ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Кибернетика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
Кафедра меңгерушісі
тех. ғыл. кандидаты,
ассоц. профессор
_____ Юнусов Р.
“_____” _____ 2020ж.

ДИПЛОМДЫҚ ЖҰМЫС

Тақырыбы: “Electronic Timesheet” электронды табель жүйесі

Мамандығы 5В070400 – Есептеу техникасы және бағдарламалық қамтамасыз ету

Орындаған Ахетова Қ.Ғ.

Ғылыми жетекші
Техн. ғыл. магистрі, сениор-лектор
_____ Куникеев А.
“____” _____ 2020ж.

Алматы 2020

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: Электронды табель - шығу табелін автоматтандыру

Название: Ахетова Қарлығаш Ғалымқызы

Координатор:Жибек Алибиева

Коэффициент подобия 1:0,6

Коэффициент подобия 2:0

Замена букв:62

Интервалы:0

Микропробелы:0

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:

……

…… ……

Дата *Подпись Научного руководителя*

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Кибернетика және ақпараттық технологиялар институты

5В070400 – Есептеу техникасы және бағдарламалық қамтамасыз ету мамандығы

Ахетова Қарлығаш Ғалымқызы

Дипломдық жоба

Тақырыбы: «Electronic Timesheet» электронды табель жүйесі

ҒЫЛЫМИ ЖЕТЕКШІНІҢ ПІКІРІ

Берілген дипломдық жобада университетіміз қызметкерлерінің еңбекақысын есептеу негізінде қолданылатын, олардың жұмыс уақыты мен жауапты қызмет иелерінің растау ретінде қол қоюын қамтитын құжаттардың толтыру барысын электронды жүйеге толық ауыстыру қарастырылған. Жүйе Сәтбаев университеті қызметіне арналып жасалынған. Бірқатар қызметкерлердің жұмыс барысын жеңілдетуге және басты ерекшелігі – электронды қол қою жүйесін енгізуге негізделгендіктен пайдасы тиері сөзсіз.

Дипломдық жобаны Ахетова Қарлығаш Ғалымқызы және Берікқалиева Айтолқын Бекбергенқызы командалық жүйеде жоспарлы түрде жасап шықты. Зерттеу барысында жүйенің іске асу барысын толық анықтап, оқу орнымыздың осы жүйеге қатысты барлық дерлік қызметкерлерімен кездесіп, жобаға қатысты сұрақтарына жауап алып, қосымшаны қалай жүзеге асыру қажет екенін, қандай ақпараттарды, функционалдарды қамту керектігін жобалады. Қазіргі таңда көп қолданысқа ие озық технологияларды салыстыра келе жобаны әзірлеуге Angular, Django және PostgreSQL бағдарламалар аясын таңдады. Әр бағдарламаның ерекшеліктерін қарастыра отырып, олардың қажетті мүмкіндіктерін пайдаланып, электронды табель жүйесін бірлесе жасап шығарды.

Электронды табель жүйесі акт, жұмыс табелі және демалыс табельдерін қамтиды. Тікелей қолданушылары – құжаттарды құрастырушы және дұрыстығына қарай растайтын басқарма мүшелері. Осыған орай, жүйеде табельші, растаушы және авторизация бөліктері жүзеге асырылды.

Команда мүшесі Берікқалиева Айтолқын Бекбергенқызы растаушы, авторизация интерфейсін және серверлік бөлігін, ал Ахетова Қарлығаш Ғалымқызы табельші интерфейсі мен оның серверлік бөлігін әзірледі.

Студенттер тарапынан ұсынылған жоба ыңғайлы, түсінікті және қарапайым интерфейске ие. Нәтиже құжаттары қолданыстағы құжаттың толтырылу стандарттарына толығымен сай.

Жұмыс барысында бар зейініндерін сала отырып, әрбір бөліктерін мұқият зерттеп, қойылған талаптарға сай және жүйелі түрде жасап, жақсы нәтиже көрсете білді.

Осы нәтижелер негізінде, Ахетова Қарлығаш Ғалымқызы Есептеу техникасы және бағдарламалық қамтамасыз ету – 5В070400 бакалавры академиялық дәрежесін беруге лайық деп есептеймін.

Ғылыми жетекші: «Программалық инженерия» кафедрасының техн. ғыл. магистрі, сениор-лектор _____ Куникеев А.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Кибернетика және ақпараттық технологиялар институты

«Программалық инженерия» кафедрасы

Мамандығы 5В070400– Есептеу техникасы және бағдарламалық қамтамасыз ету

БЕКІТЕМІН
Кафедра меңгерушісі
тех. ғыл. кандидаты,
ассоц. профессор
_____ Р. Юнусов
“___” _____ 2020ж

Дипломдық жұмыс орындауға
ТАПСЫРМА

Білім алушы Ахетова Қарлығаш Ғалымқызы
Тақырыбы : «Electronic Timesheet» электронды табель жүйесі
Университет Ректорының *20___жылғы "___" ____ №__-б* бұйрығымен бекітілген
Аяқталған жұмысты тапсыру мерзімі *20 ___жылғы "___"_____*
Дипломдық жұмыстың бастапқы берілістері: ұсынылатын дипломдық жобада электронды табель және акт құжаттарын толтыру веб-қосымшасын жасау.
Дипломдық жұмыста қарастырылатын мәселелер тізімі:
а) аналитикалық бөлім;
б) технологиялық бөлім;
в)жобалау бөлімі;
г) жоба құрылымы;
д) А Қосымшасы. Техникалық тапсырма;
е) Б Қосымшасы. Бағдарлама мәтіні.
Сызба материалдар тізімі (міндетті сызбалар дәл көрсетілуі тиіс)
Сызба материалдарының _____ слайдта көрсетілген.
Ұсынылатын негізгі әдебиет 12 әдебиеттер тізімінен тұрады.

Дипломдық жұмысты (жобаны) дайындау
КЕСТЕСІ

Бөлімдер атауы, қарастырылатын мәселелер тізімі	**Ғылыми жетекші мен кеңесшілерге көрсету мерзімдері**	**Ескерту**
1. Дипломдық жоба тақырыбының қолданылу аясы мен өзектілігін зерттеу, жүйеге қатысты тұлғалармен кездесіп, сұрақтарға жауап алу.	20.01.2020	
2. Кіріс ақпараттарын жинау, UML тілінде жобалау: тізбек және прецеденттер диаграммаларын сызу. Деректер қорының жобасын құру.	01.02.2020	
3. Жоба интерфейсін құру, әзірлеу технологияларын таңдау.	05.02.2020	
4. Жобаның веб-қосымшасын әзірлеу.	10.03.2020	
5. Дипломдық жобаны диплом алды есеп алу комиссия мүшелеріне таныстыру, ескертулерін түзетіп дипломдық жобаны аяқтау. Қорытынды жасау.	27.04.2020	
6. Сипаттау мәтіні мен графикалық материалды аяқтау.	20.05.2020	

Дипломдық жұмыс (жоба) бөлімдерінің кеңесшілері мен норма бақылаушының аяқталған жұмысқа (жобаға) қойған
қолтаңбалары

Бөлімдер атауы	**Кеңесшілер, аты, әкесінің аты, тегі (ғылыми дәрежесі, атағы)**	**Қол қойылған күні**	**Қолы**
Бағдарламалық бөлім	Куникеев А. Техн. ғыл. магистрі, сениор-лектор		
Нормалық бақылаушы	Марғұлан Қ. Техн. ғыл. магистрі, лектор		

Ғылыми жетекші _____ Куникеев А.Д.

Тапсырманы орындауға алған білім алушы _____ Ахетова Қ.Ғ.

Күні "_____"_____2020 ж.

АҢДАТПА

Бұл дипломдық жобада Сәтбаев университеті қызметіне әзірленген «Electronic Timesheet» веб-қосымшасы қарастырылады. Қосымша университет қызметкерлерінің жұмыс уақытын есепке алу және бақылау жүйесін (табель және акт құжаттары) автоматтандыруға бағытталған.

Қазіргі таңда аталған жүйе қолмен жүзеге асырылатындықтан ұсынылған бағдарламалық қамтама құжаттарды толтыру мен растау уақытын азайтады, толтыру барысында кездесетін қателіктерді болдырмайды, қағазбастылықты жояды. Осылайша университет қызметкерлерінің жұмысын жеңілдетуге мүкіндік береді.

Жобаны сипаттау мәтіні кіріспе, төрт тараудан тұратын негізгі бөлім (аналитикалық, технологиялық, жобалау, жоба құрылымы), қорытынды, пайдаланылған әдебиеттер тізімін қамтиды.

Аналитикалық бөлімде таңдалынған жобаның өзектілігі қарастырылды, қолданыстағы жүйеге талдау жүргізілді. Әзірленетін бағдарламалық қамтама түрі анықталып, сипатталды.

Технологиялық бөлімде жоба әзірлеуде қолданылатын технологиялар таңдалынды, талдау жүргізілді. Бағдарламалау ортасын баптау мен әзірлеу барысы сипатталды.

Жобалау бөлімінде проектілеу барысында қолданылған әдістер сипатталды.

Жоба құрылымы бөлімінде әзірленген жүйе нәтижесі сипатталды.

Жалпы дипломдық жұмыс 51 бет, 11 сурет, 2 қосымшадан тұрады. Жүзеге асыру барысын сипаттауға 12 әдебиеттер қолданылды.

АННОТАЦИЯ

В данном дипломном проекте рассматривается веб-приложение «Electronic Timesheet», разработанное для университета Сатпаев. Приложение направлено на автоматизацию системы учета и контроля рабочего времени сотрудников (табельные и актовые документы).

Так как в настоящее время данная система осуществляется вручную, предлагаемое веб-приложение уменьшает время заполнения и подтверждения документов, устраняет волокиту и ошибки при заполнении. Таким образом, это позволит облегчить работу сотрудников университета.

Пояснительная записка проекта включает введение, основную часть из четырех разделов (аналитическую, технологическую, проектную, проектную структуру), заключение, список использованной литературы.

В аналитической части была рассмотрена актуальность выбранного проекта, проведен анализ существующей системы. Определен и описан тип разрабатываемого программного обеспечения.

В технологической части были выбраны технологии, применяемые при разработке проекта, проведен анализ. Описан ход разработки и настройки среды программирования.

В проектной части описаны методы, использованные при проектировании.

В разделе структура проекта описан результат разработанной системы.

Общая дипломная работа состоит из 51 страниц, 11 рисунков, 2 приложения. В описании хода реализации использовано 12 литературы.

ANNOTATION

This diploma project examines the web application "Electronic Timesheet", developed for Satbayev University. The application is aimed at automating the system of accounting and control of employees ' working hours (time sheets and act documents).

Since this system is currently implemented manually, the proposed web application reduces the time for filling out and confirming documents, eliminates red tape and errors when filling in. Thus, this will make it easier for University employees to work.

The explanatory note of the project includes an introduction, the main part of the four sections (analytical, technological, design, project structure), conclusion, and a list of references.

In the analytical part, the relevance of the selected project was considered, and the existing system was analyzed. The type of software being developed is defined and described.

In the technological part, the technologies used in the development of the project were selected and analyzed. The course of development and configuration of the programming environment is described.

The design part describes the methods used in the design.

The project structure section describes the result of the developed system.

The General thesis consists of 51 pages, 11 figures, 2 appendices. The description of the implementation process uses 12 references.

МАЗМҰНЫ

КІРІСПЕ

Қазіргі заман – техника мен ақпараттың дамыған кезеңі. Адамзаттың қоршаған әлеуметтік ортамен тұрақты байланысынан келіп түсетін және өңделетін ақпараттың саны, көлемі артып, мазмұны мен құрылымы бұрыңғыдан өзгеруде. Өз кезегінде ақпараттық технологиялар да жылдам қарқынмен дамуда, адамдардың көп бөлігі Интернет желісіне қол жеткізе алып отыр. Көптеген мекемелер ішкі процестерді ұйымдастыру және басқару үшін ақпараттық жүйелерді енгізуде. Соның ішінде білім беру құрылымдары да бар.

Ақпараттық жүйелер (АЖ) дегеніміз компьютерлер, компьютерлік желілер, бағдарламалық өнімдер, деректер базалары, қолданушылар, әртүрлі техникалық және бағдарламалық байланыс құралдары қатысатын орта болып табылады.

Білім беру процесіне ақпараттық технологияларды енгізу ЖОО ішіндегі көптеген үдерістерді ұйымдастыруды, студенттер, оқытушылар және басқа да қызметкерлер туралы ақпарат жинауды жеңілдетуге мүмкіндік береді. Осы себептен бұл жобада «Electronic timesheet» веб-қосымшасы әзірленіп отыр. Бұл қосымша акт және табель жүйесін электронды жасауға мүмкіндік береді.

Қосымшаны әзірлеуде Django және Angular веб-фреймворктері таңдалынды. Django – Python бағдарламалау тілінде жазылған веб-фреймворк. Аталған фреймворк кез келген веб-әзірлеу жобасы үшін үлкен құндылықты ұсынады: веб-қосымшаларды құру процесін жылдамдатуға және жеңілдетуге көмектесетін дайын компоненттер жиынтығымен жеткізіледі. Ал, Angular қуатты, заманауи және түрлі платформаларды (веб, мобильді құрылғылар, нативті десктоп) қолдайды. Олардың ашықтығы мен қолжетімділігінің, жетілдіруге арналған тегін әрі ыңғайлы құралдардың бар болуының арқасында көп қолданысқа ие және осы себепті бұл жобаны әзірлеуде осы технологиялар таңдалынды.

Дипломдық жұмыстың мақсаты: қазіргі заманғы технологияларды қолдана отырып университетіміздегі қызметкерлердің жұмыс уақытын есепке алу табелі мен акт жүйесін электронды түрге көшіріп, қазіргі таңдағы қағазбастылықты жоя отырып қызметкерлердің жұмысын жеңілдетуге және оңтайландыруға бағытталған веб-қосымша әзірлеу.

Дипломдық жоба өзектілігі: әрбір мекеме үшін жұмыс уақытын есепке алу табелі маңызды есептік құжаттардың бірі. Ол – қызметкерлердің белгіленген жұмыс режимінің сақталуын бақылау және барлық жұмыс істеген уақыт кезеңі туралы ақпарат алу үшін жүргізілетін жүйе. Қазіргі таңда университетімізде бұл жүйе қолмен жүзеге асырылып отырғандықтан электронды түрге көшіру жеке бір өзекті мәселе болып отыр. Қолданушылар өздеріне ыңғайлы әрі қолжетімді және тегін қызмет түрлерін қолдануды жөн көретіндіктен веб-қосымша әзірлеуге деген қажеттілік туындады. Бұл электронды табель веб-қосымшасы университетіміздің қызметкерлерінің жұмысын жеңілдету мақсатында әзірленген бағдарламалық қамтама.

Қосымшаның қолдану аймағы – университетіміздің қызметкерлерін түгелімен қамтиды. Оның ішінде, бухгалтерия, hr, кадр бөлімі мен басқарма мүшелері, табельшілер – тікелей қолданушылар.

1 Аналитикалық бөлім

1.1 Жобаның өзектілігі

Қазіргі заманғы компьютерлік техниканы дамыту және жаңа технологияларды енгізу адамзат өмірінің жаңа бағытының басы болды. Микроэлектроника мен кибернетика дамуының қысқа уақыт аралығында көптеген өзгерістер орын алды.

Техниканың прогрессивті дамуы жаңа бағдарламалық өнімдердің пайда болуына себеп болды. Бұл, өз кезегінде, жыл сайын көптеген бағдарламалау тілдерінің енгізілуіне итермеледі.

Компьютер адам жұмысының ажырамас бөлігі болып табылады: адам өмірінің барлық саласын қамтып отыр. Компьютерлер білім беру саласында, мектептер мен университеттерде, кеңінен қолданысқа енгізілуде. Олар жұмыс үрдісі мақсатында, оқу мақсатында алынған деректерді жүйелеуге көмектеседі. Соңғы жылдары мектептен бастап коммерциялық курстарға дейін әр түрлі оқу орындарының үрдістерін автоматтандыруға деген қызығушылықтың қарқынды күшеюі байқалады.

Қазіргі заманғы ЖОО мәселелерінің бірі – бақылау. Университет қызметкерлерінің және бөлімдердің санының көптігіне байланысты оқу үрдісін есепке алу жүйесін автоматтандырудың қажеттілігі бар. Қазіргі уақытта әр кафедра меңгерушісімен жүргізілетін есепке алу мен бақылауды жүргізудің бірнеше түрлері бар: тұрақты қызметкердің жұмыс күндерін және демалыс күндерін есепке алатын табелі, уақытша қызметкерді қабылдау туралы бұйрықтар, есепке алу акті. Жұмыс уақытын есепке алуды қолмен жүзеге асырудың күрделілігі:

– құжаттардың едәуір саны;

– көп уақытты қажет етуі;

– тұтынушылар мен ақпараттың бөлінуі.

Осы себептерден университетіміздегі есепке алу жүйесін автоматтандыру мақсатында электронды табель жүйесін жүзеге асыру ұйғарылды.

1.2 Қолданыстағы есепке алу жүйесі

Жобаны іске асыруға кірісер алдында қазіргі кездегі осындай сипаттағы және тақырыптағы балама жүйелерді талдау қажет. Нәтижесінде, қаралған ресурстардың артықшылықтары мен кемшіліктері негізінде жобада жасалатын жүйенің қалай құрылуы керектігі жөнінде дұрыс қорытынды жасалынады.

Жұмыс уақытын есепке алу табелі – әрбір оқу орны үшін маңызды есептік құжаттардың бірі. Ол қызметкерлердің белгіленген жұмыс режимінің сақталуын

бақылауға және барлық істеген жұмыс кезеңі туралы ақпарат алуға мүмкіндік береді.

Қазіргі таңда университетімізде бұл жүйе қолмен жүзеге асырылып отыр. Әр департамент бөлімшесінде табель немесе акт толтыруға жауапты бір қызметкер тағайындалған. Құжаттардың соңғы нәтижесі бухгалтерия бөліміне беріледі. Табель толтырудың қолмен жүргізілу реті:

1) жауапты қызметкер табель толтырады: кестеге қызметкердің аты-жөні, мамандығы, жұмыс істеу ставкасы, бір айда жұмыс істеген әр күнгі сағаттары жазылады;
2) растаушы тұлғалар тізімі жазылады;
3) дайын табель А4 парағына шығарылады;
4) растаушы тұлғалар дұрыстығын тексеріп, қол қояды;
5) дұрыс толтырылмаған жағдайда табельге керекті өзгерістер енгізіліп, А4 парағына қайта шығарылады. Тізімдегі растаушы тұлғалардың қол қойып-қоймағанына қарамай тізімнің басынан бастап растауға қайта жіберіледі;
6) барлық растаушы тұлғалар қол қойғаннан соң табель бухгалтерия бөліміне өтіп, қызметкерлерге еңбек ақы тағайындалады.

Бұл реттілік арасында табельшінің әр растаушы тұлғадан келесісіне жүріп-тұрысы және оған кететін уақыт қарастырылмады. Ал, университетіміздің аумағы едәуір үлкен болғандықтан бұл маңызды уақыт мәселесі болып отыр.

Университетіміздегі қазіргі жүйе мен электронды жүйені салыстырар болсақ, екіншісінің мынадай артықшылықтарын байқаймыз:

– қағазбастылықты жою;
– уақыт үнемдеу;
– толтыру және өзгеріс енгізудің жеңілдігі;
– электронды қол қою мүмкіндігі.

Әр қызметкердің еңбек ақысының тағайындалуы табельдің соңғы нәтижесіне негізделеді. Сондықтан табель толтыру дұрыстығы – жауапкершілікті іс. Осы себептен толтыру барысында қателіктерді болдырмас үшін электронды табель қосымшасы таптырмас шешім болмақ.

1.3 Веб vs мобильді қосымша

Электронды табель жүйесін жүзеге асыру барысында екі таңдау болды: веб немесе мобильді қосымша жасау.

WEB-қосымша – бұл ішінара әзірленген беттері бар веб-сайт, оның соңғы мазмұны пайдаланушының сұрауы бойынша анықталады [1].

Мобильді қосымша – бұл мобильді құрылғылар (смартфондар, планшеттер және сол сынды) үшін жасалған және белгілі бір платформаға (IOS, Android, Windows) бейімделген бағдарламалық қамтама [1].

Екі нұсқаны салыстыру нәтижесінде байқалды:

– веб-қосымшалардың ауқымдылығы: бір мезгілде үлкен аудиторияның пайдалану мүмкіндігі;

– мобильді қосымшаларда офлайн қолжетімділіктің болуы, ал веб-қосымшада ғаламтор – міндетті;

– веб-қосымша мобильді қосымшаларға қарағанда орнатуды талап етпейді, демек, құрылғы жадын жүктемейді;

– мобильді қосымшаның функционалы көп, бірақ оны әзірлеу үшін көп уақыт пен қаражат қажет;

– веб-қосымшалар автоматты түрде жаңартылады, ал мобильді қосымшаның әрбір жаңа нұсқасын жүктеу керек;

– мобильді қолданбалардың құрылғы жадына және басқа да деректерге қолжетімділігі бар (имеет доступ);

– мобильді қосымшада клиенттермен (хабарламалар, push-хабарламалар) байланыс үнемі байланыста болады.

Салыстыру нәтижесінде университетіміздің табель жүйесін автоматтандыруды жүзеге асыру үшін қосымшаның веб нұсқасын әзірлеу ұйғарылды.

2 Технологиялық бөлім

WEB-қосымша – глобалды Интернет желісінде жұмыс жасайтын арнайы қосымша түрі, жартылай әзірленген беттері бар және оның соңғы мазмұны қолданушының сұрауы бойынша анықталады. Нақтырақ айтқанда, веб-браузердегі қолданушының сұранысына жауап ретінде веб-беттерді ұсынатын бағдарламалық қамтама. Әдетте, бұндай сұраныстарға веб-беттегі сілтемелерді басу, браузердегі бетбелгіні (закладка) таңдау немесе мекенжай жолағына URL сілтемесін енгізу нәтижелері жатады.

Веб-қосымша 3 бөліктен құралады – backend (сервер бөлігі), frontend (клиент бөлігі) және деректер қоры.

Клиенттік бөлік қолданушы интерфейсін іске асырады, серверге сұраныстарды қалыптастырады және одан келген жауапты өңдейді. Бұл операциялар қолданушы браузерінде жүзеге асырылады. Қолданушының графикалық интерфейсін (GUI) жүзеге асыру үшін қарапайым HTML, CSS технологиялары, ал сұраныстарды қалыптастыру және өңдеу, интерактивті интерфейс құру үшін басқа жетілген технологиялар қолданылады. Мысалы:

– ActiveX;
– Adobe Flash, Adobe Flex;
– Java;
– JavaScript;
– Silverlight.

Серверлік бөлім қолданушыдан келген сұранысты өңдейді, қажет мәліметтерді базадан алады, содан кейін веб-бетті қалыптастырып, HTTP хаттамасын пайдалана отырып, желі арқылы қолданушыға қайта жібереді. Қысқаша айтқанда, клиенттік бөлік пен деректер қорының арасындағы байланысты қамтамасыз етеді. Әдетте, серверлік веб-қосымшаларды құру үшін стандартты консольге шығаруға қабілетті әр түрлі технологиялар мен программалаудың кез келген тілдері қолданылады. Мысалы:

– ASP;
– ASP.NET;
– C/C++;
– Java;
– Perl;
– PHP;
– Python;
– Ruby.

Деректер қоры – компьютердің сыртқы жадында ұзақ сақтауға және тұрақты қолдануға арналған барлық мәліметтер жиналатын файлдар немесе файлдар жиынтығы. Деректер қорымен жұмыс істейтін бағдарламаларды, бағдарламалар жүйелерін әзірлеу үшін арнайы құралдар – деректер қорын басқару жүйелері (ДҚБЖ) қолданылады. Деректер қорын басқару жүйесі – мәліметтерді әзірлеуге, енгізуге, өзгертуге және жоюға арналған бағдарламалық

құралдардың жиынтығы. Сақталатын мәліметтерге байланысты ДҚБЖ әзірлеуде әр түрлі технологиялар пайдаланылады. Мысалы:

– My SQL;
– MS SQL;
– Oracle;
– Mongo DB;
– PostgreSQL;
– SQLite.

Жобада қарастырылатын электронды табель веб-қосымшасын әзірлеу үшін серверлік бөлігіне Django, клиенттік бөлігіне Angular фреймворктері, ал деректер қорын қалыптастыруға PostgreSQL тілі таңдалынды. Жобаны бастау үшін жоба әзірленетін компьютерде таңдалынған технологиялардың әзерлеу ортасын орнатып, баптау қажет. Алдымен, Django серверлік бөлігін бастау үшін Python тілі және pip құралы қажет.

2.1 Python бағдарламалау тілі

Python – бұл жоғары деңгейлі бағдарламалау тілі, ол машиналық оқыту (машинное обучение, ML), қосымшаларды құру (разработка), web, парсинг және сол сияқты ақпараттық технологиялар саласының түрлі бағыттарында кеңінен қолданылып отыр [2].

Питонның синтаксисі оны басқа бағдарламалау тілдерімен салыстырғанда әлдеқайда ерекшеленеді. Кодта артықшылық (избыточность) жоқ, синтаксистің әдеттегі ағылшын тіліне ұқсастығы тіпті қарапайым қолданушыға кодты түсінуге мүмкіндік береді. Сонымен қатар, жазылатын код жолдары аз шығады, өйткені ";", "{", "}" секілді таңбаларды пайдаланудың қажеті жоқ: кіріспелік (вложенность) шегініспен (отступ) көрсетіледі. Өз кезегінде, бұл кодтың түсініктілігін, оңай оқылуын арттырады және бағдарламашыны (программист) код синтаксисін алдағы уақытта жалпы дұрыс жазуға үйретеді.

2.1.1 Python рейтингі

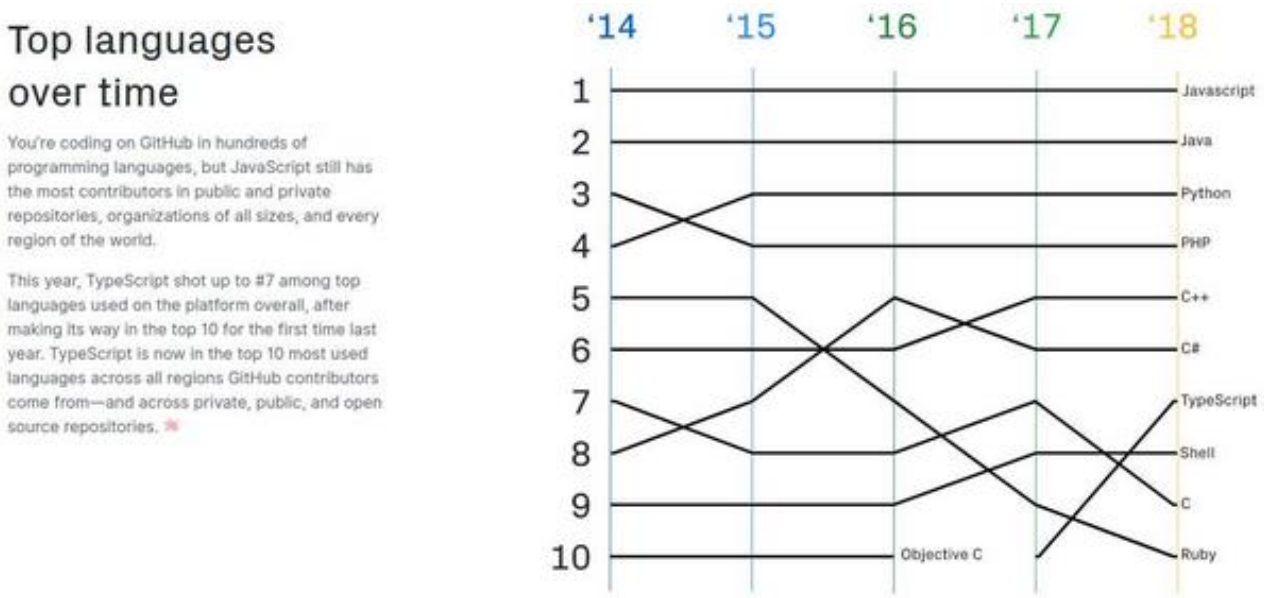
Бағдарламалау тілінің танымалдығын, қолданысын бағалау тәсілдерінің бірі – TIOBE индексі. Ол Google және басқа іздеу жүйелеріндегі сұраныстарының саны негізінде есептеледі және бағдарламалау тілдерінің атауы қамтылатын сұраныстар есепке алынады [2].

TIOBE индексіне сәйкес, 2019 жылдың тамыз айында Python ең танымал бағдарламалау тілдері тізімінде 10% танымалдылықпен үшінші орын алып отыр. Ол JavaScript, PHP, Swift және басқа да танымал тілдерді басып озды. 2019 жылға шығарылған TIOBE рейтингі төменде 2.1-суретте көрсетілген.

Aug 2019	Aug 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.028%	-0.85%
2	2		C	15.154%	+0.19%
3	4	^	Python	10.020%	+3.03%
4	3	v	C++	6.057%	-1.41%
5	6	^	C#	3.842%	+0.30%
6	5	v	Visual Basic .NET	3.695%	-1.07%
7	8	^	JavaScript	2.258%	-0.15%
8	7	v	PHP	2.075%	-0.85%
9	14	^^	Objective-C	1.690%	+0.33%
10	9	v	SQL	1.625%	-0.69%

2.1-сурет – 2019 жылғы TIOBE рейтингі

Github Octoverse рейтингі GitHub қолданушылары арасындағы тілдің танымалдығын көрсетеді. 2018 жылы GitHub Octoverse рейтингісінде Python тек JavaScript және Java технологияларына жол берді. 2018 жылға есептелген Github Octoverse рейтингі төменде 2.2-суретте көрсетілген.

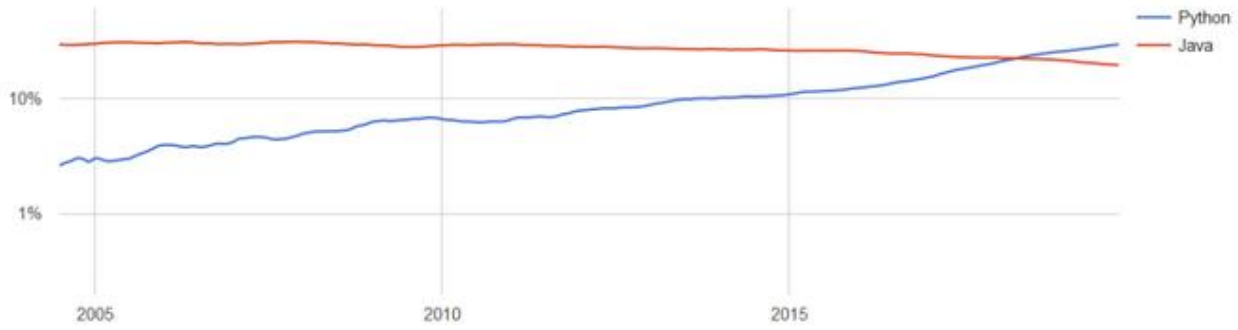


2.2-сурет – 2018 жылғы Github Octoverse рейтингі

RedMonk рейтингісінде тіл үшінші орын алады. RedMonk негізін қалаушы, Джеймс Гавернер, Data Science саласында Python – лингва франкаға айналғанын атап өтті. Яғни бұл тіл осы сала үшін негізгі тіл болды.

"Лингва франка" – түрлі тілдік топтардың өкілдері қарым-қатынас жасау үшін қолданылатын жалпыға бірдей белгілі тілдер [2]. Мысалы, ағылшын тілін халықаралық қарым-қатынастағы лингва франка деп атауға болады.

PYPL индексі оқу материалдарына қатысты іздеу сұраныстарының ішінде бағдарламалау тілдерінің санына негізделеді. Бұл индекс мәліметтері бойынша Python 29% танымалдылықпен бірінші орынды алып, 10% үстемдікпен Java тілін басып озып отыр. Төменде 2.3-суретте PYPL рейтингі көрсетілген.



2.3-сурет – PYPL рейтингі

2.1.2 PIP инструменті

Электронды табель қосымшасының серверлік бөлігін әзірлеуде Django фреймворкі таңдалынғандықтан онымен жоба құру үшін алдымен pip инструменті орнатылу қажет.

PIP – Python бағдарламаларын басқару (орнату, жаңарту және жою) үшін қолданылатын технология [3, 12]. Pip – консоль утилитасы, яғни графикалық интерфейсі жоқ. Python-ның соңғы нұсқаларынан бастап (python 2.7.9 және python 3.4 бастап) pyhton-ды орнату барысында pip автоматты түрде орнатылады. Бірақ, кейбір себептермен pip орнатылмаған болса, оны қолмен жасауға болады. Оны жүктеп және орнатқаннан кейін, ол компьютерде автоматты түрде PATH айнымалы ортасында тіркелінеді.

pip install django – django пакетін орнату [3, 12]. Бұл жағдайда pip пакетінің ең соңғы нұсқасы орнатылады. Сонымен қатар, нақты нұсқасын *pip install django==3.2* командасын жаза отырып орнатуға немесе *pip install django>=3.1* логикалық сөйлемді пайдалануға болады: көрсетілген нұсқадан төмен емес нұсқаның қажет екендігі анықталады. Сондай-ақ, PyPI репозиторийнан пакеттерді орнатуға мүмкіндік бар: *pip install –index-url PacketURL.*

pip list – барлық орнатылған пакеттердің тізімін шығару үшін қолданылатын команда [3, 12]. Біздің жағдайда, бұл команда Django, Django-cors-headers, Django-filter, djangoRestFramework және psycopg2 пакеттері орнатылғанын көрсетеді.

pip show django – пакет жөнінде толық ақпаратты көрсетеді: аты, нұсқасы, қысқа анықтамасы, ресми сайты, орналасқан орны және қажетті пакеттерін.

Django әзірлеу ортасына қажетті пакеттер орнатылды, келесі кезек – жаңа жоба құру.

2.2 Django фреймворкі

Django – бұл Python тілінде жазылған веб-қосымшалар әзірлеуге арналған тегін платформа [4]. Ол техникалық күрделі жобаларды: сайттарды, веб-сервистерді және бизнес-қосымшаларды жасауды және қолдауды жеңілдетеді.

Django жоғары өнімді веб-қосымшаларды жылдам жасауда кірістірілген функциялардың едәуір санын ұсынады және күрделіні оңтайландыруға мүмкіндік береді. Фреймворк веб-қосымшаларда табылатын ең көп таралған функцияларды іске асыратын ыңғайлы компоненттерді дайын күйінде ұсынады. Мысалы, қолданушы аутентификациясы, әкімшілік басқару тақтасы, файлдарды жүктеу және тағы басқа.

Жұмыс істеу құрылымы MVC паттерні негізінде жүзеге асады. MVC архитектурасы әзірлеушіге визуалды көрініспен және қосымшаның жеке бизнес-логикасымен жұмыс істеуге мүмкіндік береді. Django-мен жұмыс кезінде мамандар MVT – Model-View-Template немесе «модель-көрініс-үлгі» терминін жиі қолданады. MVT құрамдастарын бір-біріне қарамастан қолдануға болады. Төмендегі 2.4-суретте фреймворк архитектурасының графикалық бейнесі көрсетілген.

2.4-сурет – MVT архитектурасы

Модельдер деректер туралы ақпаратты қамтиды: бір модель – бір кесте. Ол деректерді басқаруға байланысты бизнес-логикаға, әдістерге, қасиеттерге элементтерге жауап береді. Нақтырақ айтқанда, модельдер әзірлеушілерге деректер қорындағы нысандарды жасауға, оқуға, жаңартуға және жоюға мүмкіндік береді. Модель қарапайым класс болғандықтан, ол басқа да Django деңгейлері туралы ештеңе білмейді. Деңгейлер арасындағы өзара іс-қимыл API арқылы жүреді.

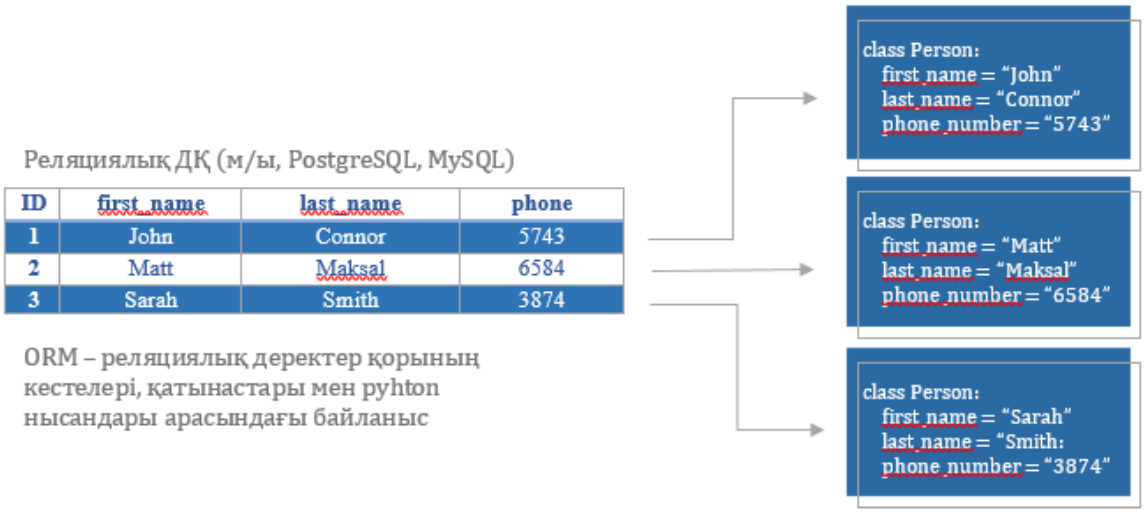
Көрініс (view) үш мәселені шешеді: HTTP-сұрауларды қабылдайды, әдістермен және қасиеттермен анықталған бизнес-логиканы жүзеге асырады, сұрауларға жауап ретінде HTTP-жауапты қайтарады. Яғни, көрініс модельден деректерді алады және үлгілерге (templates) осы деректерді ұсынады немесе деректерді алдын ала өңдеп үлгілерге ұсынады.

2.3 Фреймворк артықшылықтары

Электронды табель веб-қосымшасын әзірлеуде Django фреймворкі таңдалуының бірнеше себептері төменде қарастылады.

Қауіпсіздік. Django, әдетті жағдайда, көптеген осалдықтардан қорғауды қамтамасыз етеді. Мысалы, SQL-инъекция, сайтаралық скриптинг, жалған сайтаралық сұраныстар, кликджекинг және осы сынды осалдықтар.

ORM. Django-да деректер қорымен (ДҚ) қосымшаның өзара әрекеттесуін қамтамасыз ететін объектілі-реляциялық бейнелеу (ORM) жүзеге асырылған. ORM деректерді автоматты түрде ДҚ-дан, мысалы, Postgresql немесе MySQL, бағдарлама кодында қолданылатын объектілерге жібереді. Төмендегі 2.5-суретте ORM жүйесі көрсетілген.



2.5-сурет – ORM жүйесі

Кітапханалар. Танымал бағдарламалау тілдерінде арнайы есептерді шешуге ыңғайлы кітапханалар бар. Кітапханаларда дайын шешімдерді табуға болады: функциялар, класстар, конфигурациялар және сол іспеттес мүмкіндіктер. Осындай шешімдердің арқасында бағдарламалау тілі мүмкіндіктері кеңейтіліп, қосымшаларды жасау жеңілдетіледі. Django веб-қосымшаларды әзірлеу кезінде кітапханалардың едәуір санын қолдануға мүмкіндік береді.

Электронды табель қосымшасын әзірлеуде қолданылған кітапханалар:

1) django REST Framework – API-мен жұмысты жеңілдетеді [5];
2) PyMySQL – Pyhton тілінен MySQL деректер қорына қосылуды жеңілдетеді;
3) django CORS headers – CORS-қа (cross-origin resource sharing) қажетті сервер тақырыптарын өңдейтін Django кітапханасы. Бұл басқа домендерде өз ресурстарға қол жеткізуге мүмкіндік береді.
4) django psycopg2 – PostgreSQL тілімен байланысу кітапханасы, драйвер [9].

Әзірлеудегі жеңілдік. Django орнату, баптау тәсілі мен орнатылу орны тұрғысынан өте икемді және көп мүмкіндіктер ұсынады:

– орнатылу ортасында операциялық жүйе таңдамайды, барлық ОЖ-да жұмыс істейді (Windows, Linux, Mac OS);

– python package index (PyPI) репозиторийнен немесе басқа да пакеттер менеджерінен орнатылады (мысалы, pip);

– әр түрлі деректер қорымен байланыстырылған, қолдану және баптау өте оңай;

– жобаны Python ортасының негізгі жүйесінде немесе жеке виртуалды ортада іске қосуға болады.

Django-да жобаны құру үшін startproject командасы қолданылады: *django-admin.py startproject eTabel* [3]. Нәтижесінде бірнеше файлдар пайда болады:

– manage.py – қосымшаларды жасау, дерекқорлармен жұмыс істеу және серверді іске қосу үшін қолданылады;

– settings.py – мұнда жоба параметрлері бар. Файл барлық баптауларды қамтиды, бірақ деректер қоры көрсетілмеген;

– urls.py – көріністерді көрсетуге арналған URL мекен-жайлар орналасқан;

– wsgi.py – Django қосымшасы мен веб-сервер арасында байланыс орнату үшін пайдаланылады.

2.4 Angular фреймворкі

Angular – Google компаниясымен ұсынылған клиенттік қосымшаларды әзірлеуге арналған фреймворк [6]. Ең алдымен, ол SPA-шешімдерді (single page application), яғни бір беттік қосымшаларды әзірлеуге бағытталған.

Angular – веб-қосымшаларды жасау үшін қолданылатын заманауи фреймворктердің бірі. Әмбебап, жылдам және қазіргі уақыттағы стандарттарды қолдайтын қосымшаларды жасау үшін үлкен мүмкіндіктер ұсынатындықтан бүгінгі таңда ең танымал және сұранысқа ие, бірегей технологияға айналып отыр.

Angular үлгілер (шаблоны), маршруттау және екіжақты байланыс секілді мүмкіндіктер мен функционалдарды үсынады. Angular-дың негізгі ерекшеліктерінің бірі – TypeScript бағдарламалау тілін пайдалануы. Сонымен қатар, өзге артықшылықтары:

– Google, Microsoft секілді қазіргі таңдағы ірі компаниялар қолдайды;
– әзірлеуді жеңілдететін консоль құралдары (CLI);
– жобаның бірыңғай, икемді құрылымы;
– қатаң типизацияланған код (typescript-те жазылуы);
– Rxjs бар реактивті бағдарламалау;
– HTML кеңейтуіне негізделген үлгілер;
– Http, WebSockets, Service Workers функционалдарын қолдау;
– үлкен қолдаушылар қауымы.

Angular-мен жұмыс істеу үшін Node.js сервері және npm пакеттік менеджері орнатылу қажет. Ол үшін node.js ресми сайтынан қажетті жүктеу бағдарламасы орнатылады. Бағдарламамен бірге сондай-ақ npm пакеті автоматты түрде орнатылады.

Node.js – бұл JavaScript тілінің интерпретаторы. Node.js C++ қосымшасы болып табылады, ол кіріске JavaScript кодын алып, оны орындайды. Өз кезегінде, Npm (node package manager) – жоба модульдері мен тәуелділіктерін басқаратын пакеттер менеджері.

Қолданбаны әзірлеу үшін Angular CLI инфрақұрылымы пайдаланылады. Angular CLI – Angular қосымшаларын құру, әзірлеу және қолдау үшін командалық жолдың интерфейсін іске асыратын npm модулі [7]. Жүйеде ол глобальды орнатылуы тиіс: *npm install –g @angular/cli*. *npm install* арқылы барлық керекті модульдер орнатылады. Осылайша, Angular-дың әзірлеу ортасы орнатылды, жоба құруға дайын.

Жобаны құру *ng new demo* командасымен жүзеге асырылады, бұндағы *demo* жоба аты. Нәтижесінде, төмендегі файлдар жүйесі пайда болады:

– e2e – интеграциялық тесттері бар директория;
– node_modules – орнатылған npm модульдері;
– src – бастапқы файлдар;
– angular.js – конфигурация сипаттамасы;
– package.json – метаақпарат және қажетті npm модульдер тізімі;
– read.md – камтама бойынша сипаттама;
– tsconfig.json – typescript жалпы конфигурациясы.

2.5 Бағдарлама архитектурасы

Ангулар фреймворкі бірнеше кітапханалардан және модульдерден тұрады, олардың әрқайсысы белгілі бір функционалға жауап береді. Әрбір модуль класстар жиынтығынан және олардың қасиеттері мен әдістерінен тұрады. Әр класс өзінің функционалдық мақсатына ие.

Angular қосымшасын жобалау модульден басталады. Олардың әрқайсында өз құрылымдық элементтерінің жиынтығы бар:

– component – web-беттің бөлігіне жауап береді және HTML-үлгісін, CSS-стилін және логикасын қамтиды;

– service – component үшін деректер жеткізуші;

– directive – DOM белгілі бір бөлігін берілген түрде түрлендіреді.

ng g module home – модуль құру командасы, бұндағы *home* – модуль атауы. Жобада негізгі модульден басқа home, auth атты модульдер құрылды.

Компонент (angular component) – өз логикасымен, HTML-шаблонымен және CSS-стилімен оқшауланған функционалдың бөлігі [6, 11]. Компонентті жариялау үшін @angular/core-дан @Component() декораторы жауап береді. Қабылданатын объекттер:

– selector – компонент атауы;

– template (template Url) – HTML-жол түрінде белгілеу (HTML файлына жол);

– styles – құрылған компонентке арналған мәнерлер бар CSS файлдарына жол.

ng g component login – компонент құру командасы, бұндағы login – атауы. Әдепкі жағдайда (по умолчанию), жаңа ашылған жобада бір ғана AppComponent атты компоненті болады, ол сәлемдесу мәтінін қамтиды. Жобада, сонымен қатар, login, register, auth, act save, act list, holiday list, holiday save, gape, profile list, working hours table атты компонентері бар.

Сервистер компоненттерге мәліметтерді ұсыну үшін қажет [6, 11]. Бұл серверге сұраныстар ғана емес, сонымен қатар берілген алгоритм бойынша бастапқы деректерді түрлендіретін функциялар да болуы мүмкін. Олар Angular app архитектурасына масштабталуға және икемді болуға мүмкіндік береді. Сервистің міндеті қатаң анықталған болуы тиіс.

ng g service home – сервис құру командасы, бұндағы home – сервис атауы. Жобада home, auth, profile атты сервистері қолданылады.

ng serve -o командасымен құрылған жоба іске қосылады. Енді қолданба http://localhost:4200 (әдепкі жағдайда) мекен-жайы бойынша іске қосылған. Бұндағы -o опциясы қолданба жүргізілгеннен кейін браузерде автоматты түрде ашылатынын көрсетеді.

2.6 Деректер қоры

Деректер қоры – белгілі бір түрде құрылымдалған ақпаратты сақтайтын файл (немесе файл жиынтығы) [4]. Жиналған ақпараттарды өңдеу үшін деректер қорын басқару жүйелері ДҚБЖ пайдаланылады. ДҚБЖ мысалдары: MySQL, PostgreSQL, Oracle, Microsoft SQL Server және Microsoft Access.

Ақпаратты құрылымдау тәсілі бойынша деректер қоры бірнеше түрге бөлінеді – реляционды және реляционды емес. Жобада реляциялық деректер қоры пайдаланылады: олар бір-бірімен байланысты ақпаратты кесте түрінде сақтау үшін қызмет етеді және қазіргі уақытта кең таралған. Жоғарыда аталған ДҚБЖ түрлері реляциялық деректер қорын өңдейді.

Қосымшалардағы барлық мәліметтер деректер қорында (ДҚ) сақталады. Деректерді ДҚ-да құруға, оқуға, өзгертуге және жоюға болады. Кейде осы әрекеттерді белгілеу үшін CRUD (create, read, update, delete) аббревиатурасы қолданылады.

2.7 PostgreSQL тілі

Жобада деректер қорын жасау үшін PostgreSQL тілі және PgAdmin бағдарламасы қолданылды. PostgreSQL – бұл қуатты объектілі-реляциялық деректер қорын басқару жүйесі. PostgreSQL көптеген кеңейтілген функцияларды қамтиды, өте жылдам жұмыс істейді және қазіргі таңдағы стандарттарға сәйкес келеді. Ол C, C++, Python, Java, PHP, Ruby сияқты көптеген бағдарламалау тілдерімен байланыстырылған. Қарапайым веб-қосымшалардан бастап миллион жазбалары бар үлкен деректер қорына дейін кез келген бағдарламаны қолдау үшін пайдаланылуы мүмкін.

PostgreSQL үлкен көлемді күрделі операциялар үшін қолайлы шешім, басқа ДҚБЖ-ға қарағанда функциялардың едәуір санына ие. Сонымен қатар, PostgreSQL – кеңейтілетін жүйе, оның жұмысы каталогтарға негізделген (catalog-driven тәсілі). Нақтырақ айтқанда, ол тек кестелер мен бағандар туралы ғана емес, сонымен қатар деректер мен индекстер түрлері, функционалдық тілдер туралы ақпаратты сақтайды.

PostgreSQL деректер қоры – ACID принциптерімен үйлесімді объектілі-реляциялық жүйе:

– atomicity – атомдық (атомарность);
– consistency – үйлесімділік (согласованность);
– isolation – оқшаулану (изолированность);
– durability – тұрақтылық (стойкость).

2.7.1 Негізгі артықшылықтары

1) реляциялық ғана емес, объектілі-реляциялық ДҚБЖ. PostgreSQL – бұл объектілі-реляциялық жүйе, ал оны бағдарламалау объектілі-бағытталған. Бұл кестелердің нысандары мен мұрагерлерін анықтауға мүмкіндік береді, деректердің күрделі құрылымын тудырады. ОРДҚБЖ қатаң реляциялық модельге сай келмейтін деректер үшін оңтайлы шешім.

2) күрделі сұраныстар үшін өте қолайлы, оқу-жазу операцияларымен қатар деректерді валидациялау қажет болған кезде күрделі операциялар орындау мүмкіндігі. Дегенмен, ОРДҚБЖ тек оқу операцияларын жақсы орындай алмайды.

3) NoSQL және көптеген өзге деректер типтерін қолдайды: NoSQL функциялары үшін танымал таңдау болып табылады. Ол бастапқыдан JSON, XML және hstore сияқты көптеген деректер типтерін қолдануға мүмкіндік береді. Сонымен қатар, стандартты емес функцияларды іске қосуға болады.

4) үлкен көлемдегі ДҚ басқару үшін жобаланған. PostgreSQL функциялары деректер қорының өлшемін шектемейді. Мысалы, шамамен төрт петабайт көлеміндегі деректерді басқарады, секундына 100-ден 250 мыңға дейін сыртқы сұраныстарды өңдейді.

5) көпверсиондық көмегімен параллельді қатынауды (MVCC) басқару. MVCC оқу және жазу үшін көптеген қолданушыларға бір мезгілде ДҚ-ға қатынауды ұсынады. Әр қолданушы деректермен жұмыс жасаған кезде оқу-жазуды блоктау қажеттілігі жойылады. Бұл ДҚБЖ басқару тиімділігі және өнімділігін айтарлықтай арттырады.

6) ACID стандартына сай келуі. PostgreSQL деректердің бұзылуын болдырмайды және транзакциялық деңгейде олардың тұтастығын қамтамасыз етеді.

7) көптеген программалау тілдерін қолдайды. C / C++, Delphi, Erlang, Go, Java, JavaScript, Lisp, .Net, Python, R, Tcl және басқа бағдарламалау тілдері.

2.7.2 PostgreSQL – Django байланысы

PostgreSQL ортасын баптау үшін ресми сайтынан Windows операциялық жүйесіне арналған нұсқасы жаздырылып, орнатылды. Джанго серверлік бөлігімен байланысты Pyhton тіліне арналған psycopg2 кітапханасы жүзеге асырады, ол *pip install psycopg2* командасымен орнатылды. Келесі кезек – деректер қорын құру: консольда немесе бағдарлама көмегімен құрылады, біздің жағдайда PgAdmin бағдарламасында. Құрылған деректер қорымен жобаны байланыстыру үшін қорға қатысты негізгі мәліметтер setting.py баптау файлында көрсетіледі. Келесі кезек – модель құру.

Серверлік бөліктің моделдері models.py файлында анықталады. Модельде әртүрлі класстар сипатталады, олардың әрқайсысы бір ДҚ кестесін сипаттайды, ал класс сипаты — бұл кестенің бір бағаны. Класс Django пакетінде сипатталған Model.db.models кітапханасынан шақырылады, ал класс қасиеттері сол пакетте сипатталған типтерімен анықталады. Мысалы, жобада мына типтер қолданылды:

1) models.DateField () – күнді қамтитын өріс (поле) [4, 8];
2) models.CharField (max_length= xx) – ұзындығы шектеулі мәтін өрісі. Жол ұзындығы = xx таңбалар [4, 8];
3) models.BooleanField () – булево мәні бар өріс [4, 8];
4) models.ForeignKey() – сыртқы кестеге сілтеме [4, 8];
5) models.IntegerField () – бүтін сандар [4, 8];
6) models.DateField () – мәліметтер бар өріс [4, 8].

python manage.py migrate командасымен жазылған модельдер негізінде ДҚ-да таблицалар құрылады.

3 Жобалау бөлімі

3.1 Бірыңғай модельдеу тілі

Жобаны құрастыру және әзірлеу барысында анализдеу мен модельдеу жүргізу міндетті саналады, ал ол процестерді графикалық бейнелеу маңызды орын алады. Нақтырақ айтқанда, жобаның құрылуынан бастап аяқталуына дейінгі өмірлік циклін сызбалармен және диаграммалармен көрсету. Бағдарламалық қамтаманы әзірлеу барысының схемасын, жобасын құрастыруда қазіргі таңда көп қолданысқа ие стандарт – UML, бірыңғай модельдеу тілі.

UML-диаграмма – бағдраламалық қамтаманы әзірлеу кезінде объектілі модельдеуге арналған графикалық сипаттаманың арнайы тілі. Бұл тіл – жүйенің абстрактілі үлгісін жасауға керекті әртүрлі графикалық белгілер қолданатын ашық стандарт. UML барлық бағдаралмалық жүйелерді жобалау үшін, сондай-ақ анықтау, визуализация, құжаттау үшін қолданылады. Алайда, тек бағдарламалық қамтаманы үлгілеумен ғана шектеліп қоймай, бүгінде бизнес саласында да маңызды орын алады: әртүрлі бизнес-процестерді модельдеу, жүйелік жобалауды жүргізу, сондай-ақ ұйымдық құрылымдарды бейнелеу үшін белсенді қолданылады.

UML технологиясының негізінде бірыңғай ақпараттық модель құрылады. Модель бизнес жобаның, бағдарламалық қамтаманың схемалық көрінісін анықтайды. Сол арқылы жобалау барысында бейнелерді визуалды түрде көре отырып кемшіліктерді анықтап, мүмкін жағдайдағы қателіктерді алдын-ала болжауға мүмкіндік береді. Сондықтан UML-диаграммалары жобалау сатысындағы маңызды бөліктердің біріне айналып отыр.

UML-диаграммалардың атқаратын қызметіне қарай бірнеше түрі бар, маңыздыларын атап өткенде: прецеденттер, күй, мінез құлқын модельдеу (use case diagram), тізбек диаграммалары. Олардың негізгі құраушы бөліктері – болмыстар, класстар, түйіндер, интерфейстер және компонеттер.

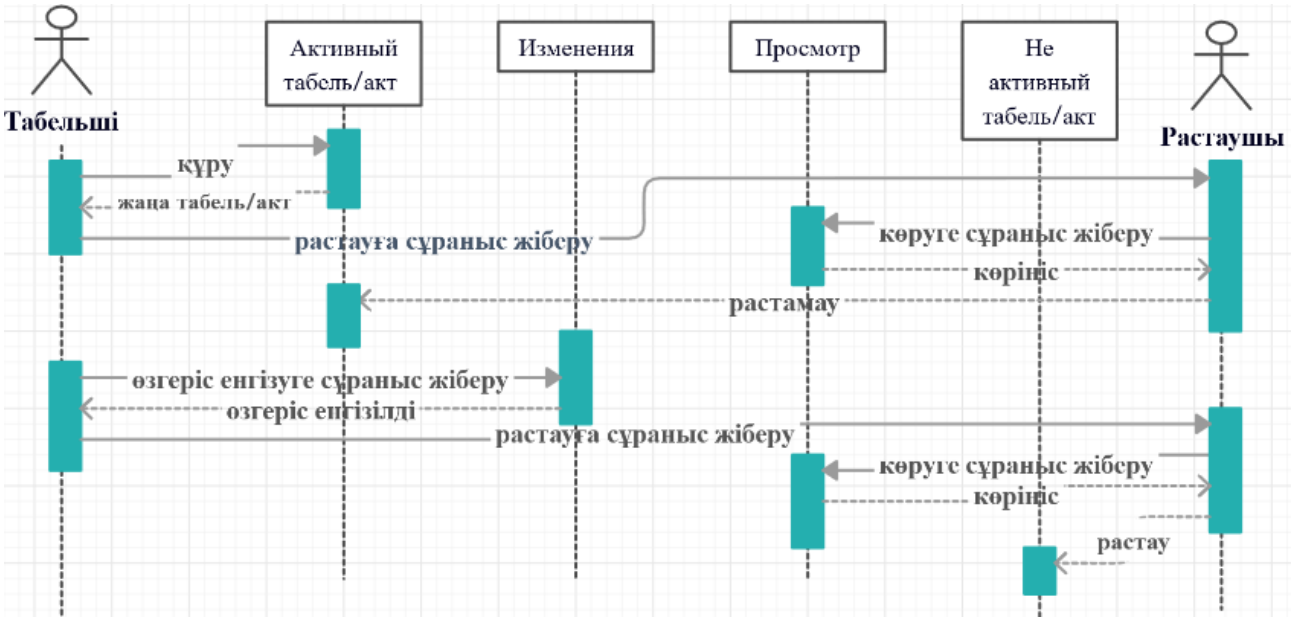
3.2 Тізбек диаграммасы

Тізбек диаграммасы әзірлеушілер қауымдастығында ғана емес, бизнес-процесстерді жобалауда да көп қолданысқа ие, танымал диаграмма. Ол хабарламаларды уақыт бойынша олардың орындалу уақытына сәйкес тәртіпке келтіруді айқындайтын диаграмма, яғни бір уақыт мезетіндегі бірнеше объектілер арасындағы өзара іс-қимылды жобалайды. Атап айтқанда, онда өзара іс-қимылға қатысатын объектілер және олармен алмасатын хабарламалардың толық тізбегі көрсетіледі.

Тізбек диаграммасының құраушы элементтері – объектілер, хабарламалар және оған қайтарылатын нәтижелер. Объектілер – атаулары жазылатын

тікбұрыштармен, хабарламалар (әдістер шақырулары) – тік сызықтармен, қайтарылатын нәтижелер – нүктелі сызықтармен белгіленеді.

Жобаның төмендегі 3.1-суретте көрсетілген тізбек диаграммасында табель не акттың құрылуынан бастап расталып аяқталғанға дейінгі процесстері сипатталды. Табельші, расталып аяқталмаған табель/акт, енгізілетін өзгерістер, табель не акт көрінісі, расталып аяқталған табель/акт және растаушы тұлғалар объект рөлін атқарады. Табельші табель/акт құруға сұраныс жібереді, нәтижесінде жаңа құрылған табель/акт қайтарылады. Құру барысында таңдалынған растаушы тұлғаларға растауға сұраныс жіберіледі. Өз кезегінде, растаушы тұлға өз тізіміне жаңадан қосылған табель/актті көруге сұраныс жіберіп, растау не растамайтынын шешеді. Растамаған жағдайда растаушы тұлға табельшіге енгізілу керек өзгерістерді хабарлайды. Расталмаған табель/акт табельшінің және растаушының аяқталмаған тізімінде қалады. Табельші өзгеріс енгізіп, растаушыға қайта сұраныс жібереді. Нәтижесінде, дұрыс болған жағдайда растаушы тұлға растап, табель/акт аяқталған табель/акттар тізіміне көшеді.



3.1-сурет – Жобаның тізбек диаграммасы

4 Жоба құрылымы

4.1 «Electronic timesheet» жобасы

«Electronic timesheet» жобасы көпфункционалды веб-қосымша ретінде жүзеге асырылды. Қосымша қызметкерлердің нақты жұмыс істеген уақытын енгізуге, белгіленген жұмыс режимінің сақталуын бақылауға және барлық жұмыс істеген уақыт кезеңі туралы ақпарат алуға мүмкіндік береді. Жұмыс уақытын есепке алу табеліне енгізілген қажетті мәліметтер негізінде бухгалтерия қызметкерлері ұйым қызметкерлеріне жалақы есептейді және өзге де төлемдерді жүзеге асырады. Қазіргі сәтте университетімізде бұл жүйе қолмен жүргізіліп отырғандықтан алдағы уақытта бұл веб-қосымша университет қызметкерлерінің жұмысын бір сатыға жеңілдетеді.

Жобада екі қолданушы жағы бар – табель/акт толтыратын табельші және толтырылған табельді растайтын басқарма мүшелері. Бұл бөлімде табельші жағы сипатталатын болады.

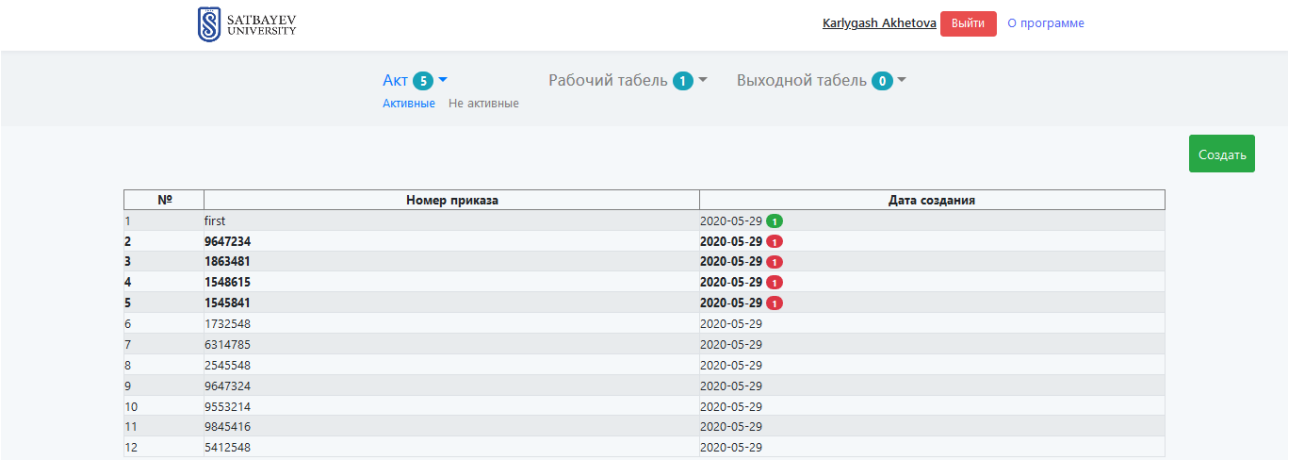
4.2 Қолданушы интерфейсі

Жүйедегі әзірленген негізгі нысан – университет қызметкерлерінің жұмыс уақытын есепке алу құжаттары – акт, жұмыс және демалыс күндері табелі. Бұл құжаттарды толтыратын табельші ретінде әр департамент, кафедра бөлімінде бір қызметкер тағайындалады. Жүйе қолданушыларына әр түрлі деңгейлі қолжетімділікті қамтамасыз ететін логин мен құпиясөз ДИС бөлімі қызметкерлерімен беріледі.

Табельші логин мен құпиясөзін дұрыс тере отырып жүйеге кіреді. Ол тек өз департамент қызметкерлеріне ғана аталған құжаттарды толтыруға құқылы, құрылған құжаттар тізімі табельшінің негізгі бетінде көрсетіледі.

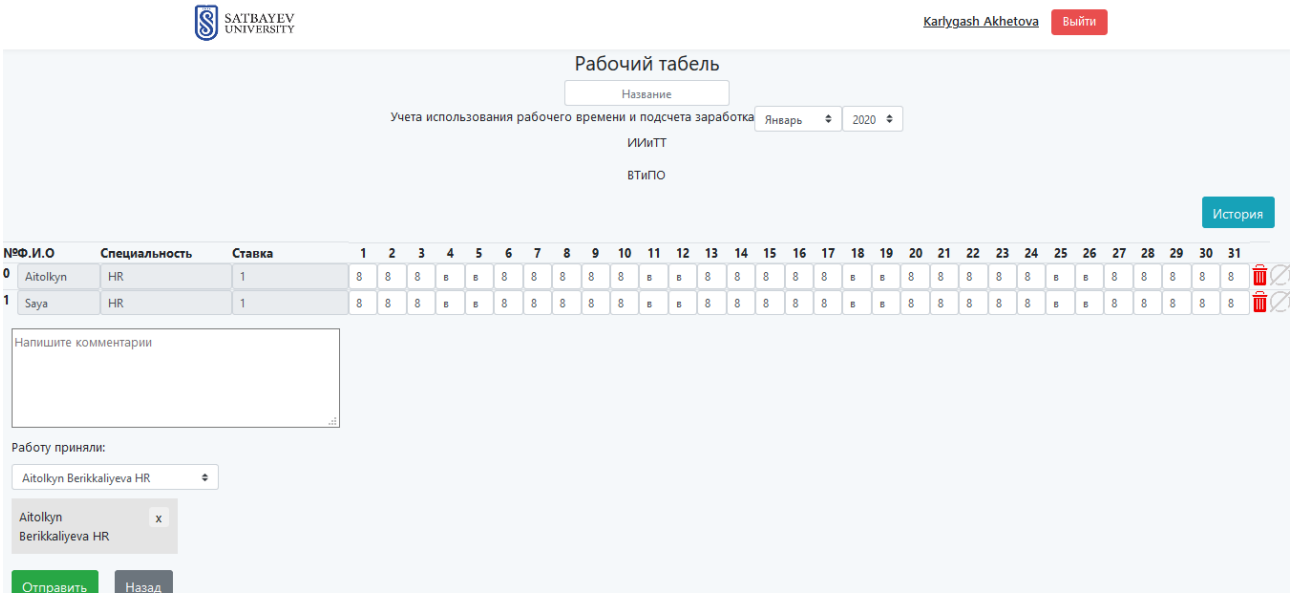
Меню бойындағы әр құжат тұсында қолданушыға келген жаңа хабарламалардың (растаушы әрекеттері) жалпы саны көрсетіледі. Әр құжат тұсында расталу статусын түспен ерекшелейтін белгі болады: жасыл белгі – растаушының бірі растаса, қызыл – растамаған жағдайда.

Құрылған табель/акт расталу статусына қарай расталып аяқталмаған және аяқталған деп екі тізімге ажыратылады. Әрбір жаңа құрылған табель/акт тізімнің жоғарысына қосылып отырады. Төмендегі 4.1-суретте табельшінің негізгі беті көрсетілген.



4.1-сурет – Табельші беті

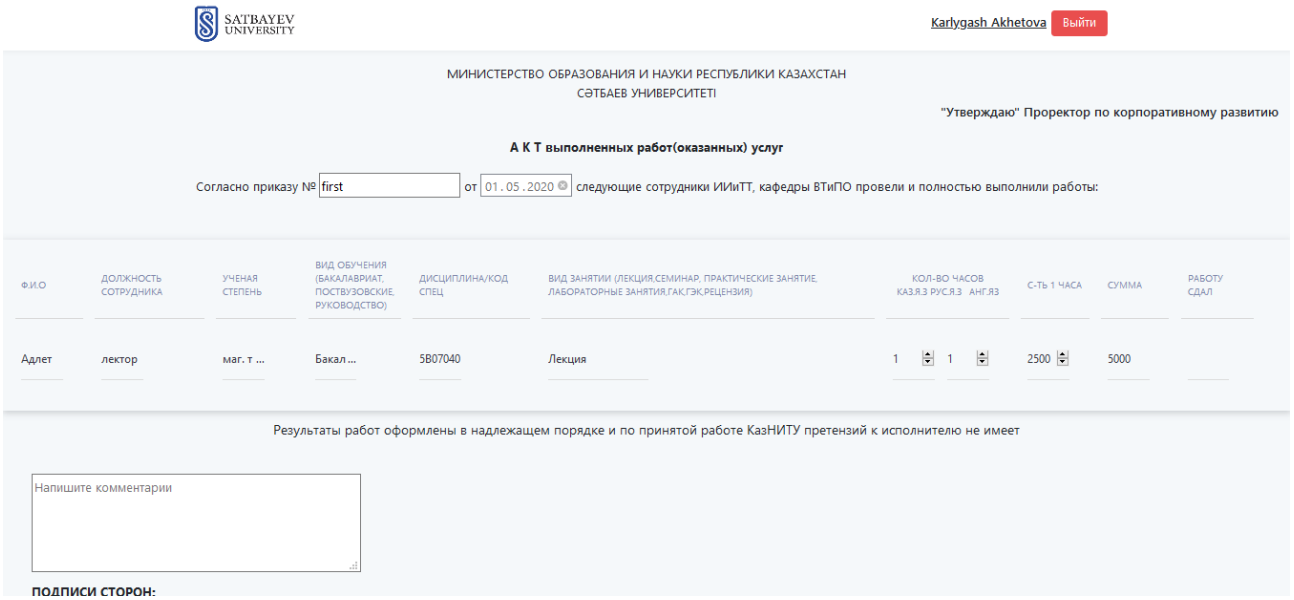
Табельші «Создать» батырмасымен жаңа табель құру бетіне көшеді. Табельді құру бетінде табель аты жазылады, ай мен жылы таңдалынады. Институт пен кафедра табельшінің жұмыс істейтін департаментіне байланысты деректер қорынан алынады. Қызметкерлер және растаушы тұлғалар тізімі де тек өз департамент мүшелерінен тұрады. Жұмыс күніне арналған табелдегі қызметкерлердің жұмыс істеу сағаты автоматты түрде толтырылады, демалыс күндері «В» әрпімен көрсетіледі. Әр қызметкердің сағат саны жұмыс істеу ставкасына байланысты есептеледі немесе оны кестеде өзгертуге мүмкіндік бар. Әр қызметкер тұсында тізімнен алып тастау (удалить) немесе сағат санын өшіру (обнуление) батырмалары тұрады. Кесте астында комментарий жазу орны бар, жазылған комментарий таңдалған растау мүшелеріне жіберіледі. (4.2-сурет)



4.2-сурет – Жұмыс күндері табелін құру беті

Жұмыс күндері мен демалыс табелі арасындағы айырмашылық – жұмыс табелінде сағат сандары автоматты түрде қызметкер ставкасына байланысты толтырылады, демалыс күніне жасалатын табелде сағаттар бос болады. Табель тек демалыс немесе мереке күндері жұмыс істеген қызметкерлерге толтырылатындықтан тек керекті күнге істелген сағат саны жазылады.

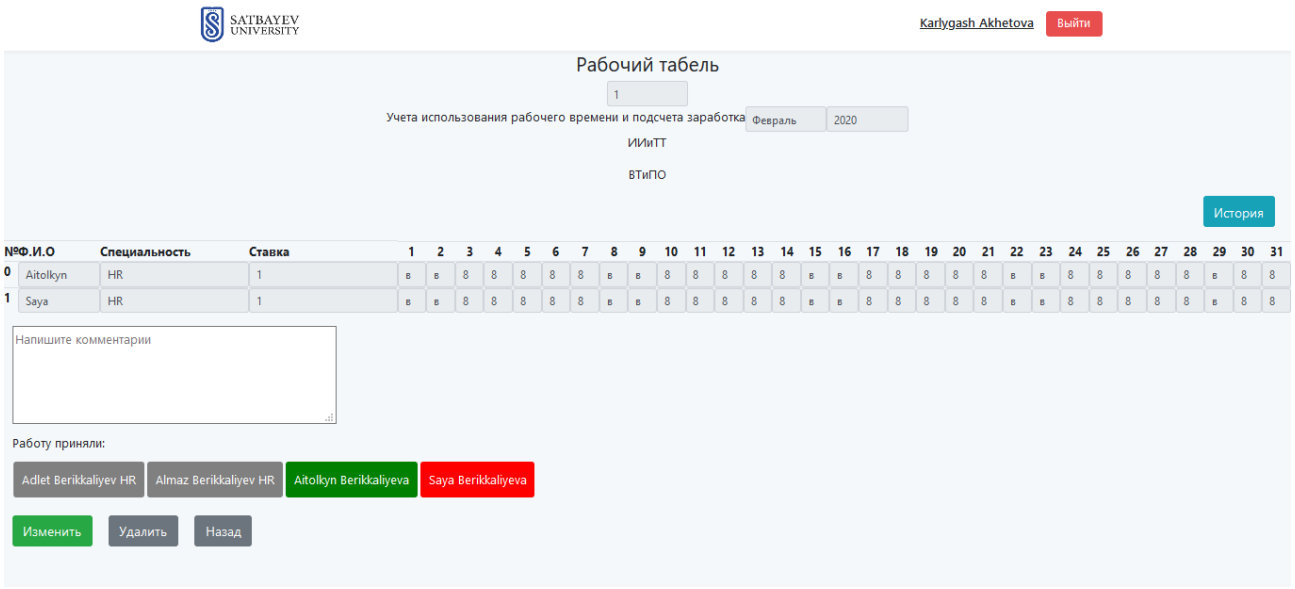
Акт құжаты департаменттің уақытша қызметкерлеріне толтырылады, сол себепті акт толтыру барысында қызметкерлер қолмен енгізіледі. Құру бетінде акт аты, бұйрық номері мен уақыты жазылады. «Добавить строку» батырмасымен жаңа қызметкер енгізіледі. Енгізу жолағына қызметкерге қатысты бірнеше мәліметтер жазылады (аты, мамандығы, ғылыми дәрежесі, оқыту түрі (бакалавриат, поствузовский), пән коды, пән түрі (лекция, семинар, лабораториялық), сағат саны, сағат бағасы). Енгізілген қызметкердің сағат саны мен бағасы негізінде суммасы автоматты түрде есептеліп, сумма атты бағанға жазылады. Растаушы тұлғалар таңдалынады. Акт құру беті төменде 4.3-суретте көрсетілген.



4.3-сурет – Акт құру беті

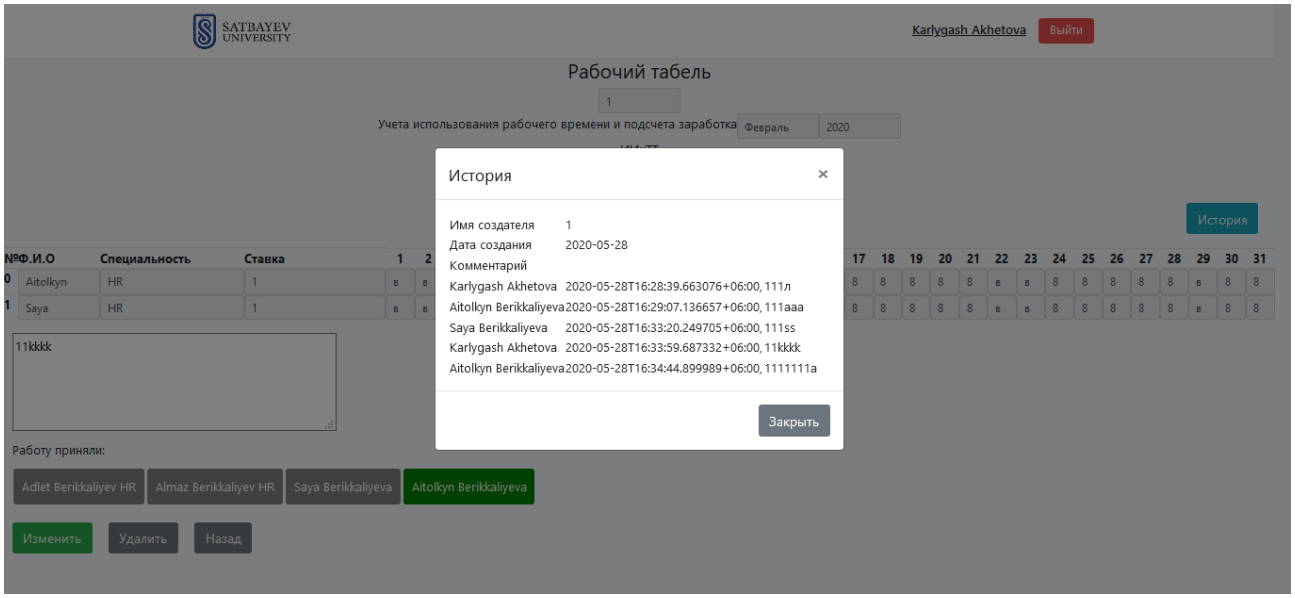
Құрылған табель/акт бетінде растау мүшелерінің расталған немесе расталмағыны көрінеді: растаған жағдайда жасыл түспен, растамаған – қызыл түспен, әлі ешқандай әрекет байқалмаса сұр түспен растаушы аттары боялады. Төменде 4.4-суретте табель беті көрсетілген.

Жасалған табель/актты өзгертуге немесе жоюға мүмкіндік бар. Өзгеріс енгізу кезінде табель атын өзгертуге, қызметкерді тізімнен алып тастауға, сағат санын өшіруге/өзгертуге болады. Алайда, растау мүшелерін, таңдалған уақытты өзгертуге мүмкіндік жоқ.



4.4-сурет – Табель беті

Жасалған табельде «История» батырмасы бар. Мұнда:
– жасаған табельші аты;
– жасалған уақыты;
– соңғы өзгеріс енгізілген уақыт;
– комментарий жазған растаушы аты;
– жазылған әрбір комментарий көрсетіледі. «История» окносы төменде 4.5-суретте көрсетілген.



4.5-сурет – «История» окносы

ҚОРЫТЫНДЫ

Дипломдық жобамен жұмыс барысында университет қызметкерлеріне акт және табель толтыру жүйесін электронды қамтамасыз ету үшін «Electronic timesheet» веб-қосымшасы әзірленді. Әзірленген жүйеде пайдаланушыларды жүйенің элементтеріне түрлі деңгейлі қолжетімділікпен топтарға бөлу қарастырылған: администратор, табельші және табельді растаушы. Табельші ортасында іске асырылған мүмкіндіктер: пайдаланушының жүйеге авторизациясы, табель мен акт құру, табелдің демалыс күндері мен жұмыс күндеріне арналған түрлері, табелді өзгерту, жою, құрылған құжаттың «История»-сын көру, жасалған табелдер мен актты расталу статусына қарап сорттау, табель ішінде растау статусын түспен көрсету, комментарий алмасу. Осы және бұлардан өзге қосымша функционалдарға ие электронды табель веб-қосымшасы университет қызметкерлерінің жұмысын оңтайландырып, жеңілдетеді, қағазбастылықты жоюға мүмкіндік береді.

Әзірлеу барысында қолданылған технологиялар тізімі: бэкенд бөлігіне Django фреймворкі, фронтенд бөлігіне Angular фреймворкі, деректер қорын жүзеге асыру мақсатында PostgreSQL тілі.

Болашаққа жоспар ретінде жүйеге электронды қол қою функционалын жүзеге асыру қарастырылып жатыр. Сондай-ақ, университетіміздегі қолданыстағы СКУД жүйесімен интеграцияны қарастыру қажет. Табелде таңдалынған қызметкердің күндік қатысуы СКУД жүйесінен алынып, автоматты түрде кестеге толтырылып, бақылау жүйесін жетілдіру мүмкіндігі. Аталған жаңалықтарды іске асыру жүйемен жұмысты жақсартуға және қамтаманың құндылығын арттыруға мүмкіндік береді.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 WEB-приложение или мобильное приложение. Что выбрать? // Сайттың электрондық нұсқасы https://punicapp.com/

2 Что такое Python и для чего он используется // Сайттың электрондық нұсқасы https://all-python.ru

3 Гринфилд Д.Р., Гринфилд А.Р. Two Scoops of Django. – Лос-Анджелес: Two scoops press, 2015. – 521 б.

4 Дронов В. Django: практика создания Web-сайтов на Pyhton – СПб.: БХВ – Петербург, 2016. – 528 б.

5 Бесплатная электронная книга Учусь Django-rest-framework [Электронды ресурс] – Кіру режимі: https://riptutorial.com/Download/django-rest-framework-ru.pdf, ашық

6 Справочник Angular // Сайттың электрондық нұсқасы https://xsltdev.ru/angular/

7 Wilken J. Angular in action – Нью-Йорк: Manning Publications, 2018. – 320 б.

8 Django documentation // Сайттың электрондық нұсқасы https://docs.djangoproject.com/en/3.0/

9 Дегтярев А., Django, начало работы с базой данных // Сайттың электрондық нұсқасы https://habr.com/ru/post/87357/

10Angular documentation // Сайттың электрондық нұсқасы https://angular.io/

11Angular documentation // Сайттың электрондық нұсқасы https://v2.angular.io/docs/ts/latest/guide/

12Установка пакетов Pyhton// Сайттың электрондық нұсқасы https://devpractice.ru/python-lesson-16-install-packages/

А Қосымшасы
(міндетті)

«Electronic Timesheet» веб-қосымшасын әзірлеуге арналған
техникалық тапсырма

А.1 Жалпы сипаттама

Бұл техникалық тапсырма университет қызметкерлерінің жұмыс уақытын есепке алу мен бақылау жүйесін автоматтандыруға арналған бағдарламалық қамтама әзірлеуге бағытталады. Аталған жүйе тұрақты қызметкерлерге толтырылатын жұмыс және демалыс күндері табельдері, уақытша қызметкерлерді есепке алу актын қамтиды. Қазіргі уақытта жүйе қолмен жүзеге асырылатындықтан университет қызметкерлерінің жұмысын жеңілдету мақсатында электронды жүйе әзірлеуге қажеттілік туындады. Бағдарламалық қамтама қолданушылары ретінде аталған құжаттарды толтыратын әр кафедра және департамент бөлімшесінің қызметкерлері және құрылған құжатты растайтын басқарма мүшелері қарастырылған. Автоматтандырылған жүйе құжаттарды толтыру мен растау уақытын азайтуға, қателіктердің алдын алуға, қағазбастылықты жоюға мүмкіндік береді.

А.2 Функционалдық сипаттамаларына талап

Жүйе келесі функциялардың орындалуын қамтамасыз етуі тиіс:
– жүйеге тек оқу орнымен белгіленген тұлғалардың кіруі;
– табель және актты толтыру, өзгерту немесе жою;
– толтыру барысында қажетті барлық ақпараттарды енгізу, өзгерту;
– растау не растамау;
– комментарий жазу;
– ағымдағы расталу статусын көрсету;
– расталып аяқталуына байланысты сорттау;
– расталу уақыты ішіндегі История-сын сақтау;
– енгізілген мәліметтердің ұзақ мерзімге сақталуы.
Қолданылатын деректер:
– оқу орны қызметкерлерінің тізімі;
– оқу орны департаменттер және бөлімшелері тізімі;
– есептік кезең ішіндегі қызметкердің жұмыс істеген уақыты туралы ақпарат, толтыру бұйрықтары.

Нәтиже:

– растаушы тұлғалармен тексеріліп, расталған акт, жұмыс және демалыс күндері табелі;

– әр құжаттың жеке История-сы;

А.3 Сенімділікке талап

Жүйемен жұмыс барысында қолданушының қарастырылмаған, қате әрекеттерін бұғаттауды қарастыру. Енгізілетін ақпаратты бақылауды және сақталатын ақпараттың тұтастығын қамтамасыз ету.

А.4 Аппараттық интерфейске талап

Қосымша әзірленетін және жүргізілетін компьютерге қойылатын жалпы талаптар:

– процессор – P4 және одан жоғары;

– оперативті жады – 512 MB;

– диск тұлғалы кеңістік – ~ 52 MB + қолданушылар файлдарын сақтауға қажет орын;

– Internet желісіне қатынау;

– қатты диск 40 Gb; 1024×768 рұқсатта жұмыс қолдайтын видеокарта;

– пернетақта, манипулятор тышқан.

А.5 Бағдарламалық үйлесімділікке талап

Бағдарламалық қамтама Linux операциялық жүйесімен үйлесімді жұмыс істеуі қажет.

А.6 Программалық интерфейске талап

Жүйе әзірлеуге қажетті бағдарламалық компоненттер:

– visual studio code бағдарламасы;

– PostgreSQL мен PgAdmin бағдарламалық құралы;

– python мен pip менеджерлік пакетінің соңғы нұсқасы;

– django пакеті;
– node.js пен npm менеджерлік пакетінің соңғы нұсқасы;
– angular пакеті.

А.7 Бағдарламалық құжатқа талап

Жүйедегі табель мен акт белгіленген бірыңғай стандартта қарастырылған әр талапқа сай болуы тиіс.

Әзірлеу барысында әр бағдарламалық модульге қажетті түсініктемелер жазылуы тиіс. Қолданушы түсініктілігін қамтамасыз ету үшін қосымша және оның элементтері мен функционалдық мүмкіндіктеріне анықтамалық ақпараттар болуы қажет.

А.8 Қолданушы интерфейсіне талап

Қосымша қолданушыға түсінікті, ыңғайлы және қарапайым болуы үшін:
– қаріптердің оңай оқылуы;
– графикалық элементтердің анықтылығы;
– артық дыбыс, әуендердің болмауы;
– кажет емес ақпараттың болмауы;
– беттегі элементтердің графикалық және логикалық тізбекте дұрыс орналасуы;
– беттің тез жүктелуін қамтамасыз ету керек. Сонымен қатар, косымшада ұсынылатын барлық функционалдық мүмкіндіктер қолжетімді болуы тиіс.

А.9 Коммуникациялық интерфейске талап

Қолданушы компьютерінде интернет желісіне қолжетімділікді қамтамасыз ететін модем құралы немесе жылдамдығы 100 Мб/с кем емес карта болуы тиіс. Сервер мен қолданушы арасындағы байланыс TCP/IP протоколы негізінде жүзеге асырылады.

А.10 Терминдер және қысқартулар

1-кестеде жоба сипаттамасында қолданылған барлық терминдер мен қысқартулар және жобаны бағдарламалық әзірлеу кезінде пайдаланылатын технологиялар анықтамалары жазылған.

А.1-кесте – Анықтамалар, терминдер және қысқартулар

Қысқартулар мен терминдер	Анықтамалар
Қосымша, қолданба	белгілі бір пайдаланушылық тапсырмаларды орындауға және пайдаланушымен тікелей қарым-қатынас жасауға арналған бағдарлама.
Веб-браузер	беттер мен веб-құжаттардың мазмұнын көруге, веб-қосымшаларды басқаруға, басқа да міндеттерді шешуге арналған қолданбалы бағдарлама.
Бағдарламалау тілі	орындаушы үшін (мысалы, компьютер) бағдарламалар жазуға арналған формальды тіл.
Қолданушының графикалық интерфейсі (GUI)	Graphical user interface, жүйемен қарапайым өзара әрекеттесу үшін пайдаланушыға берілетін интерфейс.
HTML, CSS	HyperText Markup Language, құжаттарды белгілеудің стандартты тілі. Cascading Style Sheets, құжаттың сыртқы түрін сипаттаудың формальды тілі.
HTTP хаттамасы	HyperText Transfer, деректерді HTML гипермәтіндік құжаттар түрінде берудің қолданбалы деңгейінің хаттамасы.
Консоль	пернетақтадан мәтіндік командаларды енгізу арқылы компьютерді басқаруға мүмкіндік беретін арнайы бағдарлама.
PATH айнымалы ортасы	жүйелік айнымалы, орындалатын файлды консольден шақыру кезінде ОЖ іздейтін директориялар тізімі.
URL	Uniform Resource Locator, электрондық ресурстардың біріздендірілген мекенжайларының жүйесі немесе ресурстың орналасқан жерін біркелкі анықтаушы.
API	Application Programming Interface, сұраулар қабылдайтын және жауаптар жіберетін сервердің құрамдас бөлігі.

Б Қосымшасы
(міндетті)

Бағдарлама мәтіні

1. *Home.module.ts*

```
import { NgModule, CUSTOM_ELEMENTS_SCHEMA } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { WorkingHoursTabelComponent } from './working-hours-tabel/working-hours-
tabel.component';
import { HomeRoutingModule } from './home.routing';
import { PageComponent } from './page/page.component';

import { ActListComponent } from './act/act-list/act-list.component';
import { ActSaveComponent } from './act/act-save/act-save.component';
import { HolidayTabelListComponent } from './holiday/holiday-tabel-list/holiday-tabel-
list.component';
import { HolidayTabelSaveComponent } from './holiday/holiday-tabel-save/holiday-tabel-
save.component';
import { ProfileListComponent } from './profile-list/profile-list.component';
@NgModule({
  // tslint:disable-next-line:max-line-length
  declarations: [WorkingHoursTabelComponent, PageComponent, ActListComponent, ActSaveCo
mponent, HolidayTabelListComponent, HolidayTabelSaveComponent, ProfileListComponent],
  imports: [
    CommonModule,
    FormsModule,
    ReactiveFormsModule,
    HomeRoutingModule,
  ],
  schemas: [CUSTOM_ELEMENTS_SCHEMA]})
export class HomeModule { }
```

2. *Home.routing.ts*

```
import { NgModule, CUSTOM_ELEMENTS_SCHEMA } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { WorkingHoursTabelComponent } from './working-hours-tabel/working-hours-
tabel.component';
import { PageComponent } from './page/page.component';
import { HolidayTabelSaveComponent } from './holiday/holiday-tabel-save/holiday-tabel-
save.component';
import { AuthGuard } from 'src/app/core/auth.guard';
import { ActSaveComponent } from './act/act-save/act-save.component';
import { ProfileListComponent } from './profile-list/profile-list.component';
const routes: Routes = [
  {
    path: '',
```

```
    redirectTo: 'page',
    pathMatch: 'full',
    canActivate: [AuthGuard]
  },
  {
    path: 'conformer',
    component: ProfileListComponent,
    canActivate: [AuthGuard]
  },
  {
    path: '',
    children: [
      {
        path: 'working',
        component: WorkingHoursTabelComponent,
        canActivate: [AuthGuard]
      },
      {
        path: 'working/:id',
        component: WorkingHoursTabelComponent,
        canActivate: [AuthGuard]
      },
      {
        path: 'holiday',
        component: HolidayTabelSaveComponent,
        canActivate: [AuthGuard]
      },
      {
        path: 'holiday/:id',
        component: HolidayTabelSaveComponent,
        canActivate: [AuthGuard]
      },
{
        path: 'page',
        component: PageComponent,
        canActivate: [AuthGuard]
      },{
        path: 'page/:id',
        component: PageComponent,
        canActivate: [AuthGuard]},
      {
        path: 'act',
        component: ActSaveComponent,
        canActivate: [AuthGuard]
      },
      {
        path: 'act/:id',
        component: ActSaveComponent,
        canActivate: [AuthGuard]
```

```
    }
  ]}];

@NgModule({
 imports: [RouterModule.forChild(routes)],
 exports: [RouterModule],})
export class HomeRoutingModule { }
```

3. Page.component.html

```
<nav class="navbar navbar-expand-lg navbar-light navbar-laravel">
 <div class="container">
    <a class="navbar-brand" href="#">Satbayev University</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item">
          <span class="user-text">{{user}}</span>
        </li>
        <li class="nav-item exit-btn">
          <a [class.blue-link] = "isRegister" (click)="exit()">Выйти</a>
        </li>
      </ul>
    </div>
 </div>
</nav>
<nav class="navbar navbar-light bg-light justify-content-between">
<div class="container">
  <ul class="menu">
   <li [class.active] ="isAct"  (click)="onMenuChange('act')" class="nav-item dropdown">
     <a class=" dropdown-toggle" id="navbarDropdownMenuLink">Акт</a>
      <ng-container  *ngIf="isAct">
       <div style="color: #7c7c7c; display:block; font-size: 15px;">
        <a [class.active] = "isActOneActive" (click)="onAct('active')">Активные</a>
        <a [class.active] = "isActOneHistory" (click)="onAct('history')">Не активные</a>
       </div>
    </ng-container>
   </li>
   <li [class.active] = "isWork" (click)="onMenuChange('work')" class="nav-item dropdown">
     <a class=" dropdown-toggle" id="navbarDropdownMenuLink">Рабочий табель</a>
     <ng-container  *ngIf="isWorkDropdown">
       <div style="color: #7c7c7c; display:block; font-size: 15px;">
        <a [class.active] = "isTabelOneActive" (click)="onWorkTabel('active')">Активные</a>
        <a [class.active] = "isTabelOneHistory" (click)="onWorkTabel('history')">Не активные</a>
       </div>
```

```
      </ng-container>
    </li>
    <li [class.active] = "isWeekend" (click)="onMenuChange('weekend')" class="nav-
item dropdown">
    <a class=" dropdown-toggle" id="navbarDropdownMenuLink">Выходной табель</a>
     <ng-container  *ngIf="isWeekendDropdown">
        <div style="color: #7c7c7c; display:block; font-size: 15px;">
         <a [class.active] = "isTabelTwoActive" (click)="onWeekendTabel('active')">Активные</a>
         <a [class.active] = "isTabelTwoHistory" (click)="onWeekendTabel('history')">Не активные
</a>
       </div>
      </ng-container>
    </li>
  </ul>
 <!-- <form class="form-inline">
  <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search">
  <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
</form>-->
</div>
</nav>
<div *ngIf="isWork">
 <div class="content-btn" *ngIf="isTabelOneActive">
  <button  class="btn btn-success my-2 my-sm-0 submit-
btn" type="submit" (click)="createTabel()">Создать</button>
  </div>
<div style="margin:10px">
  <table class="table table-striped table-bordered" *ngIf="sourceOne">
    <thead>
      <th>№</th>
      <th>Называния</th>
      <th>Год</th>
      <th>Месяц</th>
      <th>Дата создания</th>
    </thead>
    <tbody>
      <tr *ngFor="let t of sourceOne; let i = index" class="clickable-
row" (click)="onRowSelected(t.id)">
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{i+1}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.name}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.year}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.month}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.createDate}}</td>
      </tr>
    </tbody>
  </table>
</div>
</div>
<div *ngIf="isWeekend">
<div style="padding:10px" *ngIf="isTabelTwoActive" class="content-btn">
```

```
  <button  class="btn btn-success my-2 my-sm-0 submit-
btn"  type="submit" (click)="createTabel()">Создать</button>
</div>
<div style="margin:10px">
  <table class="table table-striped table-bordered" *ngIf="source">
    <thead>
      <th>№</th>
      <th>Называния</th>
      <th>Год</th>
      <th>Месяц</th>
      <th>Дата создания</th>
    </thead>
    <tbody>
      <tr *ngFor="let t of source; let i = index" class="clickable-
row" (click)="onRowSelectedTwo(t.id)">
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{i+1}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.name}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.year}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.month}}</td>
        <td [className]="t.status === 'DECLINED'? 'active': 'onProcess'">{{t.createDate}}</td>
      </tr>
    </tbody>
  </table>
</div>
</div>
<div *ngIf="isAct">
<app-act-
list [data]="defaultSource" [isActOneActive]="isActOneActive" [isActOneHistory]="isActOneHist
ory"></app-act-list>
</div>
```

4. Page.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Router, ActivatedRoute } from '@angular/router';
import { HomeService } from 'src/app/data/services/home.service';
import { getMonthes } from 'src/app/data/utils/data-generator';
@Component({
  selector: 'app-page',
  templateUrl: './page.component.html',
  styleUrls: ['./page.component.scss']
})
export class PageComponent implements OnInit {
  id: any;
  user: any;
  postionId: any;
  isAct: boolean;
  isWork: boolean;
  isWeekend: boolean;
```

```
isTabelOneActive: boolean
isTabelOneHistory: boolean;
isWorkDropdown: boolean;
isWeekendDropdown: boolean;
isTabelTwoActive: boolean;
isTabelTwoHistory: boolean;
isActOneActive: boolean;
conformerAct = false;
isActOneHistory: boolean;
monthes = getMonthes();
defaultSource: any;
source: any;
sourceOne: any;
actSource: any;
settings = {
  noDataMessage: 'Нет данных',
  actions: false,
  pager: {
    display: false
  },
  columns: {
    id: {
      title: '№',
      width: '5%'
    },
    createDate: {
      title: 'Дата',
      width: '10%'
    },
    name: {
      title: 'Называния',
      width: '5%'
    }},
  hideSubHeader: true,
  attr: {
    class: 'table table-bordered'
  }};
constructor(private router: Router,
            private route: ActivatedRoute,
            private service: HomeService,
          ) { }
ngOnInit() {
  this.initValues();
  this.isAct = true;
  this.isActOneActive = true; }
initValues() {
      this.id = localStorage.getItem('id');
      this.user = localStorage.getItem('name');
      this.postionId = localStorage.getItem('postionId');
```

```
}
onMenuChange(type) {
  this.isAct = type === 'act';
  this.isWork = type === 'work';
  this.isWeekend = type === 'weekend';
  this.isWorkDropdown = false;
  this.isWeekendDropdown = false;
  if (type === 'work') {
    this.isWeekendDropdown = false;
    this.isWorkDropdown = true;
  } else if (type === 'weekend'){
    this.isWeekendDropdown = true
    this.isWorkDropdown = false;
  }}
onWorkTabel(type) {
 if (type === 'active') {
  this.isTabelOneActive = true;
  this.isTabelOneHistory = false;
  this.isWorkDropdown = false;
  this.getlist('CREATED');
 } else {
  this.sourceOne = 0;
  this.isTabelOneActive = false;
  this.isTabelOneHistory = true;
  this.isWorkDropdown = false;
  this.getlist('DONE');
 }}
getlist(type) {
  if(type === 'CREATED'){
  this.service.getActiveWorlTabel(this.id, 'CREATED').subscribe((res: any) => {
    this.sourceOne = res.map(x=>({
      ...x,
      month: this.getMonthValue(x.month)
    }));
    this.service.getActiveWorlTabel(this.id, 'DECLINED').subscribe((result: any) => {
      this.sourceOne.push(...result.map(x=>({
        ...x,
        month: this.getMonthValue(x.month)
      }) ));
      console.log('this is tabel:', this.sourceOne)
    this.sourceOne.reverse();
  });})}
  else if(type === 'DONE'){
  this.service.getActiveWorlTabel(this.id, 'DONE').subscribe((res: any) => {
    this.sourceOne = res.map(x=>({
      ...x,
      month: this.getMonthValue(x.month)
    })
    );
```

```
    this.sourceOne.reverse();
  })}}
onWeekendTabel(type) {
 if (type === 'active') {
   this.isTabelTwoActive = true;
   this.isTabelTwoHistory = false;
   this.isWeekendDropdown = false;
   this.getlistHolidayTabel('CREATED');
 } else {
   this.source = 0;
   this.isTabelTwoActive = false;
   this.isTabelTwoHistory = true;
   this.isWeekendDropdown = false;
   this.getlistHolidayTabel('DONE');
 }}
getlistHolidayTabel(type) {
 if(type === 'CREATED'){
  this.service.getActiveHolidayTabel(this.id, 'CREATED').subscribe((res: any) => {
    this.source = res.map(x=>({
      ...x,
      month: this.getMonthValue(x.month)
    }));
    this.service.getActiveHolidayTabel(this.id, 'DECLINED').subscribe((result: any) => {
      this.source.push(...result.map(x=>({
        ...x,
        month: this.getMonthValue(x.month)
      })));
      console.log('this is tabel:', this.sourceOne)
    this.source.reverse();
  }); }}}
 else if(type === 'DONE'){
  this.service.getActiveHolidayTabel(this.id, 'DONE').subscribe((res: any) => {
    this.source = res.map(x=>({
      ...x,
      month: this.getMonthValue(x.month)
    }) );
    this.source.reverse();
 })}}
onAct(type) {
   if (type === 'active') {
   this.isActOneActive = true;
   this.isActOneHistory = false;
   this.isAct = false;
 } else {
   this.actSource = 0;
   this.isActOneActive = false;
   this.isActOneHistory = true;
   this.isAct = false;
 }
```

```
 }
 getMonthValue(value) {
  const v = getMonthes();
  const a = v.find(x => x.value === value).name;
  return a;
}
 onRowSelected(id) {
   if (this.isWork) {
    if (id && this.isTabelOneActive) {
      this.router.navigate(['working'], {queryParams: { tabelWork: true, id}});
    } else if (id && this.isTabelOneHistory) {
      this.router.navigate(['working'], {queryParams: { tabelWork: true, id}});
    }}}
onRowSelectedTwo(id) {
   if (this.isWeekend) {
    if ( this.isTabelTwoActive) {
       this.router.navigate(['holiday'], {queryParams: { tabelWork: true, id}});
     } else if (this.isTabelTwoHistory) {
       this.router.navigate(['holiday'], {queryParams: { tabelWork: true, id}});
     }}}
  createTabel() {
    if (this.isWork) {
      this.router.navigate(['working'], {queryParams: {tabelWork: true}});
    } else if (this.isWeekend) {
      this.router.navigate(['holiday'], {queryParams: { tabelWork: true}});
    }}
  exit() {
    this.router.navigate(['auth']);
    localStorage.removeItem('name');
    localStorage.removeItem('id');
    localStorage.removeItem('postionId');
   }}
```

5. *Models of database*

```
class Working(models.Model):
    name = models.CharField(max_length = 200, blank = False)
    year = models.IntegerField(blank = False)
    month = models.IntegerField(blank = False)
    data =  models.TextField(blank = False)
    status = models.CharField(max_length = 200, blank = False)
    createdBy =  models.IntegerField(blank = False)
    createDate = models.DateField(auto_now = False, blank = False)
    lastUpdateDate = models.DateField(auto_now = False, blank = False)
    updatedBy =  models.IntegerField(blank = False)
    comment = models.CharField(max_length = 1000, blank = True)
class HolidayTabel(models.Model):
    name = models.CharField(max_length = 200, blank = False)
    year = models.IntegerField(blank = False)
```

```
    month = models.IntegerField(blank = False)
    data =  models.TextField(blank = False)
    status = models.CharField(max_length = 200, blank = False)
    createdBy =  models.IntegerField(blank = False)
    createDate = models.DateField(auto_now = False, blank = False)
    lastUpdateDate = models.DateField(auto_now = False, blank = False)
    updatedBy =  models.IntegerField(blank = False)
    comment = models.CharField(max_length = 1000, blank = True)
class Acts(models.Model):
    docNumber = models.CharField(max_length = 1000, blank = False)
    docDate = models.CharField(max_length = 1000, blank = False)
    createdBy = models.IntegerField(blank = False)
    createdDate = models.DateField(auto_now = False, auto_now_add = True, blank = False)
    lastUpdateDate = models.DateField(auto_now = True, blank = False)
    updatedBy =  models.IntegerField(blank = False)
    comment = models.CharField(max_length = 1000, blank = True)
    status = models.CharField(max_length = 1000, blank = False)
    data = models.TextField(blank = False)
class Employee(models.Model):
    firstName = models.CharField(max_length = 200, blank = False)
    lastName = models.CharField(max_length = 200, blank = False)
    middleName = models.CharField(max_length = 200, blank = False)
    speciality = models.CharField(max_length = 200, blank = False)#tester, teacher, director, dean
    rate = models.FloatField(blank = False)
    postionId = models.IntegerField(blank = False)
    password = models.CharField(max_length = 200, blank = False)
    email = models.CharField(max_length = 200, blank = False)
    def __str__(self):
        return self.password
class Position(models.Model):
    name = models.CharField(max_length = 200, blank = False)
    departmentID = models.IntegerField(blank = False)
class Department(models.Model):
    name = models.CharField(max_length = 200, blank = False)
```

6. URL addresses

```
urlpatterns = [
    path('employee/<int:postionId>/<int:id>', views.getEmployeesByPositionId),
    path('position/', views.position),
    path('register/', views.register),
    path('login/', views.login),
    path('worktabel/', views.worktabel),
    path('worktabel/<int:createdBy>/<str:type>', views.getWorkTabelByEmployeeId),
    path('worktabelByID/<int:id>', views.getWorkTabelByTabelId),
    path('working/<int:id>', views.deleteWorking),
    path('workingflag/<int:id>', views.conformerWorkingFlagEdit), #my change
    path('conformers/<int:id>', views.conformerList),
    path('conformer/',views.conformer),
```

```
path('statusByTabelId/<int:id>', views.getStatusOfTabel),
path('departments/', views.getDepartments),
path('positions/<int:id>', views.getPositionById),
path('positionNameById/<int:id>', views.getPositionName),
path('departmentNameById/<int:id>', views.getDepartmentName),
# holidayTabel
path('holidaytabel/', views.holidaytabel),
path('holidayflag/<int:id>', views.conformerHolidayFlagEdit),
path('holiday/<int:id>', views.deleteHoliday),
path('conformerholiday/',views.conformerholiday),
path('holidaytabel/<int:createdBy>/<str:type>', views.getHolidayTabelByEmployeeId),
path('holidaytabelByID/<int:id>', views.getHolidayTabelByTabelId),
path('statusByHolidayTabelId/<int:id>', views.getStatusOfHolidayTabel),
#act
path('act/', views.act),
path('actflag/<int:id>', views.conformerActFlagEdit),
path('deleteact/<int:id>', views.deleteAct),
path('conformerAct/', views.conformerAct),
path('freelancer/', views.freelancer),
path('actFreelancer/', views.actFreelancer),
path('act/<int:createdBy>/<str:type>', views.getActByEmployeeId),
path('act/<int:id>', views.getActDetail),
path('actfreelancer/<int:id>', views.getActsFreelancer),
path('freelancer/<int:id>', views.getFreelancerById),
path('conformersAct/<int:id>',views.getActConformers),
path('actById/<int:id>', views.editAct),]
```

Форматы	Зонасы	Поз-сы.	Белгілеулер	Аталулары	Саны	Қорытынды

					Дипломдық жоба	Бет
						51
Өлш	Бет	Құжат №	қолы	күні		