МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____Скарга-Бандурова І.С.
«____»_____ 2019 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Розроблення інтелектуальної системи продажу квитків

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям 6.050101 – "Комп'ютерні науки"

Керівник проекту:	_____ (підпис)	ст.викл. Лифар О. К. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	ст.викл. Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Севостьянов А.М. (ініціали, прізвище)
Група:		КН-15бд

Сєвєродонецьк 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050101 Комп'ютерні науки
(шифр і назва)
Спеціальність
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри
І.С. Скарга-Бандурова
«_____» _____2019 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Севостьянову Андрію Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення інтелектуальної системи продажу квитків

керівник проекту (роботи) Лифар О. К., ст.викл.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " 14 " 05 2019 р. №
2. Термін подання студентом роботи
3. Вихідні дані до роботи матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз предметної області і постановка задачі.
Розробка інтелектуальної системи продажу квитків.
Опис використаних програмних та технічних засобів.
Результати застосування розробленої програмної системи.
Охорона праці та безпека в надзвичайних ситуаціях.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст.викл. кафедри КНІ Критська Я.О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання до роботи	14.05.19-16.05.19	
2	Аналіз завдання, огляд літератури	17.05.19-20.05.19	
3	Аналіз технічних засобів	21.05.19-24.06.19	
4	Розробка алгоритму	25.06.19-02.06.19	
5	Програмна реалізація	03.06.19-06.06.19	
6	Оформлення пояснювальної записки	07.06.19-09.06.19	
7	Підготовка презентації та доповіді	10.06.19-13.06.19	

Здобувач вищої освіти _____ Севостьянов А.М.
(підпис) (прізвище та ініціали)

Керівник _____ Лифар О.К.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту (роботи) бакалавра: 79 с., 19 рис., 5 табл., 19 бібліографічних джерел посилань, 3 додатки.

Об'єкт розробки: процеси організації продажу квитків.

Мета роботи: розроблення інтелектуальної системи продажу квитків.

В проекті виконано:

1. Огляд методів, підходів, технологій до проектування та розробки інформаційних систем, сформульована постановка задачі.
2. Проектування та розробку бази даних.
3. Вибір засобів розробки програмного забезпечення.
4. Проектування та розробку програмної системи.
5. Тестування роботи системи.
6. Здійснений аналіз потенційних небезпечних і шкідливих виробничих чинників проектованого об'єкта, що впливають на персонал.

Отримано наступні результати: розроблена інформаційна система забезпечує в клієнтській частині швидкий пошук необхідних даних по запиту користувачів, а також здійснення запитів з використанням даних сервера. Структура управління системою реалізована у вигляді трирівневої архітектури «клієнт-сервер додатків – сервер бази даних»: Web-сервер, що виробляє розмежування доступу і розподіляє запити; сервер додатків, що керує бізнес-логікою і реалізує необхідну сукупність процесів; БД і СКБД для збору, зберігання, обробки і управління даними.

Практичне значення, галузь застосування роботи: розроблений сервіс дає можливість розширити та поліпшити якість послуг продажу квитків.

Ключові слова: БАЗА ДАНИХ, МОДЕЛЬ ДАНИХ, C#, MVC, ENTITY FRAMEWORK, AJAX, RAZOR, SQL SERVER.

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А, м. Сєвєродонецьк, 93400

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних

ІС – інформаційна система

ПО – предметна область

СКБД – система керування базами даних

AJAX – Asynchronous Javascript And Xml

CRUD – Create Read Update Delete

CSS – Cascading Style Sheets

DOM – Document Object Model

ER-модель – Essence-Relation-модель

HTML (XHTML) – мова розмітки гіпертексту

MVC – Model-View-Controller

SQL – Structured Query Language

RAZOR – це синтаксис програмування ASP.NET, який використовується для створення динамічних веб-сторінок з мовами програмування C # або Visual Basic .NET.

EF – Entity Framework

ВСТУП

У сучасному світі важлива економія часу тому web-сервіси з продажу квитків є дуже актуальними. Вони мають ряд переваг в порівнянні з покупкою квитків в касі:

– економія часу - немає необхідності їхати в касу і стояти в черзі за квитком, можливо просто замовити його через Інтернет;

– інформація про Вас і Ваш квиток зберігається в надійно захищеній базі даних. Не варто турбуватися про те, що можливо забути, втратити або зіпсувати квиток;

– можливо замовити і оплатити квиток для будь-якої людини з будь-якого міста чи країни, їм тільки залишиться вчасно приїхати з документами;

– електронний квиток коштує дешевше паперового за рахунок вартості самого дорогого захищеного бланка.

Насправді, нинішні рішення на ринку так чи інакше намагаються покрити необхідність сучасної людини в комфортному сервісному обслуговуванні, але всі страждають від тих чи інших недоліків - не всім зрозуміла на інтуїтивному рівні форма для введення даних, відсутність швидкої допомоги користувачеві при виникненні труднощів, відсутність системи, що спрощує вибір квитка зі списку запропонованих, враховуючи переваги користувача.

Пропонована система планується з метою позбутися від усіх вищеописаних недоліків. Треба, щоб у користувача був додаток, за допомогою якого він зможе:

– підібрати оптимальний маршрут при відсутності прямого з'єднання;

– переглянути передбачувано кращий маршрут для користувача в залежності від його переваг в комфорті, швидкості, вартості поїздки і кількості потрібних йому квитків.

1. АНАЛИЗ ВИМОГ ДО ІС ТА РОЗРОБКА КОНЦЕПЦІЇ СИСТЕМИ

1.1 Опис предметної області

Об'єктом автоматизації є компанія, яка веде діяльність з продажу автобусних квитків. А саме відносини касир-клієнт при продажу квитків.

Основна функція квиткового касира - продаж квитків клієнтам. Однак часто фахівець виконує і інші завдання.

Касир зобов'язаний надавати коректну інформацію клієнту про час заходу або відправлення транспорту, наявності вільних місць.

Касир видає квитанції сплати за багаж, проїзні документи, якщо мова йде про поїздку. Перед видачею квитка касир перевіряє документи пасажира, також дозвіл на придбання квитка в спеціальну зону.

Після успішного продажу фахівець фіксує наявність вільних місць і передає інформацію в наступну касу.

Як і будь-який касир, касир квиткової каси здає готівку і замовляє бланки суворої звітності.

Клієнт може повернути квиток за день до відправлення автобусу.

1.2 Опис недоліків

Процес купівлі-продажу квитка є не автоматизованим і тому є повільним. Через що досить часто виникають черги. Робота звичайної каси унеможливлює обслуговування великої кількості людей. Що змушує клієнтів марнувати свій час, приходячи за декілька годин перед відправленням або навіть купувати квиток за декілька днів. Тому необхідна можливість продажу квитків через інтернет сайт.

1.3 Постановка задачі

Система, що розробляється, повинна скоротити час і підвищити якість виконання основних бізнес-процесів компанії, зменшити витрати зусиль, забезпечити оперативне отримання інформації, а також надійне зберігання усіх необхідних даних в одному місці, розширити клієнтську базу.

Необхідно створити систему для продажу автобусних квитків, яка реалізує можливість покупки і бронювання квитків, систему для підбору оптимального маршруту при відсутності прямого з'єднання між пунктами відправлення та прибуття, систему знижок залежно від дня тижня, пори року, кількості пасажирів і наявних у них пільг.

Експлуатаційним призначенням системи є надання інформації про автобусні маршрути і продаж квитків.

Система покупки і бронювання повинна надавати можливість користувачу ознайомитися зі всіма можливими маршрутами в його напрямку і наявністю вільних місць.

При покупці або бронюванні користувач вибирає номер автобуса і місце, доступна можливість вибору декількох місць.

Система підбору оптимального маршруту при відсутності прямого з'єднання повинна надавати, при наявності декількох варіантів, передбачувано кращий для користувача в залежності від його переваг в комфорті, швидкості, вартості поїздки і кількості потрібних йому квитків.

Повинен бути реалізований інтерфейс для адміністратора, який дозволяє додавати, редагувати і видаляти маршрути.

Повинна бути реалізована можливість реєстрації користувача для надання інформації в подальшому в залежності від його побажань.

Можлива реалізація графічного відображення маршрутів.

Визначення вимог до системи.

ІС повинна бути клієнтським додатком, працювати з єдиною базою даних. Користувачи системи:

– адміністратор;

– клієнти.

На рисунку 1.1 представлено варіанти використання системи.

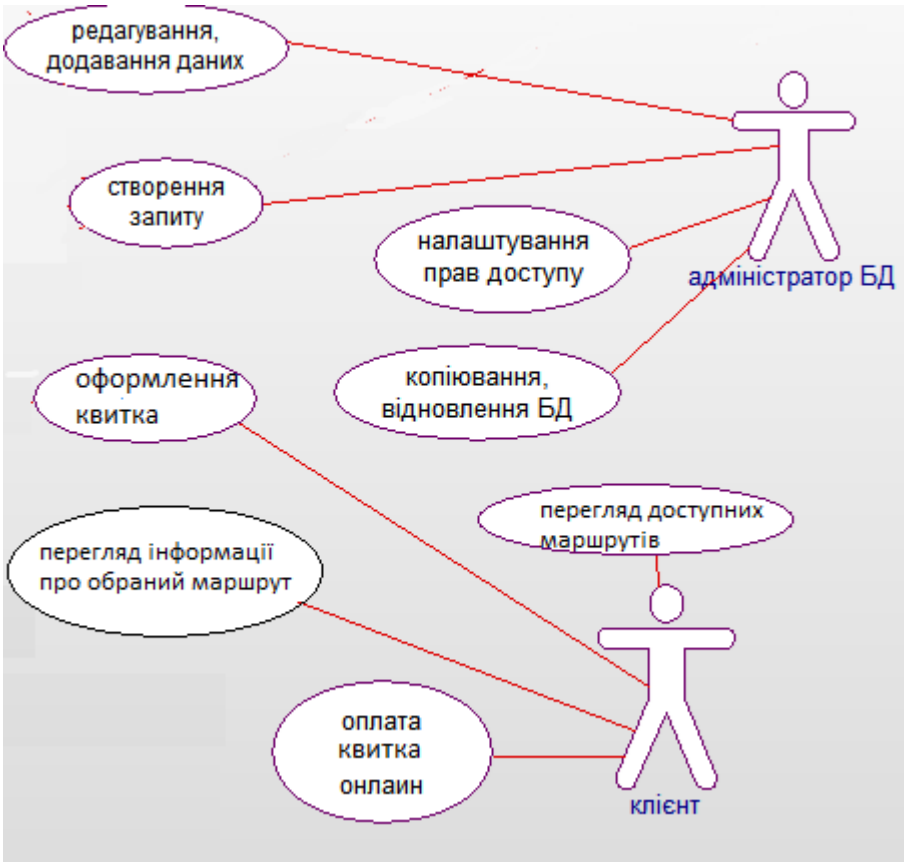

Рисунок 1.1 – Діаграма варіантів використання системи

Виходячи з аналізу діаграми варіантів використання, для кожної з осіб будуть доступні свої функції у додатку. Список функцій, автоматизація яких передбачена ІС :

а) Функції адміністратора:

1) налаштування прав доступу;\
2) відновлення/копіювання бази даних;
3) редагування даних;
4) створення запиту.

б) Функції клієнта:

5) перегляд доступних маршрутів;

6) перегляд інформації про обраний маршрут;

7) оформлення квитка;

8) оплата квитка онлайн.

Загальні вимоги:

– можливість швидкого занесення даних в систему;

– розділення прав доступу до різної інформації;

– редагування облікових записів;

– дружній інтерфейс;

– простота реалізації запитів.

Вимоги до надійності:

– контроль коректності даних, що вводяться;

– виключення ситуацій, пов'язаних одночасним виконанням однієї операції різними користувачами, за допомогою взаємних блокувань;

– захист інформації за рахунок аутентифікації користувачів, організації прав користувачів;

– резервне копіювання інформації, можливість відновлення даних у разі відмови.

2. РОЗРОБКА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ПРОДАЖУ КВИТКІВ

2.1 Методи аналізу предметної області

Загальна постановка задачі та аналіз предметної області – це перший крок проектування. Модель даних «сутність-зв'язок» або ER-модель, запропонована в 1976 П. Ченом і з того часу неодноразово піддавалася перегляду і модифікації як малоефективна модель в проектуванні баз даних, проте діаграма, що будується при проектуванні за допомогою моделі даних «сутність-зв'язок», корисна при проектуванні даталогічної схеми, тому що її наочність допомагає подальшому логічному програмуванні РБД.

«Концептуальне розуміння» предметної області так само, як «концептуальне розуміння» відповідної реляційної бази даних повинні описуватися деяким формалізованим чином. Засобом такого опису становлять інфологічні моделі. Термін «інфологічна модель» може бути застосовано як до моделювання даних, так і до моделювання власне предметної області. Різниця тут може бути дуже великою, хоча іноді ці два рівні моделювання «зливаються» в одне. В інформаційно-логічних моделях акцент робиться на «устрій» предметної області та «логіку» її функціонування, тоді як інформаційні аспекти відіграють лише допоміжну роль. Інфологічні моделі даних призначені для вирішення куди більш скромних завдань, що зводяться до побудови ефективної в деякому сенсі сукупності даних, що зберігаються, і процесів їх обробки. Інфологічна модель застосовується на етапі передпроектних досліджень, системного аналізу та формулювання концепції інформаційної системи.

У моделі виділяються два види об'єктів – сутності та зв'язки. Об'єкти наділяються властивостями, причому сутності розглядаються як агрегати властивостей, а зв'язки – як агрегати сутностей і, може бути, власних

властивостей. Межа між цими категоріями часом виявляється вельми розмитою.

Основні правила структурування в ER- моделі (ER-діаграмі):

а) властивість завжди пов'язана з одним певним типом об'єктів, тобто одна і та ж властивість не може характеризувати кілька типів (якщо тільки вони не є підлеглими загальному типу);

б) одні й ті ж об'єкти-сутності можуть агрегуватися в будь-яку кількість зв'язків;

в) сутність може агрегувати в деякому зв'язку кілька разів. У цьому випадку кожне входження сутності в зв'язок характеризується її роллю в цьому зв'язку;

г) зв'язки можуть агрегувати в собі будь-яку кількість сутностей (рівне або більше двох). Зв'язки не можуть агрегувати зв'язки.

Властивості в ER-моделі можуть бути простими (подібно атрибутам реляційної моделі даних) або агрегатними (агрегат властивостей). Властивості можуть бути однозначними, що мають одне скалярне значення для кожного об'єкта, або багатозначними - мають кілька скалярних значень. Аналогом домену реляційної моделі даних служить безліч (скалярних) значень. Кожна властивість асоціюється з типом сутності або зв'язку шляхом завдання відображення від об'єкта до безлічі значень.

Найбільш поширеним видом представлення схеми БД в ER-моделі є ER-діаграма. Це граф з трьома видами вершин: вершини-прямокутники позначають типи сутностей, вершини-ромби – типи зв'язків, вершини-овали – безлічі значень (домени). Усередині вершини записується відповідне ім'я. Зазвичай фіксується два види іменованих ребер. Ребра «сутність-зв'язок» позначають відповідну агрегацію та ім'я ребра представляє роль сутності у зв'язку. Ребра «об'єкт-безліч значень» відповідають властивостям об'єкта. Ребра-властивості можуть «гілкуватися», утворюючи в загальному випадку спрямований граф, що показує ієрархію імен агрегатів властивостей.

Подвійним ребром на діаграмі позначається багатозначна властивість. При відображенні кожен об'єкт представляється реляційної таблицею.

Відповідно до визначення, даного Ченом, об'єктом називається «предмет, який може бути чітко ідентифікований». При цьому об'єкти поділяються на правильні і слабкі об'єкти. Слабким об'єктом називається об'єкт, який знаходиться в залежності від деякого іншого об'єкта, тобто він не може існувати, якщо не існує цей інший об'єкт. Сильним називається об'єкт, який не є слабким.

Об'єкти (і відносини) мають деякі властивості. Всі об'єкти одного типу володіють деякими загальними властивостями. Деякі види властивостей та їх особливості:

– проста чи складена властивість;
– ключова властивість (унікальна в певному контексті);
– однозначна або багатозначна властивість;
– відсутня властивість («невідоме» або «не застосовується»);
– базова або похідна властивість.

Відносини в моделі сутність-зв'язок можуть мати тип один-до-одного, один-до-багатьох, багато-до-одного, багато-до-багатьох.

Кожен тип відношення показаний у вигляді ромба з назвою відносини всередині. Ромб оточений подвійною лінією, якщо відношення задано між слабким типом об'єкту і типом об'єкта, від існування якого знаходиться в залежності слабкий тип об'єкта.

Учасники кожної відносини приєднані до відповідного відношення суцільними лініями. Кожна така лінія містить напис «1» або «N» для позначення типу відношення. Подвійна лінія позначає повну участь.

2.2 Опис діаграми «сутність – зв'язок»

Модель «сутність-зв'язок» (ER-модель) – модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків.

ER-модель – це мета-модель даних, тобто засіб опису моделей даних. Вона зручна при проектуванні інформаційних систем, баз даних, архітектур комп'ютерних додатків та інших систем (моделей). За допомогою такої моделі виділяють найсуттєвіші елементи (вузли, блоки) моделі і встановлюють зв'язки між ними.

Існує ряд моделей для представлення знань. Одним з найзручніших інструментів уніфікованого представлення даних, незалежного від реалізовуючого його програмного забезпечення, є модель «сутність-зв'язок».

Модель «сутність-зв'язок» ґрунтується на якійсь важливій семантичній інформації про реальний світ і призначена для логічного представлення даних. Вона визначає значення даних в контексті їх взаємозв'язку з іншими даними.

Важливим є той факт, що з моделі «сутність-зв'язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об'єктна), тому вона є найбільш загальною. Будь-який фрагмент наочної області може бути представлений як безліч сутностей між якими існує деяка безліч зв'язків.

Оптимальна модель даних повинна задовольняти критеріям, представленим в таблиці 2.1. Проте іноді ці критерії несумісні, тому доводиться йти на деякий компроміс. Наприклад, в гонитві за найбільшою виразністю моделі даних можна втратити її простоту.

Таблиця 2.1 – Критерії оцінки моделі даних

Критерій	Опис
Структурна достовірність	Відповідність способу визначення і організації інформації на даному підприємстві
Простота	Зручність вивчення моделі як професіоналами в області розробки інформаційних систем, так і звичайними користувачами
Виразність	Здатність представляти відмінності між даними, зв'язки між даними і обмеження
Відсутність надмірності	Виключення зайвої інформації, тобто будь-яка частина даних повинна бути представлена тільки один раз
Цілісність	Узгодженість із способом використання і управління інформацією усередині підприємства
Здатність до спільного використання	Відсутність приналежності до якогось особливого застосування або технології і, отже, можливість використання моделі у багатьох застосуваннях і технологіях
Розширюваність	Здатність розвиватися і включати нові вимоги з мінімальною дією на роботу вже існуючих застосувань
Схематичне представлення	Можливість представлення моделі за допомогою наочних схематичних позначень

ER-модель – це одна з найбільш простих візуальних моделей. Вона дозволяє осягнути структуру об'єкта «крупними мазками», в загальних рисах. Такий загальний опис структури називається ER-діаграмою або онтологією вибраної предметної області, яка представлена на рисунку 2.1.

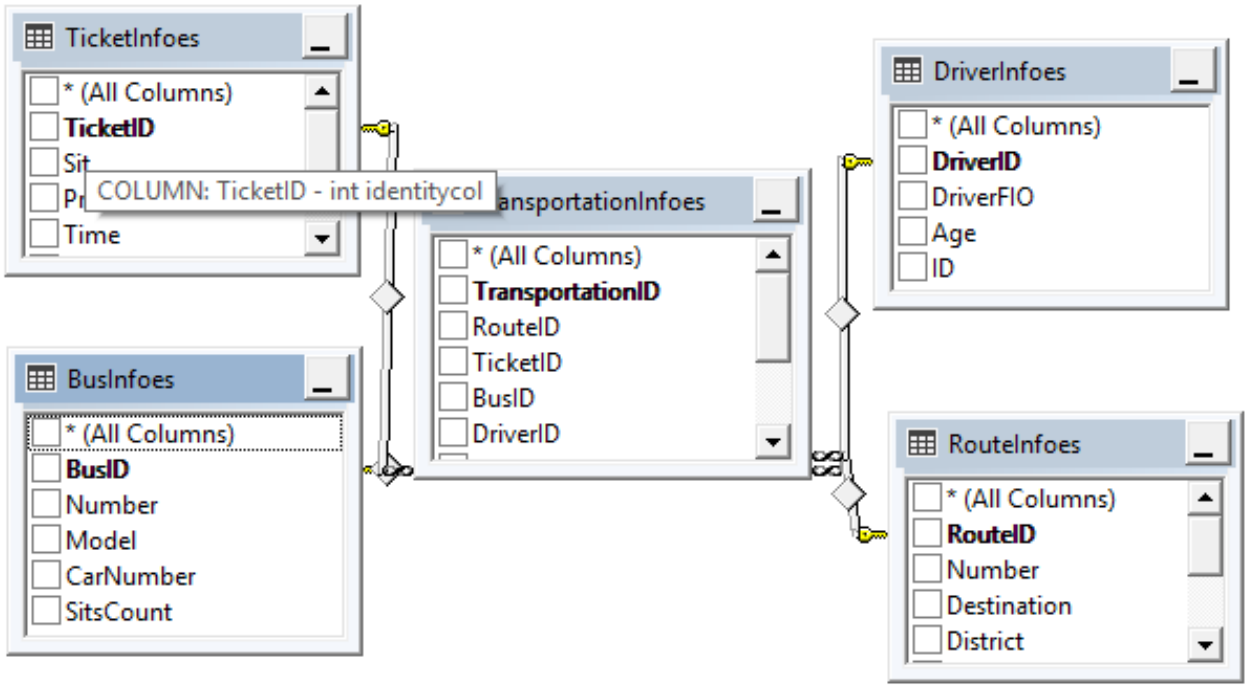

Рисунок 2.1 – Схема БД

TransportationInfoes – основна таблиця бази даних, яка має такі атрибути: TransportationID, RouteID, TicketID, BusID, DriverID. Містить інформацію про усі маршрут та додаткову інформацію. Формується та доповнюється адміном.

DriverInfoes – таблиця бази даних, яка має такі атрибути: DriverID, DriverFIO, Age, ID. Містить інформацію про водія.

BusInfoes – таблиця бази даних, яка має такі атрибути: BusID, Number, Model, CarNumber, SitsNumber. Містить інформацію про автобус.

RouteInfoes – таблиця бази даних, яка має такі атрибути: RouteID, Number, Destination, district, ArrivalTime, DestinationTime. Містить інформацію про маршрут.

TicketInfoes – таблиця бази даних, яка має такі атрибути: TicketID, Sit, Price, Time, PassagerID. Містить інформацію про клієнта.

2.3 Створення відношень за допомогою мови SQL

Мова SQL орієнтована на виконання дій з таблицями БД і даними в цих таблицях, а також деяких допоміжних дій. На відміну від процедурних мов програмування, в ній немає операторів управління, обчислювальних процесів і засобів вводу/виводу. SQL програму називають SQL-запитом.

Відношення можуть бути подані прямокутною таблицею, рядки якої відповідають кортежам відношень, а стовпці її атрибутам [6]. У таблицях, як завжди, виділяється заголовок і тіло. У заголовку таблиці розміщують імена атрибутів, в тілі – значення атрибутів, тобто самі дані.

Для створення таблиць служить оператор CREATE TABLE, який має наступний формат:

CREATE TABLE ім’я таблиці

({<визначення стовпця>} [, …n]

[<обмеження таблиці> [, …n]])

Визначення стовпця містить:

а) ім'я стовпця;

б) тип даних або AS <вирази для обчислення>;

в) наявність лічильника;

г) обов'язковість значень (NULL | NOT NULL);

д) обмеження стовпця:

1) значення за умовчанням (DEFAULT);

2) унікальність значень (PRIMARY KEY | UNIQUE) (цілісність сутностей);

3) значення, що посилається (REFERENCES) (посилальна цілісність);

4) обмеження, задане логічним виразом (CHECK) (обмеження домена, корпоративні обмеження).

Обмеження таблиці (накладаються відразу на декілька стовпців):

а) унікальність значень (PRIMARY KEY | UNIQUE) (цілісність сутностей);

б) значення, що посилається (REFERENCES) (посилальна цілісність);

в) обмеження, задані логічним виразом (CHECK) (обмеження домена, корпоративні обмеження).

2.4 Заповнення бази даних

Таблиці БД є фізичними об'єктами. Для операції з даними, що містяться в таблиці, використовуються набори даних.

У термінах системи SQL набір даних представляє собою сукупність записів, взятих з однієї або декількох таблиць. Набір даних є логічною таблицею, з якою можна працювати при виконанні програм. Взаємодія таблиці і набору даних нагадує взаємодія фізичного файлу, файлової змінної.

Таблиці заповнюються відповідно до їхніх полів і обмежень, які встановили. Якщо встановили маску вводу, то дані будуть вводитися відповідно з цією маскою, якщо встановили обмеження до кількості символів, то дані будуть вводитися відповідно до цієї кількості, а якщо встановили, що записів, які повторюються, не повинно бути, то при їх наявності буде видаватися помилка. Таблиці даної БД були заповнені відповідним чином, з урахуванням всіх обмежень.

2.5 Створення запитів в розподіленій базі даних

За допомогою оператора SELECT дані можна вибирати з однієї або декількох таблиць [7].

Управління полями складається у вказівці полів таблиці (таблиць), що включаються в результуючий набір даних. Як зазначалося вище, при відборі

всіх полів таблиці, умови відбору записів не вказуються якщо замість списку полів вказати «*», то в наборі даних виявляються всі поля записів. При цьому можна не замислюватися про назви полів записів. Порядок проходження в наборі даних відповідає порядку проходження фізичних полів таблиці, визначеному при її створенні.

Оператор SELECT – найважливіший оператор мови SQL. Він використовується для відбору записів, що задовольняють складним критеріям пошуку.

Результат виконання SQL-запиту, заданого оператором SELECT, являє собою вибірку записів, що відповідають заданим у ньому умовам.

Спрощений вигляд команди SELECT наступний:

SELECT [DISTINCT] {<стовпець> [AS] {<замінник>}, ...}

FROM [<схема.] таблиця [@<зв’язок _БД>]

WHERE <критерій_пошуку>

ORDER BY {<стовпець>, <вираз>, ...} [ASC | DESC];

Оператор WHERE задає умови (критерії) відбору, яким повинні задовольняти записи в результуючому наборі даних. Вираз, що описує умови відбору, є логічним. Його елементами можуть бути імена полів, операції порівняння, арифметичні і логічні операції, дужки, спеціальні функції LIKE, NULL, IN та інше.

Операнд GROUP BY дозволяє виділяти групи записів з однаковими значеннями в полях.

Операнд HAVING діє спільно з операндом GROUP BY і використовується для відбору записів всередині груп. Правила запису умов групування аналогічні правилам формування умов відбору в операнді WHERE.

Оператор ORDER BY містить список полів, що визначають порядок сортування записів результуючого набору даних. За замовчуванням сортування по кожному полю виконуються у порядку зростання значень;

якщо необхідно задати для поля сортування за спаданням, то після імені цього поля вказується описувач DESC.

Оператор SELECT використовується також всередині інших операторів, наприклад операторів модифікації записів, забезпечуючи для їх виконання необхідний відбір записів.

Критерієм відбору записів є логічне вираження, в якому можна використовувати операції: порівняння, "=" – рівності, ">" – більше, "<" – менше, "> =" – більше або дорівнює, "<=" – менше або дорівнює, "<>" або, "! =" – нерівно, "!>" – не більше, "! <" – не менше, "LIKE" – порівняння за шаблоном, "IS NULL" – перевірка на нульове значення, " IN "– перевірка на входження," BETWEEN "– перевірка на входження в діапазон.

Складний критерій (логічний вираз) складається з простих умов, круглих дужок, логічних операцій: AND – логічне і, OR – логічне або, NOT – логічне ні.

Записи набору даних можуть бути згруповані за певною ознакою. Групу утворюють записи з однаковими значеннями в полях, перелічених у списку операнда GROUP BY. При групуванні записів їх простіше аналізувати і обробляти, наприклад, за допомогою статистичних функцій.

Групування записів автоматично виключає повтор у полях, заданих для групування, так як записи з однаковими значеннями цих полів об'єднуються в одну групу.

3 ОПИС ВИКОРИСТАНИХ ПРОГРАМНИХ ТА ТЕХНІЧНИХ ЗАСОБІВ

3.1 Огляд мови запитів SQL

SQL – декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних і її модифікація, система контролю за доступом до бази даних. Сам по собі SQL не є ні системою керування базами даних, ні окремим програмним продуктом. Не будучи мовою програмування в тому розумінні, як C або Pascal, SQL може формувати інтерактивні запити або, будучи вбудованою в прикладні програми, виступати в якості інструкцій для керування даними. Стандарт SQL, крім того, вміщує функції для визначення зміни, перевірки і захисту даних [7].

SQL – це діалогова мова програмування для здійснення запиту і внесення змін до бази даних, а також управління базами даних. Багато баз даних підтримує SQL з розширеннями до стандартної мови. Ядро SQL формує командна мова, яка дозволяє здійснювати пошук, вставку, обновлення, і вилучення даних, використовуючи систему управління і адміністративні функції. SQL також включає CLI (Call Level Interface) для доступу і управління базами даних дистанційно.

Перша версія SQL була розроблена на початку 1970-х років у IBM. Ця версія носила назву SEQUEL і була призначена для обробки і пошуку даних, що містилися в реляційній базі даних IBM, System R. Мова SQL пізніше була стандартизована Американськими Держстандартами (ANSI) в 1986. Спочатку SQL розроблялась як мова запитів і управління даними, пізніші модифікації SQL створено продавцями системи управління базами даних, які додали процедурні конструкції, control-of-flow команд і розширення мов. З

випуском стандарту SQL:1999 такі розширення були формально запозичені як частина мови SQL через Persistent Stored Modules (SQL/PSM).

На початку 1970-х років в одній з дослідницьких лабораторій компанії IBM була розроблена експериментальна реляційна СКБД System R, для якої потім була створена спеціальна мова SEQUEL, що дозволяла відносно просто управляти даними в цій СКБД [8]. Абревіатура SEQUEL розшифровувалася як Structured English QUEry Language – «структурована англійська мова запитів». Пізніше з юридичних міркувань мова SEQUEL була перейменована в SQL.

Метою розробки було створення простої не процедурної мови, якою зміг би скористатися будь-який користувач, що навіть не має навиків програмування. Власне розробкою мови запитів займалися Дональд Чемберлін (Donald D. Chamberlin) і Рей Бойс (Ray Boyce). Пет Селінджер (Pat Selinger) займалася розробкою вартісного оптимізатора, Реймонд Лорі (Raymond Lorie) займався компілятором запитів.

Незважаючи на наявність діалектів і відмінностей в синтаксисі, в більшості своїй тексти SQL-запитів, що містять, DDL і DML, можуть бути досить легко перенесені з однієї СКБД в іншу.

За допомогою SQL програміст описує тільки те, які дані потрібно витягнути або модифікувати. Те, яким чином це зробити, вирішує СКБД безпосередньо при обробці SQL-запиту. Проте не варто думати, що це повністю універсальний принцип – програміст описує набір даних для вибірки або модифікації, проте йому при цьому корисно уявляти, як СКБД розбиратиме текст його запиту. Особливо критичні такі моменти стають при роботі з великими базами даних і зі складними запитами – чим складніше сконструйований запит, тим більше він допускає варіантів написання, різних за швидкістю виконання, але тих самих за набором даних.

Автор реляційної моделі даних Едгар Кодд, Крістофер Дейт та їхні прихильники вказують на те, що SQL не є істинно реляційною мовою [7].

Зокрема вони вказують на такі проблеми SQL:

– рядки, що повторюються;
– невизначені значення (null);
– явна вказівка порядку стовпчиків зліва направо;
– стовпці без імені та імена стовпчиків, що дублюються;
– відсутність підтримки властивості «=»;
– використання вказівників;
– висока надлишковість.

3.2 Загальні відомості про C#

C# – об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою Microsoft Research (при фірмі Microsoft).

Синтаксис C# близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато що від своїх попередників — мов C++, Delphi, Модула і Smalltalk — C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад множинне спадкування класів (на відміну від C++).

Особливості мови C#:

а)Портативність.

C# розроблялась як мова програмування прикладного рівня для CLR і тому вона залежить від можливостей самої CLR. Це стосується, перш за все, системи типів C#. Присутність або відсутність тих або інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльована у відповідні конструкції CLR. Так, з розвитком CLR від

версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі. Проте ця закономірність буде порушена з виходом C# 3.0, що є розширеннями мови, які не спираються на розширення платформи .NET. CLR надає C#, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C#, а проводиться CLR для програм, написаних на C# точно так, як і це робиться для програм на VB.NET, J# тощо.

б)Типи даних .

C# підтримує строго типізовані неявні оголошення змінних з ключовим словом var і неявно типізовані масиви з ключовим словом new [], за яким слідує ініціалізатор колекції.

C# підтримує суворий тип даних Boolean, bool. Вирази, які приймають умови, такі як while та if, вимагають висловлювання, що реалізує оператор true або false. Хоча C++ також має тип Boolean, він може бути вільно перетворений в цілі числа та навпаки, а вирази, такі як if(a), вимагають тільки того, щоб a був конвертований в bool, що дозволяє бути a int-типу або вказівником. C# забороняє «ціле значення означає справжній або помилковий підхід» на тій підставі, що примус програмістів використовувати вирази, які повертають точно bool, можуть створювати деякі типи помилок програмування, наприклад if (a = b)(використання присвоювання = замість рівності == , які, хоча і не є помилкою на C або C++, все одно будуть спіймані компілятором).

C# безпечніший в порівнянні з C++. Єдиними неявними перетвореннями за замовчуванням є ті, які вважаються безпечними, наприклад, розширення цілих чисел. Це застосовується під час компіляції, під час JIT і, в деяких випадках під час виконання. Не відбувається неявних перетворень між булевими і цілими числами, а також між членами перерахування і цілими числами (крім літерала 0, який може бути неявно перетворений в будь-який нумерований тип). Будь-яке призначене для

користувача перетворення повинно бути явно позначене як явне або неявне, на відміну від конструкторів копіювання C++ і операторів перетворення, які за умовчанням є неявними.

C# має явну підтримку коваріації та контраваріантності в родових типах, на відміну від C++, яка має певний рівень підтримки контраваріантності просто через семантику типів, що повертаються, на віртуальні методи.

Мова C # не допускає глобальних змінних або функцій. Всі методи і члени повинні бути оголошені всередині класів. Статичні члени відкритих класів можуть замінювати глобальні змінні та функції.

в)Метапрограмування.

Метапрограмування через атрибути C# є частиною мови. Багато з цих атрибутів дублюють функціональні можливості директив препроцесора, орієнтованих на платформу GCC і VisualC ++.

г)Методи та функції.

Методи в мові програмування є членами класу в проекті, деякі методи мають підписи, а деякі не мають підпису. Методи можуть бути недійсними або можуть повертати щось на зразок рядка, цілого, подвійного, десяткового, float і bool. Якщо метод недійсний, це означає, що метод не повертає жодного типу даних.

Подібно C++, і на відміну від Java, програмісти на C# повинні використовувати ключове слово virtual, щоб дозволити перевизначати методи підкласами.

Методи розширення в C# дозволяють програмістам використовувати статичні методи, як якщо б вони були методами з таблиці методів класу, дозволяючи програмістам додавати методи до об'єкта, який, на їхню думку, повинен існувати на цьому об'єкті і його похідних.

Динамічний тип dynamic допускає прив'язку методу під час виконання, що дозволяє використовувати JavaScript-подібні виклики методів і склад часу виконання.

У C# є підтримка строго типізованих покажчиків функцій через delrgate ключового слова. Подібно псевдо-C++ - signal і slot фрейма Ct, C# має семантику, спеціально пов'язану з подіями стилю публікації-підписки, хоча C# використовує делегати для цього.

C# пропонує Java-подібні синхронізовані synchronized виклики методів через атрибут [MethodImpl (MethodImplOptions.Synchronized)] і підтримує взаємовиключні блокування за допомогою блокування ключових слів.

3.3 HTML – мова розмітки гіпертексту

HTML – стандартна мова розмітки документів у Всесвітній павутині. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді [10].

HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документу та ідеологію структурної розмітки тексту.

HTML разом із CSS та скриптингом – це три основні технології побудови веб-сторінок [11].

Веб-сторінка – інформаційний ресурс доступний в мережі World Wide Web (Всесвітня павутина), який можна переглянути у веб-браузері зазвичай, ця інформація записана в форматі HTML або XHTML, і може містити гіпертекст з навігаційними гіперпосиланнями на інші веб-сторінки. Веб-сторінки можуть зберігатись на локальному комп'ютері або отримуватись із віддаленого веб-сервера.

HTML впроваджує засоби для:

– створення структурованого документу шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;

– отримання інформації із Всесвітньої мережі через гіперпосилання;

– створення інтерактивних форм;

– включення зображень, звуку, відео та інших об'єктів до тексту.

3.4 CSS – каскадні таблиці стилів

Каскадні таблиці стилів – це спеціальна мова, що використовується для відображення сторінок, написаних мовами розмітки даних. Найбільш часто CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів [11]. Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої павутини.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже існуючи функції. Рівні позначаються як CSS1, CSS2, CSS3 та CSS4.

Профілі – сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, смартфонів тощо.

CSS (каскадна або блочна верстка) прийшла на заміну табличній верстці веб-сторінок. Головна перевага блочної верстки – розділення змісту сторінки (даних) та їх візуальної презентації [11].

Переваги каскадних таблиць стилів:

– інформація про стиль для цілого сайту або його частин може міститися в одному .css-файлі, що дозволяє швидко робити зміни в дизайні та презентації сторінок;

– різна інформація про стилі для різних типів користувачів: наприклад великий розмір шрифту для користувачів з послабленим зором, стилі для виводу сторінки на принтер, стиль для смартфонів;

– сторінки зменшуються в об'ємі та стають більш структурованими, за рахунок того, що інформація про стилі відділена від тексту та має певні правила застосування і сторінка побудована з їх урахуванням;

– прискорення завантаження сторінок і зменшення обсягів інформації, що передається, навантаження на сервер та канал передачі. Досягається за рахунок того, що сучасні браузери здатні кешувати (запам'ятовувати) інформацію про стилі і використовувати для всіх сторінок, а не завантажувати для кожної.

3.5 ASP.NET MVC Framework

ASP.NET MVC Framework — фреймворк для створення веб-додатків, який реалізує шаблон Model-view-controller. Цей фреймворк доданий Microsoft в ASP.NET. Він виконується у веб браузері та надає допомогу у створенні сучасних, односторінкових веб-додатків, які використовують технологію AJAX (Asynchronous Javascript And Xml) [14].

Драйвери перегляду, що використовуються в рамках ASP.NET MVC 3 та MVC 4, - це Razor та Web Forms. Обидва двигуни перегляду є частиною структури MVC 3. За замовчуванням, движок перегляду в рамках MVC використовує Razor .cshtml та .vbhtml або веб-форми .aspx-сторінки для розробки макета сторінок інтерфейсу користувача, на які створені дані. Проте різні двигуни перегляду можуть бути використані. Крім того, замість типової поштової моделі ASP.NET Web Forms, будь-які взаємодії спрямовуються до контролерів за допомогою механізму маршрутизації ASP.NET. Перегляди можуть бути віднесені до різних URL-адрес.

Інші двигуни перегляду:

Бібліотека MVCContrib містить 8 альтернативних движків перегляду: Brail, NDjango, NHaml, NVecocity, SharpTiles, Spark, StringTemplate і XSLT.

Двигун StringTemplate View Engine використовує порт .NET для шаблону Java, StringTemplate.

Іскра - це механізм перегляду для ASP.NET MVC (та проекту Castle Project MonoRail) .

NDjango є портом мови шаблону веб-структури Django для .NET. Вона написана на F # і поставляється з розширенням Visual Studio, включаючи повну підтримку Intellisense

Архітектурний шаблон Model-View-Controller (MVC) відділяє програму на три основні компоненти: модель, вид та контролер. Рамки ASP.NET MVC забезпечують альтернативу шаблону ASP.NET Web Forms для створення веб-додатків. Рамки ASP.NET MVC - це легка, дуже перевірена система презентації, яка, як і з додатками на базі Web Forms, інтегрована з існуючими функціями ASP.NET, такими як майстер-сторінки та автентифікація на основі членства. Рамки MVC визначені в збірці System.Web.Mvc.

Шаблон дизайну контролера моделі View

MVC - це стандартний шаблон дизайну, з яким знайомі багато розробників. Деякі типи веб-додатків отримають користь від системи MVC. Інші продовжуватимуть використовувати традиційний шаблон додатків ASP.NET, який базується на веб-формах та зворотних відсилах. Інші типи веб-додатків об'єднають два підходи; жоден підхід не виключає іншого.

Моделі. Модельні об'єкти є частинами програми, які реалізують логіку для домену даних програми. Часто модельні об'єкти отримують та зберігають стан моделі в базі даних. Наприклад, об'єкт Product може отримати інформацію з бази даних, працювати на ньому, а потім оновлювати інформацію знову на таблицю Products у базі даних SQL Server.

У невеликих програмах модель часто є концептуальним розділенням, а не фізичним.

Перегляди - це компоненти, що відображають користувацький інтерфейс програми (інтерфейс користувача). Як правило, цей інтерфейс створюється з даних моделі.

Контролери. Контролери - це компоненти, які обробляють взаємодію з користувачем, працюють з моделлю, і, в остаточному підсумку, вибирають представлення, яке відображає інтерфейс користувача. У програмі MVC у представленні відображається інформація; контролер обробляє та реагує на вхід користувача та взаємодію. Наприклад, контролер обробляє значення рядка запиту і передає ці значення до моделі, яка, у свою чергу, може використовувати ці значення для запиту до бази даних.

Шаблон MVC допомагає створювати програми, що відокремлюють різні аспекти програми (логіка введення, бізнес-логіка та логіка інтерфейсу користувача), одночасно забезпечуючи вільний зв'язок між цими елементами. Шаблон визначає, де кожний вид логіки має бути розташованим у додатку. Логіка інтерфейсу користувача належить до вигляду. Вхідна логіка належить контролеру. Бізнес-логіка належить до моделі.

Розрив зв'язку між трьома основними компонентами додатка MVC також сприяє паралельному розвитку. Наприклад, один розробник може працювати над виглядом, другий розробник може працювати над логікою контролера, а третій розробник може зосередитись на бізнес-логіці моделі.

Крім керування складністю, шаблон MVC спрощує тестування додатків. Наприклад, у веб-застосунку ASP.NET на базі Web Forms один клас використовується як для відображення виводу, так і для відповіді на введення користувача. Написання автоматизованих тестів для додатків ASP.NET на основі веб-форм може бути складним, оскільки для тестування окремої сторінки ви повинні екзаменувати клас сторінки, всі його дочірні елементи та додаткові залежні класи у програмі. Оскільки так багато класів створюються для запуску сторінки, може бути важко писати тести, які зосереджені виключно на окремих частинах програми. Тому тести для додатків ASP.NET на основі веб-форм можуть бути складнішими, ніж тести в додатку MVC. Крім того, для тестів у веб-формах на базі додатків ASP.NET потрібен веб-сервер. Рамки MVC відокремлюють компоненти та значно

використовують інтерфейси, що дає змогу тестувати окремі компоненти окремо від решти структури.

Треба ретельно враховувати, чи слід реалізовувати веб-додатки, використовуючи або ASP.NET MVC, або модель ASP.NET Web Forms. Рамка MVC не замінює модель Web Forms; можливо використовувати будь-яку структуру для веб-додатків.

Перш ніж прийняти рішення використовувати модель MVC або веб-форми для певного веб-сайту треба зважити переваги кожного підходу.

Рамки ASP.NET MVC мають такі переваги:

– Полегшує управління складністю, розділяючи додаток на модель, на вигляд та на контролер.
– Не використовує стан перегляду або серверні форми. Це робить концепцію MVC ідеальною для розробників, які хочуть повний контроль за поведінкою програми.
– Використовує шаблон Front Controller, який обробляє запити веб-додатків за допомогою одного контролера. Це дає змогу спроектувати програму, яка підтримує багату інфраструктуру маршрутизації. Це забезпечує кращу підтримку для тестування (TDD).
– Добре працює для веб-додатків, які підтримуються великими групами розробників та веб-дизайнерами, які потребують високого рівня контролю за поведінкою програми.

На основі Web Forms є наступні переваги:

– Підтримує модель події, яка зберігає стан через HTTP, що дає перевагу розробці веб-додатків для бізнесу. Програма на основі веб-форм надає десятки подій, які підтримуються в сотнях серверних елементів керування.
– Використовує шаблон контролера сторінки, який додає функціональність до окремих сторінок. Щоб отримати додаткові відомості.

- Використовує стан перегляду на серверних формах, що полегшує керування інформацією про стан.
- Добре працює для невеликих команд веб-розробників і дизайнерів, які хочуть скористатися великою кількістю компонентів, доступних для швидкого розробки додатків.

Особливості ASP.NET MVC Framework

Рамки ASP.NET MVC забезпечують такі функції:

Розділення завдань застосування (вхідна логіка, бізнес-логіка та логіка інтерфейсу користувача), тестованість і тестування (TDD). Всі основні контракти в рамках MVC є інтерфейсом і можуть бути перевірені за допомогою макета об'єктів, які імітують поведінку реальних об'єктів у додатку. Можливо протестувати додаток без необхідності запускати контролери в процесі ASP.NET, що робить пристрій тестуванням швидким та гнучким. Можливо використовувати будь-які модулі тестування, сумісні з .NET Framework.

Розширювана і підключена система. Компоненти структури ASP.NET MVC розроблені таким чином, що їх можна легко замінити або налаштовувати. Можливо підключити свій власний движок перегляду, політику маршрутизації URL-адреси, серіалізацію параметрів методу дії та інші компоненти. ASP.NET MVC Framework також підтримує використання контейнерних моделей Injection Dependency Injection (DI) та Inversion of Control (IOC). DI дозволяє вставляти об'єкти в клас, а не покладатися на клас, щоб створити сам об'єкт. МОК вказує, що якщо об'єкт вимагає іншого об'єкта, перші об'єкти повинні отримати другий об'єкт із зовнішнього джерела, такого як файл конфігурації. Це спрощує тестування.

Широка підтримка маршрутизації ASP.NET, яка є потужним компонентом відображення URL-адреси, що дозволяє створювати програми, які мають зрозумілі та доступні для пошуку URL-адреси.

Підтримка використання розмітки у існуючих сторінках ASP.NET (файли .aspx), файли розмітки для користувача (файл .ascx) та головної

сторінки (.master файли) як шаблони перегляду. Можливо використовувати існуючі функції ASP.NET за допомогою ASP.NET MVC-структури, наприклад, вкладені майстер-сторінки, вбудовані вирази (<% =%>), декларативні елементи керування серверами, шаблони тощо.

Підтримка існуючих функцій ASP.NET. ASP.NET MVC дозволяє використовувати такі функції, як автентифікація форм, автентифікація Windows, авторизація URL-адреси, членство та ролі, вивід та кешування даних, сеанс і режим управління профілем, моніторинг здоров'я, система конфігурації та архітектура постачальника.

3.6. Робота з базами даних в SQL SERVER

SQL SERVER відноситься до СУБД, що підтримують активні правила. Активні СУБД об'єднують технології баз даних з програмуванням «на логічних правилах» для додання до баз даних можливості реагування на деякі (можливо, зовнішні) стимули, звані подіями.

Активність - це здатність БД виконувати деякі дії над даними опосередковано, без явного втручання користувача.

Притаманні активним базам даних можливості реагування на події можна використовувати в широкому спектрі додатків, що включають безпеку, матеріалізацію представлень, перевірку цілісності, а також інтеграцію гетерогенних баз даних. Дослідження в області активних баз даних ведуться з кінця 80-х р.р.

Компоненти СУБД MS SQL Server реалізується у вигляді кількох самостійних служб, кожна з яких відповідає за виконання певного кола завдань [6]:

1. MS SQL Server;
2. SQL Server Agent;
3. Microsoft Search (MSSearch);

4. Microsoft Distributed Transaction Coordinator (MS DTC).

Основне ядро SQL Server запускається як служба операційної системи MS SQL Server. Ця служба реалізує більшість основних функцій SQL Server: виконання процедур, що зберігаються, управління файлами баз даних і журналу транзакцій, перевірку облікових записів користувачів, виконання запитів і команд Transact-SQL. Якщо служба MS SQLServer не запущено, то користувачі не зможуть підключитися до баз даних і ніякі адміністративні завдання не зможуть бути виконані.

Служба SQL Server Agent реалізує частину SQL Server, що відповідає за автоматичне виконання завдань і повідомлення операторів про помилки в роботі сервера. SQL Server Agent можна розглядати як службу, залежну по відношенню до MS SQL Server. Для успішної роботи SQL Server запуск служби SQL Server Agent не обов'язковий.

Компонент Microsoft Search, реалізований у вигляді служби MS Search, дозволяє реалізувати пошук символьної інформації в полях таблиць баз даних. Можливості пошуку були значно перероблені в порівнянні з попередніми версіями SQL Server. Microsoft Search реалізує підтримку повнотекстових (full-text) каталогів та індексів, створених для реалізації пошуку текстової інформації в базі даних. За допомогою цієї служби можна здійснювати пошук слів чи фраз в таблицях баз даних. В результуючій множині будуть відображені форми дієслів та іменників, які відміняються. Можна здійснювати пошук слів чи фраз, що стоять близько по відношенню один до одного.

Повнотекстові каталоги не зберігаються за допомогою баз даних. Вони реалізуються у вигляді окремих файлів, до яких звертається тільки Microsoft Search. Це означає, що вони не зберігаються разом з базою даних в процесі резервного копіювання. Повнотекстові каталоги повинні архівуватися і відновлюватися окремо. Зв'язок між SQL Server і Microsoft Search реалізується через спеціального full-text-постачальника.

Управління розподіленими транзакціями. Використання MS DTC дозволяє в єдиній транзакції використовувати кілька джерел даних. При відкаті транзакції служба MS DTC відновлює початкові значення на всіх віддалених і локальних джерелах інформації.

3.7 Методи побудови та організації інтелектуальних систем

3.7.1 Принципи побудови та організації інтелектуальних систем

Вивчення ІС дозволяє зробити спробу сформулювати загальні принципи, які, не будучи достатніми, відображають необхідні моменти в їх організації та функціонуванні:

а) Принцип системності.

ІС можуть бути тільки складними системами, функції всіх їх елементів повинні бути узгоджені з призначенням системи та їх місцем у них, а також між собою. Саме взаємна узгодженість і взаємозалежність елементів системи забезпечує цілісність і функціональну повноту найбільш досконалих ІС. Це може також призводити до структурної або функціональної надмірності.

б) Принцип ієрархічності.

Складна ієрархічна багаторівнева структура є основою для одночасного протікання безлічі процесів. Рівень неординарності підсумкового процесу залежить від характеру сукупності складових процесів. Складна сукупність процесів принципово характеризується і складною структурою. Таким чином, в деякому роді рівень складності системи і її структури визначає і потенційний рівень її інтелекту.

в) Принцип адаптивності.

Принцип адаптивності передбачає наявність у ІС потенційних можливостей поліпшення роботи: в умовах апріорної і поточної невизначеності на основі навчання на досвіді.

Адаптація може відбуватися шляхом самонастроювання, самонавчання або самоорганізації. Адаптивні здібності можуть визначатися обсягом інформації (пам'яттю) системи і потреб витратами часу на її обробку.

г) Принцип взаємності функціональних і структурних властивостей.

Природно, що призначення системи, її функції безпосередньо впливають на структуру системи. Однак і структура системи повинна сприяти найбільш повній реалізації функцій.

д) Принцип динамічного самопрограмування.

Найчудовіша особливість нервового управління, найбільш яскраво виражена в цілеспрямованому творчому розумі людини, полягає в здатності на підставі різноманітного аналізу ситуацій миттєво створювати складні і разом з тим оптимальні програми діяльності, які безперервно перебудовуються і коригуються з урахуванням минулих подій, поточної дійсності і прогнозування майбутнього.

3.7.2 Розробка та обґрунтування структурної специфікації реляційної бази даних

Для побудови структурної схеми БД використовуємо традиційні засоби структурної специфікації реляційної моделі даних (РМД). Як відомо, основною структурною одиницею даних в РМД є n-арне відношення, тобто підмножина кортежів декартова добутку n доменів [12].

Нехай:

R – кінцева множина імен відношень БД;

D – $\{D_1,...,D_l\}$ – множина доменів, де всякий домен є іменована (може бути нескінченна) множина атомарних значень елементів даних;

A – кінцева множина імен атрибутів відношень;

dom – відображення з A в D, що визначає, з якого домену черпаються значення атрибутів.

Нехай $< A_i, domA_i >$, де $A_i \in A$, назвемо атрибутом.

Структурною схемою S_i відношення R_i $(R_i \in R)$ називається вираз $R_i(A_1,...,A_n)$, в якому всі A_i різні.

Відношення r_i можна визначити як розширення схеми S_i, представлене у такому виді:

$$r_i \subseteq domA_1 \times ... \times A_n.$$

Перестановка атрибутів у схемі не породжує нового розширення і, таким чином, множина $\{A_1,...,A_n\}$ атрибутів відношення R_i, задає тип відношення.

Носій відношень надалі будемо позначати $[R_i]$. Для специфікації складу носія використовуємо вираз $R_i = A_1,...,A_n$. Структурна схема U реляційної БД – це специфікація виду $(R_1,...,R_p)$ де всі $R_i \in R$ і всі R_i різні.

Реляційною БД називається розширення схеми U виду $\{r_1,...,r_p\}$, де r_i – розширення відношення R_i.

Концептуально реляційна БД (інтенсіонал) є інформаційною моделлю деякої предметної області, такою, що кожне розширення відповідає деякому стану предметної області в певний момент дискретно поточного часу. Кожен стан моделюється впорядкованою сукупністю значень елементів даних, що відповідають значенням властивостей об'єктів ПО. Об'єкту певного типу відповідає кортеж відношень певного типу. РМД припускає сильну типізацію об'єктів, тобто використання цілком певних категорій, таких як тип об'єкта, властивість (атрибут) об'єкта, домен, і віднесення кожного значення і впорядкованої сукупності значень до однієї з цих категорій. Об'єкти певного типу мають певний набір властивостей (що задається в РМД схемою відношень), а властивості мають певні набори можливих значень (що задається в РМД відображенням *dom*). Оскільки об'єкти кожного типу

володіють власним унікальним набором властивостей, носії відповідних відношень також є унікальними. Іншими словами, кожне відношення (тип) має набір атрибутів, повністю відмінний від набору атрибутів іншого відношення.

У найпростішому випадку можна припустити, що схема *U* просто задає склад об'єктів без взаємозв'язків, а концептуальна цілісність БД визначається тим, що кілька відношень можуть використовуватися для вирішення однієї прикладної задачі. Очевидно, такий підхід може виявитися неефективним і навіть неадекватим цілям моделювання ПЗ. Деякі відношення, спочатку задумані як модель типу об'єкта, доцільно реструктуризувати, представляючи їх наборами взаємозалежних відношень реляційної БД.

На рівні структурної компоненти моделі БД такий зв'язок відношень може бути встановлений єдиним способом – використанням «одних і тих же» атрибутів у пов'язаних відношеннях. Подоба або тотожність атрибутів у свою чергу визначається, по-перше, рівністю використовуваних доменів і, по-друге, рівністю імен атрибутів. Таким чином, взаємозв'язок відношень проявляється в можливості застосування до них операцій еквіз'єднання або природного з'єднання. Більш екзотичним є спосіб отримання нових відношень шляхом з'єднання вихідних.

Вираз $R_i(A_i,...,A_n)$ у структурній специфікації РМД називається носієм відношень. Для спрощення будемо вважати, що R_i визначає схему відношення з ім'ям R_i. Кожній схемі відношення R_i модель зіставляє множину кортежів декартового добутку:

$$r_i^* = domA_1 \cdot ... \cdot domA_n .$$

Таким чином, відношення в реляційному підході розглядається як реалізація схеми. Число n атрибутів відношення називається його ступенем («арності»), а число кортежів – потужністю. Зрозуміло, що ступінь відношення (при незмінній схемі) не змінюється від реалізації до реалізації,

тоді як потужні відношення, відповідні різним реалізаціям однієї і тієї ж схеми, можуть бути різними.

Будь-яка операція над даними в моделях даних супроводжується операцією над типами, тобто над описом структури операндів. Зокрема, в РМД операндами операцій є відношення, задані структурної компонентою моделі, результатом також є відношення; операції визначені при виконанні певних угод щодо типів операндів і результату. У той же час, над типами, тобто схемами відношень, можна робити свої специфічні операції (які можуть приводити до зміни даних, якщо схеми відношення вже мають реалізацію в БД).

4. ЗАСТОСУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

4.1 Розгортання системи

Для запуску системи треба:

а) комп’ютер та програмне забезпечення: сервер Apache, SQL Server;

б) створення БД з файлу TicketDB.sql;

в) у браузері запустити веб-сайт http://BusTicket/ (рис. 4.1).

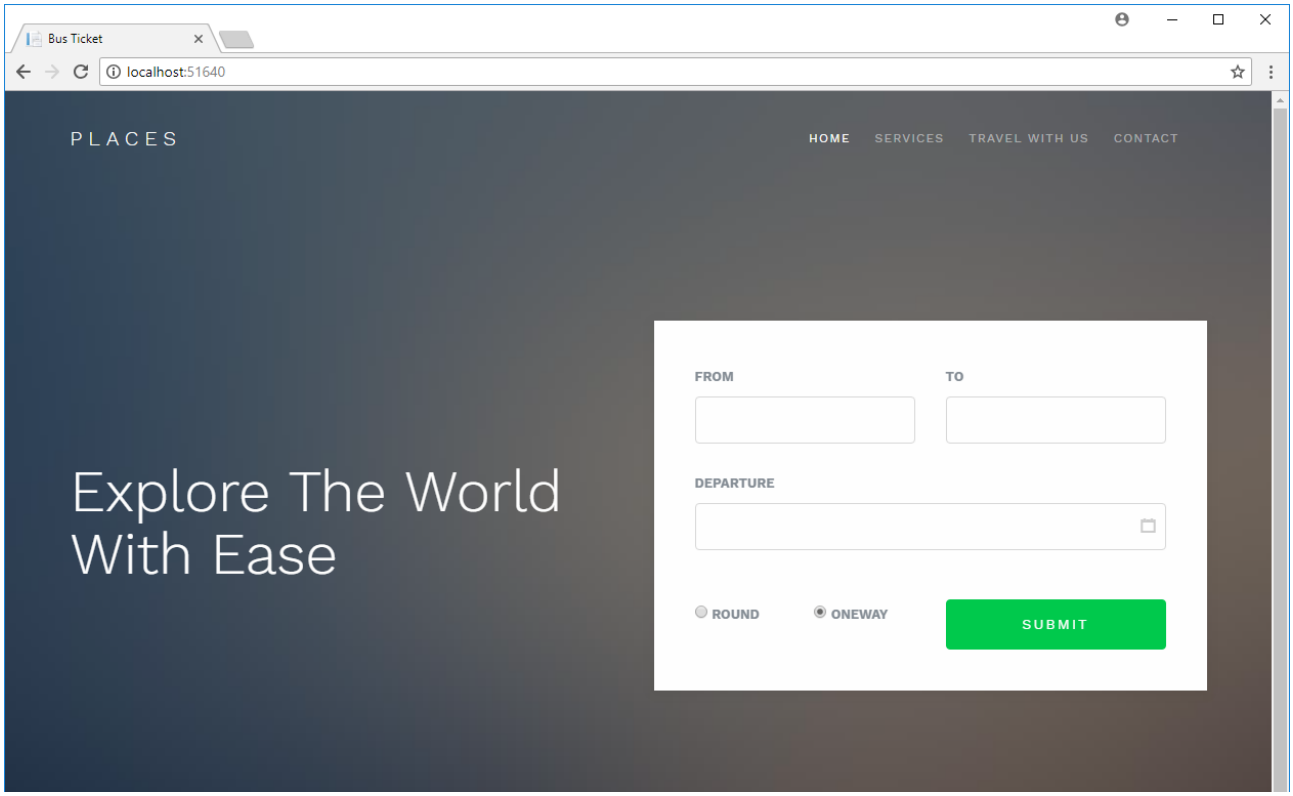

Рисунок 4.1 – Головна форма сайту

4.2 Режими роботи з системою

Режими роботи адміністратора (admin) можливо:

– створювати запити;

– встановлювати порядок доступу та паролі;

– відновлювати базу даних;
– редагувати дані у базі даних.

Режими роботи користувача (user) можливо:

– переглядати інформацію у БД (рис. 4.3);
– оформлювати квиток (рис. 4.4);
– оплачувати замовлення в режимі онлайн;
– звязуватися за будь-якими питаннями.
– вивчати інформацію про маршрут.

4.3 Опис роботи сервісу

Головна сторінка сайту містить меню з розділами: «Home», «Services», «Travel with us», «Contact» (рис. 4.1, дод. А). На головній сторінці клієнт може ввести дані про маршрут, що його інтересує (рис. 4.2), у поле «FROM» звідки, у поле «TO» куди та дату у поле «DEPATUTE».

Також він може обрати пошук зворотного квитка і натиснувши Submit побачити всі наявні маршрути (рис. 4.3).

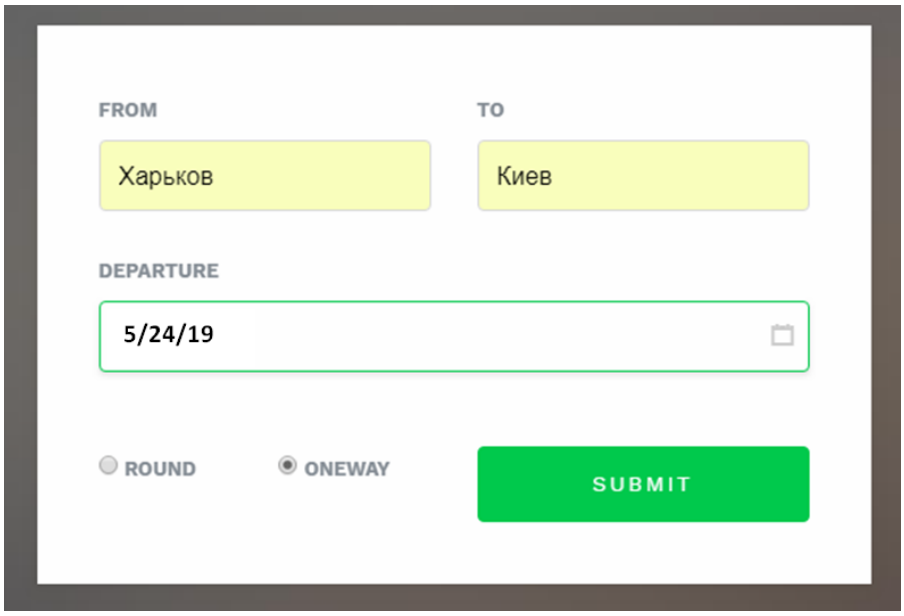

Рисунок 4.2 – Головна сторінка (вибір пунктів відправки та призначення)

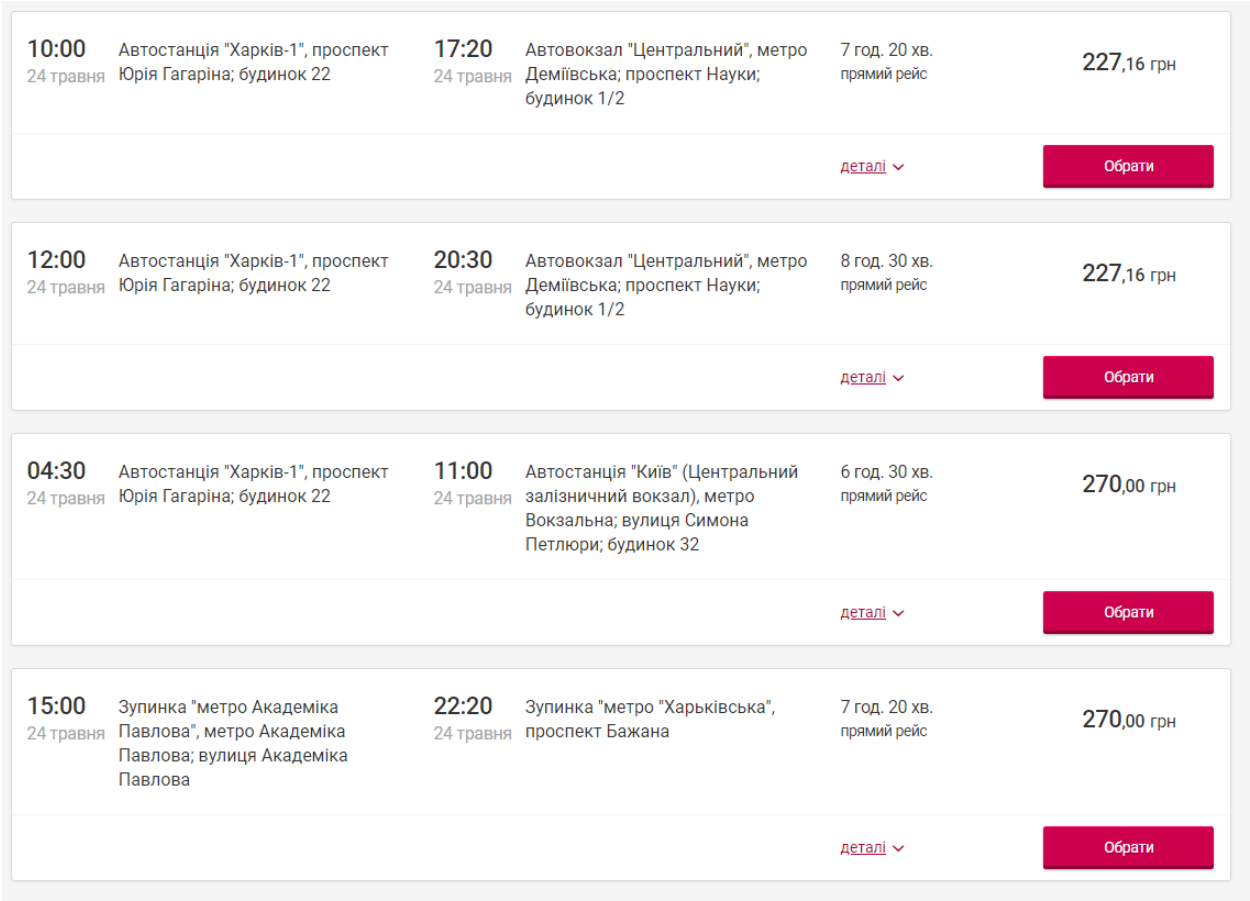

Рисунок 4.3 – Вибір маршруту

Обравши потрібний маршрут, покупець переходить на сторінку оформлення заказу (рис. 4.4)

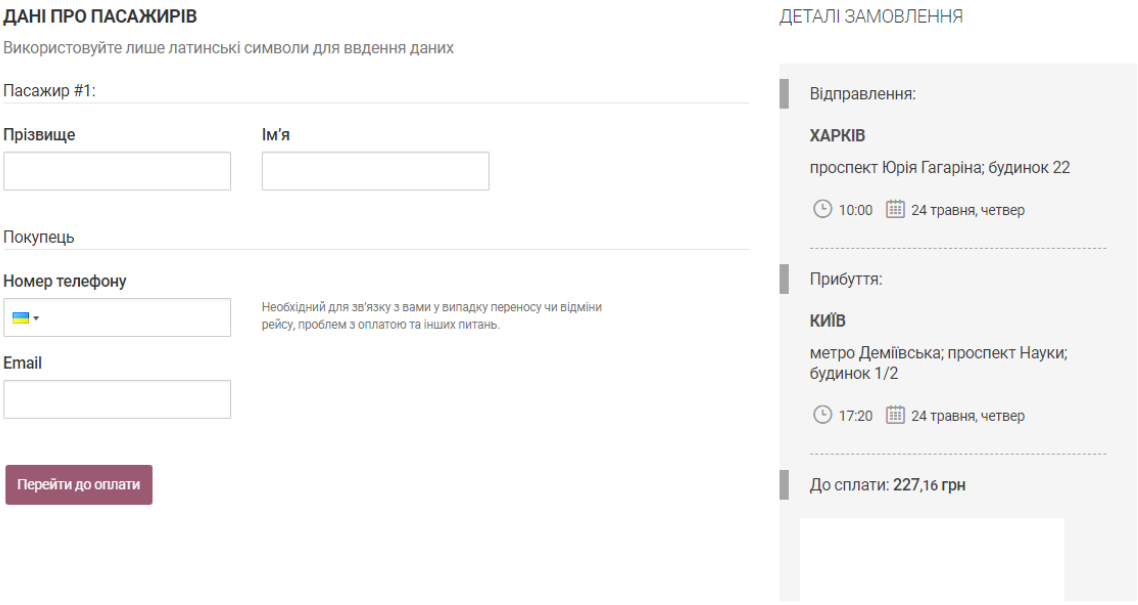

Рисунок 4.4 – Сторінка «Оформлення заказу»

На сторінці адміністратора можна редагувати БД (рис.4.5).

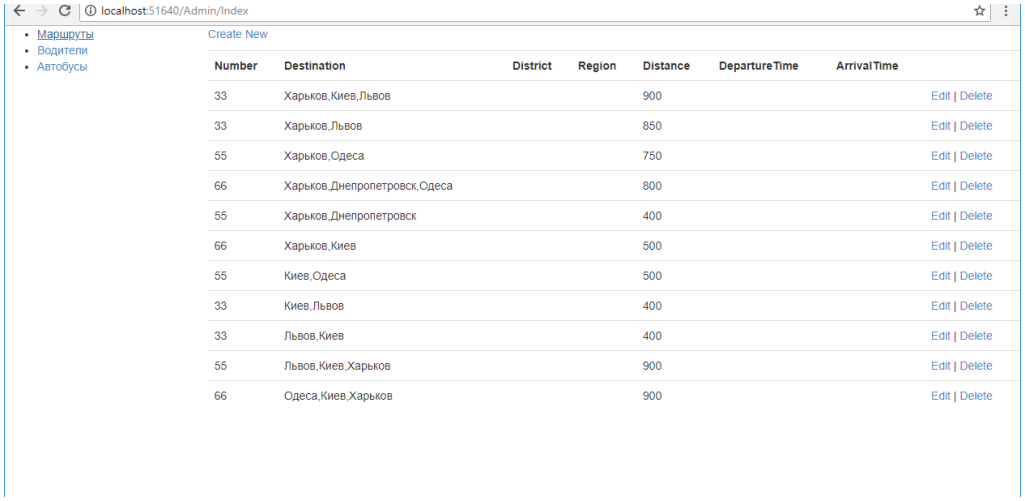

Рисунок 4.5 – Сторінка адміністратора

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯЇХ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даної роботи бакалавра була розробка інтелектуальної системи продажу квитків. Аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера, на якому буде виконуватися розробка, так як в процесі проектування використовувалося комп’ютерне обладнання.

5.1 Загальні питання з охорони праці

Санітарно-побутові умови, умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником повинні відповідати вимогам нормативних актів про охорону праці.

В законі України «Про охорону праці» [1] визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

5.1.1 Правові та організаційні основи охорони праці

Основним організаційним напрямом у здійсненні управління в сфері охорони праці є усвідомлення пріоритету безпеки праці і підвищення соціальної відповідальності держави, і особистої відповідальності працівників.

5.1.2 Організаційно-технічні заходи з безпеки праці

В організації/підприємстві проводиться навчання і перевірка знань з питань охорони праці відповідно до вимог Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці, затвердженого наказом Держнаглядохоронпраці України від 26.01.2005 N 15, зареєстрованого в Міністерстві юстиції України 15.02.2005 за N 231/10511 [2].

Обов’язковими вимогами враховане наступне:

– ознайомлення з правилами безпеки праці, одержання відповідних інструктажів засвідчується у журналі інструктажів.

– перед допуском до самостійної роботи кожен працівник має право на навчання з питань охорони праці і роботодавець зобов’язаний, і проводить таке навчання у вигляді двох інструктажів з питань охорони праці:

1) вступного, який проводять працівники служби охорони праці об’єкта господарювання з усіма працівниками, яких приймають на роботу незалежно від їхньої освіти та стажу роботи за програмою, в якій подають загальні питання охорони праці із врахуванням її особливостей на об’єкті господарювання;

2) первинного, який проводять керівники структурних підрозділів на місці праці з кожним працівником до початку їхньої роботи на цьому робочому місці.

Проходження працівником цих інструктажів з питань охорони праці підтверджується записами у відповідних журналах обліку інструктажів і

скріплюється підписами осіб, які проводили інструктажі та осіб, які отримали інструктажі.

3) Повторний (не рідше одного разу в 6 місяців);

4) Позаплановий (при зміні правил охорони праці);

5) Поточний (проводять з працівниками перед виконанням робіт, на яких оформляється наряд-допуск)

– обов'язкові організаційні заходи перед початком, під час і після завершення роботи повинні включати перевірку (візуально) наявності і справності електрообладнання та його заземлення, а під час виконання роботи вимогу «не залишати без нагляду обладнання, яке працює». Після закінчення роботи - вимагається прибирання робочого місця, відключення всіх електроприладів від електромережі.

5.2 Аналіз стану умов праці

Робота над створенням системи продажу квитків проходитиме в приміщенні багатоквартирного будинку. Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп'ютером.

5.2.1 Вимоги до приміщень

Геометричні розміри приміщення зазначені в табл. 5.1.

Таблиця 5.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	5
Ширина, м	5
Висота, м	2.7

Продовження табл. 5.1

Найменування	**Значення**
Площа, $м^2$	25
Об'єм, $м^3$	67.5

Згідно з [3] розмір площі для одного робочого місця оператора персонального комп'ютера має бути не менше 6 кв. м, а об'єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

5.2.2 Вимоги до організації місця праці

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за [4] (табл. 5.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 5.2 - Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	760	680 ÷800
Висота простору для ніг, мм	650	не менше 600
Ширина простору для ніг, мм	550	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	430	400 ÷500
Ширина сидіння, мм	420	не менше 400
Глибина сидіння, мм	350	не менше 400
Висота поверхні спинки, мм	400	не менше 300
Ширина опорної поверхні спинки, мм	400	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	750	700 ÷800

Приміщення кабінету знаходиться на четвертому поверсі чотирьох поверхової будівлі і має об’єм 67.5 $м^3$, площу – 25 $м^2$. У цьому кабінеті обладнано два місця праці, обидва укомплектовані ПК.

Температура в приміщенні протягом року коливається у межах 17–26°С, відносна вологість — близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум в лабораторії знаходиться на рівні 50 дБА. Система вентилювання приміщення — природна неорганізована, а опалення — централізоване.

5.2.3 Навантаження та напруженість процесу праці

За фізичним навантаженням виконання випускної роботи бакалавра відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням. Щодо характеру організування виконання дипломної роботи, то він підпадає під нав’язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені конкретні терміни.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів захисту.

Роботу над дипломним проектом визнано, такою, що займає 50% часу робочого дня та при восьмигодинній робочій зміні рекомендовано встановити додаткові регламентовані перерви - для розробників програм тривалістю 15 хв. через кожну годину роботи.

5.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені

питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

5.3.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 5.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00-7.15-18 [7] «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Основними робочими характеристиками персонального комп'ютера є:

- робоча напруга U=+220В +-5%;
- робочий струм I=2А;
- споживана потужність Р=350 Вт.

Робочі місця мають відповідати вимогам Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджених постановою Головного державного санітарного лікаря України від 10.12.98 N 7 [4].

Таблиця 5.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
фізичні			
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	[5]

Продовження табл. 5.3

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
- недостатність природного світла	порушення умов праці (вимог до приміщень)	2	[6]
- недостатнє освітлення робочої зони	порушення гігієнічних параметрів виробничого середовища	3	[6]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - пошук та аналіз аналогів і літератури; - пошук наявних технологій, моделювання та аналіз алгоритмів; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	[7] [4]
- фізичні (статичне – сидіння)	порушення умов праці (організації місця праці-сидіння користувача,) та організації робочого часу - безпервна робота)	2	[7] [4]

5.3.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ,

вентиляції і кондиціювання.

Запобігти утворенню горючого середовища (замінити горючі речовини і матеріали на негорючі і важкогорючі) не надається технічно можливим. Тому проектом передбачаються способи і засоби запобігання утворення (або внесення) в горюче середовище джерел запалювання, таких як:

1) застосування електроустаткування, відповідної пожежонебезпечної і вибухонебезпечної зонам відповідно до ПУЕ;

2) застосування в конструкції швидкодійних засобів захисного відключення можливих джерел запалення;

3) виключення можливості появи іскрового розряду в горючому середовищі з енергією, рівної і вище мінімальної енергії запалення.

Згідно ДСТУ Б В.1.1-36:2016 [8]_таке приміщення, площею 25 м2, відноситься до категорії "В" (пожежонебезпечної) та для протипожежного захисту в ньому проектом передбачено устаткування автоматичною пожежною сигналізацією із застосуванням датчиків-сповіщувачів РІД-1 (сповіщувач димовий ізоляційний) в кількості 1 шт., і застосуванням первинних засобів пожежогасіння.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол.

5.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ

(правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

5.4 Гігієнічні вимоги до параметрів виробничого середовища

5.4.1 Мікроклімат

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. Оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають ДСН 3.3.6.042-99 [3] і наведені в табл. 5.4:

Таблиця 5.4 – Норми мікроклімату робочої зони об’єкту

Період року	**Категорія робіт**	**Температура C^0**	**Відносна вологість %**	**Швидкість руху повітря, м/с**
Холодна	легка-1 а	22 - 24	40 – 60	0,1
Тепла	легка-1 а	23 - 25	40 – 60	0,1

Дане приміщення обладнане системами опалення, кондиціонування повітря або припливно-витяжною вентиляцією. У приміщенні на робочому місці забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря. Рівні позитивних і негативних іонів у повітрі мають відповідати ДСН 3.3.6.042-99 [3].

5.4.2 Освітлення

Світло є природною умовою існування людини. Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Хороше освітлення діє тонізуюче, створює гарний настрій, покращує протікання основних процесів вищої нервової діяльності.

У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає ДБН В.2.5-28:2018 [6]. Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n, \quad (5.1)$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$S_n = a \cdot b = 5 \cdot 5 = 25$ м^2,

$S = 1/8 \cdot 25 = 3{,}125$ м^2.

Приймаємо 2 вікна площею S=1,6 м^2 кожне.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні.

Розрахунок кількості світильників n виробляється по формулі (5.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \quad (5.2)$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 25 м^2;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (5.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1{,}1 \cdot 1{,}5}{5400 \cdot 0{,}575 \cdot 2} \approx 2$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

5.4.3 Шум та вібрація, електромагнітне випромінювання

Рівень шуму, що супроводжує роботу користувачів персональних комп'ютерів (зумовлений як роботою системних блоків, клавіатури, так і друкуванням на принтерах, а також зовнішніми чинниками), коливається у межах 50–65 дБА ДСН 3.3.6.042-99 [3]. У залах опрацювання інформації та комп'ютерного набору рівні шуму не повинні перевищувати 65 дБА.

Віброізоляція можливо здійснювати за допомогою спеціальної прокладки під системний блок, який послаблює передачу вібрацій робочого столу. Вібрація на робочому місці в приміщенні, що розглядається, відповідає нормам ДСН 3.3.6.042-99 [3].

5.4.4 Вентилювання

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти).

Також має здійснюватися провітрювання приміщення в залежності від погодних умов. Тривалість повинна бути не менше 10 хв. Найкращий обмін повітря здійснюється при наскрізному провітрюванні.

5.5 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання наводимо приклади деяких заходів безпеки.

1) Заходи безпеки під час експлуатації персонального комп'ютера та периферійних пристроїв передбачають:

- правильне організування місця праці та дотримання оптимальних режимів праці та відпочинку під час роботи з ПК;
- експлуатацію сертифікованого обладнання;
- дотримання заходів електробезпеки;
- забезпечення оптимальних параметрів мікроклімату;
- забезпечення раціонального освітлення місця праці (освітленість робочого місця не перевищувала 2/3 нормальної освітленості приміщення);
- облаштовуючи приміщення для роботи з ПК, потрібно передбачити припливно-витяжну вентиляцію або кондиціювання повітря.

2) Заходи безпеки під час експлуатації інших електричних приладів передбачають дотримання таких правил:

- постійно стежити за справним станом електромережі, розподільних щитків, вимикачів, штепсельних розеток, лампових патронів, а також мережевих кабелів живлення, за допомогою яких електроприлади під'єднують до електромережі;
- постійно стежити за справністю ізоляції електромережі та мережевих кабелів, не допускаючи їхньої експлуатації з пошкодженою ізоляцією;
- не тягнути за мережевий кабель, щоб витягти вилку з розетки;
- не закривати меблями, різноманітним інвентарем вимикачі, штепсельні розетки;
- не підключати одночасно декілька потужних електропристроїв до однієї розетки, що може викликати надмірне нагрівання провідників, руйнування їхньої ізоляції, розплавлення і загоряння полімерних матеріалів;

- не залишати включені електроприлади без нагляду;

- не допускати потрапляння всередину електроприладів крізь вентиляційні отвори рідин або металевих предметів, а також не закривати їх та підтримувати в належній чистоті, щоб уникнути перегрівання та займання приладу;

- не ставити на електроприлади матеріали, які можуть під дією теплоти, що виділяється, загорітися (канцелярські товари, сувенірну продукцію тощо).

Вимоги безпеки при надзвичайних ситуаціях:

1) При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, ВДТ, системний блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток. При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК, знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво.

2) При замиканні, перевантаженні електричного струму на електричному обладнанні, внаслідок ураження грозової блискавки та ймовірної небезпеки ураженням електричним струмом, приймають наступне:

- попередження замикання здійснюється правильним вибором, монтажем експлуатації мереж;

- застосування захисту схем у вигляді швидкодіючих реле, а також вимикачів, плавких запобіжників.

Також застосовують різні **електричні захисні засоби від ураження струмом**:

а) Ізолюючі - ізолюють людини від струмоведучих або заземлених частин, а так-же від землі. Вони діляться на основні та додаткові.

б) Основні - володіють ізоляцією, здатної довго витримувати робоче напругу електроустановки і тому ними дозволяється стосуватися струмоведучих частин, знаходячи-трудящих під напругою.

в) Запобіжні - володіють ізоляцією нездатною витримати робоча напруга електроустановки, і тому вони не можуть самостійно захищати людину від ураження струмом під цим напругою.

5.6 Розрахунок захисного заземлення (забезпечення електробезпеки будівлі)

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом, приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Коефіцієнт використання вертикальних заземлювачів $\eta_в$ в залежності від розміщення заземлювачів та їх кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача η_c.

Послідовність розрахунку.

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \qquad (5.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то $R_{шт.з.}=R_д$.

Підставивши числові значення у формулу (А.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2) Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ=40 Ом•м (табличне значення).

3) Розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом·м за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \tag{5.4}$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів I кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}$=1,7 і горизонтальних $\rho_{розр.г}$=5,5 Ом·м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м}$$

$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м}$$

4) Розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за формулою (5.5).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \tag{5.5}$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стержня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за формулою (5.6):

$$t = h_в + \frac{l_в}{2}, \tag{5.6}$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,8 м); тоді

$$t = 0,8 + \frac{3}{2} = 2,3 \text{ м}$$

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом}$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \qquad (5.7)$$

І визначається коефіцієнт використання вертикальних електродів групового заземлювача без врахування впливу з'єднувальної стрічки $\eta_в$ =0,57 (табличне значення).

6) Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \qquad (5.8)$$

7) Визначається довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \qquad (5.9)$$

де $L_в$ – відстань між вертикальними заземлювачами, (прийняти за $L_в$ = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_с = 1,05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

8) Визначається опір розтіканню струму горизонтального заземлювача (з'єднувальної стрічки) $R_г$, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_с} \cdot \ln \frac{2 \cdot l_с^2}{d_{см} \cdot h_г}, \qquad (5.10)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизантальних заземлювачів (0,5 м);

$l_с$ - довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом}$$

9) Визначається коефіцієнт використання горизонтального заземлювача $\eta_{с.}$ відповідно до необхідної кількості вертикальних заземлювачів $n_в$.

Коефіцієнт використання з'єднувальної смуги η_c=0,3 (табличне значення).

10) Розраховується результуючий опір заземлювального електроду з урахуванням з'єднувальної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_c + R_г \cdot n_в \cdot \eta_в} \le R_д. \qquad (5.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \le R_д$$

3) При виникненню пожеж при роботі на ПЕОМ від таких можливими джерел запалювання як:

- іскри і дуги коротких замикань;
- перегрів провідників, резисторів та інших радіодеталей ПЕОМ, від тривалої перевантаження та наявність перехідного опору;
- іскри при розмиканні і розмиканні ланцюгів;
- розряди статичної електрики;
- необережному поводженню з вогнем, а також вибухи газо-повітряних і паро-повітряних сумішей.

5.7 Висновки до розділу 5

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом; описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведено рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки.

Були наведені розміри приміщення та значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

5.8 Перелік корисних посилань до розділу 5

1. Закон України «Про охорону праці». Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12 - 10.14.1992 р.

2. Нормативно-правовий акт з охорони праці. НПАОП 0.00-4.12-05 «Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці» - Режим доступ: https://zakon.rada.gov.ua/laws/show/z0231-05

3. Державні санітарні норми. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» - Режим доступу: https://zakon.rada.gov.ua/rada/show/va042282-99 - 01.02.1999 р.

4. Державні санітарні правила і норми. ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» - Режим доступу: https://zakon.rada.gov.ua/rada/show/v0007282-98 - 10.12.1998 р.

5. Державний стандарт України. ДСТУ Б В.2.5-82:2016 «Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом» - Режим доступу: http://epicentre.co.ua/dstu/doc28522.html - 01.07.2016 р.

6. Державні будівельні норми. ДБН В.2.5-28:2018 «Природне і штучне освітлення» - Режим доступу: http://www.minregion.gov.ua/wp-content/uploads/2018/12/V2528-1.pdf - 03.10.2018

7. Нормативно-правовий акт з охорони праці. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0508-18 - 14.02.2018 р.

8. Державний стандарт України. ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» - Режим доступу: https://zakon.rada.gov.ua/rada/show/v0158858-16 - 15.06.2016 р.

ВИСНОВКИ

В процесі виконання даної роботи розроблена інтелектуальна система з продажу квитків.

В процесі виконання роботи зроблено огляд методів та технологій до проектування та розробки_інформаційних систем; проектування та розробку бази даних; обґрунтування та вибір засобів розробки програмного забезпечення; проектування, розробку та тестування програмної системи; здійснений аналіз потенційних небезпечних і шкідливих виробничих чинників проектованого об'єкта, що впливають на персонал.

Розроблена інформаційна система забезпечує в клієнтській частині швидкий пошук необхідних даних по запиту користувачів, а також здійснення запитів з використанням даних сервера.

Структура управління системою реалізована у вигляді трирівневої архітектури «клієнт-сервер додатків – сервер бази даних»: Web-сервер, що виробляє розмежування доступу і розподіляє запити; сервер додатків, що керує бізнес-логікою і реалізує необхідну сукупність процесів; БД і СКБД для збору, зберігання, обробки і управління даними.

Для розробки системи на базі клієнт-серверної системи застосовані програмні продукти: SQL SERVER, SQL, C#, HTML (XHTML), CSS, JavaScript, ENTITY FRAMEWORK, Asp.NET MVC.

Розроблений сервіс дає можливість розширити та поліпшити якість послуг з продажу квитків.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Буслик, М. Модели и структуры данных [Текст] / М. М. Буслик. – К. : ІСДО, 2000. – 124 с.

2. Буслік, М. Оптимальні зображення реляційних структур даних [Текст] / М. М. Буслік. – К. : ІСДО, 1993. – 84с.

3. Боуман, Дж. Практическое руководство по SQL [Текст] : учеб. пособие / Дж. Боуман, С. Эмерсон, М. Дарновски. – К. : Диалектика, 1998. – 565 с.

4. Дейт, К. Введение в системы баз данных [Текст] / К. Дейт. М. : 2006. – 1328 с.

5. Дейт, К. Введение в системы баз данных [Текст] / К. Дейт. М. : 1998. – 784 с.

6. Карпова, Т. Базы данных: модели, разработка, реализация. [Текст] : учеб. пособие / Т. Карпова. – СПб.: Питер, 2001. – 304 с.

7. Грабер, М. SQL, справочное руководство [Текст] / М. Грабер. – М. : Лори, 1998. – 292с.

8. Вескес, Дж. Access и SQL Server [Текст] : учеб. / Дж. Вескес, М. Гандерлоу. – М. : Лори, 1997. – 362 с.

9. Пфафенбергер, Б. HTML, XHTML, and CSS Bible 3rd Edition [Текст] / Б. Пфафенбергер. – WILEY, 2003. – 432 с.

10. Ши, Д. Философия CSS-дизайна [Текст] : учеб. / Д. Ши, Е. Молли. – М. : НТ Пресс, 2005. – 385 с.

11. Мейер, Д. Теория реляционных баз данных [Текст] / Д. Мейер – М. : Мир, 1987. – 608 с.

ДОДАТОК А

Лістинг коду main.js

```
                    $(document).ready(function($) {
      "use strict";
      var isMobile = false; //initiate as false
// device detection
if(/(android|bb\d+|meego).+mobile|avantgo|bada\/|blackberry|blazer|com
pal|elaine|fennec|hiptop|iemobile|ip(hone|od)|ipad|iris|kindle|Android
|Silk|lge |maemo|midp|mmp|netfront|opera m(ob|in)i|palm(
os)?|phone|p(ixi|re)\/|plucker|pocket|psp|series(4|6)0|symbian|treo|up
\.(browser|link)|vodafone|wap|windows
(ce|phone)|xda|xiino/i.test(navigator.userAgent)
    || /1207|6310|6590|3gso|4thp|50[1-6]i|770s|802s|a
wa|abac|ac(er|oo|s\-
)|ai(ko|rn)|al(av|ca|co)|amoi|an(ex|ny|yw)|aptu|ar(ch|go)|as(te|us)|at
tw|au(di|\-m|r |s
)|avan|be(ck|ll|nq)|bi(lb|rd)|bl(ac|az)|br(e|v)w|bumb|bw\-
(n|u)|c55\/|capi|ccwa|cdm\-|cell|chtm|cldc|cmd\-
|co(mp|nd)|craw|da(it|ll|ng)|dbte|dc\-
s|devi|dica|dmob|do(c|p)o|ds(12|\-
d)|el(49|ai)|em(l2|ul)|er(ic|k0)|esl8|ez([4-7]0|os|wa|ze)|fetc|fly(\-
|_)|g1 u|g560|gene|gf\-5|g\-mo|go(\.w|od)|gr(ad|un)|haie|hcit|hd\-
(m|p|t)|hei\-|hi(pt|ta)|hp( i|ip)|hs\-c|ht(c(\-|
|_|a|g|p|s|t)|tp)|hu(aw|tc)|i\-(20|go|ma)|i230|iac( |\-
|\/)|ibro|idea|ig01|ikom|im1k|inno|ipaq|iris|ja(t|v)a|jbro|jemu|jigs|k
ddi|keji|kgt( |\/)|klon|kpt |kwc\-|kyo(c|k)|le(no|xi)|lg(
g|\/(k|l|u)|50|54|\-[a-w])|libw|lynx|m1\-
w|m3ga|m50\/|ma(te|ui|xo)|mc(01|21|ca)|m\-
cr|me(rc|ri)|mi(o8|oa|ts)|mmef|mo(01|02|bi|de|do|t(\-|
|o|v)|zz)|mt(50|p1|v )|mwbp|mywa|n10[0-2]|n20[2-
3]|n30(0|2)|n50(0|2|5)|n7(0(0|1)|10)|ne((c|m)\-
|on|tf|wf|wg|wt)|nok(6|i)|nzph|o2im|op(ti|wv)|oran|owg1|p800|pan(a|d|t
)|pdxg|pg(13|\-([1-8]|c))|phil|pire|pl(ay|uc)|pn\-
2|po(ck|rt|se)|prox|psio|pt\-g|qa\-a|qc(07|12|21|32|60|\-[2-7]|i\-
)|qtek|r380|r600|raks|rim9|ro(ve|zo)|s55\/|sa(ge|ma|mm|ms|ny|va)|sc(01
|h\-|oo|p\-)|sdk\/|se(c(\-|0|1)|47|mc|nd|ri)|sgh\-|shar|sie(\-|m)|sk\-
0|sl(45|id)|sm(al|ar|b3|it|t5)|so(ft|ny)|sp(01|h\-|v\-|v
)|sy(01|mb)|t2(18|50)|t6(00|10|18)|ta(gt|lk)|tcl\-|tdg\-
|tel(i|m)|tim\-|t\-mo|to(pl|sh)|ts(70|m\-|m3|m5)|tx\-
9|up(\.b|g1|si)|utst|v400|v750|veri|vi(rg|te)|vk(40|5[0-3]|\-
v)|vm40|voda|vulc|vx(52|53|60|61|70|80|81|83|85|98)|w3c(\-|
)|webc|whit|wi(g |nc|nw)|wmlb|wonu|x700|yas\-|your|zeto|zte\-
/i.test(navigator.userAgent.substr(0,4))) isMobile = true;
      // scroll
      var scrollWindow = function() {
            var lastScrollTop = 0;
            $(window).scroll(function(){
                  var $w = $(this),
                              st = $w.scrollTop(),
                              navbar = $('.probootstrap_navbar'),
                              sd = $('.js-scroll-wrap');
                  if (st > 150) {
                        if ( !navbar.hasClass('scrolled') ) {
                              navbar.addClass('scrolled');
                        }
```

```
                }
                if (st < 150) {
                     if ( navbar.hasClass('scrolled') ) {
                          navbar.removeClass('scrolled sleep');
                     }
                }
                if ( st > 350 ) {
                     if ( !navbar.hasClass('awake') ) {
                          navbar.addClass('awake');
                     }
                }
                if ( st < 350 ) {
                     if ( navbar.hasClass('awake') ) {
                          navbar.removeClass('awake');
                          navbar.addClass('sleep');
                     }
                }
           });
     };
     scrollWindow();
     // navigation
     var OnePageNav = function() {
          var navToggler = $('.navbar-toggler');
          $(".smoothscroll[href^='#'], #probootstrap-navbar ul li
a[href^='#']").on('click', function(e) {
               e.preventDefault();
               var hash = this.hash;
               $('html, body').animate({
              scrollTop: $(hash).offset().top
            }, 700, 'easeInOutExpo', function(){
              window.location.hash = hash;
            });
          });
          $("#probootstrap-navbar ul li a[href^='#']").on('click',
function(e){
               if ( navToggler.is(':visible') ) {
               navToggler.click();
            }
          });
     };
     OnePageNav();
     var select2 = function() {
          $('.js-dropdown-multiple, .js-example-basic-
single').select2();
     }
     select2();
     var contentWayPoint = function() {
          var i = 0;
          if ($('.probootstrap-animate').length > 0 ) {
               $('.probootstrap-animate').waypoint( function(
direction ) {
                    if( direction === 'down' &&
!$(this.element).hasClass('probootstrap-animated') ) {
                         i++;
                         $(this.element).addClass('item-animate');
                         setTimeout(function(){
```

```
                                    $('body .probootstrap-animate.item-
animate').each(function(k){
                                         var el = $(this);
                                         setTimeout( function () {
                                              var effect =
el.data('animate-effect');
                                              if ( effect === 'fadeIn') {
                                                   el.addClass('fadeIn
probootstrap-animated');
                                              } else if ( effect ===
'fadeInLeft') {

      el.addClass('fadeInLeft probootstrap-animated');
                                              } else if ( effect ===
'fadeInRight') {

      el.addClass('fadeInRight probootstrap-animated');
                                              } else {
                                                   el.addClass('fadeInUp
probootstrap-animated');
                                              }
                                              el.removeClass('item-
animate');
                                         },  k * 50, 'easeInOutExpo' );
                                    });
                               }, 50);
                          }

                } , { offset: '95%' } );
           }
      };
      contentWayPoint();
  var owlCarouselFunc = function() {
        $('.js-owl-carousel').owlCarousel({
          loop : false,
          margin : 20,
          nav : true,
          stagePadding : 50,
          navText : ["<span class='ion-chevron-left'></span>","<span
class='ion-chevron-right'></span>"],
          responsive : {
              0 : {
                  items:1
              },
              600 : {
                  items:2
              },
              1000 : {
                  items:3
              }
          }
           });
           $('.js-owl-carousel-2').owlCarousel({
          loop : false,
          margin : 20,
          nav : true,
          stagePadding : 0,
```

```
          navText : ["<span class='ion-chevron-left'></span>","<span
class='ion-chevron-right'></span>"],
          responsive : {
              0 : {
                  items:1
              },
              600 : {
                  items:1
              },
              800 : {
                  items:2
              },
              1000 : {
                  items:3
              }
          }
            });
  };
  owlCarouselFunc();

  var ThumbnailOpacity = function() {
     var t = $('.probootstrap-thumbnail');
     t.hover(function(){
          var $this = $(this);
          t.addClass('sleep');
          $this.removeClass('sleep');
     }, function(){
          var $this = $(this);
          t.removeClass('sleep');
     });
  }
  ThumbnailOpacity();
  var datePicker = function() {
          $('#probootstrap-date-departure, #probootstrap-date-
arrival').datepicker({
             'format': 'm/d/yyyy',
             'autoclose': true
          });
     };
     datePicker();
});
```

ДОДАТОК Б

Лістинг коду google-map.js

```
var google;
function init() {
    // Basic options for a simple Google Map
    // For more options see:
https://developers.google.com/maps/documentation/javascript/reference#Map
Options
    // var myLatlng = new google.maps.LatLng(40.71751, -73.990922);
    var myLatlng = new google.maps.LatLng(40.69847032728747, -
73.9514422416687);
    // 39.399872
    // -8.224454
    var mapOptions = {
        // How zoomed in you want the map to start at (always required)
        zoom: 7,
        // The latitude and longitude to center the map (always required)
        center: myLatlng,
        // How you would like to style the map.
        scrollwheel: false,
        styles: [{"featureType": "all","elementType":
"geometry.fill","stylers": [{"weight": "2.00"}]},{"featureType":
"all","elementType": "geometry.stroke","stylers": [{"color":
"#9c9c9c"}]},{"featureType": "all","elementType":
"labels.text","stylers": [{"visibility": "on"}]},{"featureType":
"landscape","elementType": "all","stylers": [{"color":
"#f2f2f2"}]},{"featureType": "landscape","elementType":
"geometry.fill","stylers": [{"color": "#ffffff"}]},{"featureType":
"landscape.man_made","elementType": "geometry.fill","stylers": [{"color":
"#ffffff"}]},{"featureType": "poi","elementType": "all","stylers":
[{"visibility": "off"}]},{"featureType": "road","elementType":
"all","stylers": [{"saturation": -100},{"lightness":
45}]},{"featureType": "road","elementType": "geometry.fill","stylers":
[{"color": "#eeeeee"}]},{"featureType": "road","elementType":
"labels.text.fill","stylers": [{"color": "#7b7b7b"}]},{"featureType":
"road","elementType": "labels.text.stroke","stylers": [{"color":
"#ffffff"}]},{"featureType": "road.highway","elementType":
"all","stylers": [{"visibility": "simplified"}]},{"featureType":
"road.arterial","elementType": "labels.icon","stylers": [{"visibility":
"off"}]},{"featureType": "transit","elementType": "all","stylers":
[{"visibility": "off"}]},{"featureType": "water","elementType":
"all","stylers": [{"color": "#46bcec"},{"visibility":
"on"}]},{"featureType": "water","elementType": "geometry.fill","stylers":
[{"color": "#c8d7d4"}]},{"featureType": "water","elementType":
"labels.text.fill","stylers": [{"color": "#070707"}]},{"featureType":
"water","elementType": "labels.text.stroke","stylers": [{"color":
"#ffffff"}]}]
    };
    // Get the HTML DOM element that will contain your map
    // We are using a div with id="map" seen below in the <body>
    var mapElement = document.getElementById('map');
```

```
50      // Create the Google Map using out element and options defined above
51      var map = new google.maps.Map(mapElement, mapOptions);
52          var addresses = ['New York'];
53      for (var x = 0; x < addresses.length; x++) {
54  $.getJSON('http://maps.googleapis.com/maps/api/geocode/json?address='+add
55  resses[x]+'&sensor=false', null, function (data) {
56              var p = data.results[0].geometry.location
57              var latlng = new google.maps.LatLng(p.lat, p.lng);
58              new google.maps.Marker({
59                  position: latlng,
60                  map: map,
61                  icon: 'img/loc.png'
62              });
63          });
64      }
65  }
66  google.maps.event.addDomListener(window, 'load', init);
```

Додаток В

Комп’ютерна презентація

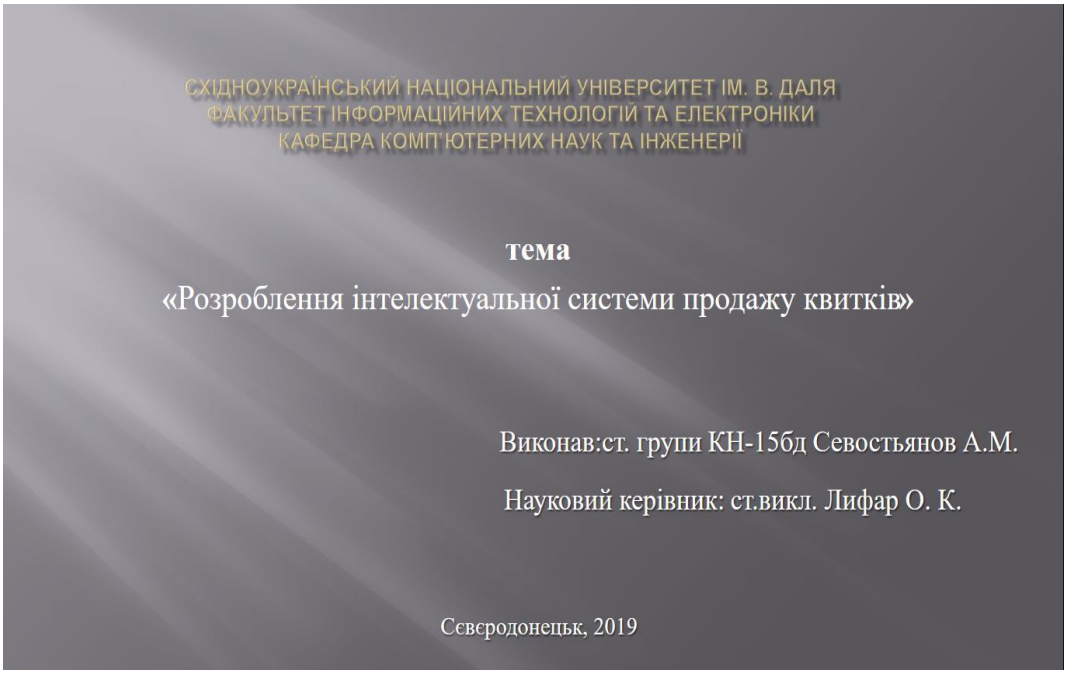

Рисунок В.1 – Слайд №1

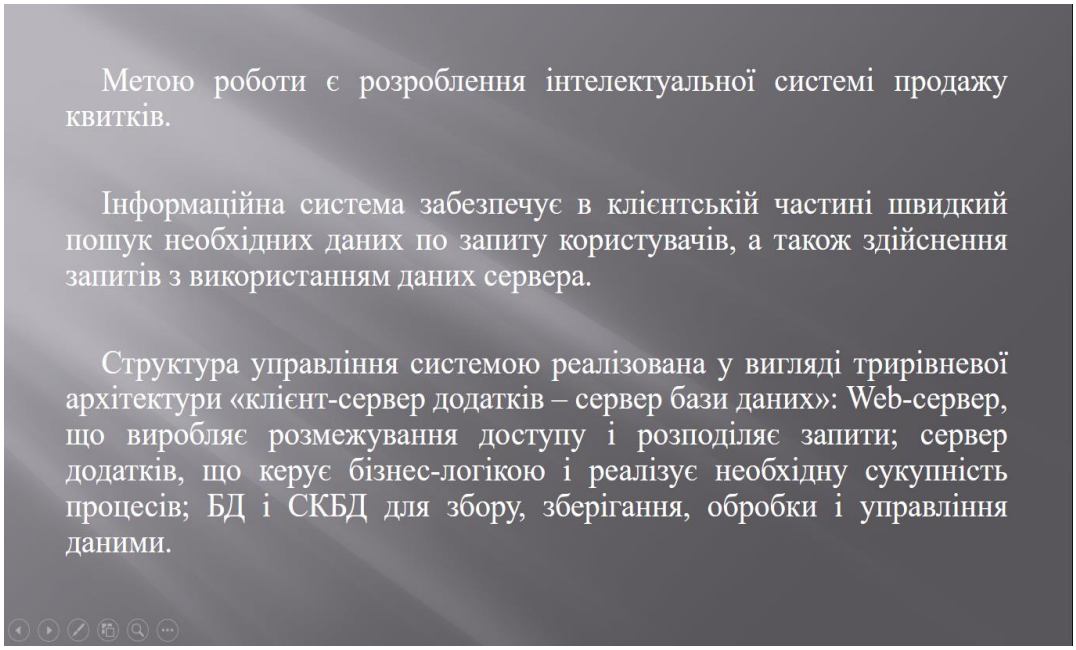

Рисунок В.2 – Слайд №2

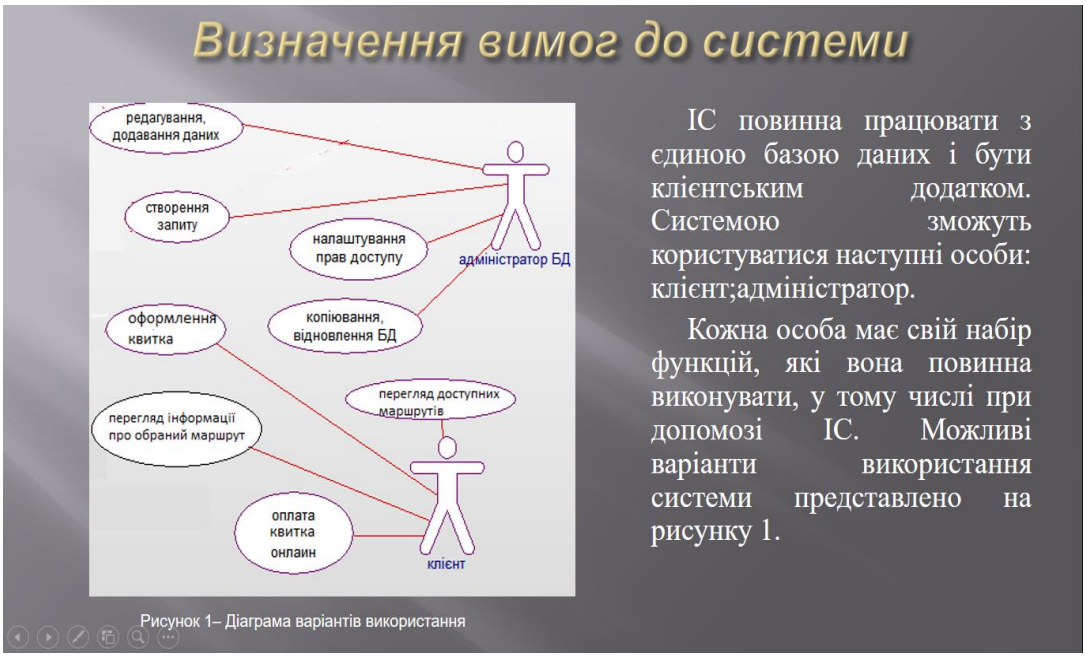

Рисунок В.3 – Слайд №3

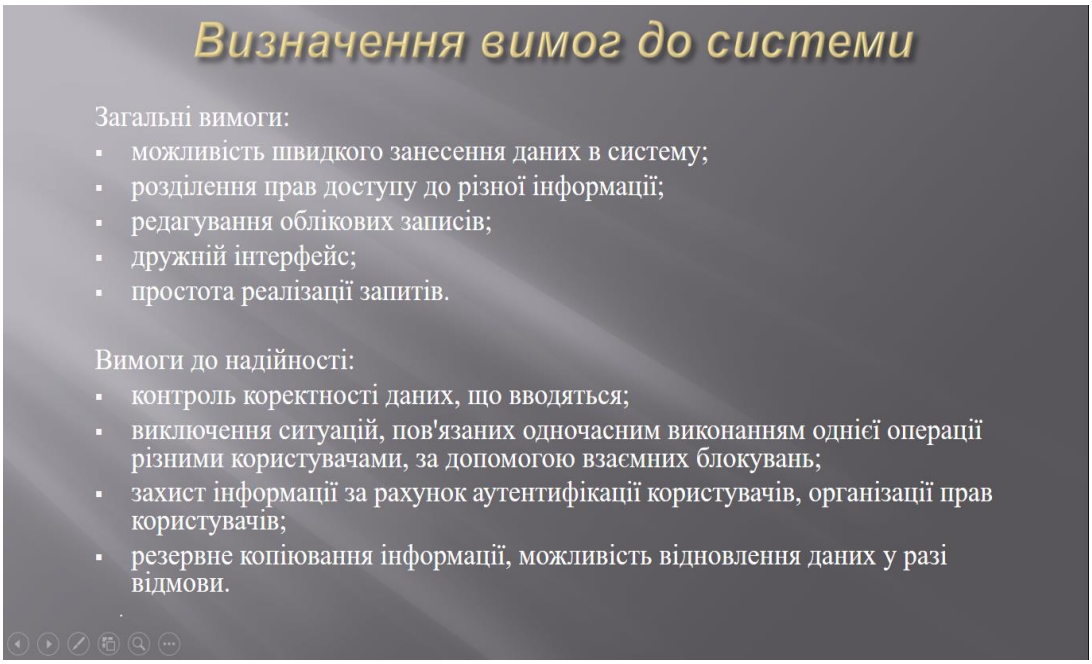

Рисунок В.4 – Слайд №4

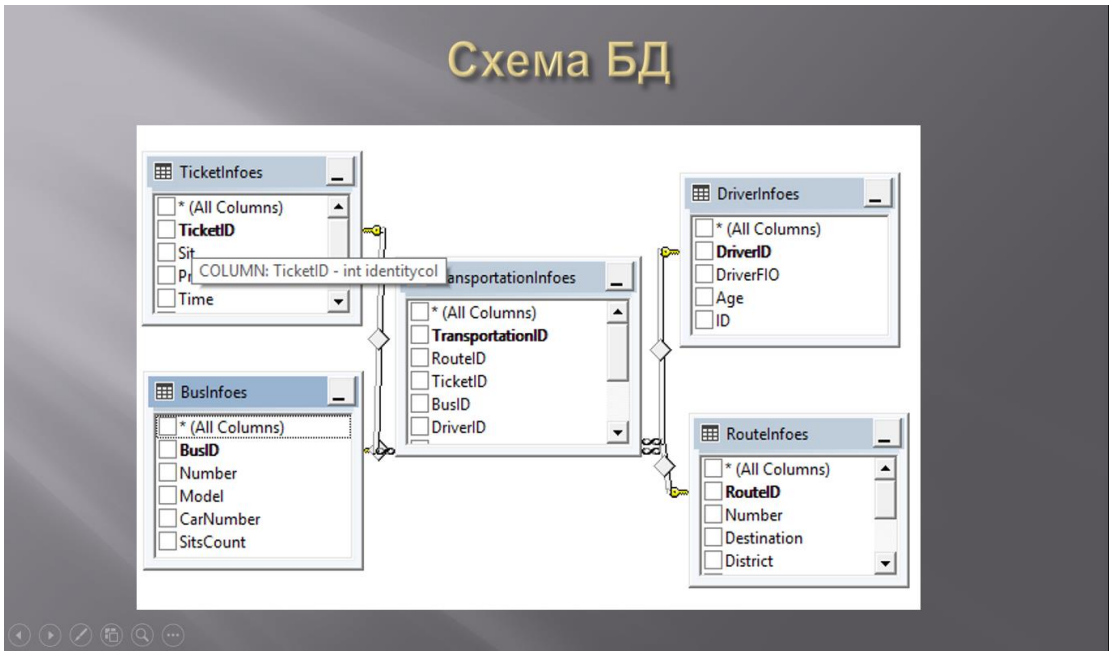

Рисунок В.5 – Слайд №5

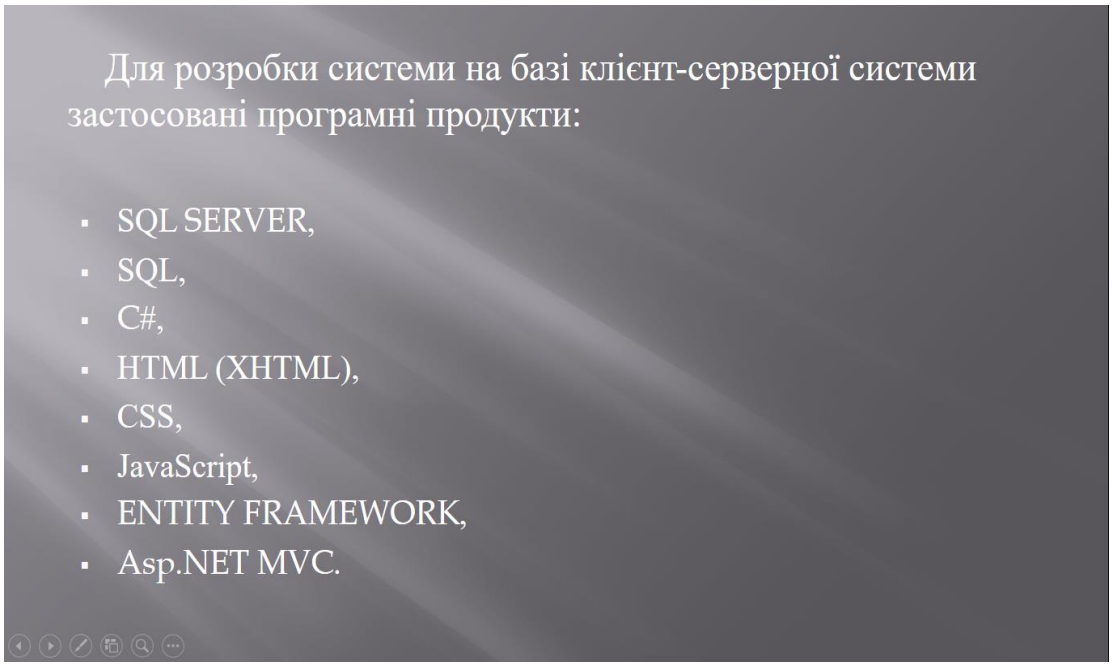

Рисунок В.6 – Слайд №6

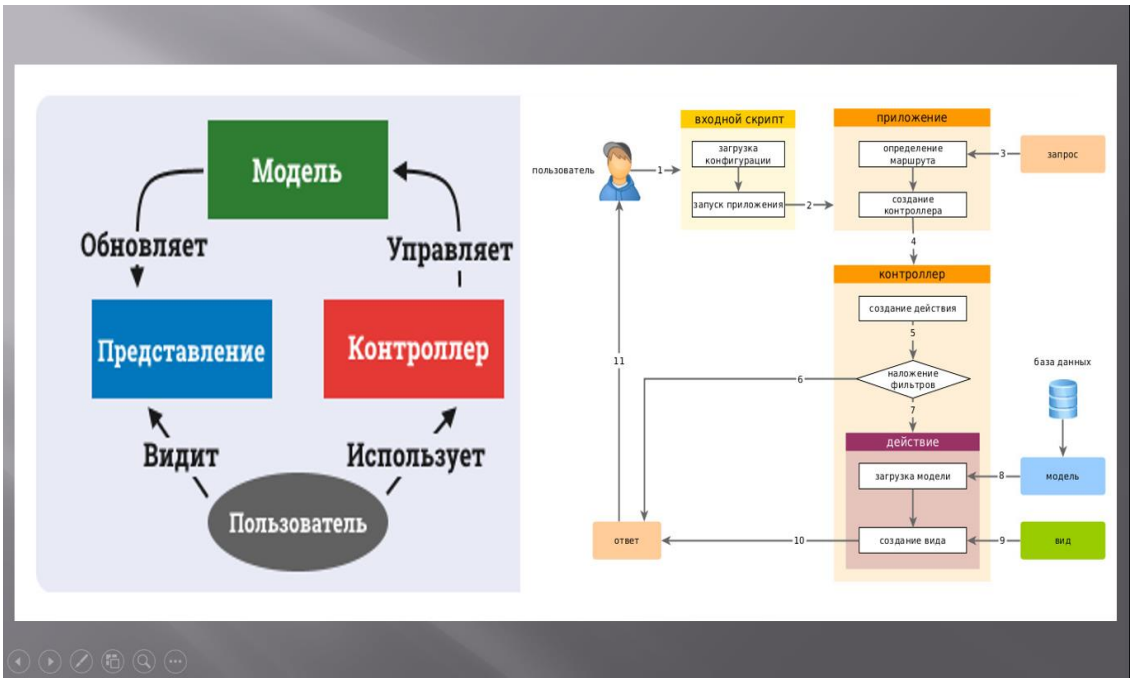

Рисунок В.7 – Слайд №7

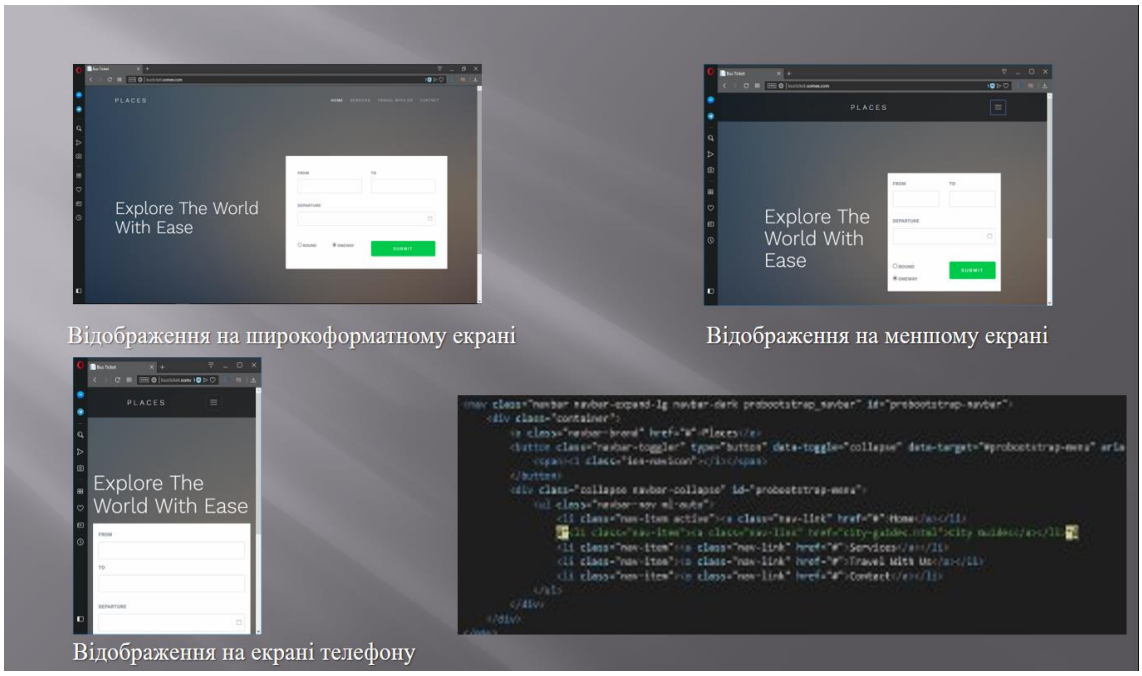

Рисунок В.8 – Слайд №8

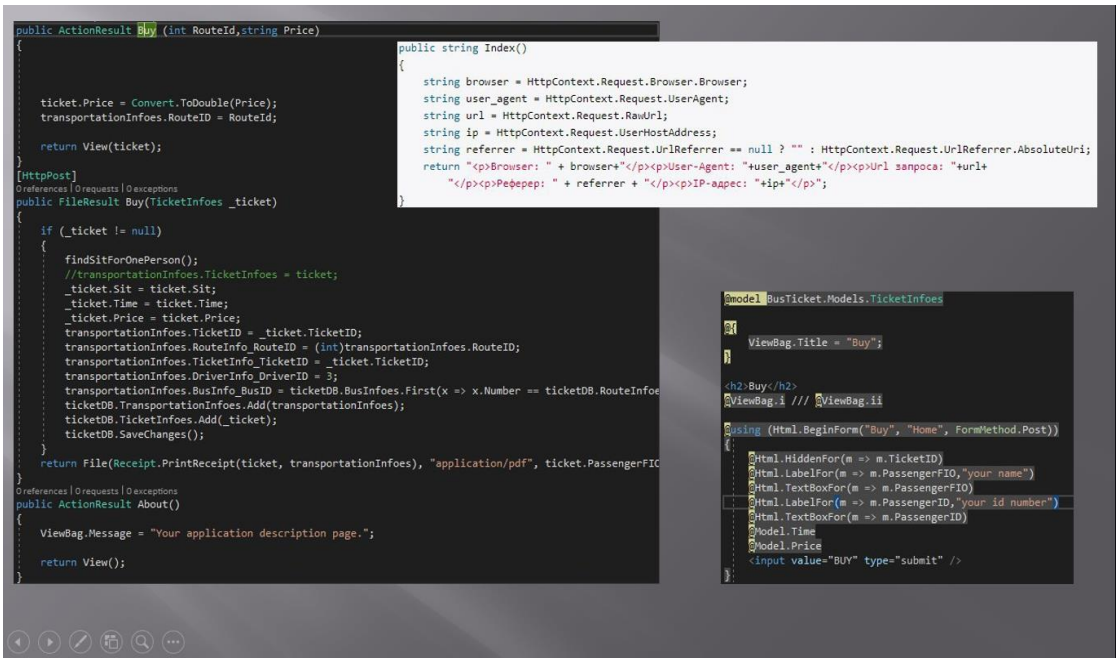

Рисунок В.9 – Слайд №9

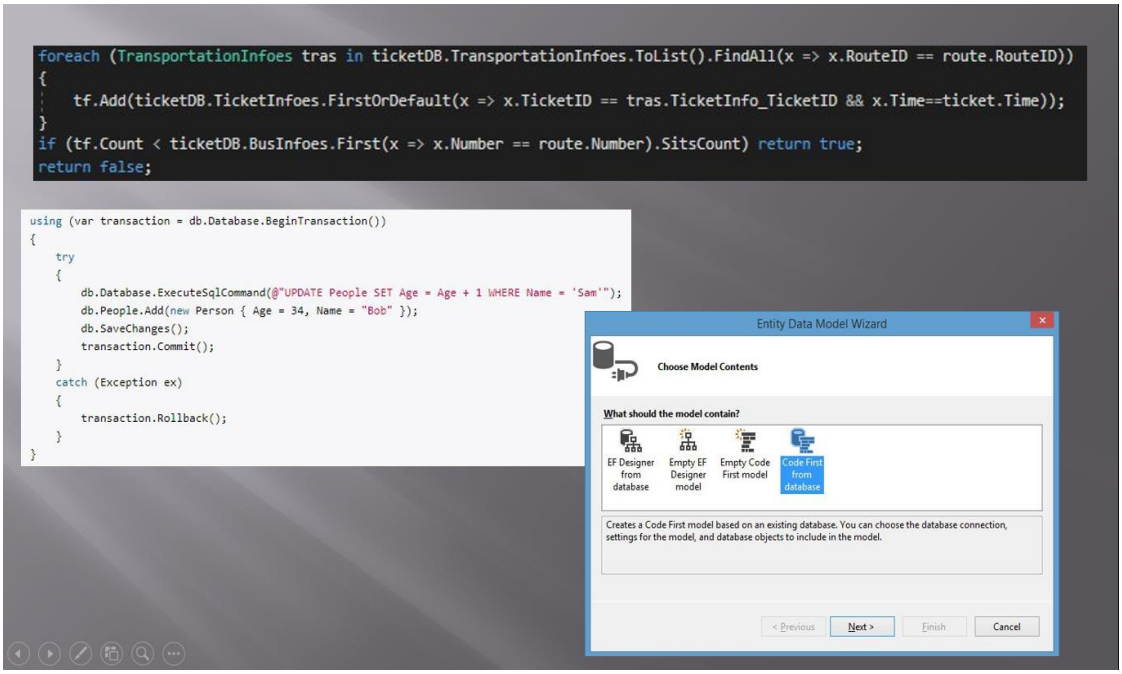

Рисунок В.10 – Слайд №10

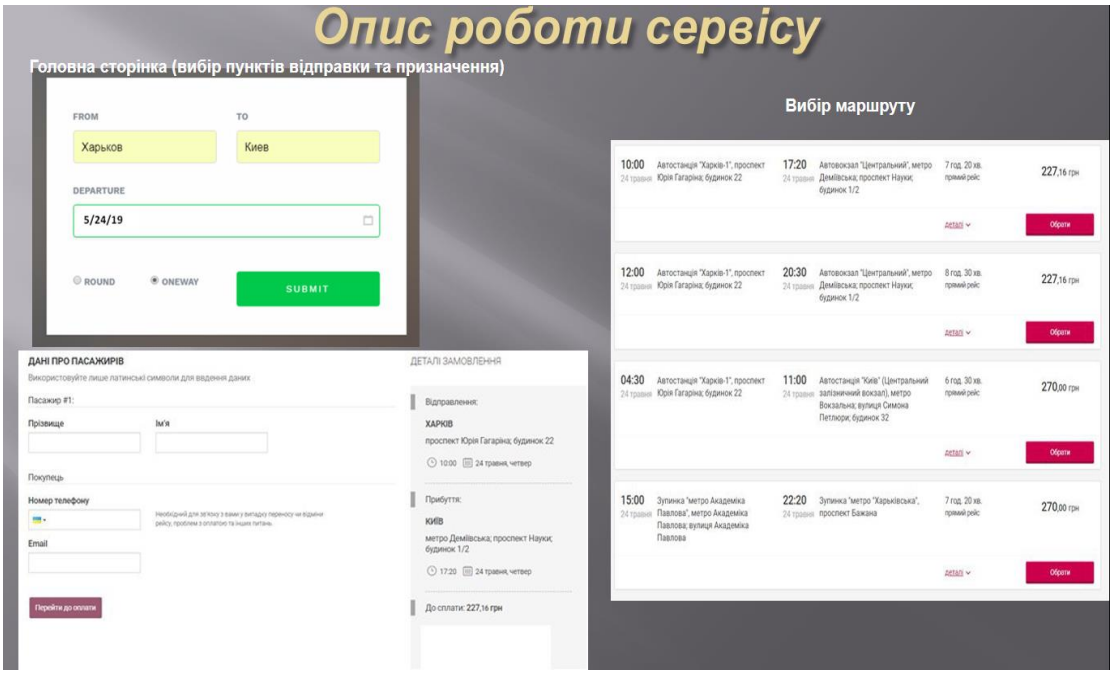

Рисунок В.11 – Слайд №11

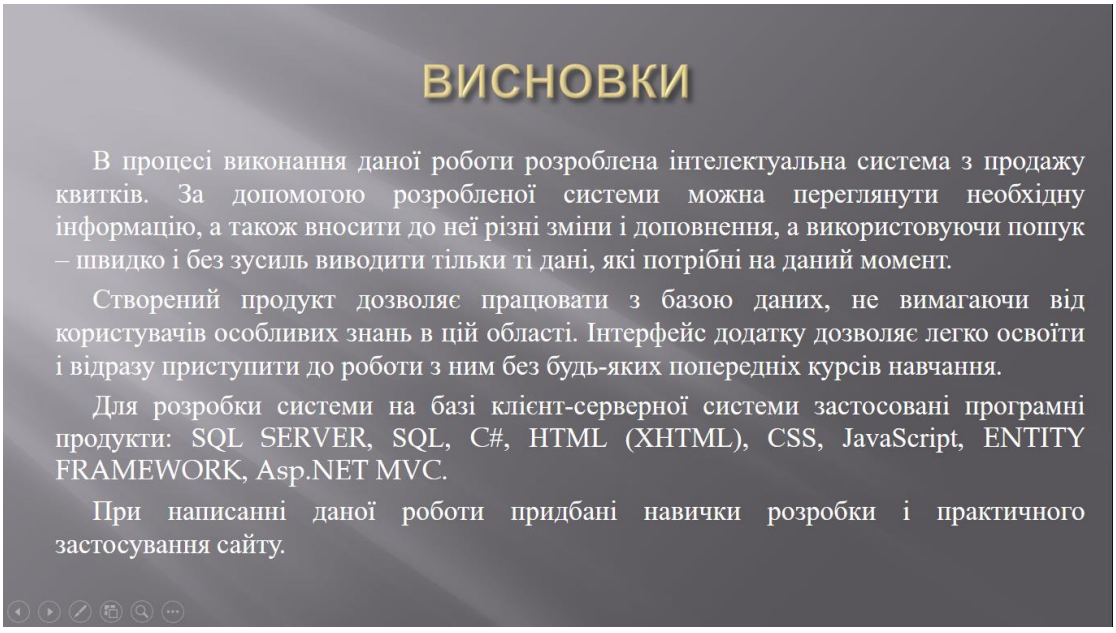

Рисунок В.12 – Слайд №12