ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
Қ.И. Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті
Автоматика және ақпараттық технологиялар институты
Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

Қасымбаева Гүлдана Бауыржанқызы

Туристік фирманың ақпараттық жүйесін құру

ДИПЛОМДЫҚ ЖОБА

5В070300 – «Ақпараттық жүйелер» мамандығы

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.И. Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
КАӨжС кафедрасы меңгерушісі
техн. ғыл. канд., ассоц.
профессор
_____Р.Ж.Сатыбалдиева
«____»_____2022 ж.

ДИПЛОМДЫҚ ЖОБА

Тақырыбы: «Туристік фирманың ақпараттық жүйесін құру»

5В070300 – «Ақпараттық жүйелер» мамандығы

Орындаған Қасымбаева Гүлдана Бауыржанқызы

Рецензент
техн.ғыл.канд., ассоц. проф., доцент
_____Балгабаева Л.Ш.

«____»_____2022 ж.

Ғылыми жетекші
техн.ғыл.канд., сениор-лектор
_____Байматаева Ш.М.

«10» мамыр 2022 ж.

Алматы 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.И. Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Автоматика және ақпараттық технологиялар институты

Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

5В070300 – Ақпараттық жүйелер

БЕКІТЕМІН
КАӨжС кафедрасы меңгерушісі
техн. ғыл. канд., ассоц.
профессор
_____Р.Ж.Сатыбалдиева
«_19_»_____05_____2022 ж.

Дипломдық жобаны орындауға
ТАПСЫРМА

Білім алушы Қасымбаева Гүлдана Бауыржанқызы
Тақырыбы: «Туристік фирманың ақпараттық жүйесін құру»
Университет Ректорының 2021 жылғы «24» желтоқсан №489П/Ө бұйрығымен бекітілген.
Орындалған жұмыстың өткізу мерзім «26» ___мамыр___2022 ж.

- Дипломдық жұмыстың бастапқы мәліметтері: туристік компания үшін автоматтандырылған ақпараттық жүйені жасау және бағдарламалық қамтамасыздандыру.

Дипломдық жұмыста қарастырылатын мәселелер тізімі:

1. Жобаға жалпы шолу;
2. Техникалық тапсырманы әзірлеу;
3. Дизайн үшін модельдер жасау;
4. Бағдарламаны іске асыру;
5. Әзірленген ақпараттық жүйенің фукционалдығын тексеру;

Графикалық материалдардың тізімі (міндетті түрде қажет сызбалар көрсетілген): жұмыстың 20 слайдтан тұратын презентациясы көрсетіледі.
Ұсынылған негізгі әдебиет 20 кітаптан тұрады.

Дипломдық жұмысты даярлау

КЕСТЕСІ

Бөлім атауы, қарастырылатын мәселелер тізімі	Ғылыми жетекші мен кеңесшілерге көрсету мерзімі	Ескерту
1. Тақырыпқа байланысты арнайы әдебиеттерді қарастыру	12.01.2022	
2. Дипломдық жобаның жоспарын құру	17.01.2022	
3. Негізгі бөлімді қарастыру	24.01.2022	
4. Бағдарламалық жасақтаманы іске асыру	27.02.2022	
5. Дипломдық жобаны қорытындылау	16.04.2022	

Дипломдық жұмыс бөлімдерінің кеңесшілері мен норма бақылаушының аяқталған жұмысқа қойған

қолтаңбалары

Бөлімдердің атауы	Кеңесшілер (аты-жөні, тегі, ғылыми дәрежесі, атағы)	Қол қойылған мерзімі	Қолы
Негізгі бөлім	Байматаева Ш.М., (техн.ғыл.канд., сеньор-лектор)	10.05.2022	
Норма бақылаушы	Аристомбаева М.Т., (техн.ғыл.магистрі, лектор)	17.05.2022	

Ғылыми жетекшісі_____Байматаева Ш.М.

Тапсырманы орындауға қабылдаған білім алушы_____ Қасымбаева Г.Б.

Күні «25» 12 2021 ж.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Сәтбаев университеті

Рецензент пікірі

Дипломдық жоба

Қасымбаева Гүлдана Бауыржанқызы
5В070300 – Ақпараттық жүйелер

Тақырыбы: Туристік фирманың ақпараттық жүйесін құру

Бұл дипломдық жоба өзінің логикалық құрылымымен ерекшеленген. Түсіндірме жобаның құрамы кіріспеден, 3 бөлімнен, қорытындыдан, әдебиеттер тізімінен құрылған.

Дипломдық жобаның мақсаты туристік компания үшін автоматтандырылған ақпараттық жүйені жасау және бағдарламалық қамтамасыздандыру.

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.

Қасымбаева Гүлдана дипломдық жобаны орындау барысында ізденіс жасап, әдебиеттермен жұмыс істеген. Дипломдық жоба 5В070300 мамандығының бітіру жұмыстарына қойылатын талаптарға сәйкес.

Қорытындылай келе, Қасымбаева Гүлдананың дипломдық жобасына жақсы деген баға беруге және оның орындаушысына 5В070300 – «Ақпараттық жүйелер» мамандығы бойынша «бакалавр – техник» біліктілігін беруге болады деп есептеймін.

Рецензент
тех. ғыл. кандидаты, "Ақпараттық жүйелер
және киберқауіпсіздік" кафедрасының оқытушысы,
Алматы техника университеті
_____Балғабаева Л. Ш.
«____»_____2022 ж.

АТУ
2022 ж./г.

Университеттің жүйе администраторы мен Академиялық мәселелер департаменті директорының ұқсастық есебіне талдау хаттамасы

Академиялық жетекшінің ұқсастық туралы есебі

Жүйе администраторы мен Академиялық мәселелер департаментінің директоры көрсетілген еңбекке қатысты дайындалған Плагиаттың алдын алу және анықтау жүйесінің толық ұқсастық есебімен танысқанын мәлімдейді:

Авторы: Қасымбаева Гүлдана

Тақырыбы: Туристік фирманың ақпараттық жүйесін құру

Жетекшісі: Байматаева Шолпан

1-ұқсастық коэффициенті: 5,46%

2-ұқсастық коэффициенті: 2,20%

Әріпті ауыстыру: 15

Интервалдар: 0

Микрокеңістіктер: 32

Ақ белгілер: 0

Есепті талдағаннан кейін:

- ☐ Жұмыста анықталған ұқсас алынған сөздерде плагиат белгілері жоқ. Осыған байланысты бұл жұмысты өз бетінше орындаған және оны қорғауға жіберемін.
- ☐ Жұмыста анықталған ұқсас сөздерде плагиат белгілері жоқ, бірақ олардың ұқсастығының шамадан тыс саны жұмыстың құндылығына және оның авторының дербестігінің болмауына қатысты күмән тудырады. Осыған байланысты жұмыс қайта редакциялансын.
- ☐ Жұмыста көшіріп алу және плагиат белгілері бар жұмысты жасыру әрекеттерін көрсете отырып, плагиаттық белгілерге немесе мәтінді қасақана бұрмалауға әкелетін әрекеттер бар. Осыған байланысты, жұмыс қорғауға жіберілмейді.

Негіздемесі:

Күні:

техн.ғыл.маг., сеньор-лектор

_____Байматаева Ш.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
Сәтбаев университеті

ҒЫЛЫМИ ЖЕТЕКШІНІҢ СЫН-ПІКІРІ

Қасымбаева Гүлдана Бауыржанқызы

Дипломдық жұмыстың тақырыбы: «Туристік фирманың ақпараттық жүйесін құру»

Дипломдық жұмыстың мақсаты туристік фирманың ақпараттық жүйесін құру болып табылады.

Дипломдық жұмыс кіріспеден, үш бөлімнен, қорытындыдан, әдебиеттер тізімінен және қосымшадан тұрады.

Дипломдық жұмысты жасау барысында пәндік аймақ бойынша шолу жасалды; жүйенің бизнес процестерінің үлгісі жасалды; туристік фирманың мәліметтер базасы құрылды; жасалған мәліметтер базасы негізінде ақпараттық жүйе құрылды.

Сондай ақ, UML тілін қолданып келесілер жасалды:

- прецеденттер диаграммасы;
- BPMN диаграммалары;
- жүйенің интерфейсі;
- бағдарламалық қамтама.

Жүйе интерфейсін жасауға Figma құралдары, деректер базасын жобалау үшін реляциялық PostgreSQL қолданылды және JAVA платформасы негізінде веб-қосымша іске асырылды.

Дипломдық жұмысты жасау барысынды Қасымбаева Г. жақсы теориялық дайындық және ақпараттық жүйелерді жасай алатын маман ретінде көрсетті. Нәтижесінде туристік фирманың жұмыс барысы туралы деректерді жинау және сақтау бойынша бизнес-процестерді автоматтандыруға арналған программалық қамтама құрылды. Жасалған программалық қамтама туристік фирмамен қолданылуы мүмкін.

Жоғарыда айтылғандарды ескеріп Қасымбаева Гүлдана Бауыржанқызының «Туристік фирманың ақпараттық жүйесін құру» тақырыбындағы жұмысы қорғауға жіберілуі мүмкін.

Ғылыми жетекші,
сеньор-лектор, т.ғ.к. Байматаева Ш.М.

Университеттің жүйе администраторы мен Академиялық мәселелер департаменті директорының ұқсастық есебіне талдау хаттамасы

Бөлім меңгерушісінің/ құрылымдық бөлімшесінің басшысының ұқсастығы туралы есебі

Жүйе администраторы мен Академиялық мәселелер департаментінің директоры көрсетілген еңбекке қатысты дайындалған Плагиаттың алдын алу және анықтау жүйесінің толық ұқсастық есебімен танысқанын мәлімдейді:

Авторы: Қасымбаева Гүлдана

Тақырыбы: Туристік фирманың ақпараттық жүйесін құру

Жетекшісі: Байматаева Шолпан

1-ұқсастық коэффициенті: 5,46%

2-ұқсастық коэффициенті: 2,20%

Әріпті ауыстыру: 15

Интервалдар: 0

Микрокеңістіктер: 32

Ақ белгілер: 0

Есепті талдағаннан кейін:

- ☐ Жұмыста анықталған ұқсас алынған сөздерде плагиат белгілері жоқ. Осыған байланысты бұл жұмысты өз бетінше орындаған және оны қорғауға жіберемін.
- ☐ Жұмыста анықталған ұқсас сөздерде плагиат белгілері жоқ, бірақ олардың ұқсастығының шамадан тыс саны жұмыстың құндылығына және оның авторының дербестігінің болмауына қатысты күмән тудырады. Осыған байланысты жұмыс қайта редакциялансын.
- ☐ Жұмыста көшіріп алу және плагиат белгілері бар жұмысты жасыру әрекеттерін көрсете отырып, плагиаттық белгілерге немесе мәтінді қасақана бұрмалауға әкелетін әрекеттер бар. Осыған байланысты, жұмыс қорғауға жіберілмейді.

Негіздемесі:

Күні:

КАӨжС кафедрасы меңгерушісі
т.ғ.к., ассоц. профессор
_____Р.Ж.Сатыбалдиева

АҢДАТПА

Бұл жоба туристік фирманың жұмысын басқару арналған бағдарламалық қамтамасыз етуді дамытуға арналған. Ұсынылған шешім тез саяхаттаушы мен турагенттердің арасындағы байланысты жеңілдетуге, турды таңдау және жаңарту процесстерін жылдамдатуға мүмкіндік береді. Саяхатшы үшін турфирма кеңсесіне келместен, қашықтан таңдау мүмкіндіктері ұйымдастырылады.

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

АННОТАЦИЯ

Данный проект посвящен разработке программного обеспечения для управления работой туристической фирмы. Предлагаемое решение позволяет упростить связь между быстро путешествующим и турагентами, ускорить процесс выбора и обновления тура. Для путешественника организуются возможности удаленного выбора без посещения офиса турфирмы.

В данном дипломном проекте рассматриваются вопросы автоматизации бизнес-процессов сбора и хранения данных о ходе работы туристской фирмы. Для визуального отображения роли и возможностей пользователей используются диаграммы Use Case, BPMN. Для создания интерфейса системы используются инструменты Figma. Для проектирования базы данных используется реляционный PostgreSQL, реализовано веб-приложение на базе платформы JAVA.

ANNOTATION

This project is dedicated to the development of software for managing the work of a travel company. The proposed solution makes it possible to simplify communication between a fast traveler and travel agents, speed up the process of choosing and updating a tour. For the traveler, remote selection opportunities are organized without visiting the office of a travel agency.

This diploma project deals with the automation of business processes for collecting and storing data on the progress of the work of a travel company. Use Case and BPMN diagrams are used to visually display the role and capabilities of users. Figma tools are used to create the system interface. Relational PostgreSQL is used for database design, a web application based on the JAVA platform is implemented.

МАЗМҰНЫ

КІРІСПЕ

Экономикадағы нарықтық қатынастардың дамуы және ғылыми-техникалық прогресс қоғамның әлеуметтік - экономикалық өмірінің барлық салаларына ақпараттық технологиялар саласындағы соңғы жетістіктерді енгізу қарқынын жеделдетті. Соңғы жылдары өндіріс пен басқарудың барлық салаларын ақпараттандырудың өсу қарқыны дамудың шешуші факторына айналып, тауарлар мен қызметтерді өндіру процесімен тығыз байланысты болып, практикалық маңыздылығын едәуір арттырды.

Қазіргі әлемдегі ақпарат ең маңызды ресурстардың біріне айналды, ал ақпараттық жүйелер адам қызметінің барлық салаларында қажетті құралға айналды. Ақпараттық жүйе іс жүзінде кез келген ұйымның жұмыс істеуінің ажырамас бөлігіне айналды, сондықтан ақпараттық жүйелерді әзірлеу мен енгізудің өзектілігі мәселесін талқылаудың қажеті жоқ. Алайда, оны жобалау тәсілдерінің жүйелілігі және әзірлеу сапасы мәселесі әлі де өзекті болып табылады [1].

Туризм - көптеген басқа салалармен тығыз әрекеттесетін және олардың дамуына ықпал ететін сала. Қызмет көрсету сапасын арттыру қызметті автоматтандырудың заманауи жүйелерін енгізбей жұмыс істеуі мүмкін емес [2].

Қазір туристік компанияның тиімді жұмысын жеке веб-сайтсыз елестету қиын. Туристік агенттік үшін бұл тұтынушыларға жылдам жету жолы. Туристік агенттік веб-сайтына қойылатын басты талап – тиімділік. Туристік агенттік сайтында, ең алдымен, нақты ұсыныстар туралы ақпарат қажет.

Веб-сайтты әзірлеу - бұл нақты әрекеттер тізбегі. Қазіргі уақытта веб-сайт жасаудың бірнеше жолы бар. Сондай-ақ сайттарды әзірлеу және дамытумен айналысатын көптеген үшінші тарап компаниялары бар. Веб-сайтты әзірлеу және дамыту - бұл бөлек қарастырылатын әртүрлі ұғымдар.

Жұмыстың өзектілігі - тұтынушыға сапалы туристік өнімді сәтті жүзеге асыруды анықтайтын туристік агенттіктің клиенттерімен жұмыс істеудің автоматтандырылған жүйесін жасау.

Дипломдық жұмыстың зерттеу объектісі туристік агенттік болып табылады.

Дипломдық жұмыстың мақсаты туристік компания үшін автоматтандырылған ақпараттық жүйені жасау және бағдарламалық қамтамасыздандыру, сонымен қатар пәнді оқу кезінде алған білімдерін бекіту және заманауи технологиялар мен құралдарды қолдана отырып ақпараттық жүйелерді жобалаудың практикалық дағдыларын алу болып табылады.

Жобаның жоспары:

1) осы тақырыпқа байланысты арнайы әдебиеттерді қарастыру;

2) техникалық тапсырманы әзірлеу;

3) дизайн үшін модельдер жасау;

4) бағдарламаны іске асыру;

5) әзірленген ақпараттық жүйенің функционалын тексеру.

Автоматтандырылған ақпараттық жүйелерді әзірлеу және өндіріске енгізу барлық жұмыстардың орындалуының жеделдігі мен дұрыстығын қамтамасыз ете отырып, туристік агенттік қызметкерлерінің еңбек тиімділігін арттырады.

Бірінші бөлім пәндік аймақты талдауға және осы жүйені модельдеуге арналған.

Екінші бөлімде техникалық тапсырманы, пайдалану моделін, мәліметтер базасының және бағдарламаны іске асыратын құралдар сипатталған.

Үшінші бөлімде автоматтандырылған ақпараттық жүйені бағдарламалық іске асыру жүргізілді.

1 Пәндік саланы талдау

1.1 Пәндік саланың сипаттамасы

Туристік компания халыққа туристік қызмет көрсетумен айналысады. Туристік компанияның негізгі мақсаты туроператорлар ұсынатын туристік өнімді жүзеге асыру және сату болып табылады. Бүгінде мұндай кәсіпорындар жер шарының түкпір-түкпіріне туристік саяхатты қамтамасыз ететін көптеген туроператорлармен ынтымақтасады.

Туристік компания ұсынатын турлар, демалыс орнына қарамастан, ерекшеленеді:

а) турдың мақсатына қарай:

– жағажай демалысы:

– тау шаңғысы курорты;

– экскурсиялық турлар;

– емдік демалыс;

– іскерлік сапар;

– аралас турлар және т.б.

б) клиенттердің талаптары бойынша:

– клиент туроператордың ұсынысында ештеңені өзгертпей, стандартты турда демалуға барады;

– жеке – жеке саяхат жоспарын құра отырып, клиенттің талаптарын ескере отырып, турды таңдау;

– клиент басқа туристер тобымен бірігіп топтық тур таңдау (көбінесе бұл экскурсиялық турлар).

Компанияның қызмет аясына мыналар кіреді:

– тур таңдау кезінде тұтынушыларға туристік қызмет көрсету;

– клиенттердің дербес деректерін есепке алу және сақтау: тегі, аты, әкесінің аты, мекенжайы, телефон нөмірі, төлқұжат деректері;

– таңдалған турларды тіркеу, соның ішінде клиенттермен келісім-шарттар жасау;

– клиенттерге әуе және темір жол билеттерін сатып алуға көмектесу;

– турларды жүзеге асыру үшін бухгалтерлік есеп пен кассалық есеп беру;

– есепті кезеңге қаржылық есеп беру құжаттарын жасау;

– саяхатқа, әуе билеттерін сатып алуға, визалар беруге және медициналық сақтандыруға қатысты барлық мәселелер бойынша білікті кеңестер беру, соның арқасында қазіргі заманғы нарық талаптарына барынша сәйкес келетін демалыс бағдарламаларын әзірлеуге болады;

– туроператорлармен жұмыс;

– әкімшілік функцияларды орындау.

Туристік агенттік ұсынатын туристік қызметтер туралы мәліметтерді аналитикалық талдау процесінде басымдықтары анықталды және пайыздық мәнде анықталды:

– туристік агенттік клиенттерінің демалыс орындары;

– клиенттер үшін маусымдық демалыс орындары;

– клиенттердің жас санаттары;

– туристік өнімді тұтынушылардың құрылымы;

– клиенттердің шетелге саяхатының мақсаттары.

Туристік агенттік клиенттерінің шетелдегі негізгі мақсаттары демалыс, емделу, экскурсиялар, іссапарлар және басқа да мақсаттар болып табылады. Сонымен қатар, сапардың басты мақсаты демалу болып табылады. Туристік қызмет пакетінің құрамы бойынша тұтынушылар әдеттегі турпакетті қалайды. Туристік фирмада турларды ұйымдастыруды туроператорлар мен туристік фирманың менеджерлері жүзеге асырады [1].

Туроператордың негізгі міндеттері:

– туристік агенттік сайтында турлар туралы ақпаратты орналастыру: ел, қала, турдың басталу және аяқталу мерзімі, қонақүйлер, экскурсиялар және т.б.;

– клиенттерге олар сатып алған турларды жүзеге асыру бойынша қызмет көрсету: әуежайда қарсы алу, гидпен қамтамасыз ету және тұру үшін қонақүйден орын беру.

Менеджердің негізгі міндеттері:

– тур таңдау туралы сұрақтар бойынша клиенттерге кеңес беру;

– турдың дизайны.

Менеджер туристердің қалауы бойынша реттелуі мүмкін турлардың, тұрмыстық жағдайлардың, мәдени бағдарламалардың бағалары көрсетілген туроператорлардың веб-сайттарында орналасқан ақпарат негізінде клиенттерге кеңес береді. Турды сатып алу кезінде ресімделетін құжаттар оның түріне, клиенттердің санатына (ересектер, балалар) және тапсырыс түріне (топтық немесе жеке тур) байланысты емес. Қазақстан бойынша және кіру кезінде виза талап етілмейтін басқа елдерде турды тіркеу мыналарды қамтиды:

– ақылы туристік қызметтерді көрсетуге шарт жасасу;

– клиентке туристік жадынама беру;

– тур клиентінің төлемі.

Ақылы туристік қызметтерді көрсету шарты мынадай міндетті тармақтарды қамтиды:

– таңдалған турдың параметрлері, туристік өнімнің жалпы құны және төлем тәртібі, төлқұжат деректері және сенім білдірушінің байланыстары көрсетілген шарттың мәні. (тур тұтынушы);

– агенттің (туристік агенттік) құқықтары, міндеттері мен жауапкершілігі;

– сенім білдірушінің құқықтары, міндеттері және жауапкершілігі;

– туроператордың құқықтары, міндеттері мен жауапкершілігі және оның қаржылық кепілдіктері;

– ерекше шарттар: мысалы, рейстердің жөнелтілетін уақытын өзгерту кезінде авиатасымалдаушы жауапты болады.

– туристік өнімді сату шарты бойынша міндеттемелерді орындамағаны үшін;

Шарт екі данада жасалады, оған бір жағынан клиент, екінші жағынан туристік компанияның өкілі қол қояды. Қолдар мөрмен бекітіледі. Шарттың бір данасы компанияда қалады, екіншісі клиентке беріледі. Егер шартта клиенттің сапардан қайтып оралу фактісі бойынша шартқа бекітілетін клиенттің нақты келген күні көрсетілсе, шарт орындалды деп есептеледі. Туристке арналған жадынамада тур туралы көрсетілетін мәліметтер:

– демалыс орны, саяхаттың ұзақтығы, қонақ үй және т.б.

– тамақтану, экскурсиялар, қосымша ақылы қызметтер;

– тұтынушының төлқұжат деректері;

– турист талап ететін қосымша деректер:

– турдың басталу және аяқталу күні;

– қонақүйдің атауы;

– қонақ үй санаты, тамақ санаты;

– қонақ үйге ауыстыру;

– гид қызметтерінің тізбесі;

– турист ұшуға мәжбүр болған жағдайда әуежайда туроператордың өкілін қалай табуға болатыны туралы ақпарат.

Турдың құнын туроператор анықтайды, ол туристік компанияға шот-фактура жазып, оны факс немесе электронды пошта арқылы жібереді. Тур төлемінің сомасына турдың өзіндік құны және субагенттік комиссия (туристік компанияның қызметтері үшін төлем) кіреді (әдетте турдың жалпы құнының белгілі бір пайызы). Шот-фактурада көрсетілген сома шартта көрсетілген.

Клиенттің тур ақысын төлеуі бірнеше жолмен жүзеге асырылуы мүмкін:

– қолма-қол төлем: клиент шартқа қол қойған күні туристік компанияда толық төлейді;

– аванстық төлем: клиент тур құнының 50% төлейді, ал қалған соманы үш банктік жұмыс күні ішінде қосымша төлеу керек;

– турды несиеге сатып алу: шартта турдың құны, клиент төлейтін сома және несиеге алған сомасы көрсетіледі. Турды несиеге сатып алған жағдайда, тур үшін төлем сомасы үш күн ішінде туристік компанияның есеп айырысу шотына несие берілген банктен субагенттік комиссияның сомасын шегеріп түсуі тиіс. туроператордың шотына аударылады. Бұл жағдайда турдың ақысын төлеу негізі банктен факс арқылы жіберілетін төлем тапсырмасы болып табылады. Төлем тапсырмасында туристік компанияның шотына аударылған сома, ақша аударылған күні көрсетіледі. Кәсіпорынның кассасы арқылы турдың төлемін растайтын құжаттар (қолма-қол ақшамен) туристік қызмет көрсету бойынша ақылы қызмет көрсету шартымен расталған чек және кредиттік жазба

болып табылады. Клиенттің туристік сапарға төлегені туралы ақпарат кассаға жазылады.

Тек визамен кіруге рұқсат етілетін елдерге тур жасағанда (Шенген елдері, АҚШ және т.б.) туристер сәйкесінше виза алуы керек. Виза құны турдың жалпы құнына кіреді.

Визаға өтініш беру үшін клиент келесі құжаттарды ұсынуы керек: 2 түрлі-түсті фотосурет, жұмыс немесе оқу орнынан анықтама, жұмыс орнынан табыс туралы анықтама немесе банктен үзінді көшірме, төлқұжат. Клиент тур басталғанға дейін кемінде екі апта бұрын туристік компанияға виза алу үшін құжаттарды тапсыруы керек. Клиенттің визаны ресімдеуге ұсынылған құжаттарын туристік агенттік қызметкері турды ұсынатын туроператорға курьерлік пошта арқылы жібереді. Құжаттарды қарау нәтижесін туристік агенттік қызметкері туроператордан жауап алғаннан кейін клиентке хабарлайды. Виза алу үшін құжаттарды қарау мерзімі, әдетте, 10 күн. Визадан бас тартқан жағдайда, клиент тур үшін төленген барлық соманы жоғалтады. Тур үшін соманы қайтару сақтандыру компаниясымен күйіп кетпеуден сақтандыру шартын жасаған жағдайда мүмкін болады. Бұл келісімді жасау, ең алдымен, Шенген елдеріне барған кезде ұсынылады. Бұл жағдайда виза беруден бас тартқан жағдайда сақтандыру компаниясы клиентке тур төлемінің сомасын қайтарады. Туристік компания әуе және теміржол билеттерін сатып алу қызметтерін де көрсетеді. Бұл қызметті жүзеге асыру туристік фирманың көптеген көлік компанияларымен жасасқан шарттық міндеттемелері негізінде жүзеге асырылады. Әуе және теміржол билеттерін сатып алу үшін көлік түрін, сапар күндерін, туристердің төлқұжат деректерін көрсете отырып, тиісті көлік компаниясына өтініш жіберіледі. Билеттер туристік компанияның кеңсесіне жеткізіледі және менеджерлер төлемге байланысты клиенттерге береді.

Туристік компанияның табысы тікелей тұтынушылардың қанағаттануына байланысты. Сондықтан компанияның имиджін сақтау және тұтынушыларды ұстап тұру үшін туристік агенттік менеджерлері әртүрлі акциялар ұйымдастырады, мысалы, оларды туған күнімен, Жаңа жылмен және басқа мерекелермен құттықтайды, тұрақты клиенттерге жеңілдіктер жасайды және т.б. Сыртқы қаржылық есептер қала әкімшілігі мен туроператорларға жіберіледі. Бұл есептер саяхат орны мен субагенттік сыйақы сомасын көрсете отырып, белгілі бір уақыт кезеңі ішінде туристік қызмет көрсету бойынша ақылы қызметтерді көрсетуге жасалған шарттардың саны туралы мәліметтерді көрсетеді.

1.2 Ақпараттық жүйені жобалау

Қазіргі жағдайда қоғамның дамуын Интернет өркениет игілігін пайдаланбай елестету қиын. Қазір әрбір ұйымның, клубтың немесе кез келген адамның жеке веб-сайты бар.

Бұл дипломдық жұмыстағы туристік компания веб сайты әртүрлі елдердегі компания ұсынатын курорттарда белгілі бір уақыт аралығында клиенттерге демалыс қызметтерін ұсынады. Әлемнің әртүрлі елдеріндегі ең танымал курорттарда клиенттерге өз қызметтерін ұсынатын типтік туристік компанияны қарастыру ұсынылады.

Клиентке қызмет көрсету оны тіркеуден басталады. Тіркеу кезінде клиент қажетті құжаттарды ұсынады, демалыс кезеңі келісіледі, кейбір анкета деректері толтырылады. Клиент демалыс орнын таңдаған кезде оған компания ұсынған жолдамалар және олардың құны туралы тиісті ақпарат беріледі. Демалыс орны мен жолдаманы таңдағаннан кейін клиенттің демалыс кезеңі келісіледі [5].

Жоғарыда айтылғандардың бәрін таңдап, клиент билетті төлегеннен кейін, барлық ақпарат дерекқорға енгізіледі. Компания байланыста тасымалдаушы компаниялармен, сондай-ақ қонақ үй кешендерімен демалыс орындарында және тікелей клиенттің қалауымен орындайды, сондықтан клиенттің демалыс процесінде компания бақылайды барлық процесі клиенттің жүріп-тұру: клиенттің курортқа жеткізуі, клиенттің қонақ үйге және кері үйге жеткізілуін бақылайды [1].

Нәтижесінде туристік компанияда көптеген процестер жүретіні байқалады, олар қолмен көп уақытты алады. Бірақ қазіргі уақытта ақпараттық технологиялар әр түрлі деңгейдегі барлық ұйымдардың қызметінде кеңінен қолданылады. Тиісінше, туристік компанияның жұмысын автоматтандыруға болады [5].

Ұтымды басқарудың мақсаты қолайлы турды таңдаудан бастап құжаттарды рәсімдеуге дейін, сондай-ақ қызметкерлердің қателіктерін азайту арқылы барлық кезеңдерде қызмет көрсету мерзімдерін қысқарту арқылы клиенттерге қызмет көрсету сапасын арттыру болып табылады. Мұның бәрі, сайып келгенде, кәсіпорынның кірісінің өсуіне әкелуі керек. "Туристік фирма" ақпараттық жүйесі орындалатын жұмысты реттеуге мүмкіндік береді.

1.3 Туристік фирмаға арналған автоматтандырылған ақпараттық жүйесін әзірлеуге арналған техникалық тапсырма

Бұған дейін туристік фирманы ұтымды және өнімді басқару үшін оны автоматтандыру қажет екендігі анықталды. Осы тақырып бойынша негізгі процестерді анықтағаннан кейін техникалық тапсырманы әзірлеу қажет [9].

Бұл техникалық тапсырма туристік фирма саласында қызмет көрсетуді ұйымдастыру үшін қолданылатын бағдарламалық өнімді әзірлеуге қолданылады. Бұл бағдарламалық жасақтаманың негізгі міндеті - туристік фирманы автоматтандыру және басқару.

Бұл бағдарламалық өнім дипломдық жұмысқа тапсырма негізінде әзірленеді.

Функционалдық сипаттамаларға қойылатын талаптар. Бағдарламалық жасақтама келесі функцияларды орындауы керек:

– клиенттерді тіркеу;

– туристік жолдамаларды тіркеу;
– клиенттің турды таңдауы;
– шарт жасасу.

Жүйенің кіріс ақпараты:
– туристік фирмадағы турлар туралы ақпарат;
– туристік фирманың қызметкерлері туралы ақпарат;
– туристік компанияның клиенттері және олардың тапсырыстары туралы ақпарат.

Жүйенің шығыс ақпараты:
– клиенттің таңдаған турды төлеуі туралы ақпарат

Сенімділік талаптары. Әзірленген жүйе сенімділік талаптарына сәйкес келуі керек тұрақты жұмыс істеуін қамтамасыз ету:
– қателердің болмауы;
– мүмкін қателіктерге төзімділік;
– енгізілген ақпаратты автосақтау;
– деректердің тұтастығын қамтамасыз ету.

Пайдалану шарттары. Жүйені орташа және төмен білікті пайдаланушылар пайдаланады. Жүйенің интерфейсі ұқсас жүйелердің интерфейстеріне мүмкіндігінше жақын болуы керек. Ақпаратты енгізу неғұрлым біріздендірілген нысандарда жүзеге асырылуы тиіс [3].

Әзірленген бағдарламалық өнім келесі жағдайларда пайдаланылуы керек:
– операциялық жүйе мен жабдықтың ақаусыздығы мен жұмысқа қабілеттілігін тұрақты тексеру, қайта орнату және қажеттілігіне қарай ауыстыру;
– жұмыс күнінің соңында қателер мен ақаулардың бар-жоғын тексеру;
– жабдықты тазалау;
– бөлмедегі ауа температурасы +20 ... +25°С.

Техникалық құралдардың құрамы мен параметрлеріне қойылатын талаптар:

Бұл жүйе кез – келген компьютерлерде жұмыс істеуі керек. Әр компьютердегі жедел жад, кем дегенде 1 ГБ болуы шарт.

2 Арнайы бөлім

2.1 Жүйенің бизнес-процестерінің моделін жасау

Бизнес-процестің контекстік диаграммасын құру үшін келесі тапсырмаларды орындау қажет:

– қажетті сыртқы объектілерді, жалпыланған процестің, ағындардың атын анықтау;

– элементтердің бір-бірімен қалай әрекеттесетінін анықтау;

– осы элементтерді және олардың өзара байланысын көрсететін диаграмма құру.

Пәндік аймақты талдау негізінде жүйенің бизнес-процес моделінің контекстік диаграммасы әзірленді [7].

Деректер моделін концептуалды жобалау процесінде деректер құрылымдары, әрбір деректер құрылымы тағайындалады нысан және нысан атрибуттары анықталған, әрқайсысы үшін бастапқы кілттерсубъектілер және субъектілер арасындағы қатынастар анықталады [6].

Концептуалды жобалау нәтижесінде келесі деректер құрылымы анықталды:

– «Тапсырыс беруші»;

– «Турдың санаты»;

– «Тур»;

– «Тур түрі»;

– «Турдағы қызметтер»;

– «Қызмет түрі»;

– «Туроператор»;

– «Бағыт».

2.2 Қолданылатын бағдарламалар кешені

Туристік фирманың веб сайтын жасау үшін Java программалау кешені, Spring фреймворкі пайдаланылды. Деректер қорын құруға PostgreSQL-ді пайдаланылды, ал сайт дизайні Figma-да дайындалды.

Java – параллель, объектіге бағытталған жалпы мақсаттағы бағдарламалау тілі. Java қосымшаларды жасаушыларға «бір рет жазу, барлық жерде іске қосу» мүмкіндігін береді. Бұл құрастырылған Java кодын Java-ны қолдайтын барлық платформаларда қайта құрастырудың қажеті жоқ, іске қоса беруге болатындығын білдіреді. Java-дағы қосымшалар, әдетте, кез – келген Java виртуалды машинасында (JVM-ағылш. Java Virtual Machine) компьютерлік архитектурадан тәуелсіз. 2015 жылдан бастап Java ең танымал бағдарламалау тілдерінің бірі болып табылады, әсіресе клиент – веб-қосымшаларды әзірлеу саласында.

Java тілі бастапқыда Sun Microsystems-те (оны Oracle корпорациясы 2009 жылдың сәуірінен 2010 жылдың қаңтарына дейін сіңірген) Джеймс

Гослинг жасаған және 1995 жылы Java бағдарламалық платформасының өзегі ретінде жарияланған. Java синтаксисі көбінесе C және C++ тілдерінен алынады, бірақ олардың кез-келгенімен салыстырғанда тілдің төмен деңгейлі мүмкіндіктері айтарлықтай шектеулі. Java компиляторларын, виртуалды машиналарды және сынып кітапханаларын түпнұсқа және анықтамалық іске асыру бастапқыда Sun компаниясы меншікті лицензиялармен шығарылды. 2007 жылғы жағдай бойынша, Java Community Process спецификацияларына сәйкес, Sun өзінің Java-технологияларының көпшілігінің лицензиясын GNU ашық лицензиялық келісіміне (ағылш. GNU General Public License). Сондай-ақ, Java-ға арналған GNU Compiler (GCJ), GNU Classpath (Java тілі сыныптарының стандартты кітапханасын тегін енгізу) және IcedTee-Web сияқты үшінші тарап технологиялары бар. Соңғы нұсқасы-Java 8-Қазіргі уақытта Oracle тегін қолдайтын жалғыз нұсқа. Бұрынғы нұсқаларды Oracle және басқа компаниялар коммерциялық негізде қолдайды.

Java тілі құрылған бес негізгі қағида:

1. Тіл "қарапайым, объектіге бағытталған және таныс" болуы керек.
2. Тіл "сенімді және қауіпсіз" болуы керек.
3. Тіл "архитектурадан тәуелсіз және мобильді" болуы керек.
4. Тілді "жоғары тиімділікпен" орындау керек.
5. Тіл "түсіндірілетін, динамикалық және ағындарды қолдауы керек".

Java құрылған принциптердің бірі - ұтқырлық, яғни Java платформасы үшін жазылған бағдарламалар барлық жерде бірдей жұмыс істеуі керек, яғни оларды орындау нәтижесі аппараттық және операциялық жүйеге тәуелді болмауы керек. Бұл бастапқы кодты архитектураға тәуелді машина кодына емес, байт коды деп аталатын аралық көрініске аудару арқылы жүзеге асырылады. Байт-код нұсқаулары машиналық тіл командаларына ұқсас, бірақ олар белгілі бір сәулет үшін жазылған виртуалды машинаны орындауға арналған. Веб-шолғышта жеке Java қосымшаларын немесе Java апплеттерін іске қосу үшін соңғы пайдаланушылар JVM — Java Runtime Environment (JRE) минималды іске асыруды қолданады, ол өз машиналарында орнатылады. Стандартты кітапханалар графика, орындау ағындары және желі сияқты хостқа тән функцияларға қол жеткізудің жалпы әдісін ұсынады. Әмбебап байт кодын пайдалану тасымалдауды айтарлықтай жеңілдетеді. Алайда, байт кодын машиналық нұсқауларға түсіндіруге байланысты үстеме шығындарға байланысты, интерпретацияланған бағдарламалар әрдайым машина тіліндегі орындалатын файлдарға қарағанда баяу жұмыс істейді. Осыған қарамастан, жұмыс уақытында байт кодын машина кодына жіберетін JIT компиляторлары дамудың алғашқы сатысында енгізілді. Java тілінің өзі платформаға тәуелді емес, оны белгілі бір платформаға бейімдеу осы платформаға жазылған Java виртуалды машинасы (JVM) арқылы жүзеге асырылады. JVM Java байт кодын платформаның машина тіліне аударады.

Java-ның кез келген қолдау көрсетілетін платформада өз кодын іске қосудың ерекше мүмкіндігі оның бағдарламаларын байт код деп аталатын

аралық көріністің қандай да бір түріне аудару арқылы қол жеткізіледі. Байт-код, өз кезегінде, Java орындалу ортасы бар кез келген жүйеде түсіндірілуі мүмкін. Платформадан тәуелсіз болуға тырысқан алғашқы жүйелердің көпшілігінде өнімділікті жоғалтудың үлкен кемшілігі болды (Basic, Perl).

Қазіргі уақытта Java SE платформасының ресми иске асырылуының иесі Oracle корпорациясы болып табылады. Бұл иске асыру Sun Microsystems жасаған түпнұсқаға негізделген. Ол Microsoft Windows үшін қол жетімді (Windows XP әлі де жұмыс істейді, бірақ тек Windows-тың соңғы нұсқасы ресми түрде қолдау көрсетеді), Mac OS X, Linux және Solaris. Java үшін Ecma International, ISO / IEC, ANSI немесе басқа ұқсас ұйым таныған ресми стандарттар жоқ болғандықтан, Oracle-ді иске асыру іс жүзінде стандарт болып табылады. Oracle Java-ны екі түрлі дистрибуция түрінде жүзеге асырады: Java Runtime Enviroment (JRE), ол соңғы пайдаланушыларға арналған және Java - да жазылған бағдарламаларды іске қосу үшін қажет Java SE платформасының бір бөлігін және бағдарламалық жасақтама жасаушыларға арналған және әзірлеу құралдарын қамтитын Java Development Kit (JDK) бөлігін қамтиды, Java компиляторы, Javadoc, Jar және түзеткіш сияқты.

Сондай-ақ, Java SE платформасының тағы бір іске асырылуын атап өткен жөн — OpenJDK, ол GNU GPL лицензиясы бойынша таратылады. OpenJDK жобасын жасау Sun GPL лицензиясы бойынша Java бастапқы кодын шығарған кезде басталды. Java SE 7 нұсқасынан бастап, OpenJDK ресми түрде Java-ның анықтамалық нұсқасы болып табылады.

Java-ның мақсаты - барлық іске асыруды үйлесімді ету. Тарихи тұрғыдан алғанда, Sun Microsystems лицензиясы Java брендін кез-келген іске асыруға пайдалану оның басқалармен үйлесімділігін білдіреді. Осыдан кейін Microsoft корпорациясымен Sun Microsoft жасаған Java іске асыру RMI немесе JNI қолдамайтыны және платформаға тәуелді ерекшеліктері бар екендігі белгілі болғаннан кейін құқықтық дау туындады. Sun компаниясы 1997 жылы сотқа жүгінді және 2001 жылы істі жеңіп алды, нәтижесінде 20 000 000 $ көлемінде өтемақы алды, сонымен қатар Micsoft компаниясы Sun Лицензиялық келісімінің шарттарын сақтауға міндеттеме алды. Нәтижесінде Micrsoft енді Java-ны Windows-пен бірге жеткізбеді.

Java платформаға тәуелділігі әсіресе Java EE үшін өте маңызды. Java EE ралізациясын сертификаттау үшін қатаң тексерулер қажет, өйткені бұл орта және ірі кәсіпорындардың міндеттері үшін серверлік платформа.

Java-да жазылған бағдарламалар ұқсас c++бағдарламаларына қарағанда баяу және қымбат жад ретінде танымал. Дегенмен, Java бағдарламаларын орындау жылдамдығы 1997/1998 жылы Java 1.1 нұсқасында JIT компиляциясын енгізумен, кодты тиімді талдауға мүмкіндік беретін жаңа тілдік мүмкіндіктерді қосумен (мысалы, ішкі сыныптарды, StringBuilder класын, шартты мәлімдемелерді (asssertions) және т. б.) және JVM оңтайландырумен (мысалы, тиімді виртуалды компиляция) едәуір өсті. машина — HotSpot-Java 1.3 шығысымен негізгі болды).

Кейбір платформалар Java-ны аппараттық деңгейде қолдайды. Бар микроконтроллерлер Java виртуалды машинасының бағдарламалық жасақтамасын қолданбай Java командаларын аппарт деңгейінде орындай алатын және Jazelle технологиясы арқылы Java байтекодының орындалуын қолдайтын ARM процессорлары (бұл технологияны қолдау ARM процессорларының соңғы енгізулерінде төмендеді).

Java-да объектілердің өмірлік циклінде жадты басқару үшін автоматты қоқыс жинағыш қолданылады. Бағдарламалаушы нысандардың қашан жасалатынын анықтайды, ал Java жұмыс уақыты енді қолданылмайтын Нысандар алатын жадты босатуға жауап береді. Яғни, қоқыс жинаушы енді сілтеме жоқ жад бөлімдерін автоматты түрде босатады. Алайда, егер бағдарламашы кодында енді қажет емес нысанға сілтеме болса, жадтың ағып кетуіне ұқсас нәрсе әлі де орын алуы мүмкін. Әдетте, бұл енді қажет емес заттар әлі де қолданылатын контейнерлерде сақталған кезде пайда болады. Егер жоқ объектінің әдісі шақырылса, онда тиісті ерекшелік қозғалады — NullPointerException.

Java жадын басқарудың автоматты моделінің идеяларының бірі-бағдарламашылар жадты қолмен басқару кезінде пайда болатын қиындықтардан құтыла алады. Кейбір тілдерде объектілерді құруға арналған жад дестеде айқын көрінеді немесе үйіндіде айқын көрінеді және босатылады. Соңғы жағдайда жадты басқару жауапкершілігі бағдарламашыға жүктеледі. Егер бағдарламашы нысанды жоймаса, жад ағып кетеді. Егер бағдарлама бұрын босатылған жад аймағына кіруге немесе босатуға тырысса, онда бұл мінез-құлықтың нәтижесі Анықталмайды, яғни Бағдарлама тұрақсыз жұмыс істей ме және/немесе апатты түрде аяқталатынын болжау қиын. Бұл мәселені ақылды көрсеткіштерді (ағылш. smart pointer), бірақ бұл тәсіл күрделілік пен үстеме шығындардың артуымен байланысты. Қоқыс жинау жадтың "логикалық" ағып кетуіне, яғни бұдан былай пайдаланылмайтын сілтемелер/көрсеткіштер қалатын жағдайларға кедергі келтірмейтінін ұмытпаңыз [19].

Қоқыс жинау кез келген уақытта болуы мүмкін. Ең дұрысы, бұл бағдарлама күту режимінде болған кезде болады. Қоқыс жинауға кепілдік беріледі, егер сіз үйіндіден жаңа зат үшін жад бөлуге тырыссаңыз, бос жад жеткіліксіз болады; бұл қоқыс жинаушы жұмыс істеп тұрған кезде бағдарламаның қатып қалуына әкелуі мүмкін. Java-да жадты нақты басқару мүмкіндігі жоқ.

Java C немесе C++ стиліндегі сілтегіштермен арифметикалық амалдарды қолдамайды, мұнда нысандардың мекен-жайларын қол қойылмаған сандармен ауыстыруға болады (әдетте long int сияқты). Бұл қоқыс жинаушыға сілтеме жасалған нысандарды жылжытуға және жалпы қауіпсіздік пен қауіпсіздікті қамтамасыз етуге мүмкіндік береді.

C ++ және басқа да объектіге бағытталған тілдер сияқты, Java-дағы қарабайыр типтегі айнымалылардың мәндері тікелей объектінің бөлігі ретінде (объект өрістері жағдайында) немесе дестеде (жергілікті айнымалылар жағдайында) сақталады, сонымен қатар жергілікті әдіс

нысандары әдетте дестеде сақталады (Escape-Analysis қараңыз). Бұл Java әзірлеушілерінің өнімділігін арттыру мақсатында қабылданған саналы шешімі.

Java-да қоқыс жинағыштардың бірнеше түрлері бар. Әдепкі бойынша, HotSpot виртуалды машинасы PS Scavenge қоқыс жинағышын пайдаланады. Дегенмен, үйінділерді басқару үшін қолдануға болатын басқа да заттар бар. Java Қосымшаларының 90% үшін Concurrent Mark-Sweep қоқыс жинаушысы жеткілікті. Oracle оны Garbage-first (G1) деп ауыстыруға тырысады.

C++ Java синтаксисіне қатты әсер етті. Бірақ синтаксисі құрылымдық, жалпыланған және объектіге бағытталған бағдарламалаудың ерекшеліктерін біріктіретін с ++ - тен айырмашылығы, Java толықтай объектіге бағытталған тіл ретінде құрылды. Барлық код сыныптар ішінде жазылады және әр деректер элементі қарапайым деректер түрлерін, яғни бүтін сандарды, өзгермелі нүкте нөмірлерін, логикалық және символдық типтерді қоспағанда, өнімділікті арттыру мақсатында объект болып табылмайтын объект болып табылады.C ++ - тен айырмашылығы, Java операторлардың шамадан тыс жүктелуін немесе бірнеше кластық мұрагерлікті қолдамайды, алайда интерфейстер үшін бірнеше мұрагерлік қолдау көрсетіледі. Мұның бәрі тілді жеңілдетеді және көптеген қателіктерден аулақ болуға және жобалау кезінде антипаттерлерді қолдануға көмектеседі [19].

Spring Framework. Java негізіндегі веб-қосымшаларды құруға келетін болсақ, Spring - ең танымал фреймворктердің бірі.

Spring Framework-бұл Java корпоративті қосымшаларын құруға арналған орта. Spring Framework Род Джонсонның *Experl* One-on-One *: J 2 EE Design and Development* (Wox, 2002) кітабынан шыққан. Соңғы онжылдықта Spring Framework негізгі функционалдылығы жағынан және қауымдастықтың қолдауы тұрғысынан айтарлықтай өсті.

Spring келесі сияқты технологияларды қамтитын құрылымды ұсына отырып, Java қолданбаларын дамытудың күрделі және басқарылмайтын кәсіпорын революциясын жеңілдетуге тырысады :

Аспектілі-бағытталған бағдарламалау (АОП)

Тәуелділікті енгізу (DI)

Ескі Java нысаны (POJO)

Spring Framework көптеген мүмкіндіктерді ұсынады. Бұл бағдарлама жасаушыларға келесі функцияларды орындауға көмектеседі:

– Деректер базасының транзакциясында транзакция API көмегінсіз орындалатын Java әдісі

– Қашықтағы API көмегінсіз қашықтағы процедураны анықтайтын жергілікті Java әдісі

– JMX API көмегінсіз басқару әрекеті үшін жергілікті Java әдісі

– JMS API көмегінсіз хабарлама өңдегіші үшін жергілікті Java әдісі

Spring Framework Java-да

Spring Framework ортасын қолдана отырып, бағдарламашлар төменде көрсетілген артықшылықтарды пайдалана алады:

– *Алдын ала орнатылған үлгілер*

Spring платформасында Hibernate, JDBC және JPA технологияларына арналған әртүрлі шаблондар бар. Бұл тәсіл арқылы әзірлеушілерге күрделі кодты анықтау қажет емес.Мысалы: JdbcTemplate-мұнда нұсқаулық жасау, транзакцияны бекіту, байланыс жасау және ерекше жағдайларды өңдеу үшін логика жазудың қажеті жоқ. Бұл тәсіл уақытты едәуір үнемдейді.

– Тестілеуге оңай

Тәуелділікті енгізу механизмі бар Spring құрылымын қолдана отырып, бүкіл қосымшаны тексеру оңай. EJB немесе Struts қосымшасы серверден қосымшаны орындауды талап етеді.

– Жеңіл IoC

Бұл, әсіресе, мысалы, EJB контейнерлерімен салыстырғанда, жеңіл. Бұл жады және CPU ресурстары шектеулі компьютерлерде қолданбаларды жасауға және орналастыруға көмектеседі.

– Тұрақты транзакцияны басқару

Spring жергілікті транзакцияға дейін (мысалы, жалғыз дерекқорды пайдалану) масштабтауға және жаһандық транзакцияларға (мысалы, JTA) масштабтауға көмектесетін интерфейсті қамтамасыз етеді.

Spring ортасы- 7 модульден тұратын көп деңгейлі архитектура. Spring модулі келесі суретте көрсетілгендей, bean компоненттерін қалай құруға, конфигурациялауға және басқаруға болатындығын анықтайтын негізгі контейнердің үстіне салынған.

Spring Framework мүмкіндіктерін кез келген J2EE серверінде пайдалануға болады және олардың көпшілігі басқарылмайтын орталар үшін де жарамды. Spring нақты J2EE қызметтерімен байланысы жоқ қайта пайдалануға болатын бизнес пен деректерге қол жеткізу нысандарын қолдау болып табылады. Мұндай нысандарды әртүрлі J2EE (Web немесе EJB) орталарында, оқшау қолданбаларда және сынақ орталарында қайта пайдалануға болатынына күмән жоқ [20].

Дипломдық жұмыстың мәліметтер қорын құру үшін PostgreSQL реляциялық дерекқорды басқару жүйесі қолданылды.

PostgreSQL («Post-Gras-Q-L» деп айтылады) – объектілік реляциялық дерекқорды басқарудың ақысыз жүйесі (ДҚБЖ). Ол AIX, әртүрлі BSD жүйелері, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX және Microsoft Windows қоса алғанда, UNIX-тәрізді әртүрлі платформаларға арналған енгізулерде бар.

PostgreSQL Берклидегі Калифорния университетінде ашық бастапқы жоба ретінде әзірленген. Коммерциялық емес Postgres дерекқорына негізделген. 1986 жылы басталған Postgres-тің дамуы сол кезде Computer Associates сатып алған бұрынғы Ingres жобасының жетекшісі Майкл Стоунбрейкермен тікелей байланысты болған. Бұл атау «Post Ingres» дегенді білдіреді және көптеген алғашқы әзірлемелер Postgres құруда қолданылған. Stonebreaker және оның оқушылары 1986 жылдан 1994 жылға дейін сегіз жыл бойы жаңа ДҚБЖ әзірледі. Осы кезеңде синтаксиске процедуралар, ережелер, пайдаланушы анықтайтын типтер және басқа компоненттер енгізілді. 1995 жылы даму қайта бөлінді: Stonebreaker өз тәжірибесін сол аттас жеке компаниясы

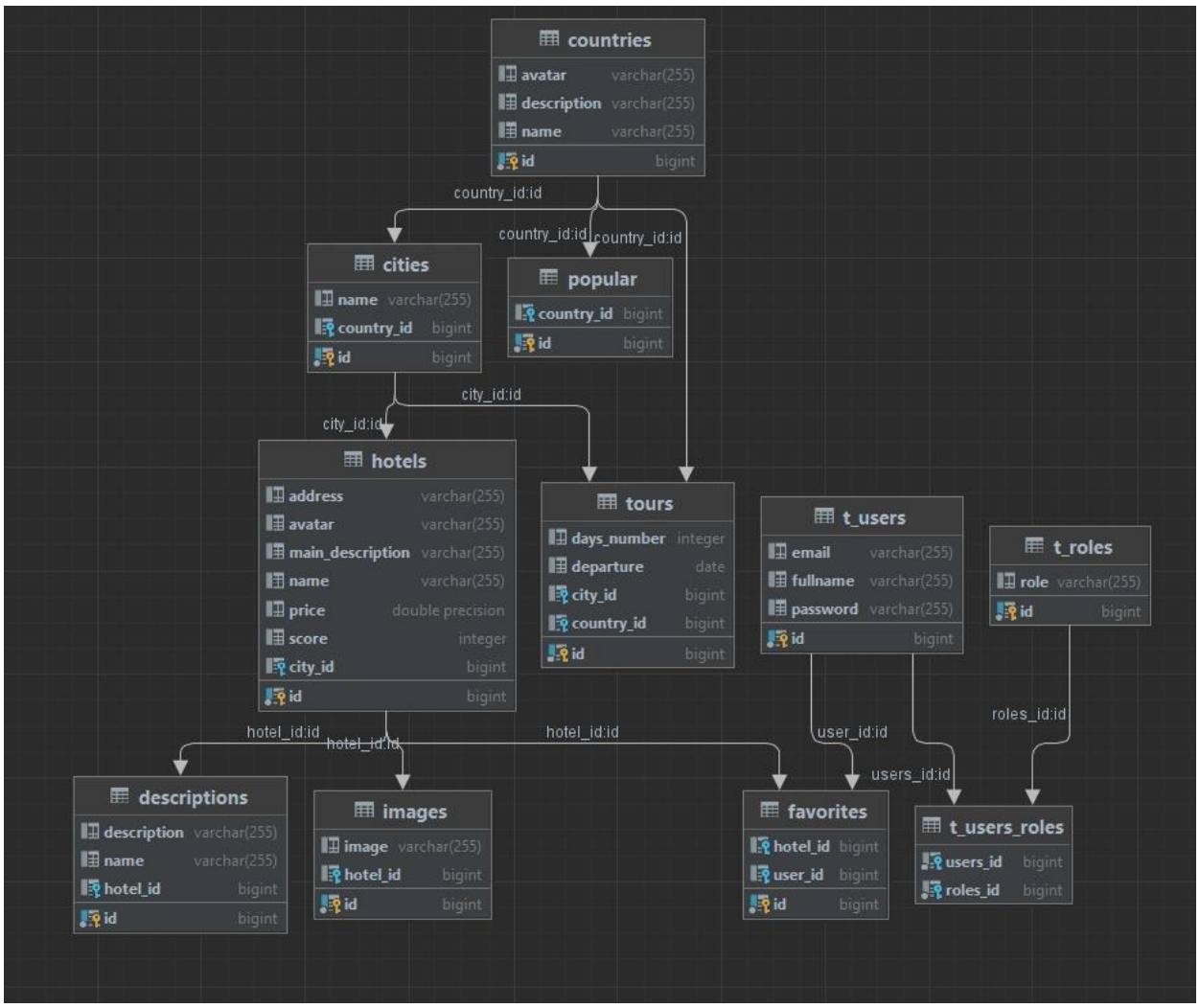
(кейінірек Informix сатып алған) сататын коммерциялық Illustra дерекқорын құру үшін пайдаланды және оның оқушылары Postgres Postgres95 жаңа нұсқасын әзірледі, онда POSTQUEL сұрауы бар.

Функциялар дерекқор клиентінде емес, серверде орындалатын код блоктары болып табылады. Олар таза SQL тілінде жазылуы мүмкін болғанымен, шартты өтулер мен циклдар сияқты қосымша логиканы жүзеге асыру SQL ауқымынан тыс және кейбір тіл кеңейтімдерін пайдалануды талап етеді. Функцияларды келесі тілдердің бірін пайдаланып жазуға болады: Кірістірілген процедуралық тіл PL/pgSQL, көп жағынан Oracle ДҚБЖ-да қолданылатын PL/SQL тіліне ұқсас; Сценарий тілдері - PL/Lua, PL/LOLCODE, PL/Perl, PL/PHP, PL/Python, PL/Ruby, PL/sh, PL/Tcl, PL/Scheme, PL/v8 (Javascript); Классикалық тілдер – C, C++, Java (PL/Java модулі арқылы); R статистикалық тілі (PL/R модулі арқылы).

Ережелер механизмі тек DML операциялары үшін ғана емес, сонымен қатар таңдау операциялары үшін теңшелетін өңдеушілерді құру механизмі болып табылады. Триггер механизмінен негізгі айырмашылығы, ережелер сұраныста талдау сатысында, оңтайлы орындау жоспарын және орындау процесінің өзін таңдау алдында іске қосылады. Ережелер кестеде SQL операциясын орындау кезінде жүйенің әрекетін қайта анықтауға мүмкіндік береді. Көріністер механизмін іске асыру жақсы мысал болып табылады: көрініс жасалған кезде, көріністе алу операциясын орындаудың орнына жүйе негізгі кестеде/кестелерде алу әрекетін орындау керектігін көрсететін ереже жасалады. Көрініс анықтамасының негізінде жатқан шарттарды алу. Жаңарту әрекеттерін қолдайтын көріністерді жасау үшін жолдарды кірістіру, жаңарту және жою ережелерін пайдаланушы анықтауы керек.

PostgreSQL келесі индекс түрлерін қолдайды: B-tree, hash, GiST, GIN, BRIN, Bloom. Қажет болса, жаңа индекс түрлерін жасауға болады. PostgreSQL-дегі индекстер келесі қасиеттерге ие: индексті тура бағытта ғана емес, кері ретпен де қарауға болады - ORDER BY ... DESC құрылысының жұмысы үшін жеке индексті құру қажет емес; кестенің бірнеше бағандары бойынша, соның ішінде әртүрлі деректер түрлерінің бағандарының үстінен индекс құруға болады; индекстер функционалды болуы мүмкін, яғни олар белгілі бір бағанның/бағандардың мәндерінің жиыны негізінде емес, мәндер жиынының функциясының мәндерінің жиыны негізінде құрылуы мүмкін; индекстер ішінара болуы мүмкін, яғни олар тек кестенің бір бөлігінде (оның кейбір проекциялары бойынша) құрылуы мүмкін; кейбір жағдайларда бұл кестенің әртүрлі (мысалы, жаңарту жиілігі бойынша) бөліктері үшін индекстердің әртүрлі түрлерін пайдалану арқылы әлдеқайда ықшам индекстерді жасауға немесе өнімділікті жақсартуға көмектеседі.

Сонымен қатар 2.2 суретте туристік фирманың дерекқоры көрсетілген.



2.2 сурет – Туристік фирманың дерекқоры

Функциялар - бұл дерекқор клиентінде емес, серверде орындалатын код блоктары.

Негізгі қозғалтқыш - бұл DML операциялары үшін ғана емес, сонымен қатар таңдау операциялары үшін де жеке процессорларды құру механизмі. Триггер механизмінен басты айырмашылығы - ережелер оңтайлы орындау жоспарын және орындау процесін таңдаудан бұрын тапсырысты талдау сатысында басталады. Ережелер кестеде SQL әрекетін орындау кезінде жүйенің әрекетін қайта анықтауға мүмкіндік береді. Пайдаланушы жаңарту әрекеттерін қолдайтын көріністер жасау үшін жолдарды кірістіру, жаңарту және жою ережелерін көрсетуі керек.

3 Туристік фирмаға арналған жүйені әзірлеу

3.1 UML тілін пайдаланып жобалау

UML – бұл диаграммалар жинағын пайдаланып бағдарламалық жасақтаманы визуализациялау тәсілі. Белгі Grady Booch, James Rumbaugh, Ivar Jacobson және Rational Software Corporation жұмысынан нысанға бағытталған дизайн үшін пайдаланылды, бірақ содан кейін ол бағдарламалық жасақтама жасау жобаларының кең ауқымын қамту үшін кеңейтілді. Бүгінгі таңда UML-ді Объектілерді басқару тобы (OMG) бағдарламалық жасақтаманы әзірлеуді модельдеу стандарты ретінде қабылдады.

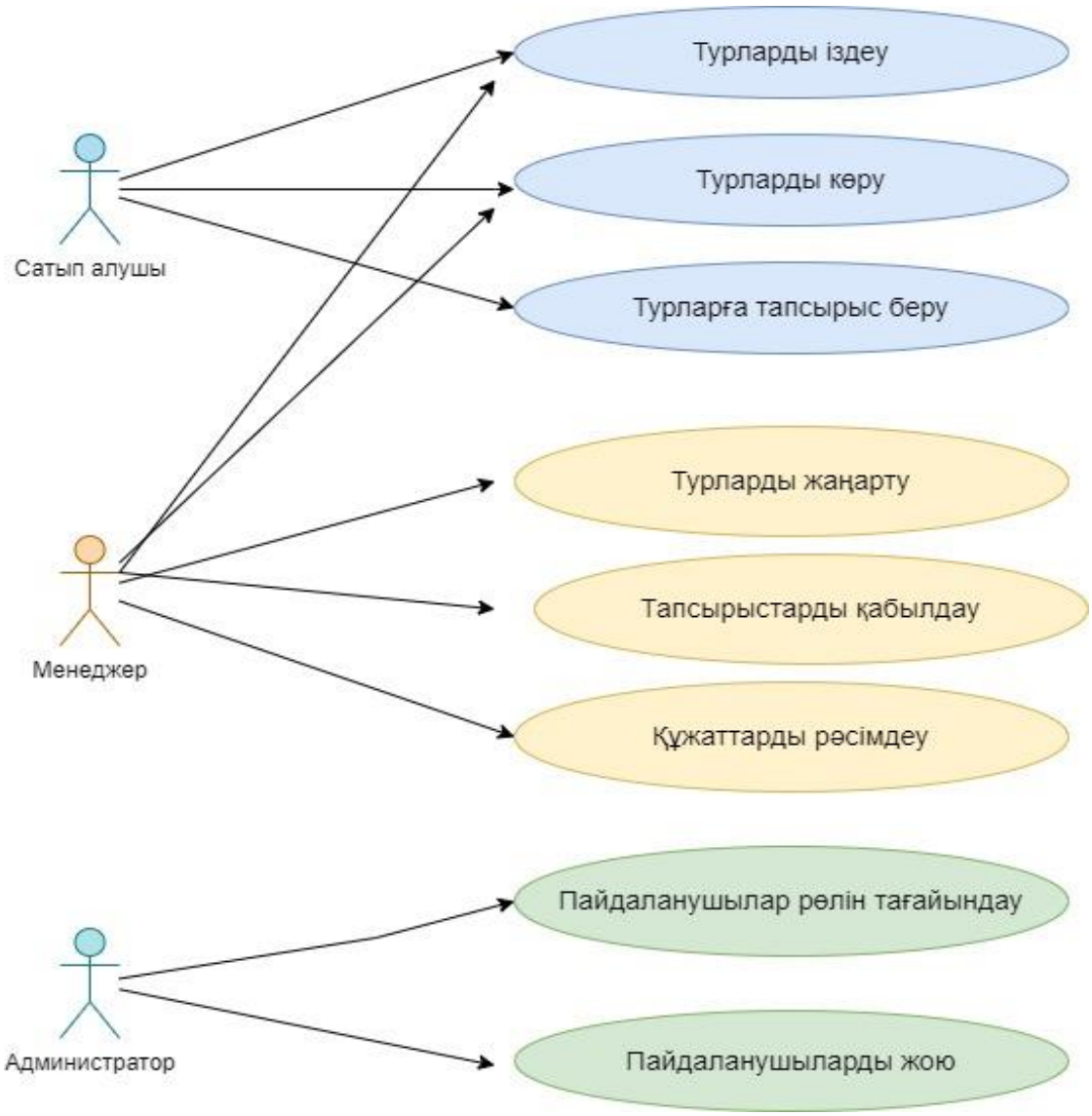
UML біртұтас модельдеу тілін білдіреді. Иерархияны анықтау және бағдарламалық қамтамасыз ету жүйесін құрамдас бөліктерге және қосалқы құрамдас бөліктерге бөлу мүмкіндігі қосылды.

UML құрылымдық диаграммалары

– класс диаграммасы
– пакеттер диаграммасы
– объектінің диаграммасы
– құрамдас диаграмма
– құрылым диаграммасы
– орналастыру диаграммасы
– белсенділік диаграммасы
– реттілік диаграммасы
– пайдалану жағдайының диаграммасы
– күй диаграммасы
– байланыс диаграммасы
– өзара әрекеттестікке шолу диаграммасы
– уақыт диаграммасы

Use case диаграммалары жүйеге қатысатын актерлердің графикалық шолуын, сол актерлер қажет ететін әртүрлі функцияларды және осы әртүрлі функциялардың өзара әрекеттесу жолын көрсетеді 3.1 сурет.

Бұл кез келген жобаны талқылау үшін бастау нүктесі болып табылады, өйткені негізгі қатысушыларды және жүйенің негізгі процестерін оңай анықтауға көмектеседі.



3.1 сурет – Use case диаграммасы

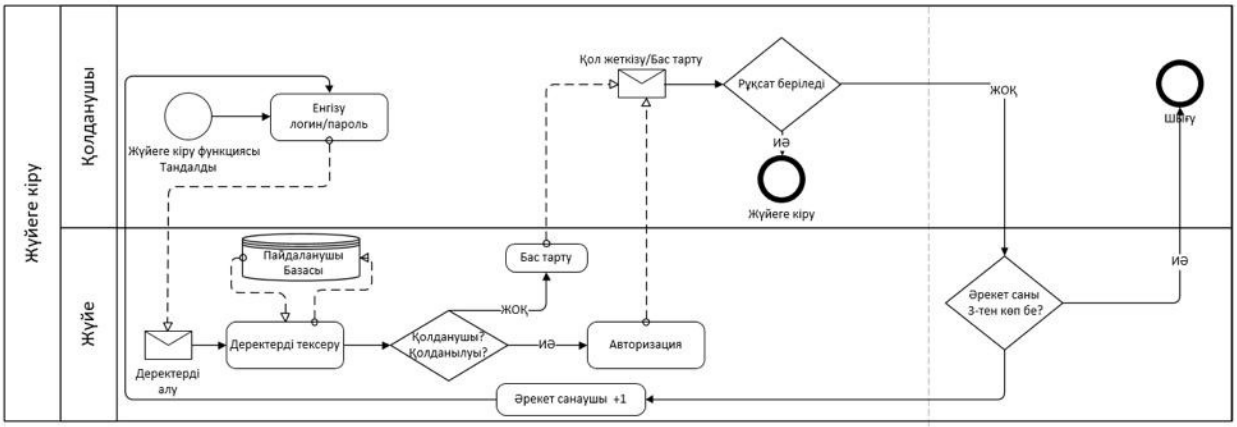
Бизнес-процестерді модельдеу нотациясы (BPMN) бизнес-процестің басынан аяғына дейінгі қадамдарды бейнелейтін ағындық диаграмма әдісі болып табылады. BPMN диаграммалары процесті аяқтау үшін қажетті жұмыс әрекеттерінің реттілігін және ақпарат ағындарының қозғалысын анық және егжей-тегжейлі көрсетеді, сондықтан бизнесті басқарудың негізгі құралдарының бірі болып табылады.

BPMN әдісін қолданудың мақсаты - жаңа жағдайларға бейімделу жолдарын, сонымен қатар тиімділік пен бәсекеге қабілеттілікті арттыру жолдарын модельдеу. Соңғы бірнеше жылда бұл әдіс стандартталған және сәл қайта қаралған атауды алды - «үлгі және бизнес-процестің белгісі», бірақ BPMN аббревиатурасы өзгеріссіз қалды.

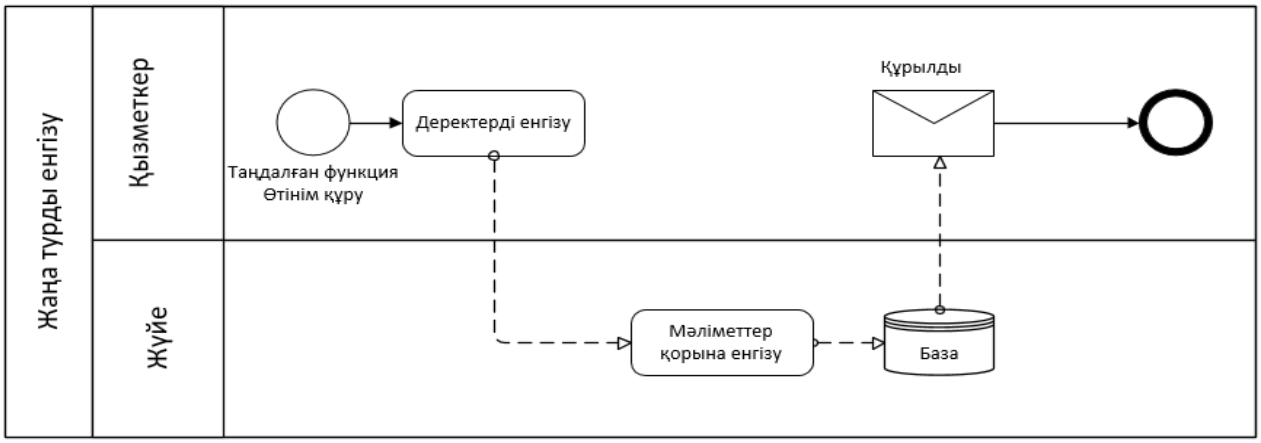
Жоғарғы деңгейдегі BPMN диаграммалары іске асырылатын қадамдардың қол жетімді суретін алу үшін пайдалана алатын бизнес-процесс қатысушылары мен басқа мүдделі тұлғаларға арналған. Неғұрлым егжей-

тегжейлі деңгей диаграммалары процесті жүзеге асыруға тікелей қатысатын және тапсырманы орындау үшін жеткілікті ақпаратты қамтитын адамдарға арналған. BPMN диаграммалары бизнес-процестерді техникалық білім деңгейіне қарамастан, барлық қатысушыларға, яғни бизнес-аналитиктерге, процесті орындаушыларға, менеджерлерге, әзірлеушілерге, сондай-ақ сыртқы қызметкерлер мен кеңесшілерге түсінікті бірыңғай стандартталған тілде сипаттайды. Ең дұрысы, бұл диаграммалар бизнес-процестерді жобалаудан іске асыруға дейінгі ыңғайлы көпірді қамтамасыз ету үшін жеткілікті егжей-тегжейлі және анық әрекеттер тізбегін көрсетуі керек.

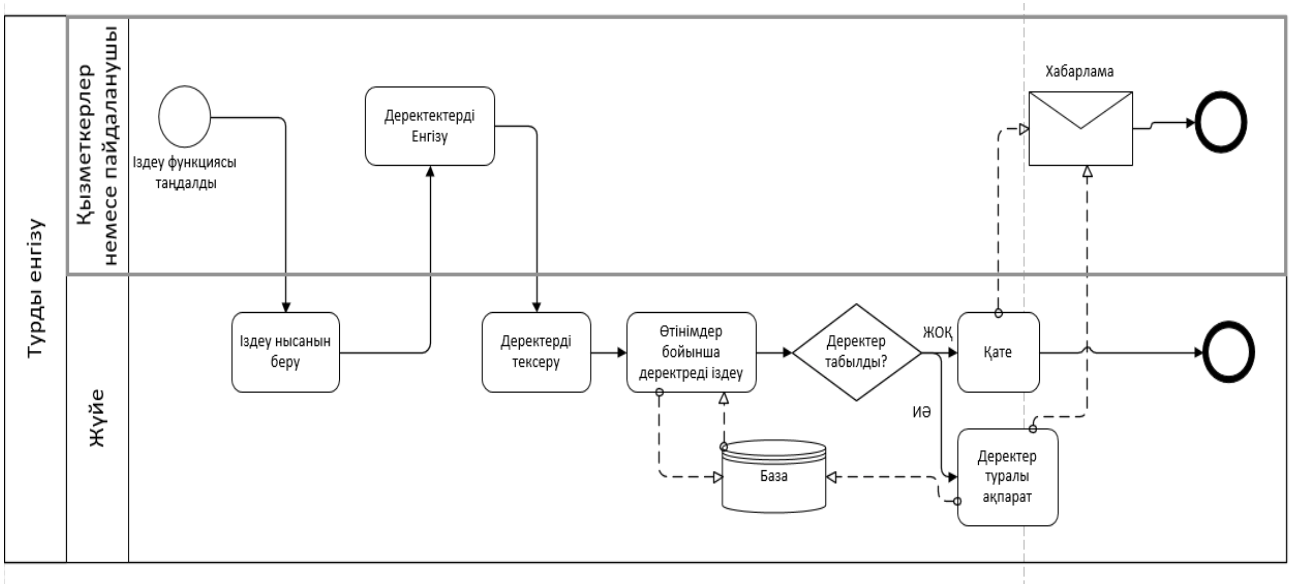
Диаграмма түрінде берілген ақпаратты мәтін түріндегі сипаттаудан гөрі түсіну оңайырақ. Схематизация ақпарат алмасуды да, сапалы нәтижеге қол жеткізу үшін тиімді процесті құру үшін бірлесіп жұмыс істеуді де жеңілдетеді. Схемалар әртүрлі процестерді орындау үшін қажетті XML құжаттарын құрастыру кезінде де талқылауды жеңілдетеді (XML - Extensible Markup Language - "extensible markup language"). 3.2-3.6 суреттерде жүйені пайдалануға байланысты әртүрлі BPMN диаграммалары көрсетілген.



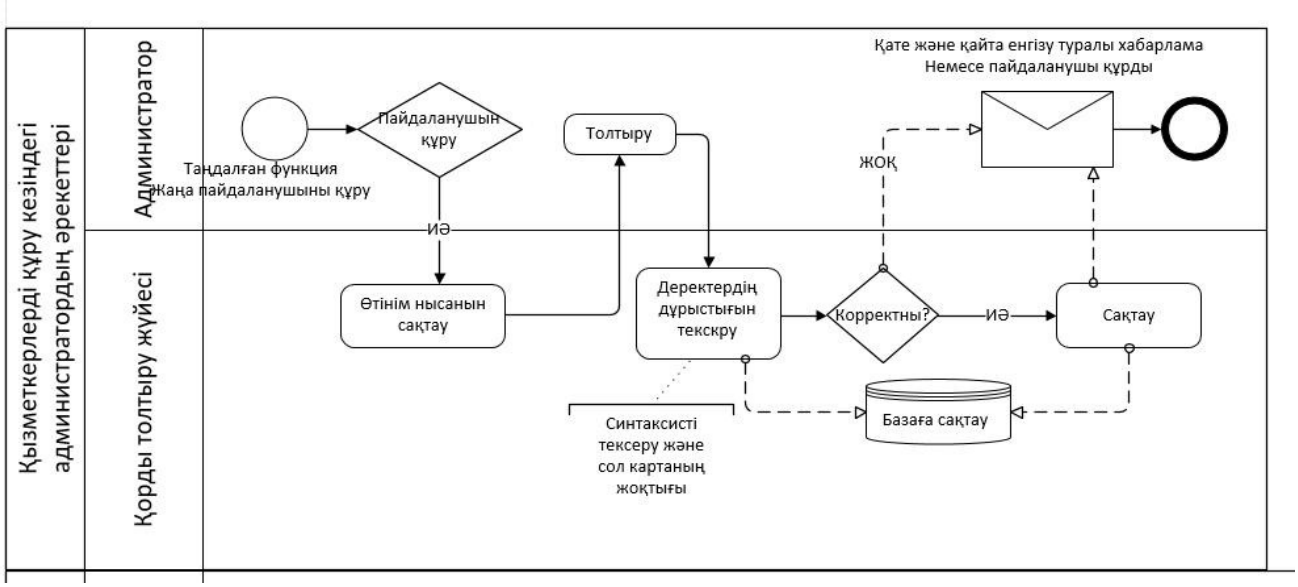
3.2 сурет – Жүйеге кіру (авторизация) бойынша BPMN диаграммасы



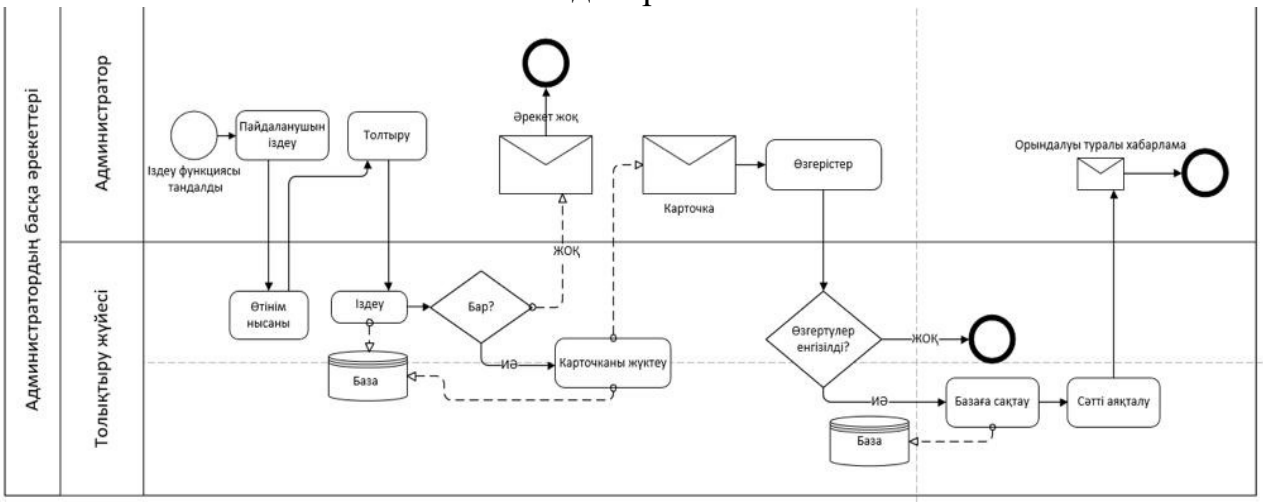
3.3 сурет – Жаңа турды жүйеге тіркеу бойынша BPMN диаграммасы



3.4 сурет – Турды іздеу бойынша BPMN диаграммасы



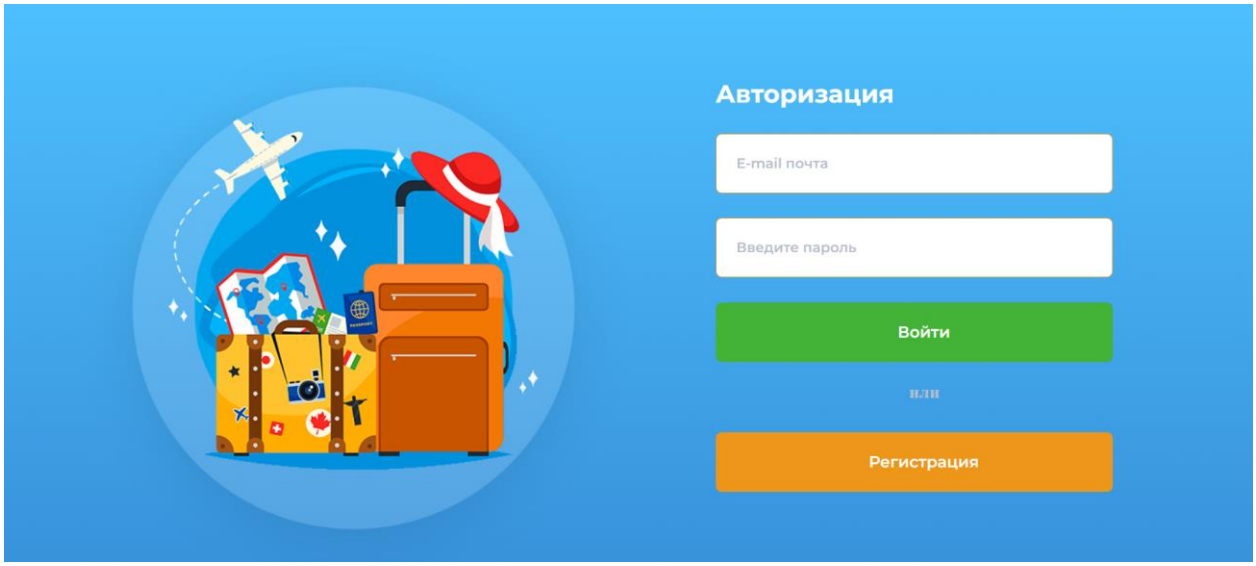
3.5 сурет – Жүйеге жаңа қолданушы тіркеу (регистрация) бойынша BPMN диаграммасы



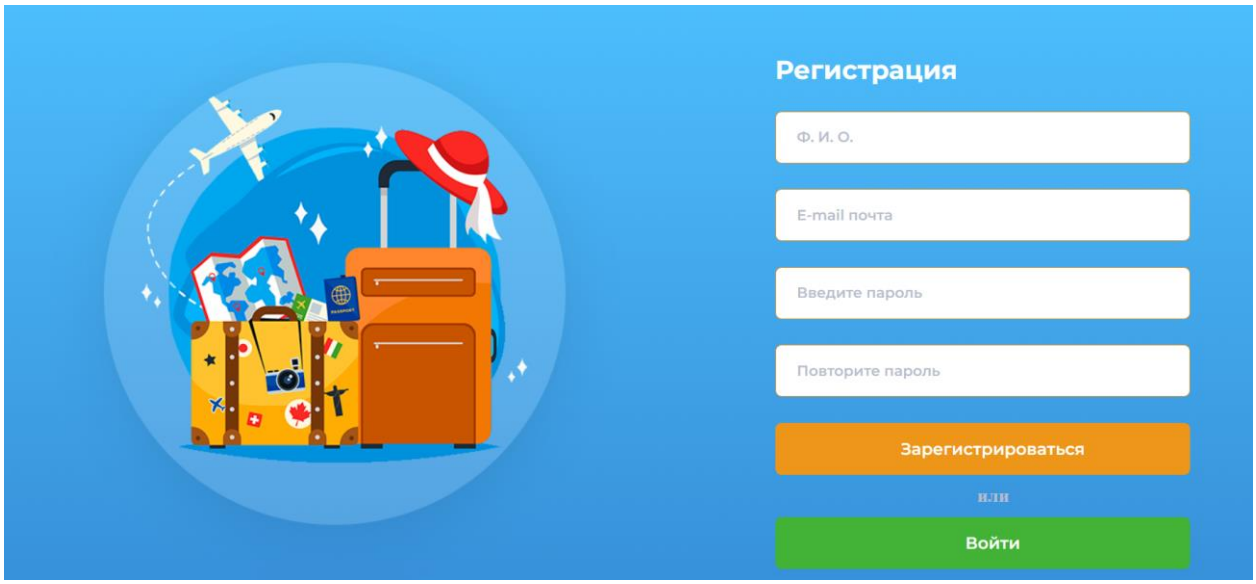
3.6 сурет – Қолданушы іздеу бойынша BPMN диаграммасы

3.2 Туристік фирмаға арналған жүйе интерфейсімен танысу

Веб сайт интерфейсі қолданушыларға ыңғайлы, түсінікті етіліп жасалды. Кез келген жүйе қолданушысы сайтқа регистрация (3.2.8 сурет) және авторизация жасай алады (3.2.7 сурет).

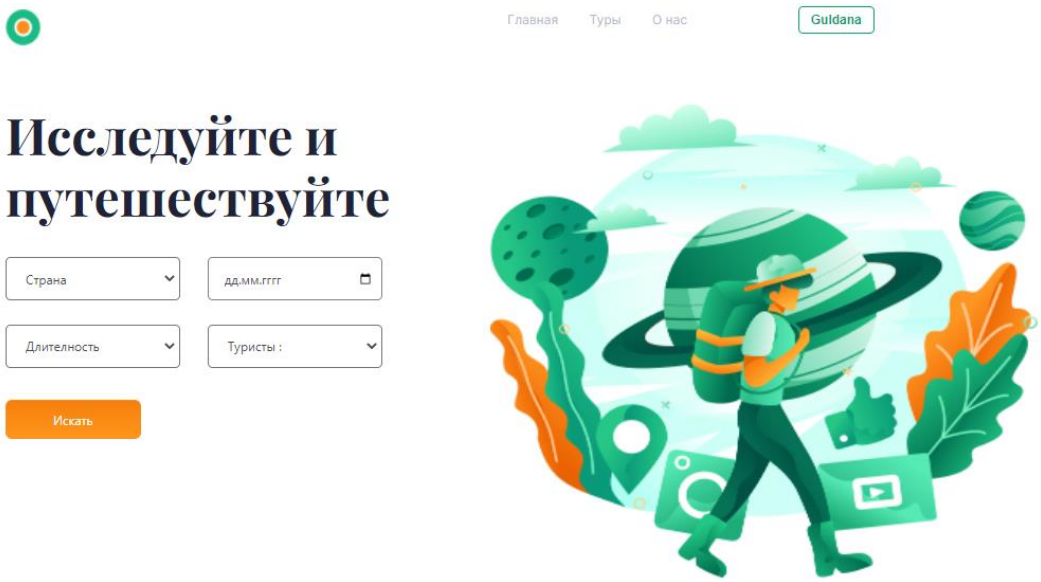


3.2.7 сурет – Авторизация парағы



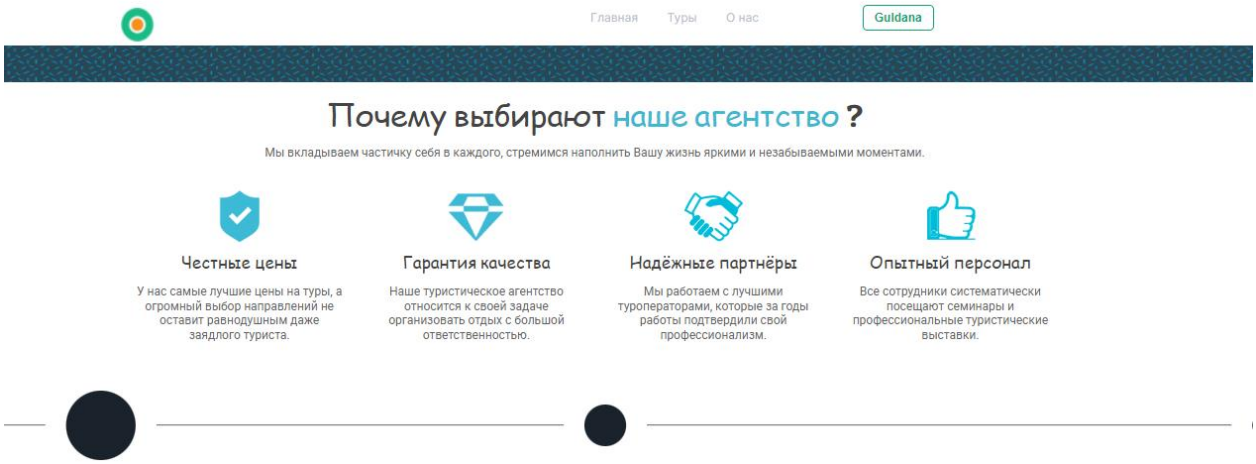
3.2.8 сурет – Регистрация парағы

Жүйеге тіркеліп, кіргеннен кейін «Басты бет» (3.2.9 сурет) ашылады. Бұл жерде саяхатшы қалаған бағыттағы мемлекетті, турдың басталатын уақыты мен созылатын уақытын, турист санын белгілеп, қажетті турды іздей алады.



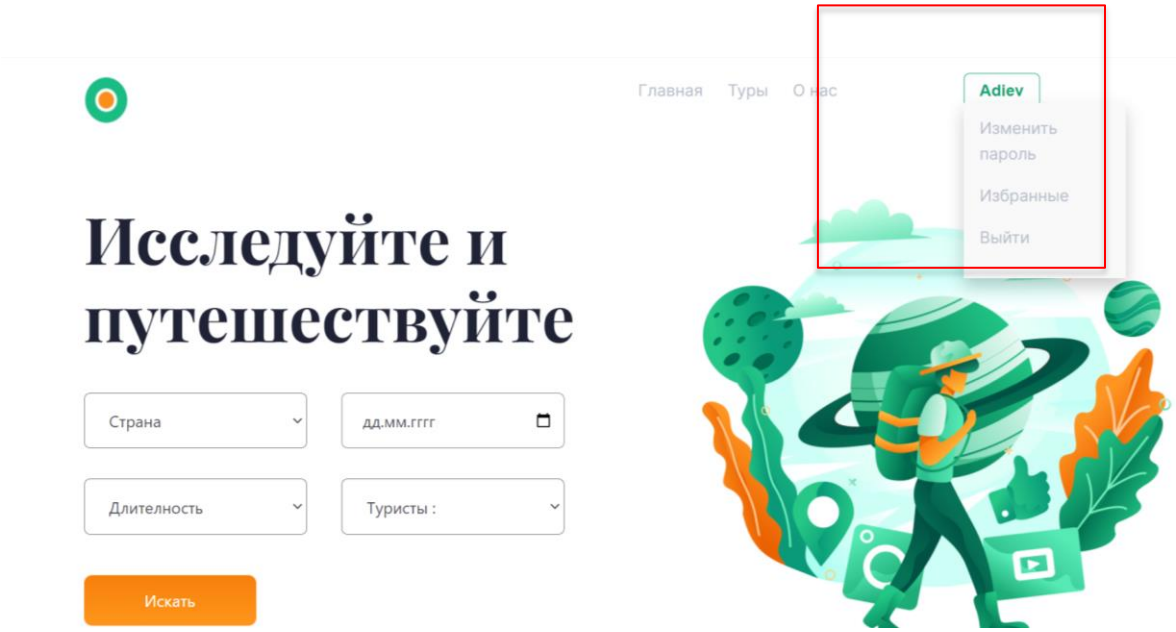
3.2.9 сурет – Басты бет

3.2.10 суретте турфирма туралы ақпараттар берілген. Оны көру үшін «О нас» батырмасын басу керек.



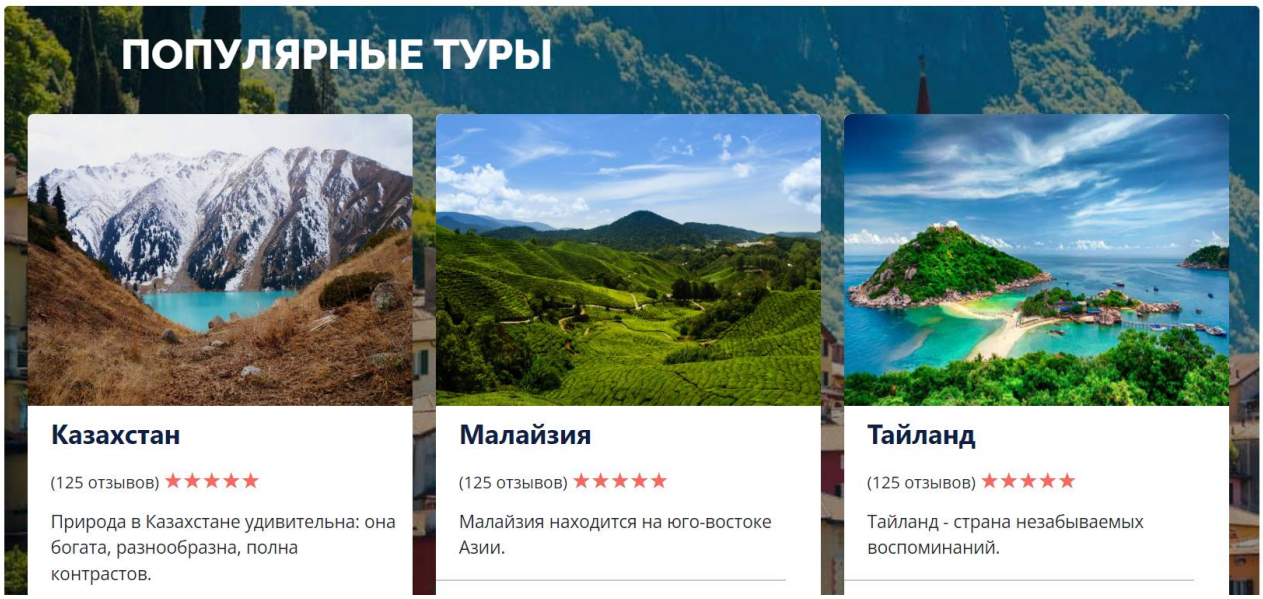
3.2.10 сурет – Тур фирма туралы ақпарат үшін «О нас» парағы

3.2.11 суретте көрсетілгендей қолданушының жеке менюі болады. Ол жерде өзіне ұнаған турларды сақтай алады, пароль өзгерте алады және жүйеден шығу үшін қолданады.



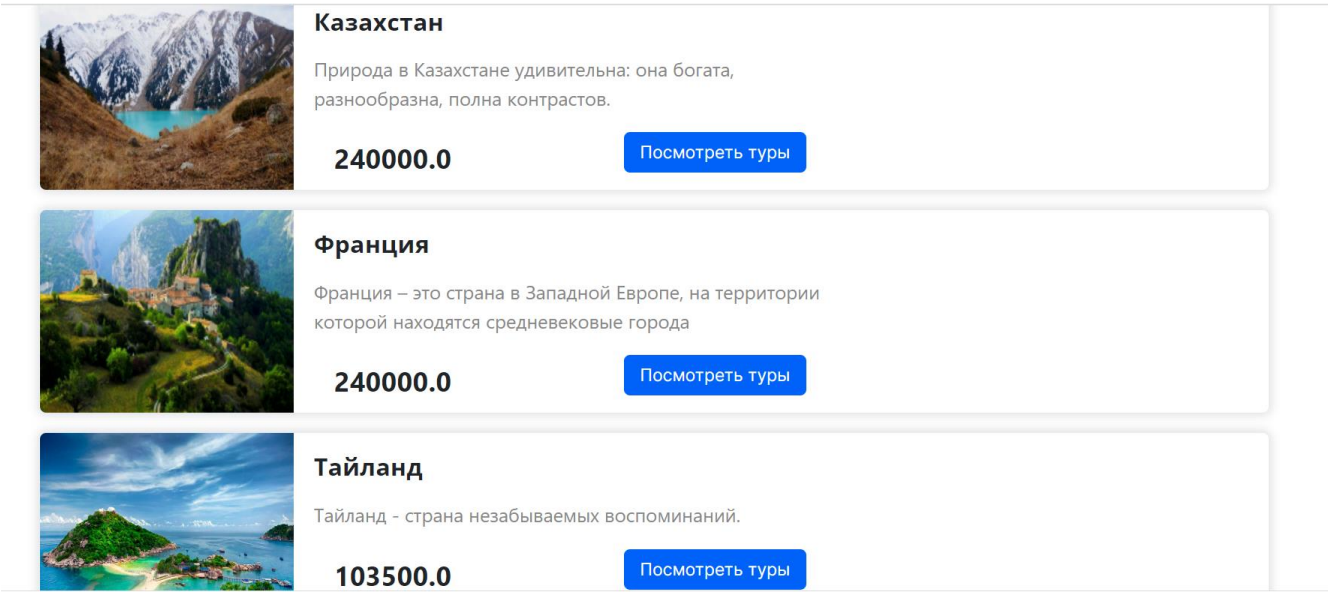
3.2.11 сурет – Қолданушы менюі

Сонымен қатар, саяхаттаушы ең танымал турларды көре алады (3.2.12 сурет).



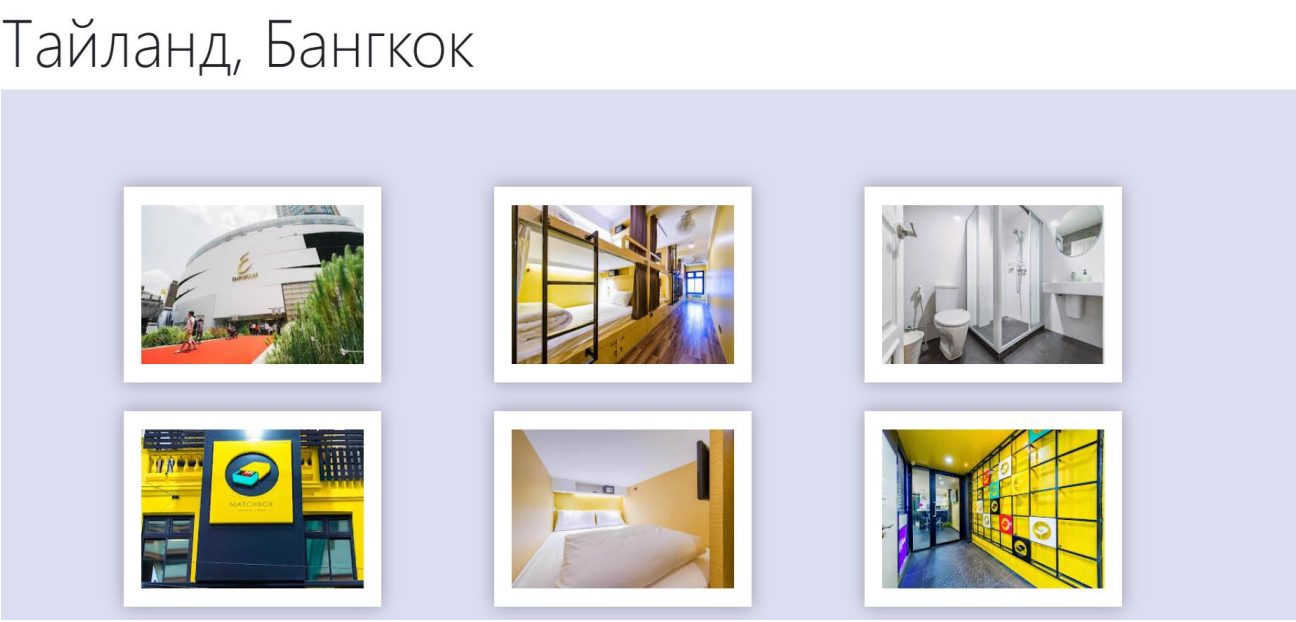
3.2.12 сурет. «Популярные туры» парағы

Саяхаттаушы берілген турларды бағасына қарай отырып, таңдай алады (3.2.13 сурет).

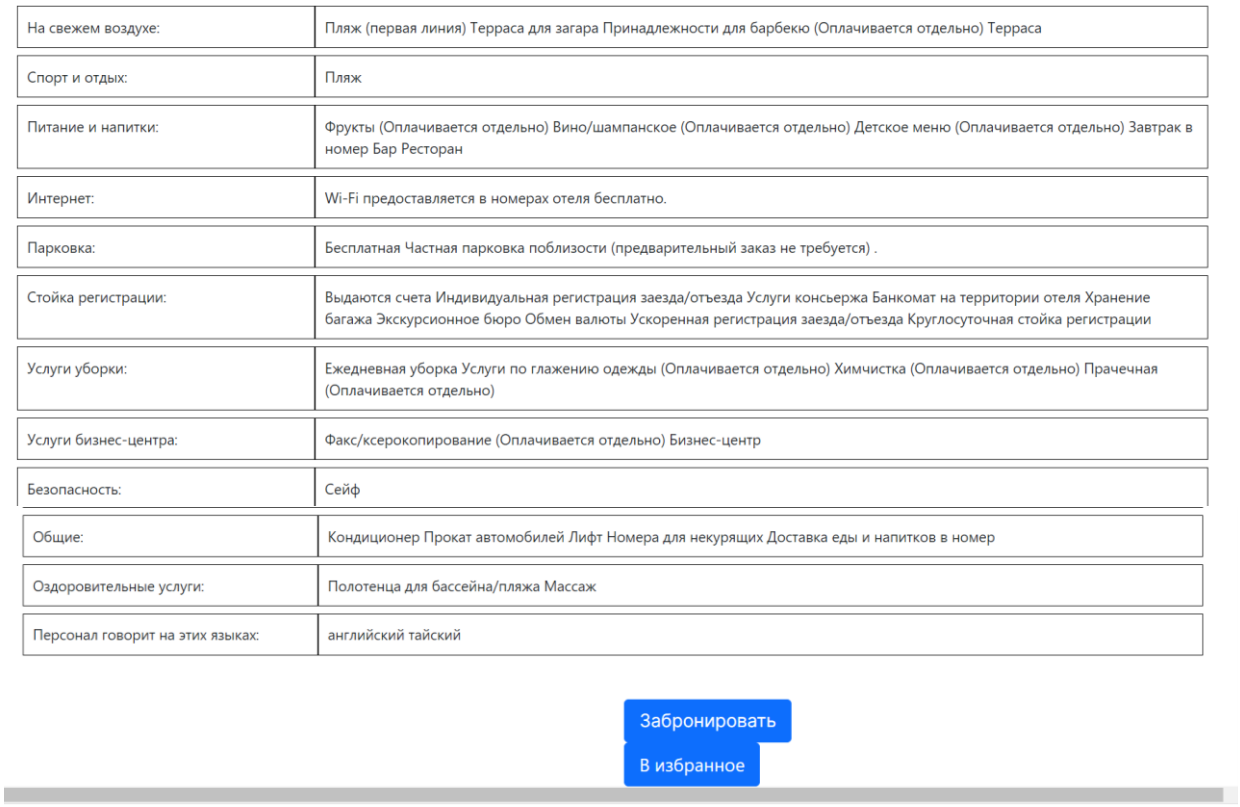


3.2.13 сурет – Турлар тізімі

Қалаған турды көру үшін «Посмотреть туры» батырмасын басамыз. Ол жерде турға байланысты барлық ақпараттар және фотосуреттері көрсетілген (3.2.14,3.2.15 сурет).

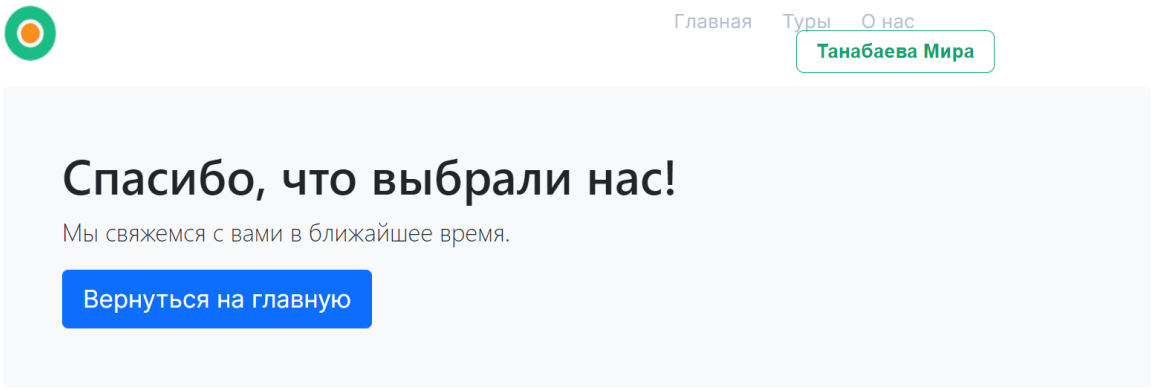


3.2.14 сурет – Тур бойынша ақпарат бөлімі

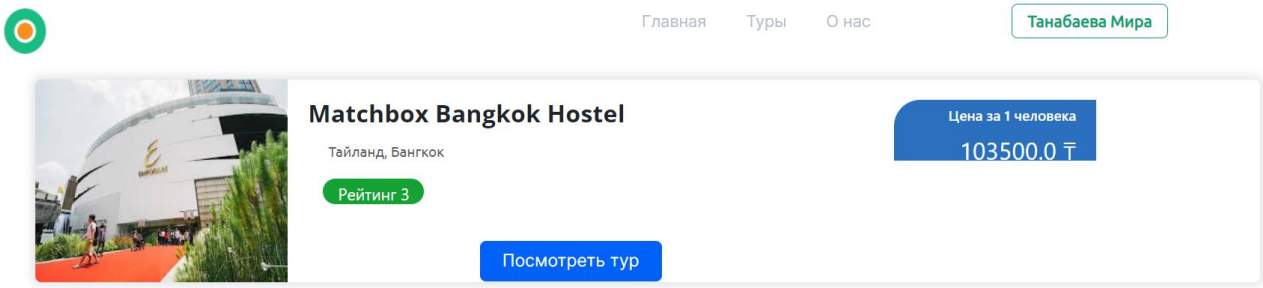


3.2.15 сурет – Тур бойынша ақпарат бөлімі

Саяхатшы тур туралы толық ақпаратпен танысып, тур ұнаған жағдайды оны «Избранные» бөліміне қосып қоюға немесе бірден брондауына болады.

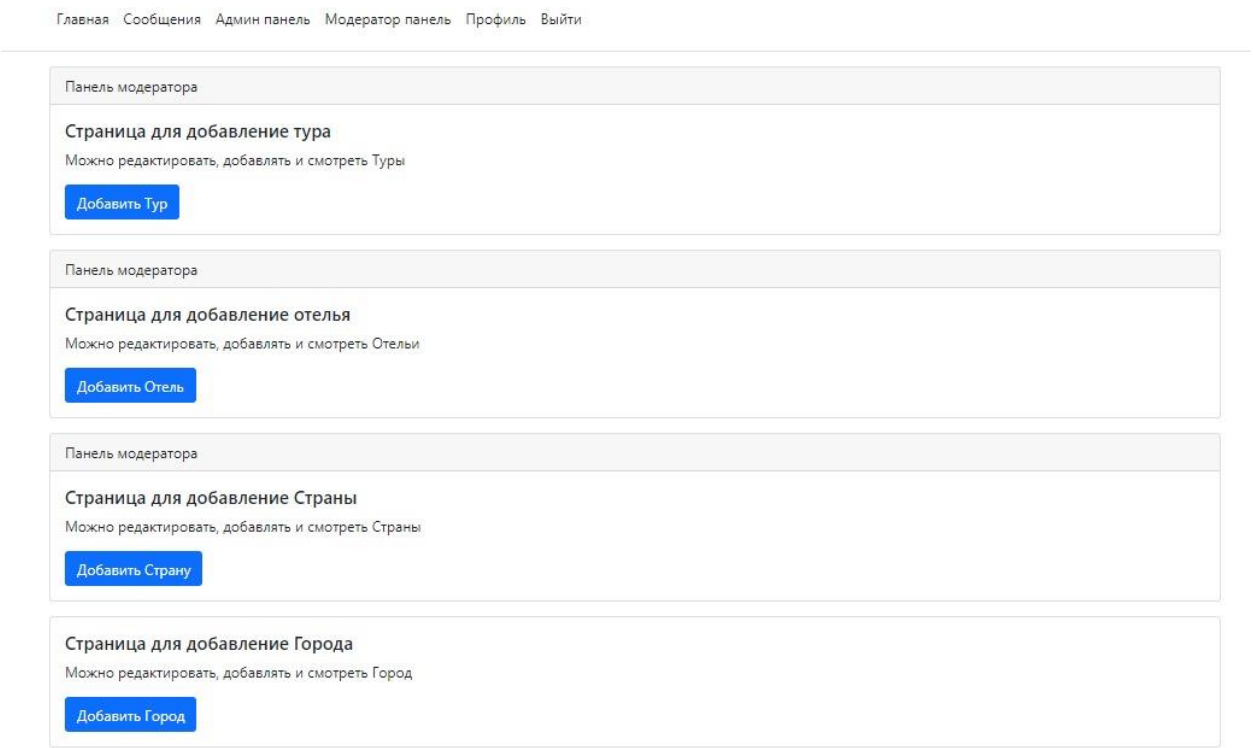


3.2.16 сурет – Турды брондау парағы

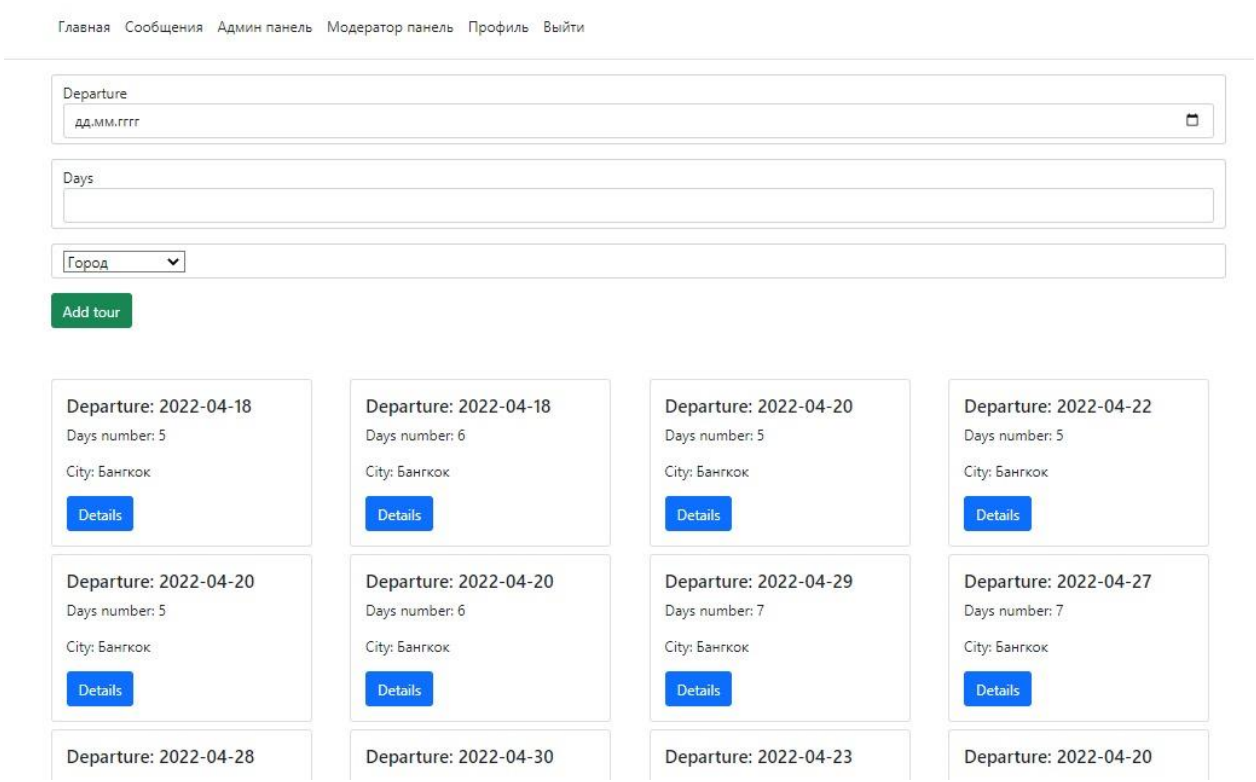


3.2.17 сурет – «В избранное» парағы

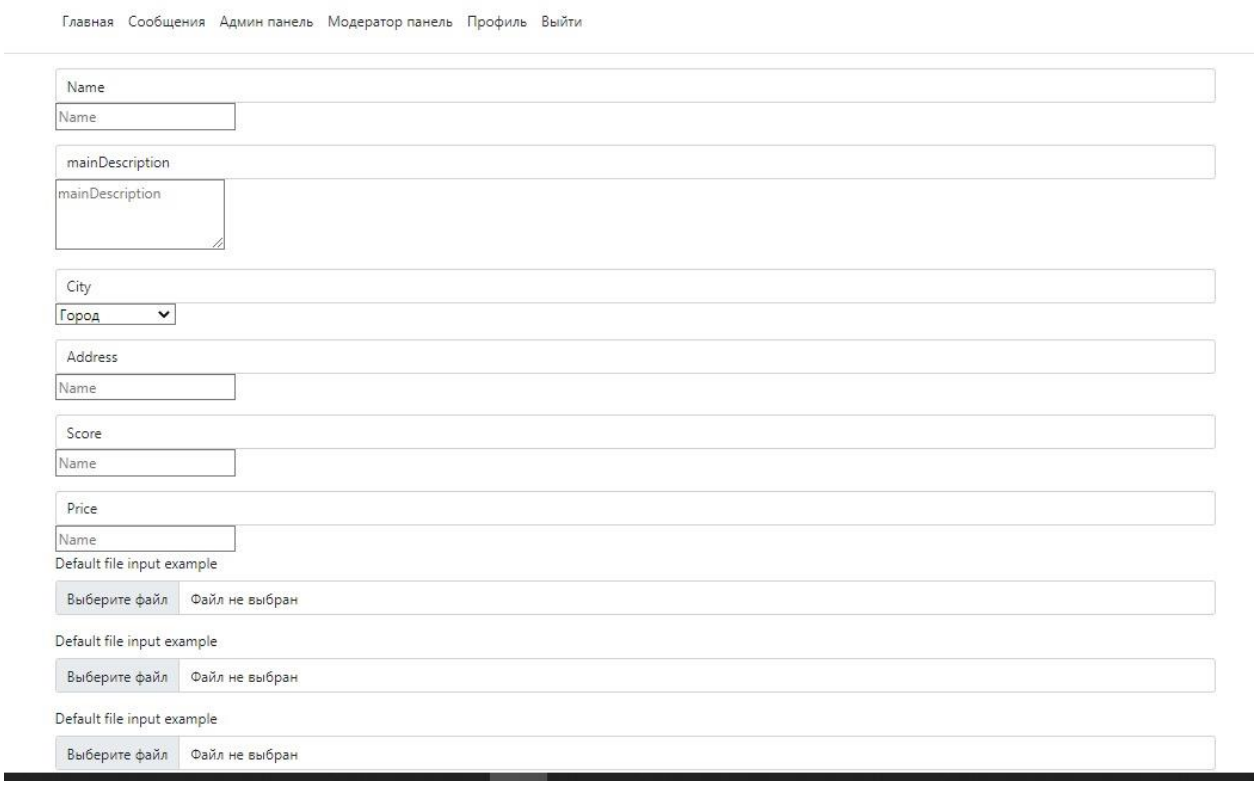
Сайт туристикалық фирманың жұмысын басқаруға барынша ыңғайлы етіп жасалды. Тек қоладанушыларға ғана емес, менеджерлердің жұмысын жеңілдетуге жағдай жасалды. Менеджер сайттағы мәліметтерді жеңіл әрі тез жаңарта алады (3.2.18-3.2.23 сурет).



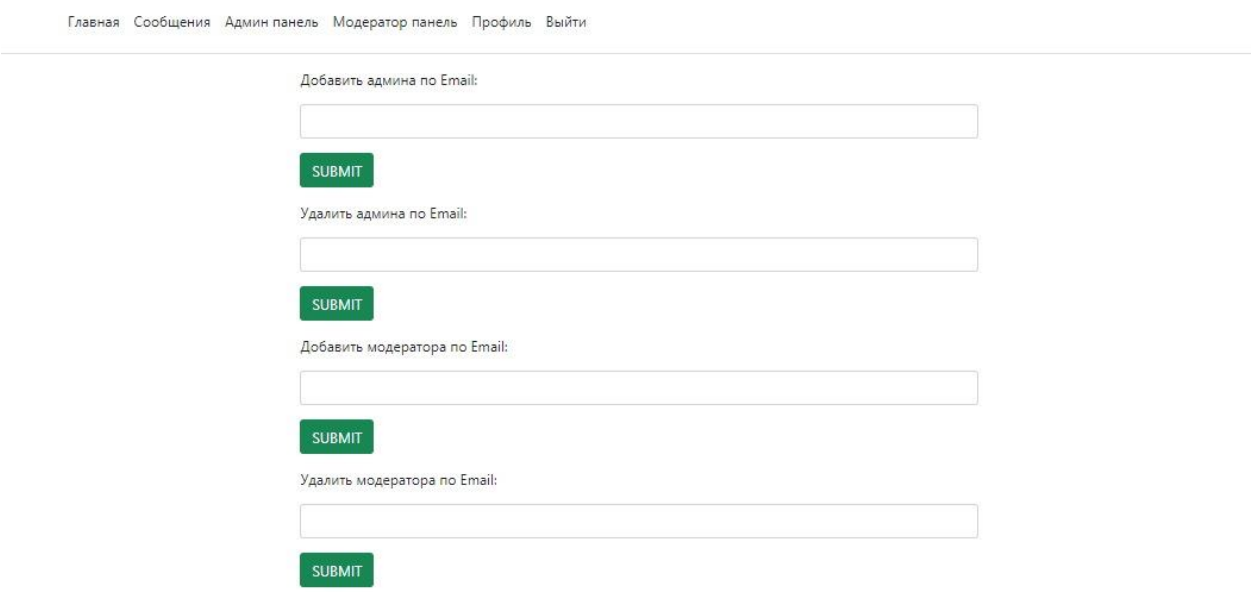
3.2.18 сурет – Менеджердің тур туралы мәліметтер жаңарту парағы



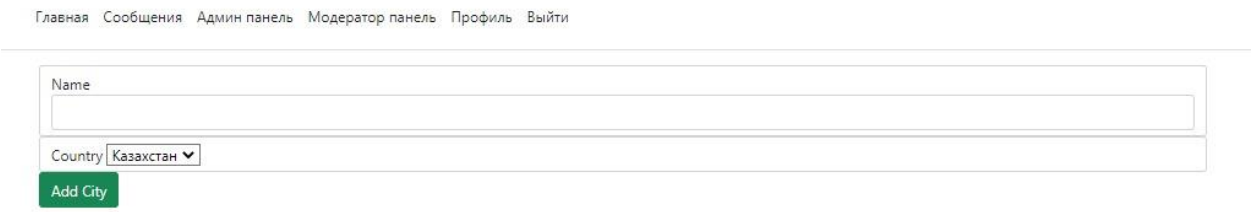
3.2.19 сурет – Менеджердің жаңа тур уақыты бойынша өзгерту парағы



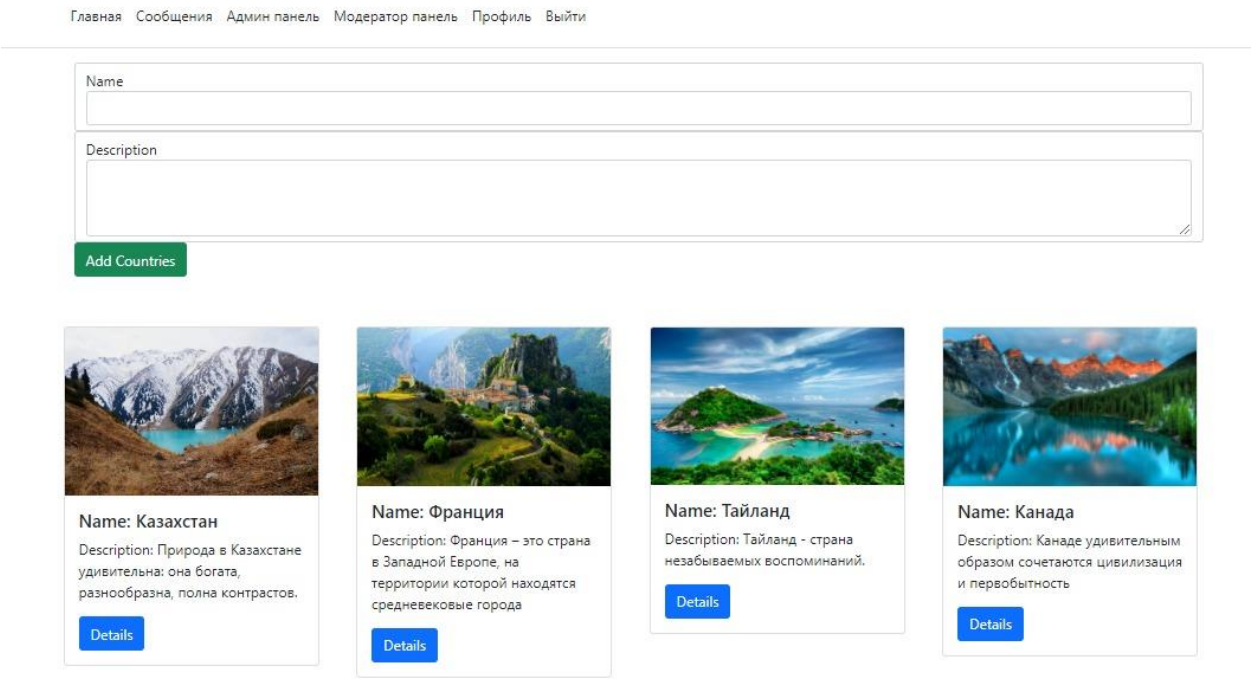
3.2.20 сурет – Менеджердің жаңа тур қосу парағы



3.2.21 сурет – Менеджердің жаңа менеджер мен админ қосу парағы



3.2.22 сурет – Менеджердің жаңа мемлекет қосу парағы



3.2.23 сурет – Менеджердің жаңа мемлекет туралы ақпарат қосу парағы

ҚОРЫТЫНДЫ

Дипломдық жұмысты жобалау процесінде пәндік аймақ талданды. Туристік фирманың жұмыс орнын автоматтандыруға арналған бағдарламалық қамтамасыз ету үшін мақсаттар мен міндеттер анықталды. Пәндік саланы талдау нәтижелері бойынша жүйенің концептуалды жобасы құрылды: пәндік саланың моделі әзірленді, концептуалды деректер моделі жасалынды.

Бұл дипломдық жобада туристік фирманың жұмыс барысы туралы деректерді жинау және сақтау бизнес-процестерді автоматтандыру мәселелері қаралды. Пайдаланушылардың рөлі мен мүмкіндіктерін визуалды түрде көрсетуге Use Case, BPMN диаграммалары пайдаланылды. Жүйе интерфейсін жасауға Figma құралдары пайдаланылды. Жүйесін дамыту үшін реляциялық PostgreSQL деректер базасын жобалау жүзеге асырылды, JAVA платформасы Spring фреймворкі негізінде веб-қосымша іске асырылды.

Әзірленген бағдарламалық қамтамасыз ету келесідей функцияларды орындауға мүмкіндік береді:

-қолдаушыларды тіркеу;

-турлар туралы ақпаратты енгізу және сақтау;

-турларды тіркеу;

-қолданушыларды тур пакеттерін брондауы.

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

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 Рубцова Н.В. Туристік қызметтің әлеуметтік-экономикалық тиімділігі. Теория, әдістеме, практика. – 2015 ж.

2. 18 Аверина А.Е. Ақпараттық жүйелерді жобалау // Қазіргі ғылым мен білім беру мәселелері. – 2015. – жоқ. 12. - 83 б.

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

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

5 Крутик А.Б. Туристік қызмет нарығындағы бәсекелестік ортаның ерекшеліктері және туристік компаниялардың бәсекеге қабілеттілігі Экономикалық ғылымдар. – 2014. – жоқ. 1. - С. 98-104.

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

7 Рябова Ю.С., Пирогов С.П. Өндірістік компанияның бизнес-процестерін модельдеу. – 2007 ж.

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

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

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

11 Кузнецов С.Д. Мәліметтер қоры // Модельдер және тілдер. Мәскеу: Бином баспасөзі. – 2008 ж.

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

13 Owens M., Allen G. PostgreSQLбойынша. – Press-LV 2010.

14 Класс диаграммалары [Электрондық ресурс]. – Қол жеткізу режимі: https://studfiles.net/preview/6214574/page:2/#5

15 Леоненков, А.V. UML және IBM Rational Rose көмегімен объектіге бағытталған талдау және жобалау / А.В. Леоненков. - М.: Бином. Білім зертханасы, Интернет ақпараттық технологиялар университеті, 2006. - 320 б.

16 Ларман, Крейг UML 2.0 және дизайн үлгілерін қолдану. Объектіге бағытталған талдауға, дизайнға және итерациялық дамуға кіріспе / Крейг Ларман. - М.: Уильямс, 2013. - 736 б.

17 Попов Н. Ақпараттық жүйелерді жобалау. – 2009 ж.

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

19 JAVA программалау тілі [Электрондық ресурс]. – Қол жеткізу режимі: https://oracle-patches.com/coding/chto-takoe-spring-obzor-frejmvorka-java

20 Spring фреймворкі туралы [Электрондық ресурс]. – Қол жеткізу режимі: https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html

Қосымша А

```
package Tourfirma.Tourfirma.controllers;

import Tourfirma.Tourfirma.entities.Roles;
import Tourfirma.Tourfirma.entities.Users;
import Tourfirma.Tourfirma.entities.tours.Countries;
import Tourfirma.Tourfirma.repositories.RolesRepository;
import Tourfirma.Tourfirma.repositories.UserRepository;
import Tourfirma.Tourfirma.repositories.tours.CountryRepository;
import Tourfirma.Tourfirma.services.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Lazy;
import org.springframework.security.access.prepost.PreAuthorize;
import
org.springframework.security.authentication.AnonymousAuthenticationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;

import java.sql.Date;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.Optional;

@Controller
public class MainController {

    private final UserRepository userRepository;
    private final RolesRepository rolesRepository;

    @Lazy
    private final UserService userService;

    private final CountryRepository countryRepository;

    public MainController(UserRepository userRepository, RolesRepository
rolesRepository, UserService userService, CountryRepository
countryRepository) {
        this.userRepository = userRepository;
        this.rolesRepository = rolesRepository;
        this.userService = userService;
        this.countryRepository = countryRepository;
    }

    @GetMapping(value = "/")
```

А қосымшасының жалғасы

```
  public String index(Model model)
    {
        if(getUser() != null)
            model.addAttribute("currentUser", getUser());
        model.addAttribute("countries", countryRepository.findAll());
        return "index";
    }

    @GetMapping(value = "/signin")
    public String signin(Model model){

        model.addAttribute("currentUser", getUser());
        return "signin";
    }

    @GetMapping(value = "/accessdenied")
    public String accessdenied(Model model){
        model.addAttribute("currentUser", getUser());
        return "403";
    }

    @GetMapping("/profile")
    @PreAuthorize("isAuthenticated()")
    public String profile(Model model){
        model.addAttribute("currentUser", getUser());
        return "profile";
    }

    @GetMapping(value = "/signup")
    public String signUpPage(Model model){
        model.addAttribute("currentUser",getUser());
        return "signup";
    }

    @PostMapping("/tosignup")
    public String signUp(@RequestParam(name = "user_email") String email,
                         @RequestParam(name = "user_password")String
password,
                         @RequestParam(name = "re_user_password")String
rePassword,
                         @RequestParam(name = "user_fullName")String
fullName){
        if(password.equals(rePassword)){

            Users newUser = userService.registerUser(new
Users(null,email,password,fullName,null));
            if(newUser!=null && newUser.getId()!=null){
                return "redirect:/signin";
            }
        }
        return "redirect:/signup?error";
    }
```

А қосымшасының жалғасы

```
    @PostMapping("/addtour")
    public String addTour(Model model){
        return "redirect:/admin";
    }

    @GetMapping("/admin")
    @PreAuthorize("hasAnyRole('ROLE_ADMIN')")
    public String adminPanel(Model model){
        List<Countries> ct = countryRepository.findAll();
        model.addAttribute("countries", ct);
        model.addAttribute("currentUser", getUser());
        return "admin";
    }

    private Users getUser(){
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        if(!(authentication instanceof AnonymousAuthenticationToken)){
            return (Users) authentication.getPrincipal();
        }
        return null;
    }


    @PostMapping("/removeadmin")
    public String removeAdmin(@RequestParam(name = "user_remove_admin")String
email, Model model){
        Users currentUser = getUser();
        Users userToDest = userRepository.findByEmail(email);
        if(userToDest == null){
            System.out.println("Cannot find user with this email!");
            return "redirect:/admin";
        }
        if(Objects.equals(userToDest.getEmail(), currentUser.getEmail())){
            System.out.println("Just you cannot!!");
            return "redirect:/admin";
        }
        List<Roles> roles = userToDest.getRoles();
        roles.remove(rolesRepository.findByRole("ROLE_ADMIN"));
        userToDest.setRoles(roles);
        userRepository.save(userToDest);
        return "redirect:/";
    }

    @PostMapping("/getadmin")
    public String setAdmin(@RequestParam(name = "user_email_admin")String
email, Model model){
        Users userToDest = userRepository.findByEmail(email);
        if(userToDest == null){
            System.out.println("Cannot find user with this email!");
            return "redirect:/admin";
        }
        if(Objects.equals(userToDest.getEmail(), currentUser.getEmail())){
```

```
            System.out.println("Just you cannot!!");
            return "redirect:/admin";
        }
        List<Roles> roles = userToDest.getRoles();
        roles.add(rolesRepository.findByRole("ROLE_ADMIN"));
        userToDest.setRoles(roles);
        userRepository.save(userToDest);
        return "redirect:/";
    }

    @PostMapping("/updatepassword")
    public String updatePassword(@RequestParam(name = "old_password")String
old_Pass,
                                 @RequestParam(name = "new_password")String
newPass,
                                 @RequestParam(name =
"retype_new_password")String reNewPass){
        if(newPass.equals(reNewPass)){
            if(userService.updatePassword(getUser(),old_Pass,newPass)){
                return "redirect:/profile?success";
            }
        }
        return "redirect:/profile?error";
    }

    @GetMapping("/about")
    public String about(
            Model model
    ){
        model.addAttribute("currentUser", getUser());
        return "aboutme";
    }

    @GetMapping("/error")
    public String error(Model model){
        model.addAttribute("currentUser", getUser());
        return "error";
    }
}

package Tourfirma.Tourfirma.controllers;

import Tourfirma.Tourfirma.entities.Users;
import Tourfirma.Tourfirma.entities.tours.*;
import Tourfirma.Tourfirma.repositories.tours.*;
import
org.springframework.security.authentication.AnonymousAuthenticationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
```

А қосымшасының жалғасы

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.*;

@Controller
@RequestMapping("/tour")
public class TourController {

    private final CountryRepository countryRepository;
    private final CityRepository cityRepository;
    private final HotelRepository hotelRepository;
    private final TourRepository tourRepository;
    private final DescriptionRepository descriptionRepository;
    private final ImageRepository imageRepository;
    private final PopularRepository popularRepository;

    private final FavoriteRepository favoriteRepository;

    public TourController(CountryRepository countryRepository, CityRepository
cityRepository, HotelRepository hotelRepository, TourRepository
tourRepository, DescriptionRepository descriptionRepository, ImageRepository
imageRepository, PopularRepository popularRepository, FavoriteRepository
favoriteRepository) {
        this.countryRepository = countryRepository;
        this.cityRepository = cityRepository;
        this.hotelRepository = hotelRepository;
        this.tourRepository = tourRepository;
        this.descriptionRepository = descriptionRepository;
        this.imageRepository = imageRepository;
        this.popularRepository = popularRepository;
        this.favoriteRepository = favoriteRepository;
    }


    @GetMapping()
    public String index(Model model){
        model.addAttribute("currentUser", getUser());
        List<Countries> countries = countryRepository.findAll();
        model.addAttribute("countries", countries);
        List<Popular> populars = popularRepository.findAll();
        List<Countries> popularCountries = new ArrayList<>();
        for(Popular popular : populars){
            popularCountries.add(popular.getCountry());
        }
        model.addAttribute("popularCountries", popularCountries);
        Map<Countries, Double> countryPrices = new HashMap<>();
        for(Countries country : countries){
            List<Cities> cities = cityRepository.getCitiesByCountry(country);
            List<Hotels> hotels = hotelRepository.findAllByCityIn(cities);
            double startPrice = 240000;
            for(Hotels hotel : hotels){
                if(hotel.getPrice() < startPrice){
```

А қосымшасының жалғасы

```
                    startPrice = hotel.getPrice();
                }
            }
            countryPrices.put(country, startPrice);
        }
        System.out.println(countryPrices);
        model.addAttribute("countryPrices", countryPrices);
        return "toursTemplates/index";
    }

    @GetMapping("/{id}")
    public String show(@PathVariable("id") long id, Model model){
        model.addAttribute("currentUser", getUser());
        Countries country = countryRepository.getById(id);
        List<Cities> cities = cityRepository.getCitiesByCountry(country);
        model.addAttribute("country", country);
        model.addAttribute("cities", cities);
        List<Hotels> hotels = hotelRepository.findAllByCityIn(cities);
        model.addAttribute("hotels", hotels);
        return "toursTemplates/show";
    }

    @GetMapping("/{id}/{hotelId}")
    public String showHotel(@PathVariable("id") long id,
@PathVariable("hotelId")long hotelId, Model model){
        model.addAttribute("currentUser", getUser());
        Hotels hotel = hotelRepository.getById(hotelId);
        model.addAttribute("hotel", hotel);
        List<Images> images = imageRepository.findAllByHotel(hotel);
        model.addAttribute("images", images);
        List<Descriptions> descriptions =
descriptionRepository.findAllByHotel(hotel);
        model.addAttribute("descriptions", descriptions);
        return "toursTemplates/show_hotel";
    }

    @GetMapping("/search")
    public String search(Model model,
                         @RequestParam("country")long id,
                         @RequestParam("departure") String departureString,
                         @RequestParam("daysNumber") int daysNumber,

                         @RequestParam("tourists") int tourists){
        model.addAttribute("currentUser", getUser());
        Countries country = countryRepository.getById(id);
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-
dd");
        LocalDate departure = LocalDate.parse(departureString, formatter);
        List<Tour> tours =
tourRepository.findAllByCountryAndAndDepartureAndAndDaysNumber(country,
departure, daysNumber);
        if(tours.size() == 0){
            return "toursTemplates/tours_not_found";
```

```
        }
        model.addAttribute("tours", tours);
        List<Cities> cities = new ArrayList<>();
        for(Tour tour : tours){
            cities.add(tour.getCity());
        }
        List<Hotels> hotels = hotelRepository.findAllByCityIn(cities);
        model.addAttribute("hotels", hotels);
        model.addAttribute("tourists", tourists);
        model.addAttribute("departure", departure);
        model.addAttribute("flyDay", departure.getDayOfMonth());
        model.addAttribute("flyMonth", MONTHS[departure.getMonthValue() -
1]);
        return "toursTemplates/search";
    }

    @GetMapping("/favorite")
    private String toFavorites(@RequestParam("id")long id, Model model){
        model.addAttribute("currentUser", getUser());
        if(getUser() == null){
            return "signin";
        }
        Hotels hotel = hotelRepository.getById(id);
        Favorite favorite = new Favorite();
        favorite.setUser(getUser());
        favorite.setHotel(hotel);
        favoriteRepository.save(favorite);
        model.addAttribute("hotel", hotel);
        List<Images> images = imageRepository.findAllByHotel(hotel);
        model.addAttribute("images", images);
        List<Descriptions> descriptions =
descriptionRepository.findAllByHotel(hotel);
        model.addAttribute("descriptions", descriptions);
        return "toursTemplates/show_hotel";
    }

    @GetMapping("/favorites")
    private String favorites(Model model){
        model.addAttribute("currentUser", getUser());
        if(getUser() == null){
            return "signin";
        }

        List<Favorite> favorites =
favoriteRepository.findAllByUser(getUser());
        List<Hotels> hotels = new ArrayList<>();
        for(Favorite favorite : favorites){
            hotels.add(favorite.getHotel());
        }
        model.addAttribute("hotels", hotels);
        return "toursTemplates/favorites";
    }
```

А қосымшасының жалғасы

```
    @GetMapping("/tours_not_found")
    private String nfTours(Model model){
        model.addAttribute("currentUser", getUser());
        return "toursTemplates/tours_not_found";
    }

    @GetMapping("/booking")
    private String booking(Model model){
        if(getUser() == null){
            return "signin";
        }
        model.addAttribute("currentUser", getUser());
        return "toursTemplates/booking_hotel";
    }

    private static final String[] MONTHS = {"Январь", "Февраль", "Март",
"Апрель", "Май", "Июнь", "Июль", "Август", "Сентябрь", "Октябрь", "Ноябрь",
"Декабрь"};



    private Users getUser(){
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        if(!(authentication instanceof AnonymousAuthenticationToken)){
            return (Users) authentication.getPrincipal();
        }
        return null;
    }
}
package Tourfirma.Tourfirma.entities.tours;

import lombok.*;
import org.hibernate.Hibernate;

import javax.persistence.*;
import java.util.Objects;

@Entity
@Table(name="cities")
@AllArgsConstructor
@NoArgsConstructor
@Getter
@Setter
@ToString
public class Cities {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "name")
```

```
    private String name;

    @ManyToOne(fetch = FetchType.EAGER)
    private Countries country;

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o))
return false;
        Cities cities = (Cities) o;
        return id != null && Objects.equals(id, cities.id);
    }

    @Override
    public int hashCode() {
        return getClass().hashCode();
    }
}
package Tourfirma.Tourfirma.entities.tours;

import lombok.*;
import org.hibernate.Hibernate;
import javax.persistence.*;
import java.util.Objects;
@Entity
@Table(name = "countries")
@Getter
@Setter
@ToString
@AllArgsConstructor
@NoArgsConstructor
public class Countries {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "name")
    private String name;

    @Column(name = "description")
    private String description;
    @Column(name = "avatar")
    private String avatar;
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o))
return false;
        Countries countries = (Countries) o;
```

Қосымша Б

```
<!DOCTYPE html>
<html xmlns:th = "http://www.thymeleaf.org"
      xmlns:layout = "http://www.ultraq.net.nz/thymeleaf/layout"
      xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
<head>
    <meta charset="UTF-8">
    <title>Real-Tour</title>
    <link rel="stylesheet" type="text/css" th:href="@{'/css/style.css'}">
</head>
<body>

<header>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css
" rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" th:href="@{'/css/style.css'}">
    <style>
        .dropdown {
            position: relative;
            display: inline-block;
            margin-left: 100px;
        }

        .dropdown-content {
            display: none;
            position: absolute;
            background-color: #f9f9f9;
            min-width: 160px;
            box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
            padding: 12px 16px;
            z-index: 1;
        }

        .dropdown:hover .dropdown-content {
            display: block;
        }

        .user-information{
            text-align: center;
            font-weight: 700;
            font-size: 16px;
            font-family: 'Ubuntu', sans-serif;
            color: #0da470;
            padding: 7px 15px 7px 15px;
            border: 1px solid #0da470;
            border-radius: 6px;

        }

    </style>
    <div class="container">
        <div class="navbar-menu">
```

Б қосымшасының жалғасы

```
            <img class="logo" th:src="@{'/img/logor.png'}" alt="logo">
            <div class="menu-bottoms">
                <ul>
                    <li><a class="menu-bottom active_c"
th:href="@{'/'}">Главная</a></li>
                    <li><a class="menu-bottom"
th:href="@{'/tour'}">Туры</a></li>
                    <li><a class="menu-bottom" th:href="@{'/about'}">О
нас</a></li>
<!--                    <li><a class="menu-bottom"
th:href="@{'/'}">Партнеры</a></li>-->
                    <li sec:authorize="isAnonymous()"><a class="login-bottom
" th:href="@{'/signin'}">Sign in</a></li>
<!--                    <li sec:authorize="isAuthenticated()"><a
class="login-bottom " href="JavaScript:void(0)" onclick="signOut()">Sign Out
</a></li>-->
                    <li sec:authorize="isAnonymous()"><a class="register-
bottom" th:href="@{'/signup'}">Register</a></li>
                    <li sec:authorize="isAuthenticated()">
                        <div class="dropdown">
                            <span class="user-information"
th:text="${currentUser.getFullName()}"></span>
                            <div class="dropdown-content">
                                <p><a th:href="@{'/profile'}" >Изменить
пароль</a></p>
                                <p><a
th:href="@{/tour/favorites}">Избранные</a></p>
                                <p><a href="JavaScript:void(0)"
onclick="signOut()">Выйти</a></p>
                            </div>
                        </div>
                    </li>
                </ul>
            </div>
        </div>
    </div>
    <script type="text/javascript" sec:authorize="isAuthenticated()">
        function signOut(){
            document.getElementById("signOutFormId").submit();
        }
    </script>
    <script type="text/javascript" th:src="@{'/js/main.js'}"></script>
    <form th:action="@{'/signout'}" method="post" id="signOutFormId"
sec:authorize="isAuthenticated()">

    </form>
</header>
</body>
<script type="text/javascript" th:src="@{'/js/main.js'}"></script>

</html>
<!DOCTYPE html>
<html lang="en" xmlns:th="https://thymeleaf.org">
```

Б қосымшасының жалғасы

```
<head>
    <meta charset="UTF-8">
    <title>Thanks for tour</title>
    <!-- CSS only -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css
" rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
</head>
<body>
    <div th:insert="~{layout/main :: header}" style="margin: 0"></div>
    <main class="container">
        <div class="bg-light p-5 rounded">
            <h1>Спасибо, что выбрали нас!</h1>
            <p class="lead">Мы свяжемся с вами в ближайшее время.</p>
            <a class="btn btn-lg btn-primary" th:href="@{/}"
role="button">Вернуться на главную</a>
        </div>
    </main>

    <!-- JavaScript Bundle with Popper -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.mi
n.js" integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>

<!DOCTYPE html>
<html lang="en" xmlns:th="https://thymeleaf.org">
<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" th:href="@{/toursStatic/css/tourspagestyle.css}">
  <title>Tours</title>
  <!-- CSS only -->
  <link rel="stylesheet" type="text/css" th:href="@{'/css/style.css'}">
  <link rel="stylesheet"
th:href="@{/toursStatic/Travelzakaz/css/tourspagestyle.css}">
  <link rel="stylesheet"
th:href="@{/toursStatic/trCountry/css/tourspagestyle.css}">
</head>
<body>
<div th:insert="~{layout/main :: header}" style="margin: 0"></div>


<!--<div class="container">-->
<!--    <h2 class="activity-tour-title">Активные туры</h2>-->
<!--    <div class="activity-tours">-->
<!--        <div class="activity-tour" th:each="hotel : ${hotels}">-->
```