге қосылған тоқ өткізгіш бөліктерді ұстауға болады. Сондықтан, бұл заттардың оқшаулағышы электр құралдарының жұмыс кернеуін сенімді ұстауы керек.

Қосымша қорғаныс құралдары негізгі қорғаныс құралдарымен бірге қолданылады, сонымен қатар қадам кернеуінен қорғану шараларына жатады.

1000 В қа дейінгі және жоғары кернеулерге арналған қондырғыларда әр түрлі қорғаныс құралдары қолданылады.

Кернеуі 1000 В электр құрылғыларды оқшаулайтын негізгі оқшаулағыштарға оқшаулағыш штангілер, оқшаулағыш және тоқ өлшегіш қысқыштар, кернеу көрсеткіштері, оқшаулағыш құрылғылар және жөндеу жұмыстарына арналған құралдар (оқшаулағыш сатылар, аудандар, габариттер және т.б.) жатады.

Қосымша оқшаулағыш құралдарға диэлектрлік қолғаптар, диэлектрлік резеңке кілемшелері және қосалқы қойғыштар жатады.

Диэлектрлік қолғаптар, галоштар және кілемшелерді арнайы маркалы резеңкелерден жасайды, олардың электрлік беріктігі жоғары болуы керек.

Диэлектрлік қолғаптарды екі түрде (1000 В – қа дейінгі және 1000 В - тан жоғары құрылғыларға) жасап шығарады және олар бірнеше өлшемдерде болады. Жұмыс алдында қолғаптарды тексереді (тесіктердің жоқтығына және т.б.).

#### 5.1.5 Өрт қауіпсіздігін ұйымдастыру

Өрт қауіпсіздік шаралары бірегей үкіметтік қаулылар, ережелер және нормалар негізінде жүргізіледі. Соларға сәйкес дербес компьютерлерді қолдану арқылы жұмыс жүргізілетін бөлмелерде өрт қауіпсіздігін міндетті түрде сақтау қажет.

ДК пайдаланушысының жұмыс орнындағы тұтану көздері болып өткізгіш сымдар, ДК-дің электронды сұлбалары, электр қоректену құрылғылары табылады. Сол себепті осындай бөлмелердің 100 шаршы метр ауданында бір көмірқышқыл типті өрт сөндіргіш орналасады.

Мекеменің барлық қызметкерлері қызметке өрт қарсы сақтық нұсқауды өткен жағдайда ғана жіберіледі. Мекеме басшылары немесе жеке кәсіпкерлер тиісті өрт қауіпсіздік ережелерін орындайтын адамдарды сәйкес қызметке тағайындауға құқылы.

Мемлекеттік органдар өз күзiреттілігі шегінде ведомстваға қарасты мекемелерде өрт қауіпсіздік шараларын жүзеге асырады, өрт сөндіру күзетіне қажетті көмек береді.

Барлық өндірістік, әкімшіліктік, қоймалық және қосалқы жайларда көрінетін жерде өрт сөндіру күзетін шақыру телефоны номері жазылған тақтайша ілініп тұрады.

Әр мекемеде бұйрық құжатпен темекі тартатын орындар анықталып, арнайы жабдықталған, өрт кезінде және жұмыс күні аяқталған кезде электр қондырғыларын тоқтан ажырату тәртібі және өрт байқалған кездегі қызметкерлердің іс-әрекеттері анықталған.

Өрт кезінде компьютерлік техниканы өрттен сөндіру үшін газдық және көмірқышқылды өрт сөндіргіштер қолдану керек. Олардың артықшылығы – өртті сөндірудің жоғары деңгейі мен электронды қондырғылардың сақталуы [17].

## 5.2 Есептік бөлім

### 5.2.1 Кітапхана АЖ-нің администраторының жұмыс орнын жасанды жарықтандыруды есептеу

АЖ қолданушысының жұмыс орнын рационалды жарықтандыру ең маңызды мақсаттардың бірі болып табылады, өйткені ол қолданушының жұмыс өнімділігіне, профессионалды ауру табуына және кездейсоқ жарақат алуына тікелей байланысты. Жарықтандырудың дұрыс ұйымдастырылуы АЖ қолданушысына қолайлы жұмыс істеу жағдайын жасайды, оның еңбекке қабілеттілігін және өнімділігін арттырады. Жұмыс орнының жарықтандырылуы кітап оқушы студенттің немесе кітапхана қызметкерінің көру мүшелерінің көп күш қаут құртпай ақ барлығын жақсы көретіндей болуы қажет. Көру мүшелерінің талуы бірнеше факторларға байланысты:

- а) жарықтандырудың нашарлығы;
- б) жарықтандырудың өте күшті болуы;
- в) жарықтың дұрыс бағытталмауы.

Жарықтандырудың нашарлығы көру мүшелерінің талуына, зейіннің нашарлауына, адамның тез шаршауына алып келеді. Жарықтандырудың өте күшті болуы көздің соқыр болуына, қозуына, шаншып ауруына алып келеді. Жұмыс орнында жарықтың дұрыс бағытталмауы қатты көлеңкелердің, көз алдында ақ дақтардың пайда болуына немесе жұмысшыны бағыттан адастыруға алып келуі мүмкін. Осы айтылған жағдайлар кездейсоқ зақым алуға немесе профессионалды ауруға алып келуі мүмкін, сондықан жарықтандыруды дұрыс есептеу өте маңызды.

Жұмыс орынды жарықтандыруды есептеу сол орынды жарықтандырытын жүйені, шамдардың санын, типін таңдауға және олардың орналасуына байланысты. Осыған байланысты жасанды жарықтандыру параметрлері есептелінеді. Негізінен жасанды жарықтандыру екі түрлі электр шамдарынан жасалынады: қыздыру шамдары және люминесценцияға негізделген шамдар. Біз люминесценцияға негізделген шамдарды қолданатын боламыз, өйткені олардың қыздыру шамдарына қарағанда бірнеше артықшылығы бар[25]:

- спектралды құрамына байланысты олар табиғи, күндік жарыққа ұқсас;

- жоғары ПӘК –ке ие (қыздыру шаманың ПӘК –нен, 1,5-2 есе үлкен);
- жарық шашуы жоғары (қыздыру шамына қарағанда, 3-4 есе үлкен);
- жұмыс істеу уақыты жоғары.

Бұл есептеу ауданы  $15\text{ м}^2$  болатын бөлмеге жасалды, оның ені 5м, биіктігі - 3 м. Жарықтық ағын әдісін қолданамыз [24]. Қолданатын шам санын анықтау үшін, бетке түсетін жарық ағынын мына формуламен анықтаймыз:

$$F = (E * K * S * 2) / n \quad (5.1)$$

мұнда,  $F$  - есептелініп отырғын жарықтық ағын, Лм;

$E$  - минималды нормалық жарықтылық, Лк (кесте бойынша анықталыды). Администратордың жұмысын, кесте бойынша, дәл жұмыстар қатарына жатқызсақ болады, сондықтан минималды жарықтылық мына мәнге тең болады:  $E = 300\text{ Лк}$ ;

$S$  - жарықтандыратын бөлме ауданы (біздің жағдайымызда  $S = 15\text{ м}^2$ );

$Z$  - орта жарықтандырудың минималды жарықтандыруға қатынасы (негізінен 1,1,1,2 тең деп алынады, біз  $Z = 1,1$  деп алайық);

$K$  - артықшылық коэффициенті, шамды қолдану барысында, шамға кір тұрып жарық ағынының азайуы алынады (оның мәні бөлменің типіне және ондағы жасалытын жұмыс характеристикасына байланысты, біздің жағдайда  $K = 1,5$ );

$n$  – қолдану коэффициенті, (есептелініп отырғын бетке түсетін, жарық ағынының барлық шам тарататын жарық ағынының қосындысына қатынасын білдіреді және бірлік бөлшегімен есептелінеді; шам характеристикасына, бөлменің көлеміне, және ( $P_K$ ) қабырғаның және ( $P_T$ ) төбенің сәуле шағылту коэффициентін анықтайтын, қабырғаның және төбенің түсіне байланысты), коэффициенттер мәні:  $P_K=40\%$ ,  $P_T=60\%$ .

$n$  мәнін әртүрлі шам қолдану коэффициентінің кестесінен аламыз. Ол үшін бөлме индексін мына формула бойынша анықтаймыз:

$$I = S / h * (A + B) \quad (5.2)$$

мұнда,  $S$  – бөлменің ауданы,  $S = 15\text{ м}^2$ ;

$h$  – есептелінген іліп қою биіктігі,  $h = 2.92\text{ м}$ ;

$A$  – бөлме ені,  $A = 3\text{ м}$ ;

$B$  – бөлме ұзындығы,  $B = 5\text{ м}$ .

Мәндерді орнына қойып мына мәнді аламыз:

$$I = 15 / 2,92 * (3 + 5) = 0,64$$

Бөлме индексі  $I$  мәні бойынша, 7 – ші кестеден [24]  $n = 0,22$  мәнін аламыз.

Барлық мәндерді жырық ағынын анықтау формуласының орнына қойамыз:

$$(300 * 1,5 * 15 * 1,1) / 0,22 = 33750 \text{ Лм}$$

Бөлмені жарықтандыруға люминесценцияға негізделген ЛБ40-1 лампасын таңдадық, оның жарық ағыны  $F = 4320 \text{ Лк}$ .

Мына формула бойынша қажетті лампалар санын есептейміз:

$$N = F / F_{\text{л}} \quad (5.3)$$

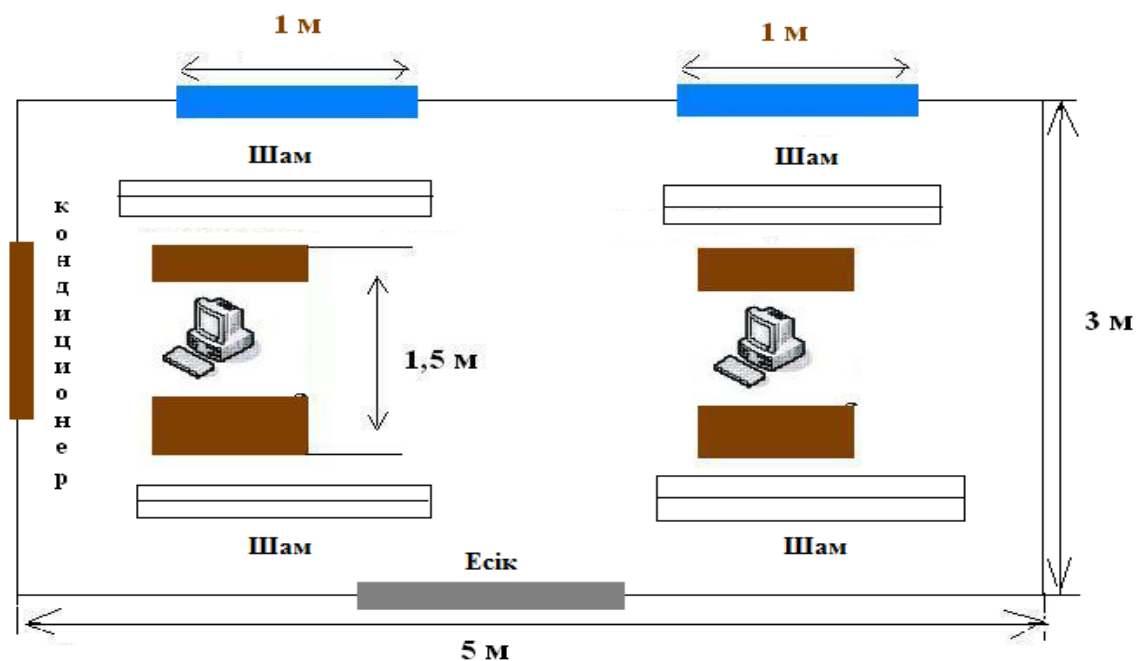
Мұнда,  $N$  – есптелініп отырған шам саны;

$F$  – жарықтық ағын,  $F = 33750 \text{ Лм}$ ;

$F_{\text{л}}$  – шамның жарықтық ағыны,  $F_{\text{л}} = 4320 \text{ Лм}$ .

$$N = 33750 / 4320 = 8 \text{ дана.}$$

Жарықтандыру құрылғысын таңдау барысында ОД типті люминесценцияға негізделген шам алынд. Әр шамда екі лампа бар.



5.4 Сурет – Шамның бөлмедегі орны

### 5.2.2 Табиғи жарықтандыруды есептеу

Адамдардың тұрақты мекендеуі бар тұрғылықты орындар табиғи жарық көзімен жарықтанады.

Табиғи жарықтандыру көлденең, үстіңгі және аралас (көлденең және үстіңгі) болып бөлінеді.

Бөлменің ішкі бет жақтарының шағылысу коэффициентінің орташа алынған есептік мәнін қоғамдық және тұрғын бөлмелер үшін 0,5-ке тең деп және өндірістік орындарда 0,4-ке тең деп алу керек [26].

Табиғи жарықтандыруды есептеу барысында бөлме терезелерінің қажетті өлшемдері мен орналасуы анықталады.

Терезе ауданы келесі формула бойынша анықталады:

$$S_0 = (S_{\Pi} * e_N * \eta_0 * K_3 * K_{3Д}) / (100 * \tau_0 * r_1) \quad (5.4)$$

Мұндағы,  $S_0$  - көлденең жарықтандыру кезіндегі жарық терезелер ауданы,  $m^2$ ;

$S_{\Pi}$  - бөлме еденінің ауданы,  $m^2$ ;

$\eta_0$  - терезелердің жарықтық сипаттамасы, 3.2-кесте бойынша [26];

$K_3$  - қор коэффициенті (коэффициент запаса), 3.11-кесте бойынша [26];

$e_N$  - табиғи жарықтандыру коэффициентінің (ТЖК) нормаланатын мәні;

$r_1$  - көлденең жарықтану кезіндегі ТЖК артуын ескеретін коэффициент, 3.9-кесте бойынша [26];

$K_{3Д}$  - қарсы тұрған мекемелермен терезелердің қараңғылануын ескеретін коэффициент, 3.8-кесте бойынша [26];

$\tau_0$  - жарық өткізудің жалпы коэффициенті, келесі формула бойынша анықталады.

$$\tau_0 = \tau_1 * \tau_2 * \tau_3 * \tau_4 * \tau_5 \quad (5.5)$$

Мұндағы,  $\tau_1$  - материалдың жарық өткізу коэффициенті, 3.3-кесте бойынша [6];

$\tau_2$  - терезелердің түптеулерінде жарықтың жұтылуын ескеретін коэффициент, 3.4-кесте бойынша [26];

$\tau_3$  - терезе конструкциясында жарықтың жұтылуын ескеретін коэффициент, көлденең жарықтану кезінде 1-ге тең деп қабылдайды;

$\tau_4$  - күннен қорғайтын құрылғыларда жарықтың жұтылуын ескеретін коэффициент, 3.6-кесте бойынша [26];

$\tau_5$  - шамдар астында орнатылатын қорғаныс торшаларында жарықтың жұтылуын ескеретін коэффициент, 0,9-ға тең деп қабылдайды.

$$S_{\Pi} = L * B = 15 m^2 \quad (5.6)$$

$$e_N = m * e_H \quad (5.7)$$

Мұндағы,  $m$  - жарық климатының коэффициенті, 3.1-кесте бойынша [26] (терезелер сыртқы қабырғаларда, бағыт О; Алматы қаласы IV);

$e_n$  - ТЖК мәні, 3.12-кесте бойынша [26] (көз жұмысының разряды IVб, көлденең жарықтану).

$\eta_0$  коэффициентін таңдаймыз:

$\eta_0$  - терезелердің жарықтық қасиеті, 3.2-кесте бойынша анықталады [26] (бөлме ұзындығының биіктігіне қатынасы  $L/l=5/3=1,7$ ; жұмыс үстелінің биіктігі  $h_p=0,725$  м).

Кестелік мән таңдаймыз:  $\eta_0 = 10,5$ .

$r_1$  - коэффициентін таңдаймыз.

Жұмыс үстел деңгейі:

$$p_z = p_{\text{ш}} + p_{\text{но}} - p_{\text{рп}} = 0,725 + 1 - 1 = 0,725 \text{ м.}$$

Бөлме биіктігінің жұмыс үстел деңгейіне қатынасы:

$$l/h_p = 3/0,725 = 4,14.$$

Есептік нүкте қашықтығының бөлме биіктігіне қатынасы:

$$l/B = 3/5 = 0,6.$$

Бөлме жарықтануы – бір жақты.

Бөлме төбесінің, қабырғаларының және еденінің шағылысу коэффициенті – 0,6.

Бөлме ұзындығының биіктігіне қатынасы:

$$L/l = 3,4;$$

$$r_1 = 2,5;$$

$$K_3 = 1,2;$$

$$(P/H_{3д} = 15/3 = 3);$$

$$K_{3д} = 1;$$

$$\tau_0 = \tau_1 + \tau_2 + \tau_3 + \tau_4 + \tau_5 = 0,9 * 0,9 * 0,8 * 1 * 0,9 = 5,583$$

мұндағы,  $\tau_1 = 0,9$  - бір қабатты әйнекті терезе;

$\tau_2 = 0,9$  - бір қабатты қатаң болат түптеулер;

$\tau_3 = 0,8$  - ағаш аркалар мен формалар;

$\tau_4 = 1$  - реттелетін жалюздер мен перделер;

$\tau_5 = 0,9$  - қорғаныс торшасында жарықтың жұтылуы.

$$S_0 = (S_n * e_n * \eta_0 * K_3 * K_{3д}) / (100 * \tau_0 * r_1) = (15 * 0,975 * 10,5 * 1,2 * 1) / (100 * 5,583 * 1,26) = 2,51 \text{ м}^2$$

Бір терезенің ауданын табамыз, себебі бөлме бір жақты екі терезе көмегімен жарықтанады:



$$S_{OK} = S_0 / 2 = 2,51 / 2 = 1,225 \text{ м}^2$$

Терезенің қажетті енін анықтаймыз:

$$L_{OK} = S_{OK} / h_{OK} = 1,225 / 2,5 = 0,502 \text{ м}$$

Қорытынды: ДК пайдаланушылар бөлмесінің қажетті жарықтануын қамтамасыз ету үшін ені 0,502 метр, ал биіктігі 2,5 метр екі терезе орнатамыз.

## Қорытынды

Кітапханалар көптеген салт-дәстүрлердің сақтаушылары болып саналады және әр түрлі халықтардың мен қоғамдық қарым-қатынастар бойынша жазба дәстүрлеріні жинап, сақтай отырып қоғамның рухани өндіру жүйенің маңызды элементтерінің бірі болып табылады. Сондықтан, кітапханалардың өлкетанулық қызметтерінің бастапқылықты бағыттардың бірі жергілікті тарихты танытуға мен өз көптеген оқырмандарын өз өлкенің мен жердің өткендігі туралы кешенді ең анық ақпаратқа қол жеткіздіру.

Қазіргі замандағы кітапханалар өз қызметтерінде тек ресми электрондық баспаларды ғана қолданбайды, сонымен бірге өз жеке қорларын да жасауға қабілеттері бар. Әриіне, бұл көп еңбекті қажет ететін процесс, алайда біз оны өзіміз үшін жасаймыз – ол біздің алдымызда қойған мақсатымызға мен іс-шарамызға сәйкес келеді.

Соңғы он жылдықта кітапханалық қызмет көрсетуде компьютерлерді, сандық жүйелер мен Ғаламторды қолдануымен байланысты маңызды өзгерістер пайда болды. Ешқашан да кітапхана қорлары қазіргі күнде сияқты ашық болған емес. Жаңа технологиялар сұранымды жіберу, ақпаратты алу мен жеткізу процесстерін жеңілдетеді. Дегенмен, электрондық қорларын пайдаланудың өсуімен кітапханалардың келімі төмендеп барады. Қазіргі ұрпақ компьютерлермен бірге бой жетіп барады және виртуальдық ақиқатта өздерін еркін сезеді. Ол үшін Web – тек ақпарат дерекнамасы ғана емес, сонымен бірге араласу, істестік пен өз ойын білдіру орыны болып табылады. Бұл кітапханаларды қызмет көрсету үшін инновациялық шешім табуға және оқырман үшін қолайлы тәсілдерді пайдаланып қызмет көрсетуге мәжбүр етеді.

Соңғы жылдарда кітапхана қызмет көрсетуіне енгізілген жаңа сервистердің көбі Web технологиялармен байланысты. Web терминнің пайда болуы мен қолдануы жаңа Ғаламтордың негізгі айырмашылықтарын сипаттаған 2005 жылда “Web деген не?” мақаланы жазған Тима О’ Рейли есімімен байланысты.

Осы дипломдық жобада мен осындай технологияны қолдана отырып АЭЖБУ кітапханасы үшін, оқырмандардың, студенттердің кітапқа қызығушылығын арттыру, қол жетімділігін оңайлату мақсатында осы Web жобаны жасадым. Бұл жоба шынайы жұмыс істейтін болса студеттерге, кітапхана қызметкерлеріне өте үлкен жеңілдік болар еді деп ойлайм. Және де осы жобаны біздегі қазіргі қолданыстағы жәлектронды кітапханамен біріктірсек болар еді.

Қазіргі замандағы әрбір кітапхананың қызмет атқаруы технологиялық инновацияларды қабылдау қабілетіне байланысты. Олардың дамуына арналған мүмкіндіктер кітапханада жаңа ақпараттық технологиялар спектрін жасайды.

## Қолданылған әдебиеттер тізімі

1. А.Г.Ни, А.К. Адильбекова, С.А. Адилгажинова. Ақпараттық жүйелер мамандығының студенттеріне дипломдық жұмыстарды (жобаларды) орындауға арналған әдістемелік нұсқау. – Алматы: АЭЖБУ, 2012. – 39 б.
3. Хорстманн, Кей С, Корнелл, Гари Java 2. Библиотека профессионала, том 1. Основы. 8-е издание.: Пер. с англ. - М.: ООО "И.Д. Вильямс", 2012. - 816 с.: ил. - Парал. тит. Англ.
4. GlassFish Server Open Source Edition Administration Guide, Release 4.0.
5. Specification: JSR-342 Java Platform, Enterprise Edition 7 Specification ("Specification").
6. The Java EE 7 Tutorial, Release 7 for Java EE Platform.
7. Oracle сайты <http://docs.oracle.com/javaee/7/tutorial/doc/home.htm>
8. Wikipedia сайты <http://kk.wikipedia.org/>
9. АЭЖБУ сайты <http://aipet.kz>
10. MySQL сайты <http://dev.mysql.com/doc/>
11. GlassFish сайты <https://glassfish.java.net/documentation.html>
12. Сюттюренко О. В Электронные информационные ресурсы: проблемы создания и использования // Научный сервис в сети Интернет: Тезисы докладов Всерос. Науч. Конф. 20-25 сент. 1999г., Новороссийск.- М.: Изд-во МГУ , 1999.
13. Экономическая информатика: Учебник / Под ред. В.П. Косарева и Л.В. Еремина. М.: Финансы и статистика, 2008.
14. Г. С. Вечканов. Экономическая теория. Учебник. 2008 год. 210 стр.
15. Безопасность жизнедеятельности. /Под ред. Н.А. Белова - М.: Знание, 2000 - 364с.
16. ГОСТ 12.1.005-88 ССБТ "Воздух рабочей зоны, общие санитарно-гигиенические требования".
17. ГОСТ РК 12.1.004-91 «ССБТ. Пожарная безопасность. Общие требования».
18. ГОСТ 12.1.030-81 «ССБТ. Электробезопасность. Защитное заземление. Зануление».
19. Дубовцев В.А. Безопасность жизнедеятельности. / Учеб. пособие для дипломников. - Киров: изд. КирПИ, 1992.
20. Белоног В.С., Калиева А.Б. Санитарные правила и нормы санпин. – Алматы, 1996.
21. ГОСТ РК 12.1.004-91 «ССБТ. Пожарная безопасность. Общие требования».
22. Самгин Э.Б. Освещение рабочих мест. – М.: МИРЭА, 1989. – 186с.
23. Справочная книга для проектирования электрического освещения. / Под ред. Г.Б. Кнорринга. – Л.: Энергия, 1976.
24. Борьба с шумом на производстве: Справочник / Е.Я. Юдин, Л.А. Борисов; Под общ. ред. Е.Я. Юдина – М.: Машиностроение, 1985. – 400с., ил.

## Қосымша А

### Бағдарлама листингі

Барлық беттерде болатын /WebPages/Web-INF/jspf папкасында орналасқан Web-парақшалар фрагменттері:

footer.jspf

```
<% @ page pageEncoding="UTF-8" %>
<div style="clear: both;">&nbsp;</div>
<div class="footer">
    Құраушы: Сергазыев Нурсултан, 2014 ж
</div>
</div><!-- end .container -->
```

```
</body>
</html>
```

header.jspf

```
<% @ page pageEncoding="UTF-8" %>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>АЭЖБУ онлайн кітапханасы</title>
    <link href="../css/style_main.css" rel="stylesheet" type="text/css">
</head>
<body>

    <%
        request.setCharacterEncoding("UTF-8");
        String searchString;
        if(request.getParameter("search_string") != null){
            searchString = request.getParameter("search_string");
            session.setAttribute("searchString", searchString);
        }
        else{
            searchString = "";
        }

        if(request.getParameter("username") != null){
            session.setAttribute("username", request.getParameter("username"));
        }
    %>
```

## Қосымша А (жалғасы)

```
%>

<div class="container"><!-- start .container-->

    <div class="header">

        <div class="logo">
            <a href="main.jsp"></a>
        </div>

        <div class="descr">
            <h3>АЭЖБУ кітапханасы<br>Барлық кітаптар - электронды
түрде!</h3>

        </div>

        <div class="welcom">
            <h5>Қош келдіңіз, <%=session.getAttribute("username")%> !</h5>
            <h6><a href="../index.jsp">Шығу</a></h6>
        </div>

        <div class="search_form">
            <form name="search_form" method="GET" action="books.jsp">
                <input type="text" name="search_string" size="110" value =
"<%=searchString%>"/>
                <input class="search_button" type="submit" value="Поиск"/>
                <select name="search_option">
                    <%if(request.getParameter("search_string") != null &&
request.getParameter("search_option").equals("Автор")){
                        %>
                        <option>Автор</option>
                        <option>Название</option>
                        <% } else { %>
                        <option>Название</option>
                        <option>Автор</option>
                        <% } %>
                    </select>
                </form>

        </div>
```

## Қосымша А (жалғасы)

</div>

left\_menu.jspf

```
<% @page import="com.sergazyyev.diplom.web.beans.Genre"%>
<% @page pageEncoding="UTF-8"%>
<%
    long selectedGenre = 0;
    if(request.getParameter("genre_id") != null){
        selectedGenre = Long.valueOf(request.getParameter("genre_id"));
        session.setAttribute("selectedGenre", selectedGenre);
    }
%>
<div class="sidebar1">
    <h3>Жанрлар:</h3>
    <ul class="nav">
        <jsp:useBean id="genreList"
class="com.sergazyyev.diplom.web.beans.GenreList" scope="application"/>
        <li><a href="books.jsp?genre_id=0">Барлық кітаптар</a></li>
        <%
            for (Genre genre: genreList.getGenreList()){
                if(genre.getId() == selectedGenre && genre.getId() != 0){
                    <li><a style="color: red;"
href="books.jsp?genre_id=<%=genre.getId()%>&name=<%=genre.getName()%>"
><%= genre.getName() %> </a></li>
                    <% } else { %>
                    <li><a
href="books.jsp?genre_id=<%=genre.getId()%>&name=<%=genre.getName()%>"
><%= genre.getName() %> </a></li>
                    <% }
                }%>
            </ul>
    </div>
```

letter.jspf

```
<% @ page pageEncoding="UTF-8" %>
<div class="letters">
```

## Қосымша А (жалғасы)

```
<%
    String searchLetter = null;
    if(request.getParameter("letter") != null){
        searchLetter = request.getParameter("letter");
        session.setAttribute("searchLetter", searchLetter);
    }
%>

<jsp:useBean id="letterList"
class="com.sergazyyev.diplom.web.beans.LetterList" scope="application"/>
<%
    char[] letters = letterList.getRussianLetters();
    for(int i = 0; i < letters.length; i++){
        if(searchLetter != null && searchLetter.toString().toUpperCase().charAt(0)
== letters[i]){
            %>
            <a style="color: red;"
href="books.jsp?letter=<%=letters[i]%>"><%=letters[i]%></a>
            <% } else{ %>
            <a href="books.jsp?letter=<%=letters[i]%>"><%=letters[i]%></a>
            <% }
            }
        %>
    }
%>

</div>
```

JSP технологиясын қолданып жазылған html web-парақшалары:  
/WebPages/index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
        <title>АЭЖБУ онлайн кітапханасы: Кіру</title>
        <link href="css/style_index.css" rel="stylesheet" type="text/css">
    </head>

    <body>
        <div class="main">

            <div class="content">
```

## Қосымша А (жалғасы)

```
<p class="title"><span class="text"></span></p>
<p class="title">Онлайн кітапхана</p>
<p class="text">АЭЖБУ онлайн кітапханасына қош келдіңіз, сіз
мұнда көптеген кітаптарды таба аласыз. Іздеу, сортировка, кітапты көру және
тағы басқа функциялар бар.</p>
<p class="text">Жоба құрылу үстінде, сол үшін дизайн және
функционал әрдайым жаңартылып отырады.</p>
<p class="text">Барлық сұрақтар бойынша мына адреске
хабарласаңыз <a
href="mailto:sergazyev@gmail.com">sergazyev@gmail.com</a></p>
<p>&nbsp;</p>
```

```
</div>
```

```
<div class="login_div">
```

```
<p class="title">Кіру үшін өзіңіздің аты жөніңізді енгізіңіз:</p>
<form class="login_form" name="username" action="pages/main.jsp"
method="POST">
  Аты: <input type="text" name="username" value="" size="20" />
  <input type="submit" value="Кіру" />
</form>
```

```
</div>
```

```
<div class="footer">
```

Құрастырушы: Сергазыев Нурсултан, 2014 ж

```
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```

/WebPages/pages/main.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
```

```
<% @include file="../WEB-INF/jspf/left_menu.jspf" %>
<% @include file="../WEB-INF/jspf/letters.jspf" %>
```



## Қосымша А (жалғасы)

```
<div style="float: left; margin-top: 20px;">
  <h3>Бір жанрды немесе іздеу өрісін пайдаланыңыз</h3>
</div>
```

/WebPages/pages/book.jsp

```
<% @page import="com.sergazyyev.diplom.web.enums.SearchType"%>
<% @page import="java.util.ArrayList"%>
<% @page import="com.sergazyyev.diplom.web.beans.Book" %>
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<%
    if((request.getParameter("search_string") == null ||
request.getParameter("search_string").trim().equals("")) &&
request.getParameter("letter") == null && request.getParameter("genre_id") ==
null){
        System.out.println(request.getContextPath());
        String url = request.getContextPath() + "/pages/main.jsp";
        response.sendRedirect(url);
    }
    else{
%>
<!DOCTYPE html>

<% @include file="../WEB-INF/jspf/left_menu.jspf" %>

<jsp:useBean id="bookList" class="com.sergazyyev.diplom.web.beans.BookList"
scope="page"/>

<% @include file="../WEB-INF/jspf/letters.jspf"%>

<div class="book_list">

    <%
        ArrayList<Book> list = null;
        if(request.getParameter("genre_id") != null){
            long genreId = Long.valueOf(request.getParameter("genre_id"));
            list = bookList.getBooksByGenre(genreId);
        }
        else if(request.getParameter("letter") != null){
```

## Қосымша А (жалғасы)

```
String letter = request.getParameter("letter");
list = bookList.getBooksByLetter(letter);
}
else if(request.getParameter("search_string") != null){
    String searchStr = request.getParameter("search_string");
    SearchType type = SearchType.TITLE;
    if(request.getParameter("search_option").equals("Автор")){
        type = SearchType.AUTHOR;
    }
    if(searchStr != null && !searchStr.trim().equals("")){
        list = bookList.getBooksBySearch(searchStr, type);
    }
}
}
%>
<h5 style="text-align: left; margin-top: 20px;">Табылған кітап саны:
<%=list.size()%></h5>
<%
    session.setAttribute("currentBookList", list);
    for(Book book: list){
%>
<div class="book_info">
    <div class="book_title">
        <p><%= book.getName() %></p>
    </div>
    <div class="book_image">
        <a
href="<%=request.getContextPath()%>/ShowImage?index=<%=list.indexOf(book)
%>" target="_blank"></a>
    </div>
    <div class="book_details">
        <br><strong>ISBN:</strong><%=book.getIsbn()%>
        <br><strong>Баспа:</strong><%=book.getPublisher()%>
        <br><strong>Бет саны:</strong><%=book.getPageCount()%>
        <br><strong>Басылып шыққан
жыл:</strong><%=book.getPublishDate()%>
        <br><strong>Автор:</strong><%=book.getAuthor()%>
        <p style="margin: 10px;"><a
href="<%=request.getContextPath()%>/PdfContent?index=<%=list.indexOf(book)
%>" target="_blank">Оқы</a></p>
    </div>
```

## Қосымша А (жалғасы)

```
</div>
<%
  }
%>

</div>
<% } %>

/src папкасында орналасқан болмыстар java класстары:
com.sergazyev.diplom.web.beans.Author.java

package com.sergazyev.diplom.web.beans;

public class Author {
    private long id;
    private String name;

    public Author(){

    }
    public Author(String name, long id){
        this.name = name;
        this.id = id;
    }

    public String getName(){
        return name;
    }

    /**
     *
     * @param name
     */
    public void setName(String name){
        this.name = name;
    }

    public long getId(){
        return id;
    }
}
```

## Қосымша А (жалғасы)

```
public void setId(long id){  
    this.id = id;  
}  
}
```

com.sergazyyev.diplom.web.beans.AuthorList.java

```
package com.sergazyyev.diplom.web.beans;
```

```
import com.sergazyyev.diplom.web.db.Database;  
import java.sql.Connection;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.ArrayList;  
import java.util.logging.Level;  
import java.util.logging.Logger;
```

```
public class AuthorList {
```

```
    private ArrayList<Author> authorList = new ArrayList<Author>();
```

```
    private ArrayList<Author> getAuthors(){
```

```
        Statement stm = null;
```

```
        ResultSet rs = null;
```

```
        Connection conn = null;
```

```
        try{
```

```
            conn = Database.getConnection();
```

```
            stm = conn.createStatement();
```

```
            rs = stm.executeQuery("select * from author order by fio");
```

```
            while(rs.next()){
```

```
                Author author = new Author();
```

```
                author.setName(rs.getString("fio"));
```

```
                author.setId(rs.getLong("id"));
```

```
                authorList.add(author);
```

```
            }
```

```
        }
```

```
        catch(SQLException ex){
```

```
            Logger.getLogger(AuthorList.class.getName()).log(Level.SEVERE, null,  
ex);
```

```
        }
```

## Қосымша А (жалғасы)

```
finally{
    try{
        if(conn != null){
            conn.close();
        }
        if(stm != null){
            stm.close();
        }
        if(rs != null){
            rs.close();
        }
    }
    catch(SQLException ex){
        Logger.getLogger(AuthorList.class.getName()).log(Level.SEVERE, null,
ex);
    }
}
return authorList;
}

public ArrayList<Author> getAuthorList(){
    if(!authorList.isEmpty()){
        return authorList;
    }
    else{
        return getAuthors();
    }
}
}
```

com.sergazyyev.diplom.web.beans.Book.java

```
package com.sergazyyev.diplom.web.beans;

import com.sergazyyev.diplom.web.db.Database;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Date;
```

## Қосымша А (жалғасы)

```
import java.util.logging.Level;
import java.util.logging.Logger;

public class Book implements Serializable {

    private long id;
    private String name;
    private byte[] content;
    private int pageCount;
    private String isbn;
    private String genre;
    private String author;
    private int publishDate;
    private String publisher;
    private byte[] image;

    public long getId(){
        return id;
    }
    public void setId(long id){
        this.id = id;
    }

    public String getName(){
        return name;
    }
    public void setName(String name){
        this.name = name;
    }

    public byte[] getContent(){
        return content;
    }
    public void setContent(byte[] content){
        this.content = content;
    }

    public int getPageCount(){
        return pageCount;
    }
    public void setPageCount(int pageCount){
```

## Қосымша А (жалғасы)

```
this.pageCount = pageCount;
}

public String getIsbn(){
    return isbn;
}
public void setIsbn(String isbn){
    this.isbn = isbn;
}

public String getGenre(){
    return genre;
}
public void setGenre(String genre){
    this.genre = genre;
}

public String getAuthor(){
    return author;
}
public void setAuthor(String author){
    this.author = author;
}

public int getPublishDate(){
    return publishDate;
}
public void setPublishDate(int publishDate){
    this.publishDate = publishDate;
}

public String getPublisher(){
    return publisher;
}
public void setPublisher(String publisher){
    this.publisher = publisher;
}

public byte[] getImage(){
    return image;
}
public void setImage(byte[] image){
```

## Қосымша А (жалғасы)

```
this.image = image;
}
public void fillPdfContent(){
    Statement stm = null;
    Connection conn = null;
    ResultSet rs = null;
    try{
        conn = Database.getConnection();
        stm = conn.createStatement();
        rs = stm.executeQuery("SELECT content FROM BOOK WHERE id = " +
this.id);
        while(rs.next()){
            this.setContent(rs.getBytes("content"));
        }
    }
    catch(SQLException ex){
        Logger.getLogger(Book.class.getName()).log(Level.SEVERE, null, ex);
    }
    finally{
        try{
            if(rs != null){
                rs.close();
            }
            if(stm != null){
                stm.close();
            }
            if(conn != null){
                conn.close();
            }
        }
        catch(SQLException ex){
            Logger.getLogger(Book.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
}
```

com.sergazyyev.diplom.web.beans.BookList.java

package com.sergazyyev.diplom.web.beans;



## Қосымша А (жалғасы)

```
import com.sergazyyev.diplom.web.db.Database;
import com.sergazyyev.diplom.web.enums.SearchType;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;

public class BookList {

    private ArrayList<Book> bookList = new ArrayList<Book>();

    private ArrayList<Book> getBooks(String str){
        Connection conn = null;
        Statement stm = null;
        ResultSet rs = null;

        try{
            conn = Database.getConnection();
            stm = conn.createStatement();
            System.out.println(str);
            rs = stm.executeQuery(str);

            while(rs.next()){
                Book book = new Book();
                book.setId(rs.getLong("id"));
                book.setName(rs.getString("name"));
                book.setAuthor(rs.getString("author"));
                book.setGenre(rs.getString("genre"));
                book.setPublisher(rs.getString("publisher"));
                book.setIsbn(rs.getString("isbn"));
                book.setPageCount(rs.getInt("page_count"));
                book.setPublishDate(rs.getInt("publish_year"));
                book.setImage(rs.getBytes("image"));
                bookList.add(book);
            }
        }
        catch(SQLException ex){
            Logger.getLogger(BookList.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

## Қосымша А (жалғасы)

```
finally{
    try{
        if(rs != null){
            rs.close();
        }
        if(stm != null){
            stm.close();
        }
        if(conn != null){
            conn.close();
        }
    }
    catch(SQLException ex){
        Logger.getLogger(BookList.class.getName()).log(Level.SEVERE, null, ex);
    }
}
System.out.println(bookList);
return bookList;
}
```

```
public ArrayList<Book> getAllBooks(){
```

```
    return getBooks(" SELECT b.id, b.name, b.isbn, b.page_count, b.publish_year,
p.name as publisher, a.fio as author, g.name as genre, b.image " +
        " FROM book b inner join author a on b.author_id = a.id inner join
genre g on b.genre_id = g.id inner join publisher p on b.publisher_id = p.id order by
b.name;");
}
```

```
public ArrayList<Book> getBooksByGenre(long genreId){
```

```
    if(genreId == 0){
        return getAllBooks();
    }
    else{
        return getBooks(" SELECT b.id, b.name, b.isbn, b.page_count,
b.publish_year, p.name as publisher, a.fio as author, g.name as genre, b.image " +
            " FROM book b inner join author a on b.author_id = a.id inner
join genre g on b.genre_id = g.id inner join publisher p on b.publisher_id = p.id " +
            " WHERE genre_id = " + genreId +
            "" order by b.name " +
            " limit 0,5; ");
    }
```

## Қосымша А (жалғасы)

```
    }  
}  
  
public ArrayList<Book> getBooksByLetter(String letter){  
  
    return getBooks(" SELECT b.id, b.name, b.isbn, b.page_count, b.publish_year,  
p.name as publisher, a.fio as author, g.name as genre, b.image " +  
        " FROM book b inner join author a on b.author_id = a.id inner join  
genre g on b.genre_id = g.id inner join publisher p on b.publisher_id = p.id " +  
        " WHERE substring(b.name, 1, 1) = '" + letter +  
        "' ORDER BY b.name " +  
        "LIMIT 0,5; ");  
}  
  
public ArrayList<Book> getBooksBySearch(String searchStr, SearchType type){  
  
    StringBuilder sql = new StringBuilder(" SELECT b.id, b.name, b.isbn,  
b.page_count, b.publish_year, p.name as publisher, a.fio as author, g.name as genre,  
b.image " +  
        " FROM book b inner join author a on b.author_id =  
a.id inner join genre g on b.genre_id = g.id inner join publisher p on b.publisher_id  
= p.id ");  
    if(type == SearchType.AUTHOR){  
        sql.append(" WHERE lower(a.fio) LIKE '%" + searchStr.toLowerCase() +  
"% ' ORDER BY b.name");  
    }  
    else if(type == SearchType.TITLE){  
        sql.append(" WHERE lower(b.name) LIKE '%" + searchStr.toLowerCase()  
+ "% ' ORDER BY b.name");  
    }  
  
    sql.append(" LIMIT 0,5;");  
  
    return getBooks(sql.toString());  
}  
}  
  
com.sergazyyev.diplom.web.beans.Genre.java  
  
package com.sergazyyev.diplom.web.beans;
```

## Қосымша А (жалғасы)

```
public class Genre {  
    private String name;  
    private long id;  
  
    public Genre()  
  
    }  
    public Genre(String name, long id){  
        this.name = name;  
        this.id = id;  
    }  
  
    public void setName(String name){  
        this.name = name;  
    }  
    public String getName(){  
        return name;  
    }  
    public long getId(){  
        return id;  
    }  
    public void setId(long id){  
        this.id = id;  
    }  
}
```

com.sergazyyev.diplom.web.beans.GenreList.java

```
package com.sergazyyev.diplom.web.beans;  
  
import com.sergazyyev.diplom.web.db.Database;  
import java.sql.Connection;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.ArrayList;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
public class GenreList {  
    private ArrayList<Genre> genreList = new ArrayList<Genre>();
```

## Қосымша А (жалғасы)

```
private ArrayList<Genre> getGenres(){
    Statement stm = null;
    ResultSet rs = null;
    Connection conn = null;

    try{
        conn = Database.getConnection();
        stm = conn.createStatement();
        rs = stm.executeQuery("select * from genre order by name");
        while(rs.next()){
            Genre genre = new Genre();
            genre.setName(rs.getString("name"));
            genre.setId(rs.getLong("id"));
            genreList.add(genre);
        }
    }
    catch(SQLException ex){
        Logger.getLogger(GenreList.class.getName()).log(Level.SEVERE, null,
ex);
    }
    finally{
        try{
            if(conn != null){
                conn.close();
            }
            if(stm != null){
                stm.close();
            }
            if(rs != null){
                rs.close();
            }
        }
        catch(SQLException ex){
            Logger.getLogger(GenreList.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }
    return genreList;
}

public ArrayList<Genre> getGenreList(){
    if(!genreList.isEmpty()){
```

## Қосымша А (жалғасы)

```
        return genreList;
    }
    else{
        return getGenres();
    }
}
}
```

com.sergazyyev.diplom.web.beans.LetterList.java

```
package com.sergazyyev.diplom.web.beans;
```

```
public class LetterList {
    public char[] getRussianLetters(){

        char[] ruLetters = new char[33];

        ruLetters[0] = 'А';
        ruLetters[1] = 'Б';
        ruLetters[2] = 'В';
        ruLetters[3] = 'Г';
        ruLetters[4] = 'Д';
        ruLetters[5] = 'Е';
        ruLetters[6] = 'Ё';
        ruLetters[7] = 'Ж';
        ruLetters[8] = 'З';
        ruLetters[9] = 'И';
        ruLetters[10] = 'Й';
        ruLetters[11] = 'К';
        ruLetters[12] = 'Л';
        ruLetters[13] = 'М';
        ruLetters[14] = 'Н';
        ruLetters[15] = 'О';
        ruLetters[16] = 'П';
        ruLetters[17] = 'Р';
        ruLetters[18] = 'С';
        ruLetters[19] = 'Т';
        ruLetters[20] = 'У';
        ruLetters[21] = 'Ф';
        ruLetters[22] = 'Х';
        ruLetters[23] = 'Ц';
        ruLetters[24] = 'Ч';
```

## Қосымша А (жалғасы)

```
ruLetters[25] = 'Ш';
ruLetters[26] = 'Щ';
ruLetters[27] = 'Ъ';
ruLetters[28] = 'Ы';
ruLetters[29] = 'Ь';
ruLetters[30] = 'Э';
ruLetters[31] = 'Ю';
ruLetters[32] = 'Я';

return ruLetters;
}

}

Деректер қорымен қарым қатынас жасауға арналған java классы:
com.sergazyyev.diplom.web.db.Database.java

package com.sergazyyev.diplom.web.db;

import java.sql.Connection;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

public class Database {

    private static Connection conn;
    private static InitialContext ic;
    private static DataSource ds;

    public static Connection getConnection(){
        try{
            ic = new InitialContext();
            ds = (DataSource) ic.lookup("jdbc/Library");
            conn = ds.getConnection();

        }
    }
}
```

## Қосымша А (жалғасы)

```
        catch(SQLException ex){
            Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);
        }
        catch(NamingException ex){
            Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);
        }
        return conn;
    }
}
```

SearchType enum классы:

```
package com.sergazyyev.diplom.web.enums;

public enum SearchType {
    AUTHOR,
    TITLE
}
```

Деректер қорынан суреттерді және кітаптарды алып web-парақшаға бейнелеуге арналған сервлет java класстары:

com.sergazyyev.diplom.web.servlets.PdfContent.java

```
package com.sergazyyev.diplom.web.servlets;

import com.sergazyyev.diplom.web.beans.Book;
import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author Sergazyyev
 */
public class PdfContent extends HttpServlet {
```



## Қосымша А (жалғасы)

```
/**
 * Processes requests for both HTTP <code>GET</code> and
<code>POST</code>
 * methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    response.setContentType("application/pdf");
    OutputStream out = response.getOutputStream();
    try{
        int index = Integer.valueOf(request.getParameter("index"));
        ArrayList<Book> list = (ArrayList<Book>)
request.getSession(false).getAttribute("currentBookList");
        Book book = list.get(index);
        book.fillPdfContent();
        response.setContentLength(book.getContent().length);
        out.write(book.getContent());
    }
    catch(Exception ex){
        ex.printStackTrace();
    }
    finally{
        out.close();
    }
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the
+ sign on the left to edit the code.">
/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
```

## Қосымша А (жалғасы)

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
} // </editor-fold>

}
```

com.sergazyyev.diplom.web.servlets.ShowImage.java

```
package com.sergazyyev.diplom.web.servlets;

import com.sergazyyev.diplom.web.beans.Book;
import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import javax.servlet.ServletException;
```

## Қосымша А (жалғасы)

```
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ShowImage extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        response.setContentType("image/jpeg");
        OutputStream out = response.getOutputStream();
        try {
            int index = Integer.valueOf(request.getParameter("index"));

            ArrayList<Book> list = (ArrayList<Book>)
request.getSession(false).getAttribute("currentBookList");
            Book book = list.get(index);
            response.setContentLength(book.getImage().length);
            out.write(book.getImage());

        }
        catch(Exception ex){
            ex.printStackTrace();
        }
        finally{
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the
+ sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
```

## Қосымша А (жалғасы)

```
processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}
}
```

## Қосымша Ә

Деректер қорын құруға арналған скриптер листингі  
library\_author.sql

```
CREATE DATABASE IF NOT EXISTS `library` /*!40100 DEFAULT
CHARACTER SET utf8 */;
USE `library`;
-- MySQL dump 10.13 Distrib 5.5.16, for Win32 (x86)
--
-- Host: localhost Database: library
--
-- Server version 5.5.29

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET
@OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `author`
--

DROP TABLE IF EXISTS `author`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `author` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `fio` varchar(300) NOT NULL,
  `birthday` date NOT NULL,
  PRIMARY KEY (`id`)
```

## Қосымша Ө (жалғасы)

```
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS
*/;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
```

-- Dump completed on 2013-03-13 14:12:21

library\_book.sql

```
CREATE DATABASE IF NOT EXISTS `library` /*!40100 DEFAULT
CHARACTER SET utf8 */;
USE `library`;
-- MySQL dump 10.13 Distrib 5.5.16, for Win32 (x86)
--
-- Host: localhost Database: library
-- -----
-- Server version 5.5.29

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@ @CHARACTER_SET_CLIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@ @CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@ @COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@ @TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@ @UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET
@OLD_FOREIGN_KEY_CHECKS=@ @FOREIGN_KEY_CHECKS,
```

## Қосымша Ә (жалғасы)

```
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `book`
--

DROP TABLE IF EXISTS `book`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `book` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `content` longblob NOT NULL,
  `page_count` int(11) NOT NULL,
  `isbn` varchar(100) NOT NULL,
  `genre_id` bigint(20) NOT NULL,
  `author_id` bigint(20) NOT NULL,
  `publish_year` date NOT NULL,
  `publisher_id` bigint(20) NOT NULL,
  `image` longblob,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id_UNIQUE` (`id`),
  UNIQUE KEY `isbn_UNIQUE` (`isbn`),
  KEY `fk_author_idx` (`author_id`),
  KEY `fk_genre_idx` (`genre_id`),
  CONSTRAINT `fk_author` FOREIGN KEY (`author_id`) REFERENCES
`author` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `fk_genre` FOREIGN KEY (`genre_id`) REFERENCES `genre`
(`id`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS
*/;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
```

## Қосымша Ө (жалғасы)

```
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2013-03-13 14:12:20

    library_genre.sql

CREATE DATABASE IF NOT EXISTS `library` /*!40100 DEFAULT
CHARACTER SET utf8 */;
USE `library`;
-- MySQL dump 10.13 Distrib 5.5.16, for Win32 (x86)
--
-- Host: localhost Database: library
-- -----
-- Server version 5.5.29

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@ @CHARACTER_SET_CLIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@ @CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@ @COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@ @TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@ @UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET
@OLD_FOREIGN_KEY_CHECKS=@ @FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@ @SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@ @SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `genre`
--

DROP TABLE IF EXISTS `genre`;
```



## Қосымша Ө (жалғасы)

```
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `genre` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS
*/;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
```

-- Dump completed on 2013-03-13 14:12:17

library\_publisher.sql

```
CREATE DATABASE IF NOT EXISTS `library` /*!40100 DEFAULT
CHARACTER SET utf8 */;
USE `library`;
-- MySQL dump 10.13 Distrib 5.5.16, for Win32 (x86)
--
-- Host: localhost Database: library
--
-- Server version 5.5.29

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
```

## Қосымша Ә (жалғасы)

```
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0 */;
/*!40014 SET
@OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `publisher`
--

DROP TABLE IF EXISTS `publisher`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `publisher` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS
*/;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2013-03-13 14:12:18
```