ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.И. Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университетіКибернетика және ақпараттық технологиялар институты Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

Аллаберган Ақмарал

«Жеке білім беру мекемелеріне арналған» WEB-сайт әзірлеу

ДИПЛОМДЫҚ ЖҰМЫС

5В070300 – «Ақпараттық жүйелер» мамандығы

Алматы 2021

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.Сәтбаев атындағы қазақ ұлттық техникалық зерттеу университеті
Ақпараттық және телекоммуникациялық технологиялар институты
Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

«Қорғауға жіберілді»
КАӨжС кафедра меңгерушісі,
_____ Н.А.Сейлова
« 27 »___мая___2021 ж.

ДИПЛОМДЫҚ ЖҰМЫС

Тақырыбы: «Жеке білім беру мекемелеріне арналған WEB-сайт әзірлеу»

5В070300 – «Ақпараттық жүйелер» мамандығы

Орындаған: Аллаберган А.А.

Ғылыми жетекші лектор
Зиро А.А.

Алматы 2021

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.Сәтбаев атындағы қазақ ұлттық техникалық зерттеу университеті

Ақпараттық және телекоммуникациялық технологиялар институты

Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

БЕКІТЕМІН
КАӨС кафедра меңгерушісі,
тех.ғыл.канд, ассоц. доцент
_____ Н.А.Сейлова
« 27 »___мая___2021 ж.

Дипломдық жұмысты орындауға
ТАПСЫРМА

Білім алушы: Аллаберган Ақмарал Айдынқызы
Тақырыбы: «Жеке білім беру мекемелеріне арналған WEB-сайт әзірлеу»
Университет Ректорының 2020 жылғы «24» 11 №2131-б бұйрығымен бекітілген
Аяқталған жұмысты тапсыру мерзімі 2021 жылғы « 20» мамыр
Дипломдық жұмыстың бастапқы берілістері: диплом алдындағы практикалық жұмыс қорытындысы, тақырып бойынша әдебиеттерге шолу нәтижелері,теориялық мәліметтердің жиыны
Дипломдық жұмыста қарастырылатын мәселелер тізімі:
а)қойылған мәселенің қазіргі жағдайын пайымдау
ә)ақпараттық қамтаманы құру
б)программалық қамтаманы құру
Сызбалық материалдар тізімі: Power Point бағдарламасындағы слайдтар
Сызба материалдар: 15 слайдпен көрсетілген
Ұсынылатын негізгі әдебиет: 11 атау

Дипломдық жұмысты дайындау

КЕСТЕСІ

Бөлімдер атауы, қарастырылатын мәселелер тізімі	Ғылыми жетекші мен кеңесшілерге көрсету мерзімдері	Ескерту
Мәселенің қазіргі жағдайына шолу және оны талдау	28.02.2021	
Ақпараттық қамтаманы құру	7.04.2021	
Программалық қамтаманы құру	3.05.2021	

Дипломдық жұмысының бөлімдерінің кеңесшілері мен норма бақылаушыларының аяқталған жобаға қойған **қолтаңбалары**

Бөлімдер атауы	**Кеңесшілер, аты, әкесінің аты, тегі (ғылыми дәрежесі,атағы)**	**Қол қойылған күні**	Қолы
Норма бақылаушы	Кабдуллин Максат		
Программалық қамтама			

Ғылыми жетекші _____  Зиро А.А.

Тапсырманы орындауға алған білім алушы_  А.А. Аллаберган

Күні « 24» 11 2021 ж.

АҢДАТПА

Дипломдық жұмыстың тақырыбы - «Жеке білім беру мекемелеріне арналған» WEB-сайт әзірлеу.

Бұл дипломдық жұмысты жобалау барысында білім алушы мен білім берушілердің пайдалануына арналған, сонымен қатар білім алушының білімін нақтылай түсуіне арналған веб-сайтын әзірлеу қарастырылады. WEB-сайт көмегімен білім алушы өзіне қажет материалдарды алып, сонымен қатар тестілер мен қосымша тапсырмаларды орындау арқылы өз білімін нығайта түседі.

Дипломдық жұмысты әзірлеу барысында HTML, CSS, JavaScript, PHP бағдарламалық тілдері және MongoDB құжатқа негізделген мәліметтер қорын басқару жүйесі пайдаланылады.

АННОТАЦИЯ

Тема дипломной работы – разработка WEB-сайта “Для частных образовательных учреждений”

Целью этой дипломной работы является разработка веб-сайта для использования студентами и преподавателями, а также для уточнения знаний студентов. С помощью WEB-сайта студент укрепляет свои знания, получая необходимые материалы, а также выполняя тесты и дополнительные задания.

При разработке дипломной работы используются языки программирования HTML, CSS, JavaScript, PHP и документированная система управления базами данных MongoDB.

ANNOTATION

The topic of the diploma work is development of the WEB-site "For private educational institutions"

The purpose of this diploma work is to develop a website for use by students and teachers, and to refine students' knowledge. With the help of the WEB-site, the student strengthens his knowledge by receiving the necessary materials, as well as completing tests and additional tasks.

When developing the diploma work, the programming languages HTML, CSS, JavaScript, PHP and the MongoDB document-oriented database management system are used.

МАЗМҰНЫ

КІРІСПЕ

Қазіргі қоғамның даму кезеңінде ақпараттық технологиялардың үлесі зор. Олар барлық салада жоғары деңгейде дамып келеді, тіпті қазір ақпараттық технологияны қолданбайтын саланы табу қиын десек те болады. Соның ішінде білім беру саласында өте кеңінен таралуда. Ақпараттық технологиялардың білім берудегі мақсатына тоқталатын болсақ, қоғамдағы білім алушылардың интеллектуалды мүмкіндіктерін арттырумен қатар, оқу үрдісін қарқынды дамыту және білім беру жүйесіндегі барлық деңгейінде білім сапасын жақсарту болып табылады.

Бүгінгі таңдағы еліміздегі жағдайға байланысты, көп мемлекеттер, сонымен қатар Қазақстанда да қашықтықтан оқытуға көшті. Қашықтықтан білім беру бастапқыда қиынға соққанымен, кейінірек барлық білім беру мекемелері, сонымен қатар ата-аналар да компьютер мен смартфонды жақсы деңгейде үйреніп қалды. Қашықтықтан білім берудің артықшылығы, оқушылар өз бетімен ізденуді үйренді, уақытты тиімді қолдануды үйренді. Оқушылар мен студенттер үшін электронды кітапханалар кеңінен тарап жатыр, оның артықшылықтарының бірі - үйден шықпай-ақ өзін өзі дамытуға уақыт та, мүмкіндіктер де жеткілікті.

Қазіргі уақытта мектептен бөлек балалардың ақыл-ойын, білім деңгейін дамытуға, бар білімін одан жоғары көтеру үшін арнайы білім беру мекемелері бар. Ол мекемелерде өздерінің арнайы білім беру платформалары және әдістері бар. Платформа әдемі дизайнмен әрленіп, ішінде ойын түріндегі және тағы басқа да тапсырмалар қарастырылған. Яғни, оқушыны білім алуға бірнеше жолдармен қызықтыруға болады дегім келеді.

Сонымен, бүгінде жеке білім беру мекемелерінің санының артуына байланысты, олар арасындағы бәсекелестік те көбейіп келеді. Барлық жеке білім беру мекемелерінің мақсаты бір, яғни олар білім алушылардың білімдерін көтеріп, сонымен қатар білім беру мекемесінің атағы мен беделін жоғары деңгейге көтеру. Ол үшін әрине мықты білім беру платформасын немесе білім беру методикасын ойлап табу қажет.

Менің дипломдық жұмысымда жеке білім беру мекемелері үшін веб-сайт әзірлеу. Мен бұл сайт әзірлеуді репетиторлық білім беру мекемелеріне арнап, болашақта сондай білім беру мекекмелеріне ұсынамын. Веб-сайттың мақсаты – білім алушының қызығушылығын арттырумен қатар, алған білімін нақтылау. Яғни бұл жерде білім алу барысында сайттан алған материалдарын, тапсырмаларды орындау арқылы есте сақтап, білімін нақтылауға үлес қосады.

1. Пәндік саланы талдау

1.1 Білім беру мекемесі ұғымына жалпы шолу

Білім беру мекемелері – бұл оқу процесстерін жүзеге асыратын, яғни бір немесе одан да көп білім беру программаларын іске асырып, сонымен қатар студенттер мен оқушылардың немесе жалпы білім алушылардың тәрбиеленуін, білім алып, дамуын қамтамасыз ететін мекемелер болып табылады.

Білім берудің мақсаты – ғылыми тұрғыда негізделген білімдерді меңгеру және де белгілі бір дәрежеде білімді игеріп, даму мен практикалық дайындық болып табылады. Басқа түрде жеткізетін болсақ, білім дегеніміз – арнайы құрастырылған қарым-қатынас негізінде алынған білім, білік және қабілеттер жиынтығы.

Білім беру мекемелерінің түрлеріне тоқталып өтейік:

- бастауыш жалпы білім беретін;
- негізгі орта;
- орта білім беретін;
- жеке пәндерді тереңірек оқытатын орта жалпы білім беретін;
- гимназия;
- лицей;
- жеке бала бақшалар;
- жеке мектептер;
- репетиторлық және тағы басқа болып бөлінеді.

Бастауыш жалпы білім беретін мекеме ол жалпы білім беру программасын жүзеге асырады. Негізгі орта мектептер бастауыш жалпы және негізгі орта білім берудің жалпы білім беру программасын іске асырады. Орта білім беретін мекеме осы үш мектептердің жалпы білім беру бағдарламасын қамтамасыз етеді. Репетиторлық білім беру мекемелері белгілі бір пәндерді жетік меңгеру немесе мектепте дұрыс түсіне алмаған пәндерді тереңдетіп оқытады. Ондай мамандарды репетиторлар деп атайды. Жеке пәндерді оқытатын арнайы даярлық мектептер, олар осы үш мектеп бойынша белгілі бір немесе бірнеше пәндерге қосымша дайындық ретінде жалпы білім беру бағдарламасын жүзеге асырады. Сол сияқты мектеп жасына дейінгі балаларды дайындайтын да жеке және мемлекеттік балабақшалар, сонымен қатар басқа да білім беру мекемелері жетіп артылады.

Жеке білім беру мекемелері мектептерге қарағанда балаларды қызықтыратын оқу бағдарламаының жаңаша методикасын ойлап тауып, балалардың білім деңгейін арттыру мақсатында құрылған.

Сонымен, оқытудың кез келген түрінің үнемділігі бірнеше компоненттерге топтастырылады: техникалық база, оқытуды ұйымдасыру кезінде пайдаланылатын, әзірленген әдістемелік материалдардың үнемділігіне, оқыту технологиясына.

1.2 Таңдалған тақырыптың өзектілігі

Менің тақырыбым Жеке білім беру мекемелеріне арналған веб-парақша құру. Қазіргі таңда көптеген білім беру мекемелері жұмыс істеуде, сонымен қатар олардың саны да көбейіп жатыр. Осы себептен де балаларын оқытуға бергілері келіп отырған ата-аналарда өте қиын таңдау болады және сұрақтар туындайды. Менің ойымша, әрбір жеке білім беру мекемелерінде өз веб-парақшалары болуы керек және ол жерде барлық ақпарат толығымен және түсінікті түрде қарастырылуы тиіс, сонымен қатар жеке білім беру мекемелерінде нақты және нәтижесі бар білім беру әдістемесі болуы керек деген ойдамын. Осы мәселелер мен талаптарды ескере отырып, өз дипломдық жұмысыма осындай тақырыпты алдым.

Білім беру мекемелеріне сайттың бүгінде өте қажет екендігіне сөз жоқ. Ол іске асыруға арналған екі маңызды функция бар. Біріншіден, сайт - бұл мекеменің Интернеттегі өкілдігі және мұғалімдер мен оқушылар туралы ақпарат көзі. Екіншіден, оқу процесінде сайт мұғалімнің тапсырмасын едәуір жеңілдететін құрал ретінде де қолданыла алады. Сонымен білім беру мекемесінің сайты кімге керек? Біріншіден, мекеме өмірімен тікелей байланысты адамдар. Сайт студенттерге немесе оқушыларға тек соңғы жаңалықтарды білуге, үй тапсырмаларын орындауға, сабақ кестесіне, сыныптан тыс жұмыстардан фоторепортаждарды көруге және интернетте сұхбаттасуға мүмкіндік беріп қана қоймай, сонымен қатар өзін-өзі көрсету құралы ретінде де қызмет етеді. Бұл сайтты мен репетиторлық білім беру мекемелеріне арнап жасадым, себебі менің сайтымда екі пәннен тұратын видеоматериалдарды көрумен қатар, соған қатысты тапсырмаларды орындау сияқты функциялар қарастырылған. Менің ұсынатын веб-сайтым репетиторлық білім беру мекемелеріне бүгінгі таңда қажет деп ойлаймын. Сол себепті де осындай сайт құруды дұрыс деп санадым.

1.3 Есептің қойылымы

Қандай да бір сайт қалыпты жұмыс жасауы үшін оның сенімді жерде сақталуы міндетті. Осындай мақсаттар үшін арнайы серверлер бар, яғни, аппараттық немесе web-серверлер деп аталады. Сондай-ақ, сайттарды сақтау

қызметтерінің атауы – web-хостинг деп аталады.

Білім беру мен сертификаттауға бағытталған web – ресурс құрастыру негізінде орналасқан орнына қарамастан онлайн түрде білімін дағдылай түсіп, сондай-ақ, сертификат алу мүмкіндігі әлдеқайда тиімді екендігіне көз жеткізу.

Дипломдық жұмыстың мақсаты - интернет-технологияларды дамыту үшін білім беру жүйесінің архитектурасын, клиент-сервер архитектурасының негіздерін, адаптивті бағдарламалық қамтамасыз ету прототипін, осындай жүйенің деректер базасын, білім берудің түрлі әдістері мен оларды дамыту. Заманауи коммуникациялық технологияларды пайдалану.

Осы мақсатқа қол жеткізу үшін төмендегі міндеттер қойылып, шешімі табылады:

- білім беру мекемесіне байланысты WEB-парақшаларды зерттеу барысында табылған олардың артықшылықтары мен кемшіліктері туралы ақпарат жинау;
- заманауи веб-сайт құрастыру және сайтты құру үшін керекті программалық құралдарды зерттеу.
- оқыту мекемесінің WEB-парақшасының сұлбасын келтіру;
- білім беру мекемелеріне арналған web-сайтты жасаудың әдістемелік және педагогикалық аспектілеріне талдау жасау;
- программалық қамтаманың жұмыс істеу алгоритмдерін жобалау;
- мәліметтер базасының құрылымын жобалау және құру;
- пайдаланушыларға сайт интерфейсінің жеңілдігін қарастыру;
- білім беру мекемесінің WEB-сайтын іске асыру және тестілеу.

2. Веб-сайттарды құруға арналған әдістемелер мен талаптар. Мәліметтер базасы

2.1 Білім беру мекемесінің веб-сайттарын құру әдістемесі

Веб-сайттың дизайны нәтижелі көрсеткішке тәуелді болатын барлық процестерді қамтиды. Сайттың мақсаттарын және міндеттерін қалыптастыру дегеніміз - бұл веб-сайтты дамытудың алғашқы кезеңі және оған барынша жауапкершілікпен қарау қажет. Сайттың негізгі функцияларына ұйымдастыру кіреді, яғни ол дегеніміз - сабақтар кестесі, мекеме қызметкерлері, түрлі хабарландырулар жайлы ақпаратқа жылдам және ыңғайлы қол жетуді қамтамасыз етеді, бұл әсіресе ата-аналар және басқа да жауапты адамдар үшін маңызды; білім беру - сайтқа нұсқаулықтарды, оқу материалдарын кірістіру, сондай-ақ білім беру көздеріне сілтемелер оқу процесі кезінде веб-сайтты пайдалануға мүмкіндік береді.

Осы кезеңдерде сайттың мазмұны ойластырылады, бұл маңызды компоненттердің бірі болып табылады. Егер сайтта пайдаланушылардың белгілі бір тобын қызықтыратын пайдалы ақпарат болмаса, онда сайттың сол сияқты болу қажеттілігі туралы сұрақ туындауы мүмкін. Сонымен, мекеме жайлы білгісі келетін аудиторияға мынадай ақпараттар болуы шарт:

- мекеме туралы ақпарат;
- іс-шаралар, курстар туралы ақпарат;
- мұғалімдер туралы ақпарат және тағы басқа да қызықты ақпараттар

бұл ақпараттар білім беру мекемесінің оқушылары мен қызметкерлері үшін ғана емес, сонымен қатар көптеген адамдар үшін қызықты болуы мүмкін және жаңа білім алушы аудиторияны тартуға мүмкіндік береді, қосымша жарнама жасайды.

Сайт құрылымын жобалау. Сайттың навигациялық схемасы оның құрылысына байланысты болады және пайдаланушының ол арқылы сіз ұсынатын ақпаратқа қалай жылдам қол жеткізетіндігін анықтайды.

Навигациямен жұмыстың қарапайымдылығы мен жеңілдігі - веб-сайт трафигін анықтайтын факторлардың бірі. Пайдаланушылар веб-сайттың кез-келген парақшасына, соның ішінде басты парақшаға тез әрі оңай өтуі керке. Дәл осы кезеңде сайт бөлімдерінің атауларын, парақтың тақырыпшаларын ойластыру қажет, ақпараттарды орналастырудың логикалық құрылымын ойластыру керек. Үшінші кезеңнің нәтижесінде сайтқа ақпаратты орналастырудың нақты логикалық құрылымы қалыптастырылуы керек.

2.2 Білім беру мекемесінің веб-парақшасына қойылатын талаптарды талдау

Біз «талаптар» дегенді қалай түсінеміз? Бұл барлық веб-ресурстар пайдаланушылардың оларға назар аударуы үшін және де олармен жұмыс жасауды ұнатуы үшін, соған сәйкес келетін талаптар.

Оларға мысал ретінде төмендегілерді алсақ болады:

- тітіркендірмейтін дизайн - түсініксіз шрифттар мен көзді ауыртатын түстер болмаған жөн. Пайдаланушыны сайттың дизайны алаңдатпауы тиіс, керісінше пайдаланушыға керегін жылдам табуға көмектесу қажет;
- түсінікті құрылым - пайдаланушы өзіне қажет бөлімге 2-3 рет басу арқылы жылдам өту керек. Егер сіздің сайтыңызда гетерогенді ақпарат көп болса, сайттан іздеу тетігін құрастырыңыз. Оның жұмыс істеуін міндетті түрде тексеріңіз;
- жүктеудің жоғары жарамдылығы - пайдаланушы веб-сайттың жүктелуін 3 секундтан артық күтпеуі керек. Бұл ереже компьютерлерде де, смартфондарда да жұмыс істейді. Пайдаланушыға сайтқа дереу кіріп, өзіне керегін табуы керек, бірақ ол ұзақ күтпес үшін жүктелу жылдамдығы жоғары болуы керек;
- кері байланыс мүмкіндігі.

2.3 Мәліметтер базасын модельдеу

MongoDB - кесте схемасын сипаттауды қажет етпейтін құжатқа негізделген мәліметтер қорын басқару жүйесі. NoSQL жүйелерінің классикалық түріндегі мысалдардың бірі болып саналады, ол JSON-ға ұқсас құжаттар мен мәліметтер базасының үлгісін қолданады. C++ тілінде жазылған. Ол веб-дамытуда, соның ішінде, JavaScript-ке негізделген MEAN стегінің шеңберінде қолданылады.

MongoDB төмендегілер үшін пайдаланылады:

- событиелерді сақтау және тіркеу;
- құжаттар мен контентті басқару жүйелері;
- электрондық сауда;
- ойындар;
- мониторинг деректері, датчиктер;
- мобильді қосымшалар;
- веб-парақшалардың деректерін сақтау (мысалы, түсініктеме, рейтингтер, пайдаланушы профилдері, пайдаланушы сессияларын сақтау болып табылады).

Мәліметтер базасын басқару үшін келесі командаларды орнатуға болады:

MongoDB Shell-ге қосылу mongo - бұл интерактивті қабық, ол әзірлеушілер мен әкімшілерге деректерді қарауға, қосуға, жоюға және жаңартуға мүмкіндік береді, сонымен қатар репликалауды, бұзуды, түйіндерді ажыратуға, JavaScript немесе кез келген басқа мәліметтер базасына сұраныстарды орнатуға мүмкіндік береді;

- mongostat - бұл орындалатын MongoDB данасының статистикасының тізімін қортындылайтын командалық жол құралы, бұл кірістіру, жаңарту, жою, сұраныстар мен командалар санын, сондай-ақ дананың ресурстарды тұтынуын визуалдауға мүмкіндік береді;
- mongotop - бұл даналардың данадан оқылатын немесе жазылатын уақытын қадағалау әдісін ұсынатын құрал. Сонымен қатар әр жинақ деңгейінде статистикалық мәліметтер келтірілген;
- mongoimport және mongoexport - JSON, CSV немесе TSV импорттауға және экспорттауға арналған құралдар, бірқатар басқа форматтарға қолдау көрсетіледі; mongodump және mongorestore - бұл резервтік көшірмені құруға және одан дерекқорды қалпына келтіруге арналған құралдар.

Төмендегі суретте деректер қоры құрылымы келтірілген, яғни нақтылай кететін болсақ,

articles – админ енгізетін ақпараттардың кестесі

images – сақталатын суреттердің кестесі

taskquestions – берілетін тапсырмалар сақталатын кесте

tasks – пәндердің кестесі

userroles – қандай рөлдер бар екендігі туралы кесте құрылымы

users – тіркелген аккаунттар кестесі

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
articles	7	1.3 KB	8.8 KB	1	16.4 KB	
images	8	95.1 B	761.0 B	1	32.8 KB	
taskquestions	30	108.9 B	3.3 KB	1	36.9 KB	
tasks	7	67.6 B	473.0 B	1	36.9 KB	
userroles	3	48.3 B	145.0 B	1	16.4 KB	
users	6	214.2 B	1.3 KB	1	36.9 KB	

2.3.1-сурет. Деректер қорының құрылымы

Төмендегі суреттерде негізгі кестелердің құрылымын мысал ретінде келтіріліп көрсетілген:

users

	_id ObjectId	email String	name String	surname String	middleName String
1	6086e98373375d37840b94f1	"akmaral@mail.ru"	"Akmaral"	"Allabergen"	"Aidynkyzy"
2	6086ea1573375d37840b94f4	"akma"	"akma"	"allabergen"	"aidynkyzy"
3	6086ea4073375d37840b94f5	"admin"	"Admin"	"Adminov"	"Adminov"
4	6086eec473375d37840b94fb	"akmarall@mail.ru"	"Akmaral"	"Allabergen"	"Aidynkyzy"
5	6086f0c873375d37840b94fc	"teacher@mail.ru"	"Мұғалім"	"Аккаунт"	"-ұ-ұ-"
6	6086f46273375d37840b9503	"baknur@mail.ru"	"Baknur"	"Baknur"	"-ұ-ұ-"

2.3.2-сурет. Users кестесінің құрылымы

articles

	_id ObjectId	title String	text String	image ObjectId	__v Int32
1	6083c29a65632a3ff4fe1cfe	"COVID--19 уақытында сессия қал	"Коронавирус адамның өкпе жолда	6083c29a65632a3ff4fe1cfd	0
2	60841fba28326747a064dfe5	"Демалатын жер"	"Lorem Ipsum is simply dummy te	60841fba28326747a064dfe4	0
3	60841fd728326747a064dfe7	"Neon edition"	"It is a long established fact	60841fd728326747a064dfe6	0
4	60841ff628326747a064dfe9	"Малиновый закат"	"There are many variations of p	60841ff628326747a064dfe8	0
5	6084201328326747a064dfeb	"Alone everything"	"The standard chunk of Lorem Ip	6084201328326747a064dfea	0
6	6084205428326747a064dfed	"Астрономия тобының жаңалықтары	""Sed ut perspiciatis unde omni	6084205428326747a064dfec	0
7	608420a428326747a064dfef	"Call Of Duty"	""But I must explain to you how	608420a428326747a064dfee	0

2.3.3-сурет. Articles кестесінің құрылымы

2.4 Прецеденттер диаграммасы

Пәндік саланың моделін құру осы өмірде бар абстракцияларды, яғни жүйеде кездесетін объектілерді анықтаудан басталады.

Прецеденттер диаграммасы – бұл жүйенің болжамды мінез құлқының құжатталған үлгісі болып табылады. Әр прецедент құжат бойынша бекітілген оқиғалар ағыны арқылы сипатталуы тиіс. Мәтіндік құжат актер прецедентті бастаған кезде жүйе не жасау керек екенін анықтайды. Прецедентті сипаттайтын құжаттың құрылымы әртүрлі, бірақ типік сипаттамасында мынандай бөлімдер болу керек:

- қысқаша сипаттамасы;
- алғышарттар;
- іс-шара ағынының егжей-тегжейлі сипаттамасы;
- соңғы шарттар.

Бұл диаграммада, 3 түрлі рөл бар, олар: админ, оқушы, мұғалім. Осы рөлдерің әрқайсысының өз атқаратын қызметі бар.

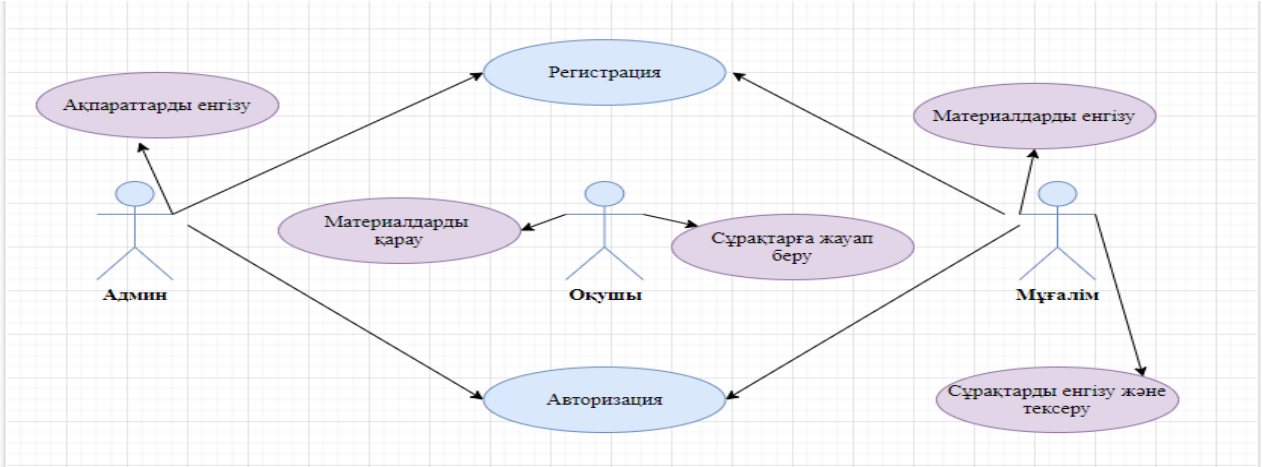
Әрбірінің өз қызметі бар:

Админ – жүйеге ақпараттарды жүктей алады, сонымен қатар мұғалім мен оқушыны енгізе алады.

Мұғалім – жүйеге материалдарды енгізеді және оқушыға тапсырмалар бере алады.

Оқушы – материалдарды қарап, тапсырмаларды орындап және өзінің дұрыс немесе дұрыс емес жауаптарын көре алады.

Үшеуінде де жүйеге авторизация/регистрация жасай алу құқығы бар.



2.4.1-сурет. Прецеденттер диаграммасы

2.5 ER диаграмма

Мәліметтер қоры объектілері бір бірімен байланысты болғандықтан, оларды қатысушылар арасындағы қатынастар ретінде қарастыруға болады.

Көпке-көп байланыс түрі – әр субъект бір немесе одан да көп басқа субъектілерге сәйкес келетін қатынас түрі.

Біреуге көп байланыс түрі – бірінші субъект екінші субъектінің тек біреуіне тиесілі, ал екінші құрылымда бірінші субъектінің бір немесе бірнешеуі болатын қатынас түрі.

Бір-біріне қатынас түрі – бұл жерде әр субъект басқа субъектінің бір данасын ғана көрсетеді.

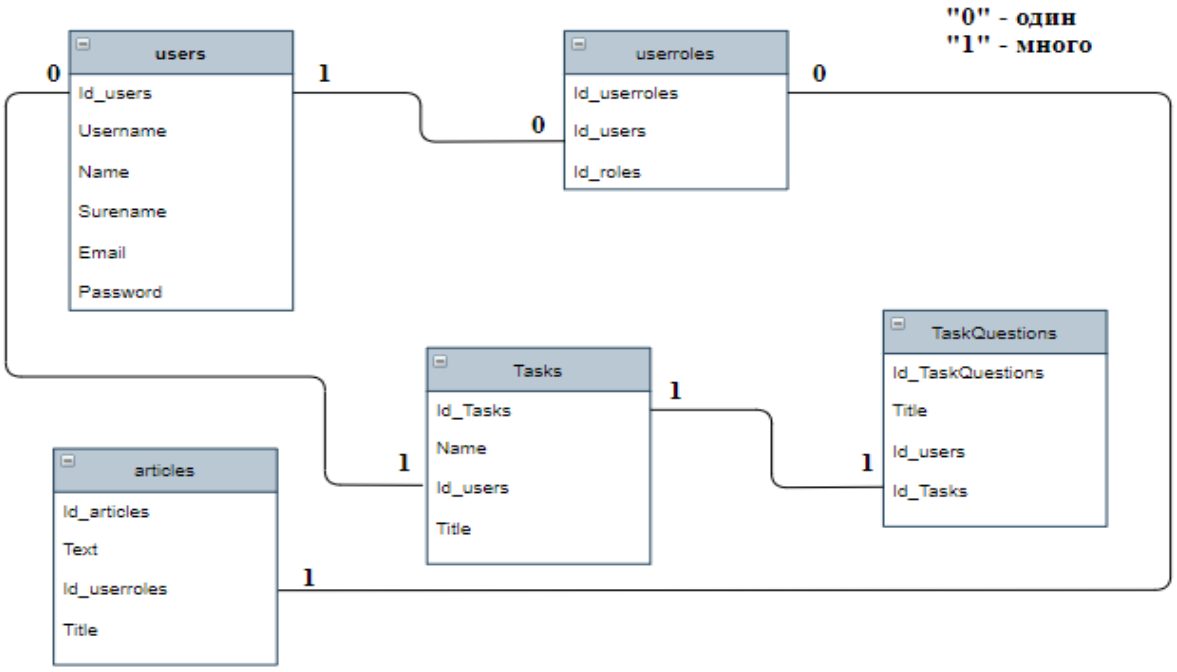
Пайдаланушы – бұл белгілі бір қимыл-әрекеттерді орындаушы.

Рөлдер – пайдаланушы орындай алатын мүмкіндік жиынтығының сипаттамасы.

Тапсырма – пайдаланушы белгілі бір уақытта орындайтын іс-әрекет.

Мақалалар/жаңалықтар – болып жатқан соңғы іс-шаралар туралы хабарлама.

Тапсырма сұрақтары – пайдаланушы орындайтын немесе құратын іс-әрекет.



2.5.1-сурет. ER диаграмма

3. Программалық қамтаманы құру және жобалау

3.1 Программалау тілін таңдауды негіздеу

Жоғарыда қойылған міндеттерді ескере келе, білім беру мекемесіне арналған веб-сайтты құру үшін HTML, CSS, JavaScript бағдарламалау тілдері таңдалып алынды Сонымен қатар Visual Studio Code, Node.js программалық орталары және MongoDB құжатқа негізделген мәліметтерді басқару жүйесі қолданылды. Осы таңдалып отырған тілдер басқа тілдерге қарағанда мүмкіндіктері мен артықшылықтарының болғаны үшін таңдалып алынды.

3.1-сурет. Программалық қамтама құрылымы

Таңдалған тілдер өзара жақсы әрекет жасайды және бірінің мүмкіндіктерін бірі толықтырады.

HTML (*HyperTextMarkupLanguage*) - бұл құжаттарды кодтау мақсатында қолданылатын гипермәтіндік белгілеу тілі. HTML парақшалары интернетте браузерден серверге, жай мәтін немесе шифрлар арқылы HTTP және HTTPS хаттамаларымен жіберіледі.

HTML тілін Тим Бернерс-Ли деген британдық ғалым 1986-1991 жылдары Женева Еуропалық ядролық зерттеу орталығында жасап шығарған. HTML басында ғылыми және техникалық құжаттарды халықтар арасында өзара алмастыруға арналған тіл ретінде қарастырылып жасалған. HTML-дің көмегімен құжатты оңай әрі жылдлам әдемі етіп жасауға болады. Құрамының өзгергенінен басқа HTML бағдарламалау тіліне гипермәтіндерді оқу қасиеті қосылған, кейіннен бейнематериалдық қасиеттер қосыла бастаған. Бағдарламалар құжаттарды құрылымдау, пішімдеу, оларды байланыстыру құралдарына қайта шығару құралы ретінде құрылды.

CSS (*Cascading Style Sheets*) - веб беттерді форматтау кезінде қолданылатын бағдарламалау тілі. Бұл тіл стильдер кестелерінің тілі деп аударылады. Егер HTML тілін тек құжаттарды құру үшін пайдалансақ, онда CSS тілін сол құжаттарды әдемі етіп форматтап, құжаттың түр-түсін өзгертуге болады. Екеуінің біреуі болмаса сайт та болмайды.

JavaScript - салыстырмалы қарапайым объектіге бағытталған скрипттік бағдарламалау тілі, ол үлкен емес клиенттік және серверлік қосымшаларды Интернет үшін құруға арналған. JavaScript тілінде жазылған бағдарламалардың барлығы HTML құжаттамаларына қосылады. JavaScript жасалған бағдарламалардың мысалы ретінде пайдаланушы енгізген мәліметтерді тексеретін не болмаса құжаттаманы ашу немесе жабу кезінде белгілі бір әрекеттерді орындауды іске асыратын бағдарламаларды қарастыруға болады. Мұндай бағдарламалар негізінен пайдаланушының әрекетіне әрекет жасай алады, яғни тышқан пернесін басу, экрандық формаға мәліметтерді беру немесе тышқанды жылжыту арқылы беттерде жылжуға мүмкіндік береді. Одан басқа, JavaScript-бағдарламалар браузерлердің өзін және құжаттамалар атрибуттарын басқара алады.

JavaScript бағдарламалау тілінің функциялары:

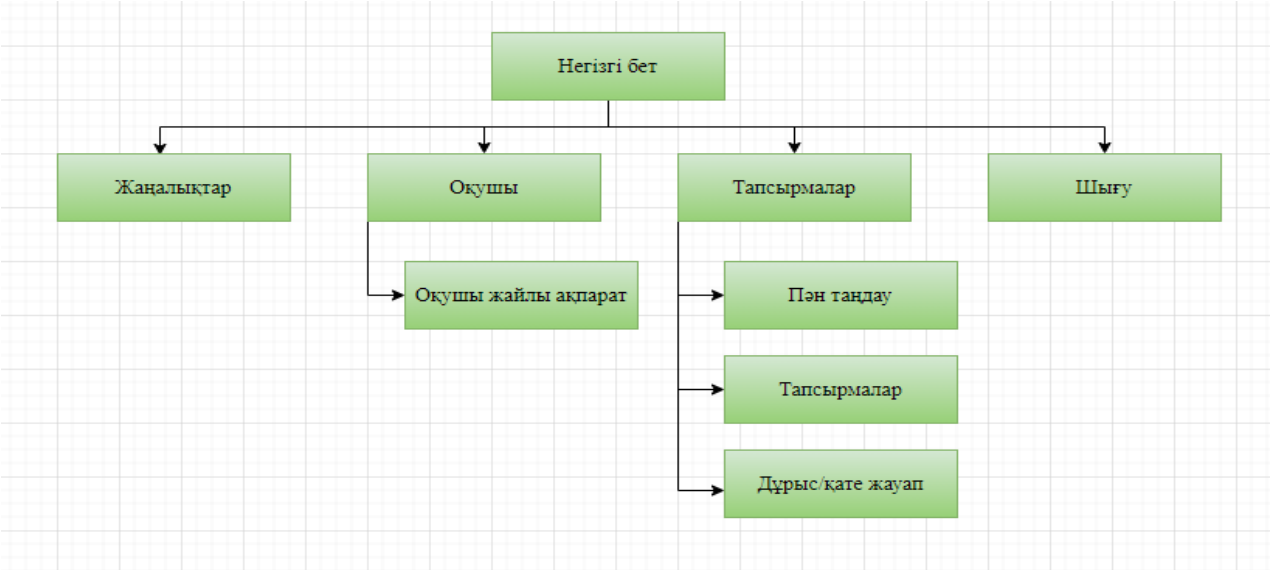
- ешбір заманауи браузер JavaScript тілінің қолдауынсыз өтпейді;
- пайдалы функционалдық параметрлер;
- үздіксіз жақсарту тілі: JavaScript2 проектінің бета нұсқасы әзірленеді;
- Microsoft Office және Open Office сияқты мәтіндік редакторларды пайдаланатын бағдарламалармен өзара әрекеттеседі.

Жалпы мағлұматтар. Осындай амал, жоспардың мақсаты - жаңа ақпараттық технологияларды, соның ішінде интернет желісін пайдалану базасында мекеменің осындай концепциясын құру болып келеді. Бұл басты теория клиент-сервердің архитектурасына және интернет желісінің қабілетіне негізделген болып табылады.

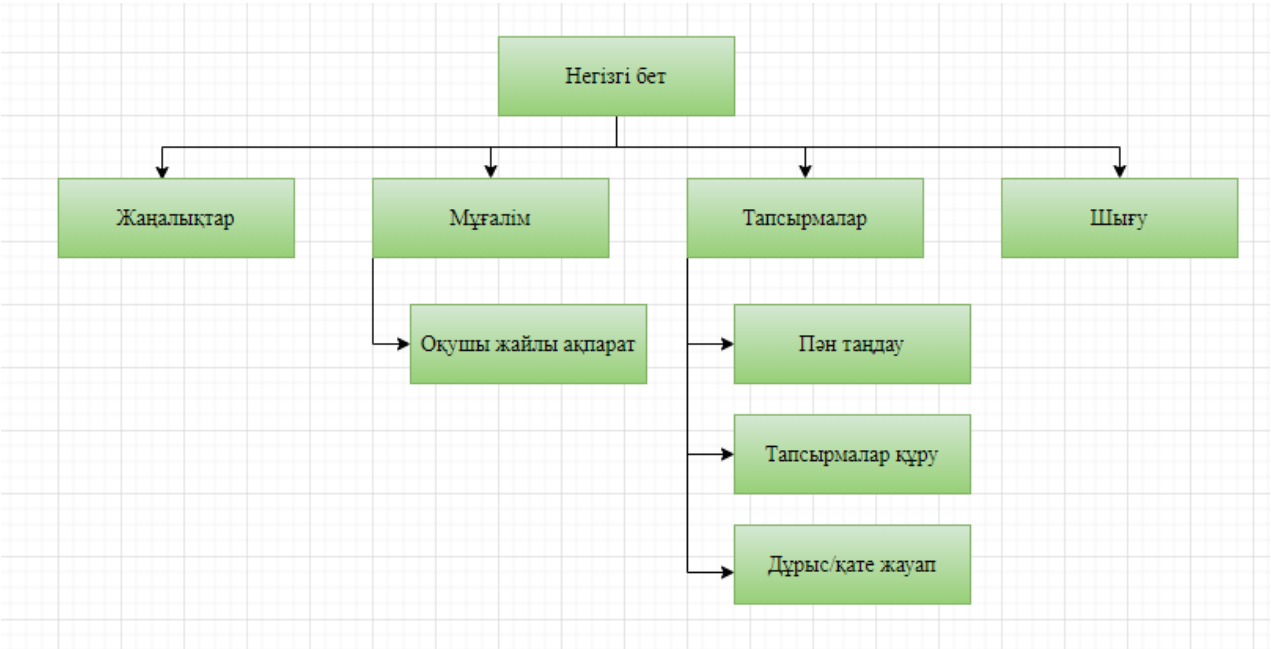
Visual Studio Code редакторы – бұл бастапқы код редакторы. Ол бағдарламалаудың бірнеше тіллдерін, синтаксисті бөлектеуді, рефакторингті, түзетуді, кодты навигацияны қолдайды. Сонымен қатар Visual Studio құжатты, жол беру символдарын және ағымдағы құжаттың бағдарламалау тілін сақтау кезінде код бетін ауыстыруға мүмкіндік береді.

3.2 Функционалдық құрылымдық сұлба

Төмендегі сызбада программа жұмысының құрылымдық сұлбасы көрсетілген. Сызбада веб-парақшаның бастапқы беттері көрсетілген.

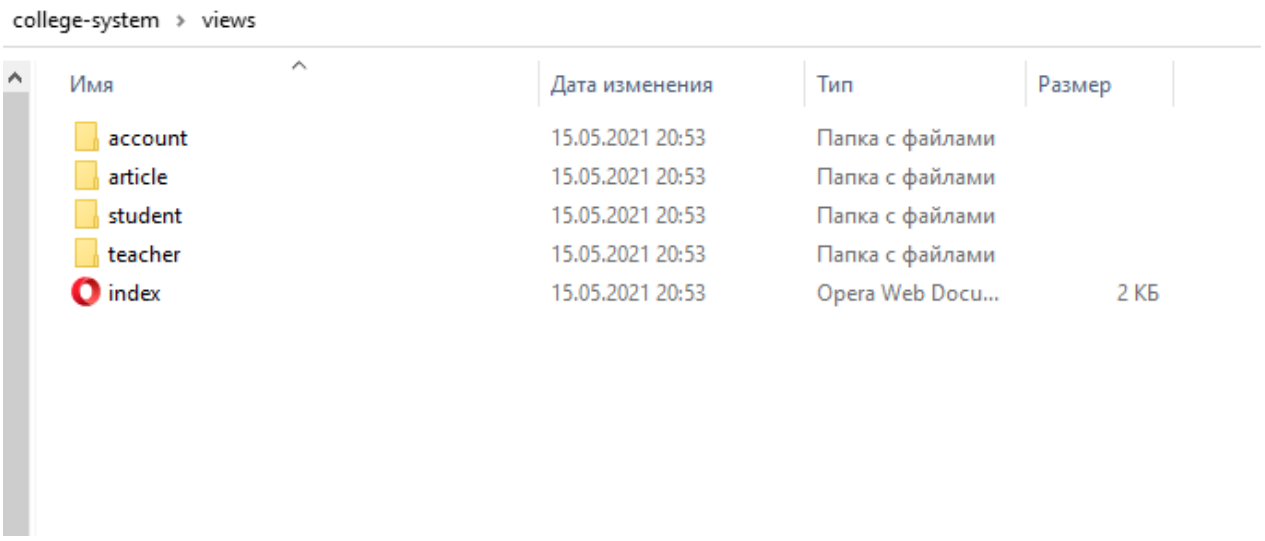


3.2.1-сурет – Веб-сайттың оқушы аты бойынша құрылымдық сұлбасы



3.2.2-сурет. Веб-сайттың мұғалім аты бойынша құрылымдық сұлба

3.3 Бағдарламаның физикалық құрылымы



3.3-сурет. Бағдарламаның негізгі физикалық құрылымы

Student – веб-сайттың студент бөлігінің файлы
Teacher – мұғалім бөлігінің файлы
Index – веб-сайттың негізгі файлы
Account – регистрация/авториация бөлігінің файлы
Article – админ бөлігің файлы

3.4 Кіріс мәліметтер

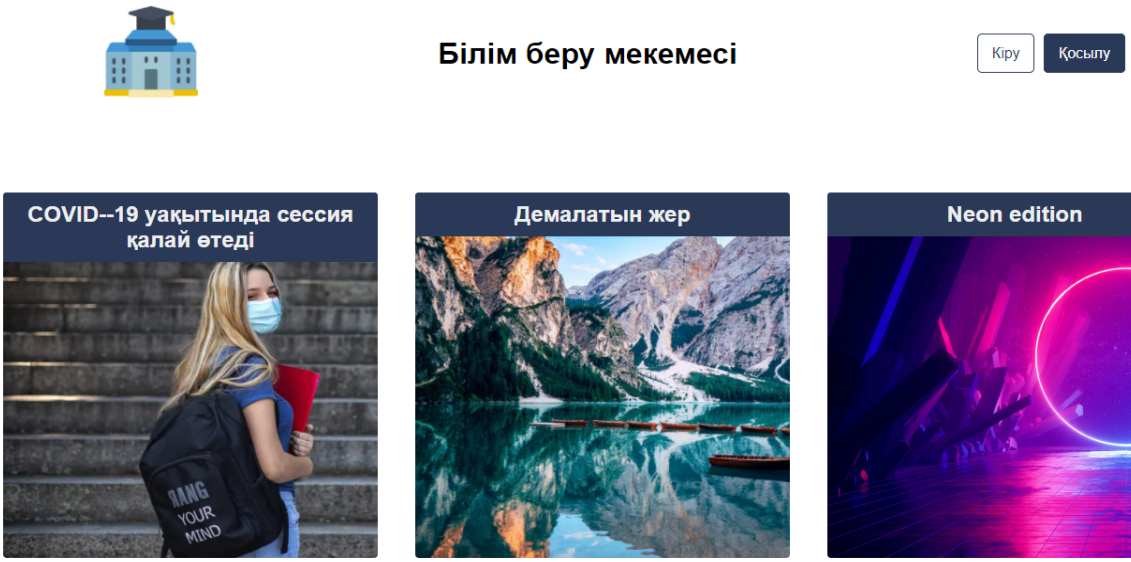
Сайттың кіріс мәліметтері ретінде бағдарламаға пайдалунышылардың негізгі функционал батырмаларын басуын келтіруге болады.

3.5 Шығыс мәліметтер

Сайттың шығыс мәліметтері деп серверден жіберілген жауапты, яғни нәтиже ретінде алуға болады.

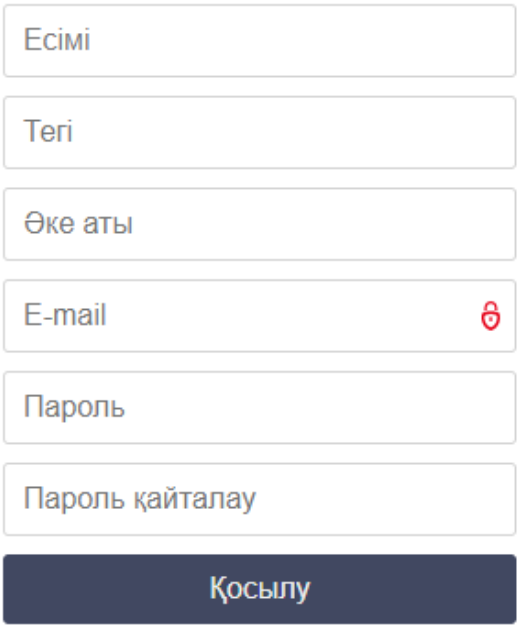
3.6 Веб-сайтты тестілеу

Веб-сайтты шақыру үшін серверде ашу үшін Visual Studio Code бағдарламалық ортасын ашып, “Terminal” – ға “npm start” жолын енгіземіз, содан кейін “Enter” басамын. Басқаннан кейін Opera Browser-де парақша ашылады. Парақшаға автоматты түрде кірмеген жағдайда, http://localhost/8080 жолын енгіземіз. Сол кезде бізге керек веб-сайт ашылады. Сайт ашылған соң, алғашқы беті төмендегі суретте көрсетілген.



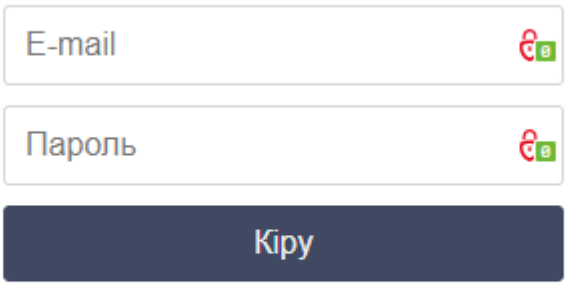
3.6.1-сурет. Веб-сайттың негізгі парақшасы

Келесі суретте, сайтқа “Қосылу” немесе, тіркелген болса сайтқа “Кіру” батырмасын басамыз. Төменде “Қосылу”, яғни тіркелу беті көрсетілген:

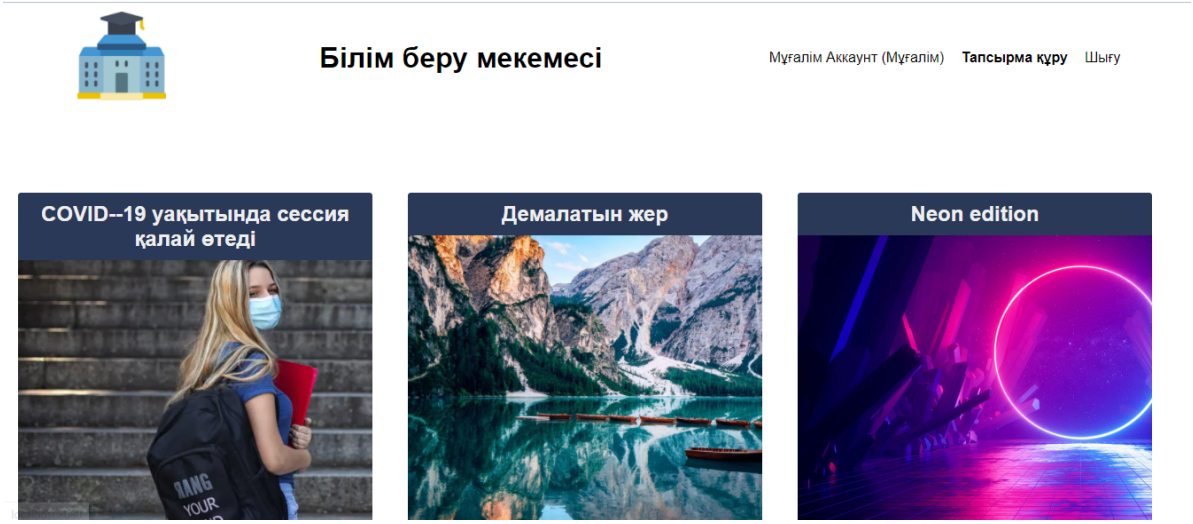


3.6.2-сурет. Веб-сайтқа тіркелу беті

Веб-сайтқа тіркеліп болғаннан кейін, жүйеге қосылу процесі орындалады.

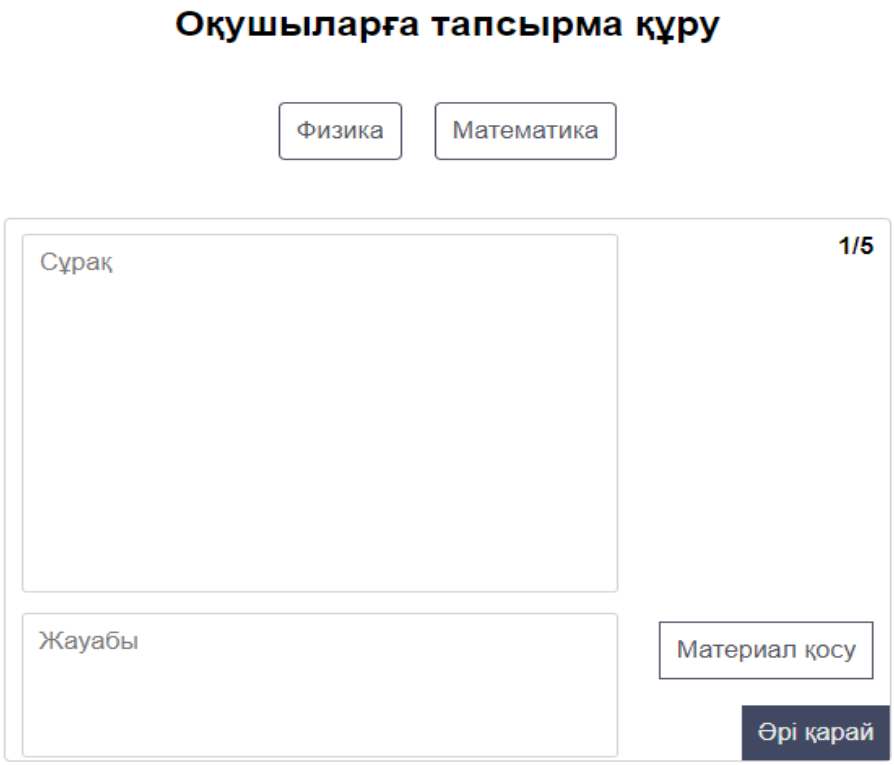


3.6.3-сурет. Қосылу(авторизация) процесі



3.6.4-сурет. Мұғалім аккаунтының беті

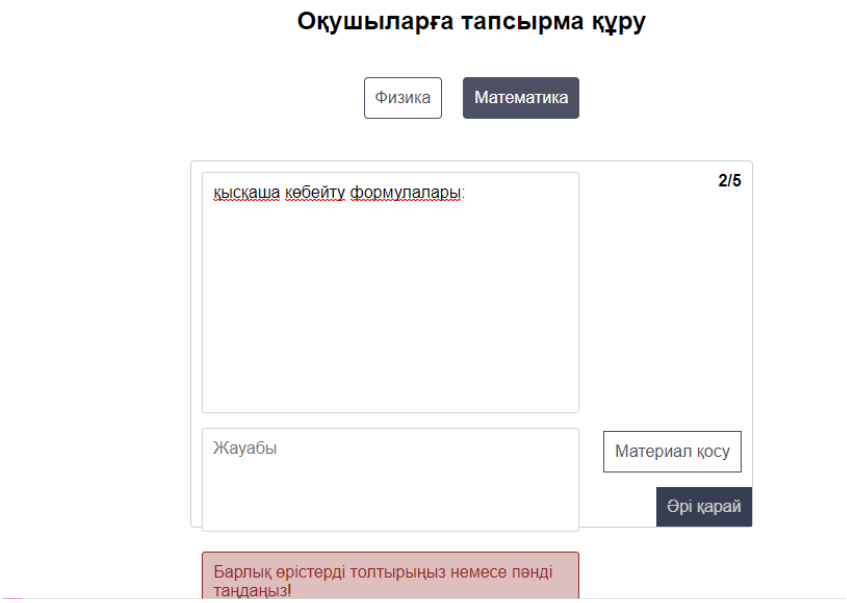
Мұғалімнің атынан кірген соң, “Тапсырма құру” тетігін басу арқылы мұғалім тапсырмалар құра алады. Сонымен қатар, тапсырма құру барысында тапсырманың үстіне видео түрінде материал жүктеп, оқушының есте сақтауы үшін, видеоға қатысты тапсырма-сұрақ қоя алады. Ол сурет төменде көрсетілген:



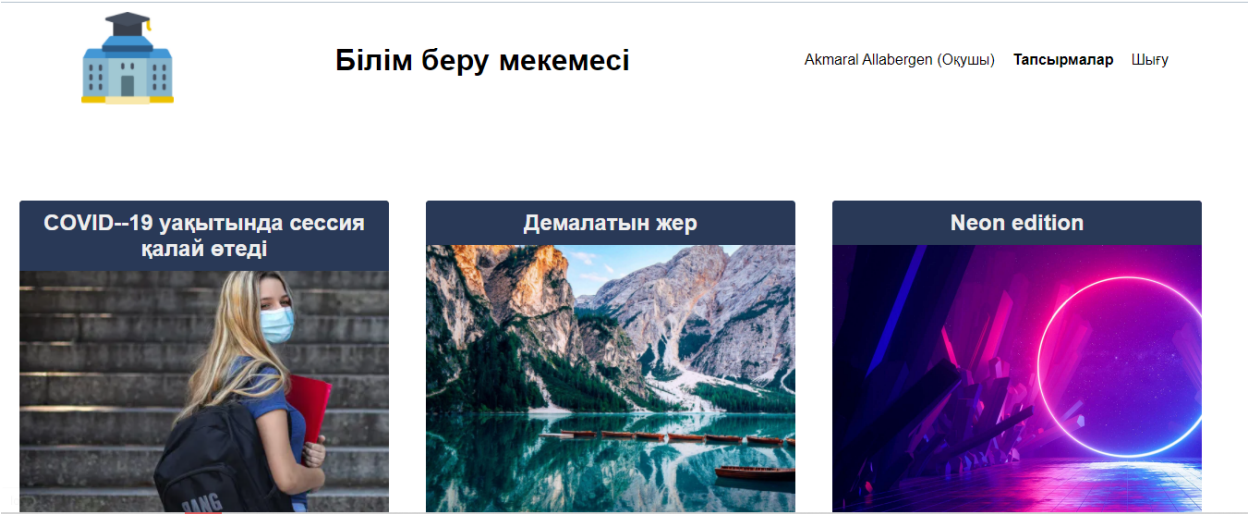
3.6.5-сурет. Оқушыларға тапсырма құру беті

Тапсырма құру барысында белгілі бір өріс толтырылмай қалған

жағдайда, өрістердің барлығын толтыру жөніндегі қате көрсетіледі, яғни төмендегі суретте көрсетілген:



3.6.6-сурет. Өрістерді тодтыру туралы қате

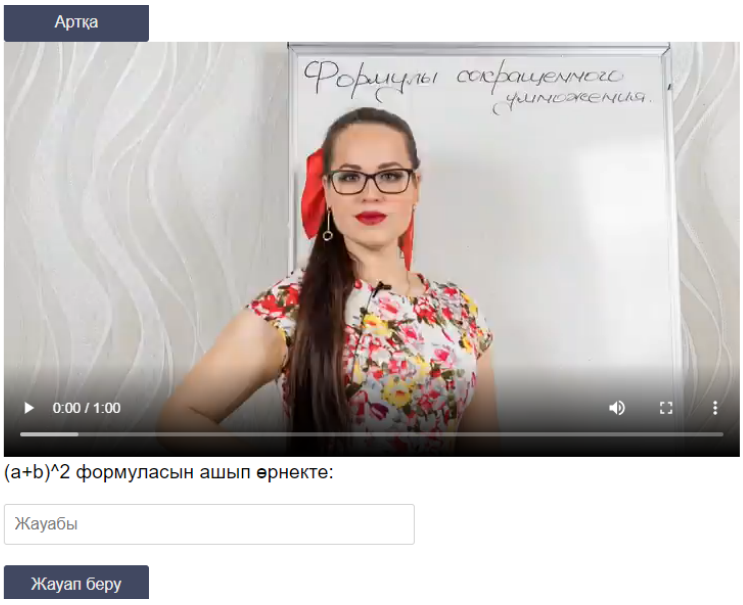


3.6.7-сурет. Оқушының аккаунтынан кіру беті

Оқушы аккаунтынан кіріп, “Тапсырмалар” тетігін басқан кезде, оқушыға мұғалім құрған тапсырмалар көрсетіледі, яғни ол жерде де пән таңдайды, пәнді таңдап болған соң тапсырмалар тізімі көрсетіледі:

3.6.8-сурет.Тапсырмалар беті

Келесі суретте жүктелген видеометариал мен тапсырмаға жауап беру беті:



3.6.9-сурет. Тапсырмаларды орындау беті

Тапсырмаларды оырндап болғаннан кейін, осындай тапсырмалардың қате және дұрыс жауаптарып көрсетіледі:

3.6.10-сурет. Тапсырмалардың дұрыс және қате жауаптарын көрсет

ҚОРЫТЫНДЫ

Қоғамның қазіргі даму кезеңі компьютерлендіру кезеңі деп аталады. Қазіргі заманғы материалдарды жаңарту, білім беру және басқа да салалар ақпараттық қызметтерді, оның көп санын өңдеуді талап етеді. Заманауи талаптарға жауап беретін ақпараттың өсуі кезеңінде білім беру технологиялары мен білім берудің жаңа әдістерін іздеу қажет. Интернет- технологияларды және онлайн-оқытуды қолдану студенттерге жаңа мүмкіндіктер ашады және оқытуды қолжетімді етеді.

Дипломдық жұмысты орындау барысында төмендегі міндеттер қойылып, шешімі табылды:

- оқыту мекемесінің WEB-парақшасының сұлбасы келтірілді;
- UML және ER диаграммалар көмегімен ақпараттық жүйе жобаланды;
- программалық қамтаманың жұмыс істеу алгоритмдері жобаланды;
- мәліметтер базасының құрылымы жобаланып құрылды;
- пайдаланушыларға сайт интерфейсінің жеңілдігі қарастырылды;
- білім беру мекемесінің WEB-сайты іске асырылды.

Ақпараттық технологиялардың даму деңгейі веб-сайтты бүкіл әлемге танытудың басты құралына айналдырады.

Дипломдық жұмысты жүзеге асыру барысында заманауи веб-технологиялар таңдалып, пайдаланылды. Веб-сайтты жасау барысында Html (HyperText Markup Language), CSS (Cascading Style Sheets), JavaScript бағдарламалау тілдері пайдаланылды. Мәліметтер жүйесін сақтау үшін MongoDB құжатқа негізделген мәліметтер қорын басқару жүйесі пайдаланылады, сонымен қатар Visual Studio Code бастапқы код редакторы пайдаланыла отырып ойдағыдай web – ресур құрылды.

Сонымен, дипломдық жұмыс білім беру жаңалықтарын қарау, электронды ресурстарды оқу және тапсырмалар орындау сияқты функцияларды ұсынатын дамыған бағдарламалық жасақтама арқылы репетиторлық білім беру мекемелерінде білім деңгейін дамытуды ұсынады.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 Web-сайттарды құру: Өзін-өзі пайдалану нұсқаулығы / И.В.Панфилов, Триумф, 2008. – 485 б.

2 Байтенова Л.М., М.К.Уандыкова. Ақпараттық жүйелерде мәліметтерді басқару. – Алмат: 2016.

3 И.В.Панфилов, Web-сайттарды құру: Өзін-өзі пайдалану нұсқаулығы / И.В.Панфилов, Триумф, 2008. – 485 б.

4 Астаубаева Г.Н. Интернет программалау. – Алматы 2016.

5 Джон Д. HTML мен CSS. Web-сайттарды әзірлеу және дизайны, 2013 – 115 б.

6 Титоров Д.Ю. Білім беру мекемесінің интранет желісін құру. Информатика және білім беру, №6, 2003 ж.

7 Гендина.Н.И. Веб-сайттардың мазмұнын жобалаудың лингвистикалық құралдары [Мәтін] / Н.И. Гендина // Ғылыми-техникалық кітапханалар. - 2008. - №3. - С. 5-14

8 Инькова. Н.А. Веб-сайттар құру: Оқу-әдістемелік құрал [Электрондық ресурс] / Инькова Н.А., Зайцева Е.А., Кузьмина Н.В., Толстих С.Г. // Кіру режимі: http://club-edu.tambov.ru/methodic/fio/p5.doc

9 Джон Д. HTML мен CSS. Web-сайттарды әзірлеу және дизайны, 2013 – 115 б

10 Рогачева, Г.И. Қазіргі ақпараттық білім беру ресурстары [Мәтін] / Г.И. Рогачева // Мектептегі білім беруді ақпараттандыру: Халықаралық материалдар. ғылыми-практикалық конф. 17-18 қыркүйек 2002 ж. - Барнаул, қалам, А. Темпера. - М., 2006. - 61 б.

11 Сорников Я.А. DreamWeaver көмегімен веб-сайт жасау үшін стильдерді қолдану. М: Интернеттегі білім беру сұрақтары, № 39.

ҚОСЫМША

Негізгі бет(index.html):

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/standard/start.css">
    <link rel="stylesheet" href="css/standard/input-button.css">
    <link rel="stylesheet" href="css/style.css">
    <script src="js/initialize.js" defer></script>
    <script src="js/article/articles.js" defer></script>
    <script src="js/article/pagination.js" defer></script>
    <title>Колледж</title>
</head>
<body>
    <header class="flex-justify-center">
        <div class="logo-link">
            <a href="/">
                <img src="img/Logo.png">
            </a>
        </div>
        <div class="header-title">
            <h1>Білім беру мекемесі</h1>
        </div>
```

```
        <div id="changable-box"></div>
    </header>
    <main>
        <div class="college-article flex-block" id="articles"></div>
    </main>
    <div id="pagination">
        <div id="pagination-btn">

        </div>
    </div>
</body>

</html>
```

Регистрация/авторизация:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/standard/input-button.css">
    <link rel="stylesheet" href="css/standard/start.css">
    <link rel="stylesheet" href="css/auth.css">
    <script src="js/showMessage.js" defer></script>
```

```
    <script src="js/account.js" defer></script>
    <title>Авторизация</title>
</head>
<body>
    <form method="POST">
        <input type="text" placeholder="E-mail" name="email"><br>
        <input type="password" placeholder="Пароль" name="password"><br>
        <button type="button" onclick="Login()">Кіру</button>
        <div id="message"></div>
    </form>
</body>

</html>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/standard/input-button.css">
    <link rel="stylesheet" href="css/standard/start.css">
    <link rel="stylesheet" href="css/signup.css">
    <script src="js/showMessage.js" defer></script>
    <script src="js/account.js" defer></script>
```

```
    <title>Регистрация</title>
</head>
<body>
    <form method="POST">
        <input type="text" placeholder="Есімі" name="name"><br>
        <input type="text" placeholder="Тегі" name="surname"><br>
        <input type="text" placeholder="Әке аты" name="middleName"><br>
        <input type="text" placeholder="E-mail" name="email"><br>
        <input type="password" placeholder="Пароль" name="password"><br>
        <input type="password" placeholder="Пароль қайталау" name="passwor
dConfirm"><br>
        <button type="button" onclick="Register()">Қосылу</button>
        <div id="message"></div>
    </form>
</body>
</html>
```

Тапсырма орындау HTML:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/standard/start.css">
    <link rel="stylesheet" href="css/standard/input-button.css">
```

```
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/tasks.css">
    <script src="js/showMessage.js" defer></script>
    <script src="js/student/taskExecute.js" defer></script>
    <title>Тапсырмалар орындау</title>
</head>

<body>
    <div class="title" align='center'>
        <h2>Тапсырмалар</h2>
        <span>Пәнді таңдаңыз</span>
    </div>
    <div class="choosen-block">
        <div class="item-btn" onclick="getTasksByItem('Физика')">
            <img src="img/atom.png">
            <p align='center'>Физика</p>
        </div>
        <div class="item-btn" onclick="getTasksByItem('Математика')">
            <img src="img/calculator.png">
            <p align='center'>Математика</p>
        </div>
    </div>
    <div id="tasks-block"></div>
    <div id="task-execute"></div>
</body>
```

```
</html>
```

Тапсырма құру HTML:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/standard/start.css">
    <link rel="stylesheet" href="css/standard/input-button.css">
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/teacher/task.css">
    <script src="js/showMessage.js" defer></script>
    <script src="js/teacher/task.js" defer></script>
    <title>Есеп құру</title>
</head>

<body>
    <div class="title" align='center'>
        <h2>Оқушыларға тапсырма құру</h2>
    </div>
    <div class="item-choose">
        <span class="item">Физика</span>
```

```
            <span class="item">Математика</span>
        </div>
        <div class="task-constructor">
            <textarea type="text" id="task-
question" placeholder="Сұрақ"></textarea><br>
            <textarea type="text" id="task-
answer" placeholder="Жауабы"></textarea><br>
            <label for="material">Материал қосу</label>
            <input type="file" id="material" name="material">
            <button onclick="AddTask()">Әрі қарай</button>
            <div class="task-count">
                <h4 id="current-task">1/5</h4>
            </div>
            <span id="message"></span>
        </div>
    </body>


    </html>
```

Admin.html:

```
    <!DOCTYPE html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/standard/start.css">
```

```
        <link rel="stylesheet" href="css/standard/input-button.css">
        <link rel="stylesheet" href="css/article/admin.css">
        <script src="js/showMessage.js" defer></script>
        <script src="js/article/send.js" defer></script>
        <title>Мақала қосу (Админ)</title>
    </head>
    <body>
        <a href="/" class="to-main-page"><img src="img/left-
arrow.png" align='center'>Бас бетке</a>
        <form method="POST" enctype="multipart/form-data">
            <input type="text" placeholder="Тақырыбы" name="title"><br>
            <textarea placeholder="Мақала мәтіні" name="text"></textarea><br>
            <label for="article-photo">Сурет қосу</label>
            <input type="file" id="article-photo" name="articlePhoto"><br>
            <button type="button" onclick="PublishArticle()">Мақаланы жариялау<
/button>
            <div id="message"></div>
        </form>
    </body>
    </html>
```

Мақалаларды көру коды:

```
    <!DOCTYPE html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Мақала тақырыбы</title>
    <link rel="stylesheet" href="../css/standard/input-button.css">
    <link rel="stylesheet" href="../css/standard/start.css">
    <link rel="stylesheet" href="../css/article/aritcleView.css">
    <script src="../js/article/getArticle.js" defer></script>
</head>
<body>
    <div class="article">
        <h1 id="title"></h1>
        <p id="article-text"></p>
        <img src="" id="article-image">
    </div>
</body>
</html>
```

Start.js

```
require("@babel/register")({
    presets: ["@babel/preset-env"]
});

module.exports = require('./app.js')
```

app.js:

```
import express from "express";
import mongoose, { mongo } from "mongoose";
import bodyParser from "body-parser";
```

```
import mainRoutes from "./server/routes/main"
import pageRoutes from "./server/routes/pages"

const config = require('./server/config');

const app = express();

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));

app.use(express.static(__dirname + "/public"));
app.use(express.static(__dirname + "/views"));

mongoose.set('useCreateIndex', true);
mongoose.connect(config.mongoURI, { useUnifiedTopology: true, useNewUrl
Parser: true, useFindAndModify: false })
    .then(() => {
        console.log("Database connected");
    })
    .catch(error => {
        console.log("Error connecting to database, ", error);
    });

const port = 8080;

app.use('/', pageRoutes)
```

```
app.use("/api/", mainRoutes);
app.get('*', (req, res) => res.status(400).send('Ошибка 404'))

app.listen(port, () => {
   console.log(`Our server is running on port ${port}`);
});
```

taskExecute.js:

```
const tasksBox = document.getElementById('tasks-block')
const executeBlock = document.getElementById('task-execute')
let answerInput = document.getElementById('answer')

let currentItem = ''
function getTasksByItem(item) {
   currentItem = item
   tasksBox.style.display = 'block'
   executeBlock.style.display = 'none'
   let videoOff = document.querySelector('video');
   if (videoOff)
   {
      videoOff.remove()
   }
   const xhr = new XMLHttpRequest()
   xhr.open('GET', 'api/tasks/' + item, true)
   xhr.setRequestHeader('Authorization', sessionStorage.getItem('token'))
   xhr.send()
```

```
    xhr.onreadystatechange = () => {
        if (xhr.readyState === 4) {
            if (xhr.status >= 200 && xhr.status < 400) {
                const response = JSON.parse(xhr.response)
                let tasksHTML = ''
                response.data.forEach((x, index) => {
                    tasksHTML += `<div class="task">
                    <p>Тапсырма#${index + 1} (${item})</p>
                    <button onclick="getTaskQuestions('${x._id}')">Өту</button>
                </div>`
                });

                tasksBox.innerHTML = tasksHTML
            }
        }
    }
}

let taskQuestions = {}
let currentQuestion = {}
let questionIndex = 0;

let wrongAnswer = 0;
```

```
const materialVideo = document.getElementById('material-video')
function getTaskQuestions(taskId) {
    const xhr = new XMLHttpRequest()
    xhr.open('GET', 'api/task/questions/' + taskId, true)
    xhr.setRequestHeader('Authorization', sessionStorage.getItem('token'))
    xhr.send()

    xhr.onreadystatechange = () => {
        if (xhr.readyState === 4) {
            if (xhr.status >= 200 && xhr.status < 400) {
                const response = JSON.parse(xhr.response)
                if (response.data.length > 0) {
                    executeBlock.style.display = 'block'
                    tasksBox.style.display = 'none'
                    let material = ''
                    if (response.material !== undefined) {
                        material = `<video src="${'../img/article/' + response.material.name}" id="material-video" controls="controls"></video>`
                    }
                    taskQuestions = response.data
                    currentQuestion = taskQuestions[questionIndex]
                    executeBlock.innerHTML = `<div class="question">
                    <button onclick="getTasksByItem('${currentItem}')" class='btn-back'>Артқа</button>
                    <br>
                    ${material}
```

```
                    <p>
                        ${currentQuestion.question}
                    </p>
                    <input type="text" name="answer" id="answer" placeholder="Жауабы">
                    <button onclick="answerTheQuestion()">Жауап беру</button>
                </div>`
                    answerInput = document.getElementById('answer')
                }
            }
        }
    }

    function answerTheQuestion() {
        if (!answerInput.value.toLowerCase().includes(currentQuestion.answer.toLowerCase())) {
            wrongAnswer++;
        }

        if (questionIndex === taskQuestions.length - 1) {
            executeBlock.innerHTML = `<br>Дұрыс жауап - ` + ((questionIndex - wrongAnswer) + 1)
            executeBlock.innerHTML += `<br>Қате жауап - ` + wrongAnswer
        } else {
            questionIndex++;
```

```
        currentQuestion = taskQuestions[questionIndex]
        executeBlock.innerHTML = `<div class="question">
        <p>
            ${currentQuestion.question}
        </p>
        <input type="text" name="answer" id="answer" placeholder="Жауабы">
        <button onclick="answerTheQuestion()">Жауап беру</button>
    </div>`
        answerInput = document.getElementById('answer')
        answerInput.value = ""
    }
}
```

Task.js:

```
const items = document.getElementsByClassName('item')
let choosenItem = "";
for (let i = 0; i < items.length; i++) {
    items[i].addEventListener('click', () => {
        choosenItem = items[i].innerHTML
        for (let i = 0; i < items.length; i++) {
            items[i].classList.remove('active')
        }
        items[i].classList.add('active')
    })
}
```

```
        let taskList = []

        const question = document.getElementById('task-question')
        const answer = document.getElementById('task-answer')
        const counter = document.getElementById('current-task')
        const maxCount = 5

        let material = document.getElementById('material')

        function AddTask() {
            if (question.value === "" || answer.value === "" || choosenItem === "") {
                ShowMessage("Барлық өрістерді толтырыңыз немесе пәнді таңдаңыз!
", true)
                return
            }

            taskList.push({
                question: question.value,
                answer: answer.value
            })
            question.value = ""
            answer.value = ""

            if (taskList.length === maxCount) {
                const token = sessionStorage.getItem('token')
                if (token) {
```

```
        const formData = new FormData()

        const xhr = new XMLHttpRequest()

        formData.append('subject', choosenItem)
        formData.append('tasks', JSON.stringify(taskList))
        formData.append('file', material.files[0])

        xhr.open('POST', '/api/task', true)
        xhr.setRequestHeader('Authorization', token)

        xhr.send(formData)
        xhr.onreadystatechange = () => {
            if (xhr.readyState === 4) {
                const response = JSON.parse(xhr.response)
                ShowMessage(response.message, xhr.status >= 400)
            }
        }
    }
    taskList = []
    counter.innerText = "1/5"
    return
}

counter.innerText = (taskList.length + 1) + "/5"
```

```
}
```

Account.js:

```
const form = document.querySelector('form')


function Register() {
    const { name, surname, middleName, email, password, passwordConfirm } =
form.elements
    const data = {
        name: name.value,
        surname: surname.value,
        middleName: middleName.value,
        email: email.value,
        password: password.value,
        passwordConfirm: passwordConfirm.value
    }
    const xhr = new XMLHttpRequest()
    xhr.open('POST', 'api/account/register', true)
    xhr.setRequestHeader('Content-Type', 'application/json;charset=UTF-8')
    xhr.send(JSON.stringify(data))


    xhr.onreadystatechange = async (e) => {
        if (xhr.readyState === 4) {
            const response = JSON.parse(xhr.response)


            if (xhr.status >= 200 && xhr.status < 400) {
                ShowMessage(response.message)
```

```
            setTimeout(RedirectTo, 2000, "http://localhost:8080/login")
        }
        if (xhr.status >= 400)
            ShowMessage(response.message, true)
    }
  }
}

function Login() {
  const { email, password } = form.elements
  const data = {
    email: email.value,
    password: password.value,
  }
  const xhr = new XMLHttpRequest()
  xhr.open('POST', 'api/account/login', true)
  xhr.setRequestHeader('Content-Type', 'application/json;charset=UTF-8')
  xhr.send(JSON.stringify(data))

  xhr.onreadystatechange = async (e) => {
    if (xhr.readyState === 4) {
      const response = JSON.parse(xhr.response)

      if (xhr.status >= 200 && xhr.status < 400) {
        ShowMessage(response.message)
```

```
                if (xhr.status === 200) {
                    const jwtDecoded = parseJwt(response.token)
                    sessionStorage.setItem('id', jwtDecoded.id)
                    sessionStorage.setItem('fullName', jwtDecoded.fullName)
                    sessionStorage.setItem('email', jwtDecoded.email)
                    sessionStorage.setItem('role', jwtDecoded.role)
                    sessionStorage.setItem('token', response.token)

                    setTimeout(RedirectTo, 1000, jwtDecoded.role === 'admin' ? '/ad
min' : '/')
                }
            }
            if (xhr.status >= 400)
                ShowMessage(response.message, true)
        }
    }
}

function parseJwt(token) {
    var base64Url = token.split('.')[1];
    var base64 = base64Url.replace(/-/g, '+').replace(/_/g, '/');
    var jsonPayload = decodeURIComponent(atob(base64).split('').map(function
(c) {
        return '%' + ('00' + c.charCodeAt(0).toString(16)).slice(-2);
    }).join(''));
```

```
    return JSON.parse(jsonPayload);
};

function RedirectTo(url = "/") {
    location.href = url
}
```

Initialize.js:

```
const changableBox = document.getElementById('changable-box')
const fullName = sessionStorage.getItem('fullName')
const role = sessionStorage.getItem('role')
SetPaginationFilterOptions()

if (sessionStorage.getItem('token')) {
    const linkName = fullName.split(' ')[0] + ' ' + fullName.split(' ')[1]
    changableBox.innerHTML = `<div class="profile-link">
                                <a href="/${role === 'admin' ? 'admin' : 'profile'}">${lin
kName + (role === 'admin' ? ' (Админ)' : role === 'teacher' ? ' (Мұғалім)' : role === '
student' ? ' (Оқушы)' :'')}</a>
                                ${role === 'teacher' ? '<a href="/createTask" class="task-
link">Тапсырма құру</a>' : ''}
                                ${role === 'student' ? '<a href="/tasks" class="task-
link">Тапсырмалар</a>' : ''}
                                <a href="/" onclick="Logout()">Шығу</a>
                            </div>`
} else {
    changableBox.innerHTML = `<div class="reg-auth">
                                <a href="/login">Кіру</a>
```

```
                    <a href="/register">Қосылу</a>
                </div>`
}

function Logout() {
    sessionStorage.removeItem('token')
    sessionStorage.removeItem('id')
    sessionStorage.removeItem('fullName')
    sessionStorage.removeItem('email')
}

function SetPaginationFilterOptions() {
    sessionStorage.setItem('page', 1)
    sessionStorage.setItem('size', 6)
}
```

showMessage.js

```
const messageBox = document.getElementById('message')

function ShowMessage(message = "", error = false) {
    messageBox.style.display = 'block'
    messageBox.className = error ? "error" : "success"
    messageBox.innerHTML = message

    setTimeout(HideMessage, 3000)
}
```

```
function HideMessage() {
    messageBox.style.display = 'none'
}
```

Controllers. articleController.js:

```
import isEmpty from "../functions/validation/is-empty";
import { Article } from "../models/ArticleModel"
import mongoose from "mongoose";
import { Image } from "../models/file/ImageModel";

export async function postArticle(req, res) {
    const { title, text } = req.body

    if (isEmpty(title) || isEmpty(text))
        return res.status(400).json({
            success: false,
            message: "Барлық өрістерді толтырыңыз"
        })
    const imageData = {
        name: req.file.filename,
        contentType: req.file.mimetype,
    }

    const image = new Image(imageData)
        .save()
```

```
.then(newImage => {
    const articleData = {
        _id: mongoose.Types.ObjectId(),
        title: title,
        text: text,
        image: newImage
    }

    const article = new Article(articleData)
    article.save()
        .then(newArticle => {
            return res.status(201).json({
                success: true,
                message: 'Мақала сәтті жарияланды!',
            });

        })
        .catch(error => {
            res.status(500).json({
                success: false,
                message: 'Ошибка сервера: ' + error.message
            })
        })
}).catch(error => {
    res.status(500).json({
```

```
            success: false,
            message: 'Ошибка сервера: ' + error.message
        })
    })
}

export async function getArticles(req, res) {
    const { page, size } = req.query
    const start = (page - 1) * size
    Article.countDocuments({})
        .then(count => {
            const pageCount = Math.ceil(count / size);
            Article.find({})
                .skip(start)
                .limit(parseInt(size))
                .populate('image')
                .then(articles => {
                    res.status(200).json({
                        success: true,
                        data: articles,
                        pageCount: pageCount,
                    })
                })
                .catch(error => {
                    res.status(500).json({
```

```
                success: false,
                message: error.message
            })
        })
    })
    .catch(error => {
        res.status(500).json({
            success: false,
            message: error.message
        })
    })
}

export async function getArticleById(req, res) {
    const { id } = req.params

    Article.findById(id)
    .then(article => {
        if (article)
            res.status(200).json({
                success: true,
                data: article
            })
    })
    .catch(error => {
```

```
            res.status(500).json({
                success: false,
                error: error.message
            })
        })
}
```

taskController.js:

```
import { Task } from "../models/TaskModel"
import mongoose from "mongoose"
import { TaskQuestion } from "../models/TaskQuestionModel"
import { Image } from "../models/file/ImageModel";

export async function createTask(req, res) {
    console.log(req.file);
    console.log(req.body);

    const imageData = {
        name: req.file.originalname,
        contentType: req.file.mimetype
    }

    console.log(imageData);

    const image = new Image(imageData)
    image.save()
```

```
.then(newFile => {
    const task = new Task({
        _id: mongoose.Types.ObjectId(),
        subject: req.body.subject,
        material: newFile
    })

    task.save()
        .then(newTask => {
            let tasks = JSON.parse(req.body.tasks)
            const waitPromise = new Promise((resolve, reject) => {
                tasks.forEach((x, index) => {
                    const taskQuestion = new TaskQuestion({
                        _id: mongoose.Types.ObjectId(),
                        question: x.question,
                        answer: x.answer,
                        task: newTask
                    })
                    taskQuestion.save()
                        .catch(err => {
                            res.status(400).json({
                                success: false,
                                message: "Ошибка: " + err
                            })
                        })
```

```
                    if (index === tasks.length - 1) resolve();
                });
            })
            waitPromise.then(() => {
                res.status(201).json({
                    success: true,
                    message: "Тапсырмалар жарияланды!"
                })
            })
        })
        .catch(err => {
            res.status(400).json({
                success: false,
                message: "Ошибка: " + err
            })
        })
    })
}

export async function getTasksByItem(req, res) {
    const { item } = req.params
    Task.find({ subject: item })
        .then(tasks => {
            return res.status(200).json({
                success: true,
```

```
            data: tasks
        })
    })
}

export async function getTaskQuestionsByTaskId(req, res) {
    const { id } = req.params
    Task.findById(id)
        .populate('material')
        .then(foundedTask => {
            TaskQuestion.find({ task: id })
            .then(taskQuestions => {
                return res.status(200).json({
                    success: true,
                    data: taskQuestions,
                    material: foundedTask.material
                })
            })
        })
}
```

userController.js:

```
import mongoose from "mongoose"
import isEmpty from "../functions/validation/is-empty"
import { UserRole } from "../models/dictionary/UserRoleModel"
import { User } from "../models/UserModel"
```

```
const express = require('express')
const router = express.Router()

const bcrypt = require('bcrypt')
const jwt = require('jsonwebtoken')
const passport = require('passport')

export async function registerUser(req, res) {
    let data = {
        _id: mongoose.Types.ObjectId(),
        email: req.body.email,
        name: req.body.name,
        surname: req.body.surname,
        middleName: req.body.middleName,
        password: req.body.password,
    }

    if (isEmpty(data.email) || isEmpty(data.name) ||
        isEmpty(data.surname) || isEmpty(data.middleName) ||
        isEmpty(data.password))
        return res.status(400).json({
            success: false,
            message: "Барлық өрістерді толтырыңыз"
        })
```

```
if (data.password !== req.body.passwordConfirm)
    return res.status(400).json({
        success: false,
        message: "Пароль сәйкес келмейді"
    })

User.exists({ email: data.email })
    .then(hasUserExists => {
        if (hasUserExists)
            return res.status(400).json({
                success: false,
                message: "Сіз терген Email-мен қазіргі өзінде тіркелген"
            })
        else {
            let role = req.body.role
            if (isEmpty(role))
                role = 'student'

            UserRole.findOne({ name: role })
                .then(foundRole => {
                    data.role = foundRole

                    const user = new User(data)
```

```
                bcrypt.genSalt(10, (err, salt) => {
                    if (err) return console.error("Bcrypt salt generate error: " + err
)

                    bcrypt.hash(user.password, salt, (err, hash) => {
                        if (err) return console.error("Bcrypt passsword hashing erro
r: " + err)

                        user.password = hash
                        user.save()
                            .then(newUser => {
                                return res.status(201).json({
                                    success: true,
                                    message: "Сіз сәтті тіркелдіңіз",
                                })
                            })
                            .catch(error => {
                                return res.status(422).json({
                                    success: false,
                                    message: "Ошибка регистрации: " + error,
                                    error: error.name
                                })
                            })
                    })
                })
            }).catch(err => {
```

```
                res.status(500).json({
                    message: err
                })
            })
        }
    })
}

export function loginUser(req, res) {
    const { email, password } = req.body
    User.findOne({ email: email })
        .populate('role')
        .then(user => {
            if (!user) {
                return res.status(400).json({
                    succes: false,
                    message: "Дұрыс емес логин немесе пароль"
                })
            }

            bcrypt.compare(password, user.password)
                .then(isMatch => {
                    if (isMatch) {
                        const name = user.name ? user.name : ""
                        const surname = user.surname ? user.surname : ""
```

```
        const middleName = user.middleName ? user.middleName : ""

        const paylaod = {
            id: user.id,
            fullName: name + " " + surname + " " + middleName,
            role: user.role.name,
            email: user.email
        }

        jwt.sign(paylaod, 'secret', {
            expiresIn: 86400
        }, (err, token) => {
            if (err)
                return console.error('There is some error in token', err)
            else {
                res.status(200).json({
                    success: true,
                    token: token,
                    message: "Авторизация..."
                })
            }
        })
    } else {
        return res.status(400).json({
            succes: false,
```

```
                        message: "Дұрыс емес логин немесе пароль"
                    })
                }
            })
        })
    }

    export function verifyToken(req, res, next) {
        var token = req.headers['authorization'];
        if (!token)
            return res.status(403).send({ success: false, message: 'No token provided.'
});
        token = token.replace('Bearer', '').trim()
        jwt.verify(token, 'secret', function (err, decoded) {
            if (err)
                return res.status(419).send({ success: false, message: 'Сессия истекла!
Авторизуйтесь снова.' });

            req.userId = decoded.id;
            next();
        });
    }

    router.get('/user/me', passport.authenticate('jwt', { session: false }), (req, res) =>
{
        return res.json({
```

```
        id: req.user.id,
        name: req.user.name,
        email: req.user.email
    });
});
```

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Сәтбаев университеті

Ғылыми жетекшінің пікірі

Дипломдық жұмыс

Аллаберган Ақмарал

5В070300 – Ақпараттық жүйелер

Тақырыбы: «Жеке білім беру мекемелеріне арналған WEB-сайт әзірлеу»

Бұл дипломдық жұмыс өзінің логикалық құрылымымен ерекшеленген. Түсіндірме жобаның құрамы кіріспеден, 3 бөлімнен, қорытындыдан, әдебиеттер тізімінен және қосымшадан тұрады.

Менің пікірімше, диплом жобалаушы алдына қойылған тапсырманы толығымен орындады және кейінгі технологияларын меңгергендігін көрсетті.

Жалпы дипломдық жоба профессионалдық деңгейде орындалған. Түсіндірме жазба сауатты бейнеленген, жоба бойынша барлық қажетті ақпараттар бар.

Кемшілік ретінде кейбір шағын стилистикалық қателерді атап кетуге болады.

Жоғарыда айтылғандарға байланысты, дипломдық жұмыс 5В070300 – «Ақпараттық жүйелер» мамандығының бітіру жұмыстарына қойылатын талаптарына сәйкес және дипломдық жұмыс қорғауға жіберіле алады, ал оның авторы Аллаберган Ақмарал бакалавр академиялық дәрежесін алуға лайықты деп есептеймін.

Ғылыми жетекші

Лектор  Зиро А.А.

«24» мая 2021 ж.

Протокол анализа Отчета подобия

заведующего кафедрой / начальника структурного подразделения

Заведующий кафедрой / начальник структурного подразделения заявляет, что ознакомился(-ась) с Полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Аллаберган Ақмарал

Название: «Жеке білім беру мекемелеріне арналған» WEB-сайт әзірлеу

Координатор: Зиро А.А.

Коэффициент подобия 1: 10,12
Коэффициент подобия 2: 7.49

Замена букв:27 Интервалы:4

Микропробелы:7

Белые знаки:0

После анализа отчета подобия заведующий кафедрой / начальник структурногоподразделения констатирует следующее:

☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, работа признается самостоятельной и допускается к защите;

☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы по существу и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;

☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста, указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, работа не допускается к защите.

Обоснование:

После анализа отчета по плагиату и работы дипломника выявлено, что заимствования являются добросовестными и не обладают признаками плагиата, так в основном связаны с применением обще известных терминов.

……  ……

Дата. 25.05.2021 *Подпись заведующего кафедрой /*

начальника структурного подразделения

Протокол анализа Отчета подобия Научным руководителем

Заявляю, что я ознакомился(-ась) с Полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Аллаберган Ақмарал

Название: «Жеке білім беру мекемелеріне арналған» WEB-сайт әзірлеу

Координатор:Зиро А.А.

Коэффициент подобия 1: 10,12

Коэффициент подобия 2: 7.49

Замена букв:27

Интервалы:4

Микропробелы:7

Белые знаки: 0

После анализа Отчета подобия констатирую следующее:

- ☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, признаю работу самостоятельной и допускаю ее к защите;
- ☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы по существу и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;
- ☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста,указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, не допускаю работу к защите.

Обоснование:

После анализа отчета по плагиату и работы дипломника выявлено, что заимствования являются добросовестными и не обладают признаками плагиата, так в основном связаны с применением обще известных терминов

…… ……

Дата 24.05.2021 *Подпись Научного руководителя*