Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни «Об’єктно-орієнтоване програмування»

Каса автовокзалу

Керівник , проф. Бондарєв В. М.

Студент гр. ПЗПІ-22-6 Афанасьєва О. М.

Комісія:

Проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В. М.

Ст. викл. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Черепанова Ю. Ю.

Ст. викл. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ляпота В. М.

Харків 2023

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра *програмної інженерії*

Рівень вищої освіти *перший (бакалаврський)*

Дисципліна *Об’єктно-орієнтоване програмування*

Спеціальність *121*  *Інженерія програмного забезпечення*

Освітня програма *Програмна інженерія*

Курс 1 .Група *ПЗПІ-22-6* . Семестр *2* .

***ЗАВДАННЯ***

***на курсовий проект студента***

***Афанасьєвої Оксани Миколаївни***

(Прізвище, Ім'я, По батькові)

1 Тема проекту:

*Каса автовокзалу*

2 Термін здачі студентом закінченого проекту: ***“16” - червня -2023 р***.

3 Вихідні дані до проекту:

*Специфікація програми, методичні вказівки до виконання курсової роботи*

4 Зміст розрахунково-пояснювальної записки:

*Вступ,* *опис вимог, проектування програми, інструкція користувача, висновки*

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |
| --- | --- | --- |
| № | Назва етапу | Термін виконання |
| 1 | Видача теми, узгодження і затвердження теми | 13.02.2023 –  14.03.2023 р. |
| 2 | Формулювання вимог до програми | 20.03.2023 –  23.03.2023 р. |
| 3 | Розробка підсистеми зберігання та пошуку даних | 01.04.2023 –  07.04.2023 р. |
| 4 | Розробка основних функцій | 21.05.2023 –  10.06.2023 р. |
| 5 | Розробка функції зберігання та завантаження даних | 25.05.2023 –  10.06.2023 р. |
| 6 | Тестування і доопрацювання розробленої програмної системи | 30.05.2023 –  10.06.2023 р. |
| 7 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 18.05.2023 –  10.06.2023 р. |
| 8 | Захист | 05.06.2023 –  16.06.2023 р. |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Афанасьєва Оксана Миколаївна

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Проф. Бондарєв В. М.

« 13 » лютого 2023 р.

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи: 71 с., 23 рис., 1 додаток.

КАСА АВТОВОКЗАЛУ, АВТОБУС, МАРШРУТ, КВИТКИ, КОРИСТУВАЧ, АДМІНІСТРАТОР, ЗВІТ, ООП, .NET, МОВА C#

Метою роботи є розробка програми «Каса автовокзалу», яка буде надавати користувачу можливість знайти необхідний маршрут, придбати або повернути квиток та переглянути усі придбані ним квитки. Також програма надає можливість адміністратору створювати, редагувати, видаляти маршрути, та переглядати посадкові відомості по маршрутах.

В результаті отримана програма, що дозволяє створювати маршрути з такими характеристиками як: номер, місто, дата та час відправлення та прибуття, місто та час зупинок, кількість місць в автобусі та ціна. Також є можливість купувати квитки, повертати квитки.

В процесі розробки було використано середовище Microsoft Visual Studio 2022, фреймворк Windows Forms, платформи .NET 6.0, мова програмування C#.

**ЗМІСТ**

[ВСТУП 6](#_Toc139262943)

[1 ОПИС ВИМОГ 7](#_Toc139262944)

[1.1 Сценарії використання програми 7](#_Toc139262945)

[1.2 Опис функцій 14](#_Toc139262946)

[2 ПРОЕКТУВАННЯ ПРОГРАМИ 25](#_Toc139262947)

[2.1 Архітектура програми 25](#_Toc139262948)

[2.2 Формат зберігання зовнішніх даних 25](#_Toc139262949)

[2.3 Діаграма класів 25](#_Toc139262950)

[3 ІНСТРУКЦІЯ КОРИСТУВАЧА 28](#_Toc139262951)

[3.1 Встановлення програми 28](#_Toc139262952)

[3.2 Використання функцій програми 28](#_Toc139262953)

[ВИСНОВКИ 40](#_Toc139262954)

[ДОДАТОК А 41](#_Toc139262955)

# ВСТУП

Кожного дня велика кількість людей подорожує у різні місця світу з допомогою громадського транспорту. Велика завантаженість на вокзалах, черги до кас, неможливість дізнатися чи є необхідний вам маршрут, або взагалі відсутність можливості дістатися до вокзалу. Не кожен хоче витрачати стільки часу та сил, щоб придбати квиток або дізнатися, що необхідного йому маршруту не існує. Саме тому програма «Каса автовокзалу» має собі за мету вирішити ці проблеми.

За допомогою програми «Каса автовокзалу» користувач може з легкістю переглянути усі можливі маршрути, знайти необхідний йому маршрут та без будь-яких проблем придбати квиток на нього. І якщо у користувача виникне потреба у повернені квитка, він також зможе це зробити прямо у застосунку. Це допомагає зекономити велику кількість часу та сил.

Програма орієнтована на широке коло користувачів. Вона не потребує значних ресурсів, тому нею може скористуватися будь-хто у кого є комп’ютер.

Застосунок має зручний інтерфейс, тому кожен зможе розібратися як ним користуватися.

# 1 ОПИС ВИМОГ

## 1.1 Сценарії використання програми

**Сценарій 1. Реєстрація**

Передумова

Користувач натиснув кнопку «Зареєструватися».

Основний сценарій

1. Користувач вводить ім’я та пароль і натискає кнопку «Зареєструватися».

2. Програма виводить повідомлення про успішну реєстрацію.

Додатковий сценарій

1. Користувач вводить ім’я та пароль і натискає кнопку «Зареєструватися».

2. Вже існує користувач з таким ім’ям.

3. Програма виводить повідомлення, що вже існує користувач з таким ім’ям.

Додатковий сценарій

1. Користувач не заповнює текстові поля та натискає кнопку «Зареєструватися».

2. Програма виводить повідомлення, що необхідно заповнити поля.

Додатковий сценарій

1. Користувач натискає кнопку «Я вже зареєстрований».

2. Програма відкриває вікно входу.

**Сценарій 2. Вхід на обліковий запис**

Передумова

Користувач вже є зареєстрованим та обрав кнопку «Увійти».

Основний сценарій

1. Користувач вводить ім’я та пароль і натискає кнопку «Увійти».

2. Програма відкриває головне вікно.

Додатковий сценарій

1. Користувач вводить ім’я та пароль і натискає кнопку «Увійти».

2. Пароль неправильний.

3. Програма виводить повідомлення, що пароль був введений неправильно.

Додатковий сценарій

1. Користувач вводить ім’я та пароль і натискає кнопку «Увійти».

2. Не існує користувача з таким ім’ям, або ім’я було введено неправильно.

3. Програма виводить повідомлення, що користувача з таким ім’ям не існує, або ім’я було введено неправильно.

Додатковий сценарій

1. Користувач не заповнює текстові поля та натискає кнопку «Зареєструватися».

2. Програма виводить повідомлення, що необхідно заповнити поля.

Додатковий сценарій

1. Користувач натискає кнопку «Зареєструватися».

2. Програма відкриває вікно реєстрації.

**Сценарій 3. Пошук маршруту**

Передумова

Користувач зайшов на свій акаунт і відкрив головне вікно програми.

Основний сценарій

1. Користувач заповнює поля на панелі пошуку і натискає кнопку «Пошук».

2. Програма знаходить маршрути, які задовольняють усім умовам пошуку, та виводить їх на екран.

Додатковий сценарій

1. Користувач заповнює поля на панелі пошуку і натискає кнопку «Пошук».

2. Програма не знаходить жодного маршруту, який задовольняє усім умовам пошуку, та виводить повідомлення, що маршрут не знайдено.

**Сценарій 4. Перегляд інформації про маршрут**

Передумова

Користувач зайшов на свій акаунт, відкрив головне вікно програми та знайшов необхідний йому маршрут.

Основний сценарій

1. Користувач двічі натискає на необхідний йому маршрут.

2. Програма показує вікно з інформацією про маршрут.

**Сценарій 5. Купівля квитка**

Передумова

Користувач зайшов на свій акаунт, відкрив головне вікно програми.

Основний сценарій

1. Користувач обрав маршрут зі списку та натиснув кнопку «Купити».

2. Відкривається вікно з полями для вводу даних.

3. Користувач заповнює поля та натискає кнопку «Придбати».

4. Квиток додається до особистого кабінету.

Додатковий сценарій

1. Користувач обрав маршрут зі списку та натиснув кнопку «Купити».

2. Відкривається вікно з полями для вводу даних.

3. Користувач заповнює не всі поля та натискає кнопку «Придбати».

4. Програма виводить повідомлення, що не всі поля були заповнені.

Додатковий сценарій

1. Користувач обрав маршрут зі списку та натиснув кнопку «Купити».

2. Відкривається вікно з полями для вводу даних.

3. Користувач натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи користувач точно хоче скасувати купівлю.

5. Користувач натискає кнопку «Так».

6. Програма відкриває головне вікно.

Додатковий сценарій

1. Користувач обрав маршрут зі списку та натиснув кнопку «Купити».

2. Відкривається вікно з полями для вводу даних.

3. Користувач натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи користувач точно хоче скасувати купівлю.

5. Користувач натискає кнопку «Ні».

6. Програма залишається на вікні купівлі квитка.

**Сценарій 6. Перегляд придбаних квитків**

Передумова

Користувач зайшов на свій акаунт та відкрив вікно «Мої подорожі».

Основний сценарій

1. Програма виводить усі придбані користувачем квитки на заплановані подорожі.

2. Користувач двічі натискає на необхідний йому квиток.

3. Програма показує вікно з інформацією про придбаний квиток.

Додатковий сценарій

1. Користувач не придбав жодного квитка.

2. Програма виводить повідомлення, що у користувача немає заброньованих подорожей.

**Сценарій 7.** **Скасування квитка**

Передумова

Користувач зайшов на свій акаунт, придбав квиток та відкрив вікно «Мої подорожі».

Основний сценарій

1. Користувач обирає квиток та натискає кнопку «Повернути квиток».

2. Програма виводить повідомлення з питанням чи користувач точно хоче скасувати бронювання.

3. Користувач натискає на кнопку «Так».

4. Бронювання скасовано.

Додатковий сценарій

1. Користувач обирає квиток та натискає кнопку «Повернути квиток».

2. Програма виводить повідомлення з питанням чи користувач точно хоче скасувати бронювання.

3. Користувач натискає на кнопку «Ні».

4. Бронювання не скасовано.

**Сценарій 8. Створення маршруту**

Передумова

Адміністратор зайшов на свій акаунт та відкрив головне вікно.

Основний сценарій

1. Адміністратор натискає кнопку «Створити».

2. Програма відкриває вікно з полями для введення даних.

3. Адміністратор вводить дані та натискає кнопку «Зберегти».

4. Програма зберігає новий маршрут.

Додатковий сценарій

1. Адміністратор натискає кнопку «Створити».

2. Програма відкриває вікно з полями для введення даних.

3. Адміністратор вводить не всі дані та натискає кнопку «Зберегти».

4. Програма виводить повідомлення, що не всі дані були введені.

Додатковий сценарій

1. Адміністратор натискає кнопку «Створити».

2. Відкривається вікно з полями для введення даних.

3. Адміністратор натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи він точно хоче скасувати створення маршруту.

5. Адміністратор натискає кнопку «Так».

6. Програма відкриває головне вікно.

Додатковий сценарій

1. Адміністратор натискає кнопку «Створити».

2. Відкривається вікно з полями для введення даних.

3. Адміністратор натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи він точно хоче скасувати створення маршруту.

5. Адміністратор натискає кнопку «Ні».

6. Програма залишається на вікні створення.

**Сценарій 9. Редагування даних про маршрут**

Передумова

Адміністратор зайшов на свій акаунт, відкрив головне вікно та знайшов необхідний маршрут.

Основний сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Редагувати».

2. Програма відкриває вікно з інформацією про маршрут.

3. Адміністратор вводить нові дані в поля та натискає кнопку «Зберегти».

4. Програма зберігає нову інформацію.

Додатковий сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Редагувати».

2. Програма відкриває вікно з інформацією про маршрут.

3. Адміністратор заповнює не усі поля та натискає кнопку «Зберегти».

4. Програма виводить повідомлення, що не всі дані були введені.

Додатковий сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Редагувати».

2. Програма відкриває вікно з інформацією про маршрут.

3. Адміністратор натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи він точно хоче скасувати редагування маршруту.

5. Адміністратор натискає кнопку «Так».

6. Програма відкриває головне вікно.

Додатковий сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Редагувати».

2. Програма відкриває вікно з інформацією про маршрут.

3. Адміністратор натискає кнопку «Скасувати».

4. Програма виводить повідомлення чи він точно хоче скасувати редагування маршруту.

5. Адміністратор натискає кнопку «Ні».

6. Програма залишається на вікні редагування.

**Сценарій 10. Видалення маршруту**

Передумова

Адміністратор зайшов на свій акаунт, відкрив головне вікно та знайшов необхідний маршрут.

Основний сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Видалити».

2. Програма виводить повідомлення з питанням чи адміністратор точно хоче видалити маршрут.

3. Адміністратор натискає на кнопку «Так».

4. Маршрут видалено.

Додатковий сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Видалити».

2. Програма виводить повідомлення з питанням чи адміністратор точно хоче видалити маршрут.

3. Адміністратор натискає на кнопку «Ні».

4. Маршрут не видалено.

**Сценарій 11. Перегляд посадкової відомості**

Передумова

Адміністратор зайшов на свій акаунт, відкрив головне вікно та знайшов необхідний маршрут.

Основний сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Посадкова відомість».

2. Програма відкриває вікно з посадковою відомістю цього маршруту.

Додатковий сценарій

1. Адміністратор обрав необхідний маршрут та натиснув кнопку «Посадкова відомість».

2. Ніхто не придбав квитки на цей маршрут.

3. Програма виводить повідомлення, що на цей маршрут поки не купленого жодного квитка.

## 1.2 Опис функцій

**Функція 1. Реєстрація**

Панель реєстрація розташована на окремому вікні (рис. 1.1) та містить такі текстові поля:

* ім’я користувача – унікальне ім’я користувача;
* пароль – пароль користувача.

Зображення, що містить знімок екрана, текст, Прямокутник, схема

Автоматично згенерований опис

Рисунок 1.1 – Вікно реєстрації

Якщо в полі відсутні дані, то, при натисканні на кнопку, програма виводить повідомлення та користувач має заповнити необхідні поля.

Внизу по центру панелі є кнопка «Зареєструватися». Також під нею є кнопка «Я вже зареєстрований», при натисканні на неї, відкриється вікно входу.

Повідомлення про помилку, якщо вже існує користувач з таким ім’ям, буде з’являтися в центрі сторінки.

Якщо реєстрація пройшла успішно, програма виводить повідомлення про успішну реєстрацію та відкриває вікно входу.

**Функція 2. Вхід**

Панель входу розташована на окремому вікні (рис. 1.2) та містить такі текстові поля:

* ім’я користувача – унікальне ім’я користувача;
* пароль – пароль користувача.

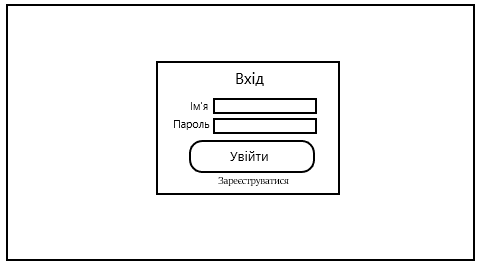


Рисунок 1.2 – Вікно входу

Якщо в полі відсутні дані, то, при натисканні на кнопку, програма виводить повідомлення та користувач має заповнити необхідні поля.

Внизу по центру панелі є кнопка «Увійти». Також під нею є кнопка «Зареєструватися», при натисканні на цю кнопку, відкриється вікно реєстрації.

Повідомлення про помилку, якщо неправильний пароль чи ім’я, або не існує користувача з таким ім’ям, буде з’являтися в центрі сторінки.

Якщо вхід пройшов успішно, програма відкриває головне вікно (рис. 1.3).

**Функція 3. Пошук маршруту**

Панель пошуку маршруту розташована на головному вікні програми (рис. 1.3) і містить такі поля:

* від – назва міста відправлення;
* до – назва пункту призначення;
* дата – дата подорожі.

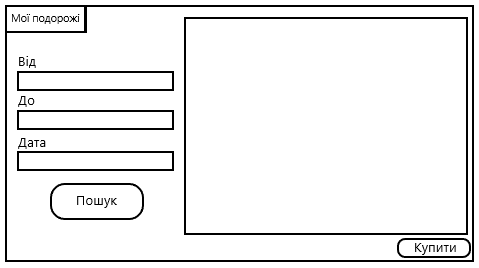


Рисунок 1.3 – Головне вікно користувача

В текстових полях регістр не важливий. Якщо в текстовому полі відсутні дані, то при натисканні на кнопку, нічого не відбувається та користувач має знову ввести дані. Якщо користувач не обирає певну дату, то автоматично обирається сьогоднішня дата.

Внизу по центру панелі пошуку є кнопка «Пошук».

Результат пошуку буде відображатися на головному вікні праворуч від панелі пошуку. Опис маршруту складається з номера маршруту, міста відправлення та прибуття, дати та часу відправлення і прибуття, кількості вільних місць і ціни.

Якщо жодного маршруту не знайдено, то користувач бачить повідомлення: “Такого маршруту не знайдено”.

**Функція 4. Перегляд інформації про маршрут**

Перегляд інформації про маршрут доступний на головному вікні. Панель інформації про маршрут розташована на окремому вікні (рис. 1.4).

Зображення, що містить текст, знімок екрана, ряд, Шрифт

Автоматично згенерований опис

Рисунок 1.4 – Вікно з інформацією про маршрут

Інформація про маршрут складається з номера маршруту, міста відправлення та прибуття, дати та часу відправлення і прибуття, зупинок, кількості вільних місць та ціни.

**Функція 5. Купівля квитка**

Кнопка купівлі квитка розташована на головному вікні (рис.1.3). Панель купівлі квитка знаходиться на окремому вікні (рис. 1.5).

В текстових полях регістр не важливий. Якщо в текстовому полі відсутні дані, то програма виводить повідомлення: “Заповніть усі поля” та користувач має знову ввести дані.

В правому нижньому кутку розташовані кнопки «Скасувати» та «Придбати».

Зображення, що містить текст, знімок екрана, схема, ряд

Автоматично згенерований опис

Рисунок 1.5 – Вікно купівлі квитка

Після успішного придбання квитка програма зберігає квиток до особистого кабінету та відкриває головне вікно.

**Функція 6. Перегляд придбаних квитків**

Панель перегляду придбаних квитків розташована на вікні «Мої подорожі» (рис. 1.6).

Зображення, що містить текст, знімок екрана, схема, ряд

Автоматично згенерований опис

Рисунок 1.6 – Вікно «Мої подорожі»

В правому нижньому кутку розташована кнопка «Повернути квиток». А в лівому нижньому кутку – кнопка «Назад».

Інформація про придбаний квиток складається з номера маршруту, міста відправлення та прибуття, дати та часу відправлення і прибуття та кількості пасажирів, на яких був придбаний цей квиток.

Якщо у користувача немає активних квитків, то користувач бачить повідомлення: “У Вас поки немає заброньованих подорожей”.

Також, натиснувши двічі на необхідний квиток, користувач може подивитися більш детальну інформацію про маршрут (рис. 1.4).

**Функція 7. Скасування квитка**

Панель скасування квитка розташована на вікні «Мої подорожі» (рис.1.6).

Повідомлення-попередження (рис. 1.7) з’являється в центрі екрана.

Зображення, що містить Прямокутник, ескіз, ряд, схема

Автоматично згенерований опис

Рисунок 1.7 – Повідомлення-попередження

В лівому краю повідомлення-попередження розташована кнопка «Так», а в правому – «Ні».

Результатом видалення квитка буде зникнення його зі списку квитків у вікні «Мої подорожі».

**Функція 8. Створення маршруту**

Функція створення маршруту доступна лише адміністратору. Панель створення нового маршруту знаходиться на окремому вікні (рис. 1.9) та містить такі текстові поля:

* маршрут № – номер маршруту;
* звідки – місто відправлення;
* куди – пункт призначення;
* дата – дата відправлення та прибуття;
* час – час відправлення, прибуття та зупинок;
* назва – назва міст зупинок;
* кількість вільних місць – число місць в автобусі;
* ціна – ціна квитка.

Кнопка «Створити маршрут» знаходиться в лівому верхньому кутку на головному вікні (рис. 1.8).

Зображення, що містить знімок екрана, схема, Прямокутник, ряд

Автоматично згенерований опис

Рисунок 1.8 – Головне вікно адміністратора

Зображення, що містить знімок екрана, текст, схема, Паралель

Автоматично згенерований опис

Рисунок 1.9 – Вікно створення маршруту

В текстових полях регістр не важливий. Якщо в текстовому полі відсутні дані, то програма виводить повідомлення: “Заповніть усі поля” та користувач має знову ввести дані.

В правому нижньому кутку розташовані кнопки «Скасувати» та «Зберегти».

При скасуванні створення, програма виводить повідомлення, що дані не будуть збережені.

**Функція 9. Редагування даних про маршрут**

Функція редагування маршруту доступна лише адміністратору. Панель редагування нового маршруту знаходиться на окремому вікні (рис. 1.9) та містить такі текстові поля:

* маршрут № – номер маршруту;
* звідки – місто відправлення;
* куди – пункт призначення;
* дата – дата відправлення та прибуття;
* час – час відправлення, прибуття та зупинок;
* назва – назва міст зупинок;
* кількість вільних місць – число місць в автобусі;
* ціна – ціна квитка.

Кнопка «Редагувати» знаходиться в правому нижньому кутку на головному вікні (рис. 1.8).

В текстових полях регістр не важливий. Якщо в текстовому полі відсутні дані, то програма виводить повідомлення: “Заповніть усі поля” та користувач має знову ввести дані.

В правому нижньому кутку розташовані кнопки «Скасувати» та «Зберегти».

При скасуванні створення, програма виводить повідомлення, що оновлені дані не будуть збережені.

**Функція 10. Видалення маршруту**

Функція видалення маршруту доступна лише адміністратору та розташована в правому нижньому кутку на головному вікні (рис. 1.8).

При видаленні, програма виводить повідомлення-попередження (рис.1.7), чи адміністратор точно хоче видалити цей маршрут.

В лівому краю повідомлення-попередження розташована кнопка «Так», а в правому – «Ні».

**Функція 11. Перегляд посадкової відомості**

Перегляд посадкової відомості доступний лише адміністратору. Кнопка «Посадкова відомість» розташована в правому нижньому кутку на головному вікні (рис. 1.8). Панель посадкової відомості розташована на окремому вікні (рис. 1.10).

Зображення, що містить текст, знімок екрана, Шрифт, ряд

Автоматично згенерований опис

Рисунок 1.10 – Вікно «Посадкова відомість»

Посадкова відомість складається з номера маршруту, імені та прізвища пасажира, кількості заброньованих місць пасажиром, сплаченої суми.

# 2 ПРОЕКТУВАННЯ ПРОГРАМИ

## 2.1 Архітектура програми

Я вирішила обрати архітектуру під назвою «Настільна з GUI». Оскільки GUI надає можливість взаємодіяти з програмою через графічні елементи, такі як: кнопка, текстове поле, списки та інше. Це робить програму інтуїтивно зрозумілою для користувача.

## 2.2 Формат зберігання зовнішніх даних

Для зберігання даних використовується формат даних Json. Коли дані зберігаються, створюються файли у форматі .json, що містять структуровані дані у формі тексту. Також Json має структуру, яку легко зрозуміти та зручно використовувати.

## 2.3 Діаграма класів

Діаграма основних класів зображена на рисунку 2.1.

Зображення, що містить текст, квитанція, Паралель, Шрифт

Автоматично згенерований опис

Рисунок 2.1 – Діаграма класів

Клас RouteData представляє конкретний об’єкт «Маршрут». Цей клас містить властивості (Number, FromName, ToName, DateTimeFrom, TimeFrom, DateTimeTo, TimeTo, Stops, TimeStop, FreeSeats, Price), які використовуються для зберігання даних маршруту.

Клас Registration містить метод, що перевіряє чи не існує користувач з таким ім’ям.

Клас UserData містить властивості (Name, Password, IsAdmin), які використовуються для зберігання даних про користувача.

Клас LogIn має три методи, які перевіряють правильність пароля та імені, та чи користувач є адміністратором.

Клас HomePages відповідає за відображення усіх маршрутів, видалення вже прострочених маршрутів, пошук маршруту та виведення знайдених маршрутів.

Клас TicketData визначає властивості (Number, Name, Surname, Seats, Price, Nickname), які використовуються для зберігання даних про квиток придбаний користувачем.

Клас BuyTicket містить один метод, який відповідає за збереження необхідних даних про маршрут, квиток на який купляється.

Клас SaveLoadData має два методи, які відповідають за перетворення списків у файл формату .json та навпаки, перетворення файлу .json у список з даними про об’єкт.

# 3 ІНСТРУКЦІЯ КОРИСТУВАЧА

## 3.1 Встановлення програми

Для роботи застосунку, на цільовому комп’ютері має бути встановлена програма Microsoft Visual Studio 2022 та версія .NET 6.0. Далі необхідно завантажити архів з вихідним кодом програми та видобути усе з архіву. Відкривши програму Microsoft Visual Studio 2022, натиснути на кнопку «Open a project or solution» і знайти та вибрати файл проекту. Коли вихідний код відкриється, натиснути на кнопку «Start» вгорі вікна.

## 3.2 Використання функцій програми

**Реєстрація**

Коли програма завантажиться, ви побачите вікно реєстрації (рис. 3.1).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.1 – Вікно реєстрації

Щоб зареєструватися, вам необхідно ввести ім’я та пароль, і натиснути кнопку «Зареєструватися». Якщо ви вже зареєстровані, ви можете натиснути на посилання «Я вже зареєстрований», що знаходиться під кнопкою.

**Вхід**

Після того, як ви зареєструєтеся або натиснете на посилання «Я вже зареєстрований», ви побачите вікно входу (рис. 3.2).

Зображення, що містить текст, знімок екрана, Шрифт, ряд

Автоматично згенерований опис

Рисунок 3.2 – Вікно входу

Щоб зайти, вам необхідно ввести ім’я та пароль, і натиснути кнопку «Увійти».

**Головна сторінка користувача**

Якщо ви увійшли на акаунт користувача, ви побачите головну сторінку (рис. 3.3).

Зображення, що містить текст, знімок екрана, програмне забезпечення, число

Автоматично згенерований опис

Рисунок 3.3 – Головна сторінка користувача

На головній сторінці користувача можна виконати пошук необхідного маршруту, перейти на сторінку «Мої подорожі», придбати квиток на маршрут та вийти з акаунта.

При подвійному натисканні на маршрут, ви можете побачити більш детальну інформацію про нього (рис. 3.4).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.4 – Інформація про маршрут

**Функція пошуку**

Пошук розташований ліворуч на головній сторінці (рис. 3.5).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.5 – Пошук

Вам необхідно ввести звідки, куди та коли ви хочете поїхати. В текстових полях регістр не важливий. Після того як ви заповните усі необхідні поля та натиснете кнопку «Пошук», результат пошуку висвітиться праворуч у списку (рис. 3.6).

Зображення, що містить текст, знімок екрана, монітор, програмне забезпечення

Автоматично згенерований опис

Рисунок 3.6 – Результат пошуку

Ви також можете знову вивести усі маршрути, для цього необхідно натиснути на кнопку «Вивести усі маршрути».

**Функція купівлі**

Щоб придбати квиток на маршрут, вам необхідно обрати маршрут зі списку та натиснути на кнопку «Купити», що розташована в нижньому правому кутку вікна (див. рис. 3.3). Після цього у вас відкриється вікно купівлі квитка (рис. 3.7).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.7 – Вікно купівлі квитка

На цьому вікні розташована невелика інформація про маршрут, квиток на який ви хочете придбати. Вам необхідно ввести свої ім’я та прізвище, та обрати на скількох людей ви бронюєте місця. Зі зміною кількості пасажирів, ціна в нижньому лівому кутку змінюється.

Якщо ви ввели усі необхідні дані, вам потрібно натиснути на кнопку «Придбати», і квиток буде додано до вашого кабінету.

Якщо ви не хочете купувати квиток, ви можете натиснути на кнопку «Скасувати» та повернутися на головну сторінку.

**«Мої подорожі»**

Якщо ви хочете переглянути придбані квитки, вам необхідно натиснути на посилання в верхньому лівому кутку головного вікна (див. рис. 3.3). Після того як ви натиснете на посилання, ви побачите вікно «Мої подорожі» (рис.3.8).

Зображення, що містить текст, знімок екрана, монітор, Шрифт

Автоматично згенерований опис

Рисунок 3.8 – Вікно «Мої подорожі»

На цій сторінці ви можете переглянути інформацію про ваші квитки та повернути їх. Також при подвійному натисканні на маршрут, ви побачите більш детальну інформацію про нього.

Якщо ви захочете повернутися на головну сторінку, вам необхідно буде натиснути на посилання «Назад», розташоване в правому нижньому кутку (див. рис. 3.8).

**Функція повернення квитка**

Щоб повернути квиток, вам необхідно обрати квиток зі списку та натиснути на кнопку «Повернути квиток», що розташована в нижньому правому кутку (див. рис. 3.8).

**Функція виходу**

Якщо ви захочете вийти зі свого акаунту, вам потрібно буде натиснути на кнопку «Вийти», що розташована в лівому нижньому кутку головної сторінки (див. рис. 3.3). Після цього ви побачите перед собою вікно входу (див. рис. 3.2).

**Головна сторінка адміністратора**

Щоб зайти на акаунт адміністратора, необхідно ввести ім’я «Tom» та пароль «963». Після цього ви побачите головне вікно адміністратора (рис.3.9).

**Зображення, що містить текст, знімок екрана, програмне забезпечення, число

Автоматично згенерований опис**

Рисунок 3.9 – Головне вікно адміністратора

На головній сторінці адміністратора можна створити новий маршрут, відредагувати, видалити або переглянути посадкову відомість маршруту зі списку.

**Функція створення маршруту**

Щоб створити маршрут, необхідно натиснути на кнопку «Створити», що розташована внизу головного вікна адміністратора (див. рис. 3.9). Після цього ви побачите вікно створення маршруту (рис. 3.10).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.10 – Вікно створення маршруту

Щоб створити новий маршрут, необхідно заповнити усі поля, окрім зупинок, вони є необов’язковими, та натиснути кнопку «Зберегти». Номери маршрутів не можуть повторюватися.

Щоб додати зупинку, потрібно ввести назву зупинки та обрати час, о котрій буде ця зупинка, та натиснути на кнопку «Додати зупинку». А щоб видалити зупинку, просто оберіть зупинку, яку хочете видалити, та натисніть кнопку «Видалити зупинку».

Якщо ви не хочете створювати новий маршрут, ви можете натиснути на кнопку «Скасувати», та вас поверне на головне вікно.

**Функція редагування**

Щоб відредагувати маршрут, потрібно обрати необхідний маршрут зі списку та натиснути на кнопку «Редагувати», розташовану внизу головного вікна адміністратора (див. рис. 3.9). Після цього, відкриється вікно редагування (рис. 3.11).

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 3.11 – Вікно редагування

Коли вікно відкриється, ви одразу побачите поточну інформацію про маршрут, ви можете її змінити. Після внесення змін, натисніть на кнопку «Зберегти».

Якщо ви не хочете редагувати цей маршрут, натисніть на кнопку «Скасувати», та вас перенесе на головну сторінку (див. рис. 3.9).

**Функція видалення**

Щоб видалити маршрут, необхідно обрати маршрут зі списку та натиснути на кнопку «Видалити», що розташована внизу головної сторінки адміністратора (див. рис. 3.9).

**Функція перегляду посадкової відомості**

Якщо ви хочете переглянути посадкову відомість до маршруту, вам потрібно обрати маршрут зі списку та натиснути на кнопку «Посадкова відомість», розташовану внизу головної сторінки адміністратора (див. рис.3.9). Після цього відкриється вікно з посадковою відомістю певного маршруту (рис. 3.12).

Зображення, що містить текст, знімок екрана, ряд, число

Автоматично згенерований опис

Рисунок 3.12 – Вікно перегляду посадкової відомості

На цьому вікні ви можете побачити дані про людей, що придбали квиток на цей маршрут.

Щоб повернутися на головну сторінку, натисніть на посилання «Назад».

# ВИСНОВКИ

Протягом виконання курсової роботи, була розроблена програма «Каса автовокзалу». З допомогою цієї програми, користувачі можуть шукати маршрути, придбати, повернути та переглянути усі придбані ним квитки. Також адміністратор може створювати, редагувати, видаляти маршрути та переглядати посадкову відомість по маршрутах.

Поставлені на початку роботи, цілі здебільшого були виконані, проте ще є що змінити та додати. Існують невеликі недоліки, які потребують подальшого вивчення та вирішення.

Оскільки це був мій перший проект, я зіштовхнулася з великою кількістю проблем. Більшість з яких була успішно вирішена.

Створення програми допомогло ще ближче ознайомитися з мовою програмування C#, навчитися працювати з формою збереження Json та отримати досвід в роботі з Windows Forms.

# ДОДАТОК А

Вихідний код програми

**Registration.cs**

using BusStationCashDesk.Windows\_Forms;

using System.Text.Json;

using System.IO;

using BusStationCashDesk.Classes;

namespace BusStationCashDesk

{

public partial class Registration : Form

{

private SaveLoadData<UserData> file;

private List<UserData> userList;

public Registration()

{

InitializeComponent();

file = new SaveLoadData<UserData>("userData.json");

userList = file.Load();

}

private void Registration\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void Form1\_Load(object sender, EventArgs e)

{

this.FormClosing += Registration\_FormClosing;

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

LogIn loginForm = new LogIn();

loginForm.Show();

this.Hide();

}

private void buttonRegistration\_Click(object sender, EventArgs e)

{

string name = nameTextBox.Text;

string password = passwordTextBox.Text;

if (string.IsNullOrEmpty(name) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Введіть ім'я та пароль.", "Помилка реєстрації",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (IsUserRegistered(name) == true)

{

MessageBox.Show("Користувач з таким ім'ям вже існує.", "Помилка реєстрації",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (userList == null)

{

List<UserData> userList = new List<UserData>();

UserData newUser = new UserData(name, password, false);

userList.Add(newUser);

file.Save(userList);

}

else

{

UserData newUser = new UserData(name, password, false);

userList.Add(newUser);

file.Save(userList);

}

MessageBox.Show("Ви зареєстровані.", "Реєстрація", MessageBoxButtons.OK,

MessageBoxIcon.Information);

LogIn form = new LogIn();

form.Show();

this.Hide();

}

private bool IsUserRegistered(string name)

{

if (userList == null)

{

return false;

}

else

{

foreach (var user in userList)

{

if (user.Name == name)

{

return true;

}

}

}

return false;

}

}

}

**LogIn.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Models;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class LogIn : Form

{

private SaveLoadData<UserData> file;

private List<UserData> userList;

private SaveLoadData<string> file1;

private List<string> nicknameList;

public LogIn()

{

InitializeComponent();

file = new SaveLoadData<UserData>("userData.json");

userList = file.Load();

file1 = new SaveLoadData<string>("nicknameData.json");

nicknameList = new List<string>();

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Registration form = new Registration();

form.Show();

this.Hide();

}

private void buttonLogIn\_Click(object sender, EventArgs e)

{

string name = nameTextBox.Text;

string password = passwordTextBox.Text;

if (string.IsNullOrEmpty(name) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Введіть ім'я та пароль.", "Помилка входу",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (IsUserRegistered(name) == false)

{

MessageBox.Show("Користувача з таким ім'ям не існує або ім'я було введено неправильно.",

"Помилка входу", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (IsPasswordRight(name, password) == true)

{

if (IsAdministrator(name) == false)

{

nicknameList.Add(nameTextBox.Text);

file1.Save(nicknameList);

HomePage form = new HomePage();

form.Show();

this.Hide();

}

else

{

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

}

else

{

MessageBox.Show("Неправильний пароль.", "Помилка входу",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

}

private bool IsPasswordRight(string name, string password)

{

if (userList == null)

{

return false;

}

else

{

for (int i = 0; i < userList.Count; i++)

{

if (userList[i].Name == name && userList[i].Password == password)

{

return true;

}

}

}

return false;

}

private bool IsAdministrator(string name)

{

if (userList == null)

{

return false;

}

else

{

for (int i = 0; i < userList.Count; i++)

{

if (userList[i].Name == name)

{

return userList[i].IsAdmin;

}

}

}

return false;

}

private bool IsUserRegistered(string name)

{

if (userList == null)

{

return false;

}

else

{

foreach (var user in userList)

{

if (user.Name == name)

{

return true;

}

}

}

return false;

}

private void LogIn\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void LogIn\_Load(object sender, EventArgs e)

{

this.FormClosing += LogIn\_FormClosing;

}

}

}

**HomePage.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Forms;

using Microsoft.VisualBasic.Devices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.ComponentModel.Design.ObjectSelectorEditor;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class HomePage : Form

{

private SaveLoadData<RouteData> route;

private List<RouteData> routeList;

private HomePages homePages;

public HomePage()

{

InitializeComponent();

route = new SaveLoadData<RouteData>("routeData.json");

routeList = route.Load();

homePages = new HomePages();

dateTimePicker.MinDate = DateTime.Today;

this.FormClosing += HomePage\_FormClosing;

}

private void HomePage\_FormClosing(object? sender, FormClosingEventArgs e)

{

if(e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void HomePage\_Load(object sender, EventArgs e)

{

homePages.RemoveExpiredRoutes(routeList);

homePages.DisplayRoute(routeList, listRoute);

}

private void linkLabelMyTravels\_LinkClicked(object sender,

LinkLabelLinkClickedEventArgs e)

{

MyTravel form = new MyTravel();

form.Show();

this.Hide();

}

private void buttonSearch\_Click\_1(object sender, EventArgs e)

{

homePages.Search(fromTextBox.Text.ToLower(), toTextBox.Text.ToLower(),

dateTimePicker.Value.Date, listRoute);

}

private void listRoute\_DoubleClick(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

InformationRoute informationRoute = new InformationRoute(selected);

informationRoute.ShowDialog();

}

}

}

}

private void buttonAllRoute\_Click(object sender, EventArgs e)

{

homePages.DisplayRoute(routeList, listRoute);

}

private void buttonBuy\_Click(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

BuyTicket form = new BuyTicket(selected);

form.Show();

this.Hide();

}

}

}

else MessageBox.Show("Виберіть маршрут, на який хочете придбати квиток.",

"Бронювання", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void buttonLogOut\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Ви впевнені, що хочете вийти з облікового запису?",

"Підтвердження виходу", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

LogIn form = new LogIn();

form.Show();

this.Hide();

}

}

}

}

**HomePageAdministration.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Forms;

using BusStationCashDesk.Models;

using Microsoft.VisualBasic.Devices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class HomePageAdministration : Form

{

private SaveLoadData<RouteData> route;

private List<RouteData> routeList;

private SaveLoadData<TicketData> file1;

private List<TicketData> ticketList;

private HomePages homePages;

public HomePageAdministration()

{

InitializeComponent();

route = new SaveLoadData<RouteData>("routeData.json");

routeList = route.Load();

file1 = new SaveLoadData<TicketData>("ticketData.json");

ticketList = file1.Load();

homePages = new HomePages();

dateTimePicker.MinDate = DateTime.Today;

this.FormClosing += HomePageAdministration\_FormClosing;

}

private void buttonCreate\_Click(object sender, EventArgs e)

{

Route form = new Route();

form.Show();

this.Hide();

}

private void HomePageAdministration\_FormClosing(object? sender,

FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void HomePageAdministration\_Load(object sender, EventArgs e)

{

homePages.RemoveExpiredRoutes(routeList);

homePages.DisplayRoute(routeList, listRoute);

}

private void buttonSearch\_Click(object sender, EventArgs e)

{

homePages.Search(fromTextBox.Text.ToLower(), toTextBox.Text.ToLower(),

dateTimePicker.Value.Date, listRoute);

}

private void listRoute\_SelectedIndexChanged(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

remove.Enabled = true;

}

else remove.Enabled = false;

}

private void buttonRemove\_Click(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

DialogResult result = MessageBox.Show("Ви впевнені, що хочете видалити цей маршрут?",

"Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if(result == DialogResult.Yes)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected)

{

routeList.RemoveAt(i);

break;

}

}

route.Save(routeList);

homePages.DisplayRoute(routeList, listRoute);

}

}

else MessageBox.Show("Виберіть маршрут, який хочете видалити.",

"Видалення", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void listRoute\_DoubleClick(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

InformationRoute informationRoute = new InformationRoute(selected);

informationRoute.ShowDialog();

}

}

}

}

private void buttonEdit\_Click(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

Edit form = new Edit(selected);

form.Show();

this.Hide();

}

}

}

else MessageBox.Show("Виберіть маршрут, який хочете відредагувати.",

"Редагування", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private void buttonAllRoute\_Click(object sender, EventArgs e)

{

homePages.DisplayRoute(routeList, listRoute);

}

private void buttonLogOut\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Ви впевнені, що хочете вийти з облікового запису?",

"Підтвердження виходу", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

LogIn form = new LogIn();

form.Show();

this.Hide();

}

}

private void buttonBoardingList\_Click(object sender, EventArgs e)

{

if (listRoute.SelectedItems.Count > 0)

{

string selected = listRoute.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

int found = 0;

for (int j = 0; j < ticketList.Count; j++)

{

if (ticketList[j].Number == selected)

{

found++;

}

}

if (found == 0)

{

MessageBox.Show("Ніхто не придбав квиток на цей маршрут.",

"Посадкова відомість", MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

else

{

BoardingList form = new BoardingList(selected);

form.Show();

this.Hide();

}

}

}

}

else MessageBox.Show("Виберіть маршрут, посадкову відомість якого хочете переглянути.",

"Посадкова відомість", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

**BoardingList.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Models;

using Microsoft.VisualBasic.Devices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.ComponentModel.Design.ObjectSelectorEditor;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class BoardingList : Form

{

private SaveLoadData<TicketData> file;

private List<TicketData> ticketList;

private string selected;

public BoardingList(string number)

{

InitializeComponent();

file = new SaveLoadData<TicketData>("ticketData.json");

ticketList = file.Load();

selected = number;

}

private void BoardingList\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void BoardingList\_Load(object sender, EventArgs e)

{

int found = 0;

numberRouteTextBox.Text = selected;

for (int i = 0; i < ticketList.Count; i++)

{

if (ticketList[i].Number == selected)

{

DisplayRoute(ticketList[i]);

found++;

}

}

if (found == 0)

{

MessageBox.Show("Ніхто не придбав квиток на цей маршрут.",

"Посадкова відомість", MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

this.FormClosing += BoardingList\_FormClosing;

}

private void linkLabelBack\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

private void DisplayRoute(TicketData ticket)

{

ListViewItem item = new ListViewItem(ticket.Name);

item.SubItems.Add(ticket.Surname);

item.SubItems.Add(ticket.Seats.ToString());

item.SubItems.Add(ticket.Price.ToString());

passengerList.Items.Add(item);

}

}

}

**BuyTicket.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Models;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

using static System.ComponentModel.Design.ObjectSelectorEditor;

using static System.Formats.Asn1.AsnWriter;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class BuyTicket : Form

{

private SaveLoadData<RouteData> file;

private List<RouteData> routeList;

private string selected;

private SaveLoadData<string> file1;

private List<string> nicknameList;

private SaveLoadData<TicketData> file2;

private List<TicketData> ticketList;

private int index;

public BuyTicket(string number)

{

InitializeComponent();

file = new SaveLoadData<RouteData>("routeData.json");

routeList = file.Load();

selected = number;

file1 = new SaveLoadData<string>("nicknameData.json");

nicknameList = file1.Load();

file2 = new SaveLoadData<TicketData>("ticketData.json");

ticketList = file2.Load();

index = 0;

}

private void BuyTicket\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

DialogResult result = MessageBox.Show("Дані не будуть збережені! " +

"Ви впевнені, що хочете закрити сторінку?", "Попередження",

MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

Application.Exit();

}

else e.Cancel = true;

}

}

private void BuyTicket\_Load(object sender, EventArgs e)

{

this.FormClosing += BuyTicket\_FormClosing;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

TextInfo title = CultureInfo.InvariantCulture.TextInfo;

numberRouteTextBox.Text = routeList[i].Number;

fromTextBox.Text = title.ToTitleCase(routeList[i].FromName ?? string.Empty);

toTextBox.Text = title.ToTitleCase(routeList[i].ToName ?? string.Empty);

fromDateTimePicker.Value = routeList[i].DateTimeFrom;

timeFromTextBox.Text = routeList[i].TimeFrom;

toDateTimePicker.Value = routeList[i].DateTimeTo;

timeToTextBox.Text = routeList[i].TimeTo;

priceTextBox.Text = routeList[i].Price;

index = i;

}

}

}

private void buttonCancel\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Ви впевнені, що хочете скасувати бронювання?",

"Попередження", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

HomePage form = new HomePage();

form.Show();

this.Hide();

}

else

{

var close = e as FormClosingEventArgs;

if (close != null)

close.Cancel = true;

}

}

private void buttonBuy\_Click(object sender, EventArgs e)

{

string? number = selected;

string? name = nameTextBox.Text;

string? surname = surnameTextBox.Text;

decimal seats = passengerNumeric.Value;

string price = priceTextBox.Text;

string? nickname = nicknameList[0];

if (string.IsNullOrEmpty(name) || string.IsNullOrEmpty(surname))

{

MessageBox.Show("Введіть усі необхідні дані.", "Збереження",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

if (routeList[index].FreeSeats >= seats)

{

if (ticketList == null)

{

routeList[index].FreeSeats = routeList[index].FreeSeats - seats;

List<TicketData> ticketList = new List<TicketData>();

TicketData newTicket = new TicketData(number, name, surname, seats,

price, nickname);

ticketList.Add(newTicket);

file.Save(routeList);

file2.Save(ticketList);

}

else

{

routeList[index].FreeSeats = routeList[index].FreeSeats - seats;

TicketData newTicket = new TicketData(number, name, surname, seats,

price, nickname);

ticketList.Add(newTicket);

file.Save(routeList);

file2.Save(ticketList);

}

}

else

{

MessageBox.Show("Недостатньо місць в автобусі.", "Помилка збереження",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

HomePage form = new HomePage();

form.Show();

this.Hide();

}

private void numericPassenger\_ValueChanged(object sender, EventArgs e)

{

priceTextBox.Text = (Math.Round(decimal.Parse(routeList[index].Price) \*

passengerNumeric.Value, 2)).ToString();

}

}

}

**Edit.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Windows\_Forms;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.ComponentModel.Design.ObjectSelectorEditor;

namespace BusStationCashDesk.Forms

{

public partial class Edit : Form

{

private SaveLoadData<RouteData> file;

private List<RouteData> routeList;

private List<string>? nameStop;

private List<string>? timeStop;

private string selected;

public Edit(string number)

{

InitializeComponent();

dateFromDateTimePicker.MinDate = DateTime.Today;

dateToDateTimePicker.MinDate = DateTime.Today;

file = new SaveLoadData<RouteData>("routeData.json");

routeList = file.Load();

selected = number;

this.FormClosing += Edit\_FormClosing;

}

private void Edit\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

DialogResult result = MessageBox.Show("Оновлені дані не будуть збережені! " +

"Ви впевнені, що хочете закрити сторінку?", "Попередження",

MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

Application.Exit();

}

else e.Cancel = true;

}

}

private void Edit\_Load(object sender, EventArgs e)

{

timeFromDateTimePicker.CustomFormat = "HH:mm";

timeToDateTimePicker.CustomFormat = "HH:mm";

timeStopDateTimePicker.CustomFormat = "HH:mm";

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

TextInfo title = CultureInfo.InvariantCulture.TextInfo;

numberRouteTextBox.Text = routeList[i].Number;

fromTextBox.Text = title.ToTitleCase(routeList[i].FromName ?? string.Empty);

toTextBox.Text = title.ToTitleCase(routeList[i].ToName ?? string.Empty);

dateFromDateTimePicker.Value = routeList[i].DateTimeFrom;

timeFromDateTimePicker.Value = DateTime.ParseExact(routeList[i].TimeFrom,

"HH:mm", CultureInfo.InvariantCulture);

dateToDateTimePicker.Value = routeList[i].DateTimeTo;

timeToDateTimePicker.Value = DateTime.ParseExact(routeList[i].TimeTo,

"HH:mm", CultureInfo.InvariantCulture);

stopList.View = View.Details;

for (int j = 0; j < routeList[i].Stops.Count; j++)

{

ListViewItem item = new ListViewItem(routeList[i].Stops[j]);

item.SubItems.Add(routeList[i].TimeStop[j]);

stopList.Items.Add(item);

}

freeSeatNumeric.Value = routeList[i].FreeSeats;

priceTextBox.Text = routeList[i].Price;

}

}

}

private void buttonCancel\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Оновлені дані не будуть збережені! " +

"Ви впевнені, що хочете скасувати редагування?", "Попередження",

MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

else

{

var close = e as FormClosingEventArgs;

if (close != null)

close.Cancel = true;

}

}

private void buttonAddStop\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(nameStopTextBox.Text) ||

string.IsNullOrEmpty(timeStopDateTimePicker.Text))

{

MessageBox.Show("Введіть назву зупинки.", "Повідомлення",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

ListViewItem item = new ListViewItem(nameStopTextBox.Text);

item.SubItems.Add(timeStopDateTimePicker.Text);

stopList.Items.Add(item);

nameStopTextBox.Clear();

}

private void buttonRemoveStop\_Click(object sender, EventArgs e)

{

if (stopList.SelectedItems.Count > 0)

stopList.Items.Remove(stopList.SelectedItems[0]);

}

private void buttonSave\_Click(object sender, EventArgs e)

{

if (nameStop == null && timeStop == null)

{

nameStop = new List<string>();

timeStop = new List<string>();

foreach (ListViewItem item in stopList.Items)

{

DateTime stopTime = DateTime.Parse(item.SubItems[1].Text);

DateTime fromTime = timeFromDateTimePicker.Value;

DateTime toTime = timeToDateTimePicker.Value;

DateTime fromDateTime = dateFromDateTimePicker.Value.Date;

DateTime toDateTime = dateToDateTimePicker.Value.Date;

if ((stopTime < fromTime || stopTime > toTime) && fromDateTime ==

toDateTime || fromDateTime != toDateTime && stopTime <

fromTime && stopTime > toTime)

{

MessageBox.Show("Час зупинок має бути після відправлення та до прибуття.",

"Помилка збереження", MessageBoxButtons.OK, MessageBoxIcon.Error);

nameStop = null;

timeStop = null;

return;

}

}

foreach (ListViewItem item in stopList.Items)

{

nameStop.Add(item.SubItems[0].Text);

timeStop.Add(item.SubItems[1].Text);

}

}

string number = numberRouteTextBox.Text;

string fromName = fromTextBox.Text.ToLower();

string toName = toTextBox.Text.ToLower();

DateTime dateTimeFrom = dateFromDateTimePicker.Value.Date;

string timeFrom = timeFromDateTimePicker.Value.ToString("HH:mm");

DateTime dateTimeTo = dateToDateTimePicker.Value.Date;

string timeTo = timeToDateTimePicker.Value.ToString("HH:mm");

List<string>? stops = nameStop!;

List<string>? timeStops = timeStop;

decimal freeSeats = freeSeatNumeric.Value;

string price = priceTextBox.Text;

if (string.IsNullOrEmpty(number) || string.IsNullOrEmpty(fromName) ||

string.IsNullOrEmpty(toName) || string.IsNullOrEmpty(price))

{

MessageBox.Show("Введіть усі необхідні дані.", "Збереження",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

if (dateTimeFrom > dateTimeTo || (timeFromDateTimePicker.Value >

timeToDateTimePicker.Value && dateTimeFrom == dateTimeTo))

{

MessageBox.Show("Дата та час прибуття мають бути пізніше за відправлення.",

"Помилка збереження", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

int repeat = 0;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == number && routeList[i].Number != selected)

{

repeat++;

}

if (repeat == 0 && i == routeList.Count - 1 && stops != null && timeStop != null)

{

RouteData newRoute = new RouteData(number, fromName, toName,

dateTimeFrom, timeFrom, dateTimeTo, timeTo, stops, timeStop,

freeSeats, price);

for (int j = 0; j < routeList.Count; j++)

{

if (routeList[j].Number == selected && selected != null)

{

routeList.RemoveAt(j);

}

}

routeList.Add(newRoute);

file.Save(routeList);

break;

}

else if (i == routeList.Count - 1)

{

MessageBox.Show("Вже існує маршурт з таким номером.", "Помилка збереження",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

}

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

private void textBoxPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsDigit(e.KeyChar) && e.KeyChar != ',' &&

e.KeyChar != '\b' && e.KeyChar != '\u007F')

{

e.Handled = true;

}

if (e.KeyChar == ',' && ((sender as TextBox)?.Text?.Contains(",") ?? false))

{

e.Handled = true;

}

}

}

}

**InformationRoute.cs**

using BusStationCashDesk.Classes;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class InformationRoute : Form

{

private SaveLoadData<RouteData> file;

private List<RouteData> routeList;

private string selected;

public InformationRoute(string number)

{

InitializeComponent();

file = new SaveLoadData<RouteData>("routeData.json");

routeList = file.Load();

selected = number;

}

private void InformationRoute\_Load(object sender, EventArgs e)

{

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

DisplayRoute(i);

}

}

}

private void DisplayRoute(int i)

{

TextInfo title = CultureInfo.InvariantCulture.TextInfo;

numberRouteTextBox.Text = routeList[i].Number;

fromTextBox.Text = title.ToTitleCase(routeList[i].FromName ?? string.Empty);

toTextBox.Text = title.ToTitleCase(routeList[i].ToName ?? string.Empty);

fromDateTimePicker.Value = routeList[i].DateTimeFrom;

timeFromTextBox.Text = routeList[i].TimeFrom;

toDateTimePicker.Value = routeList[i].DateTimeTo;

timeToTextBox.Text = routeList[i].TimeTo;

listStop.View = View.Details;

for (int j = 0; j < routeList[i].Stops.Count; j++)

{

ListViewItem item = new ListViewItem(title.ToTitleCase(routeList[i].Stops[j]

?? string.Empty));

item.SubItems.Add(routeList[i].TimeStop[j]);

listStop.Items.Add(item);

}

freePlaceTextBox.Text = (routeList[i].FreeSeats).ToString();

priceTextBox.Text = routeList[i].Price;

}

}

}

**MyTravel.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Models;

using Microsoft.VisualBasic.Devices;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.Globalization;

using System.Linq;

using System.Reflection;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

using static System.ComponentModel.Design.ObjectSelectorEditor;

using static System.Formats.Asn1.AsnWriter;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class MyTravel : Form

{

private SaveLoadData<RouteData> file;

private List<RouteData> routeList;

private SaveLoadData<string> file1;

private List<string> nicknameList;

private SaveLoadData<TicketData> file2;

private List<TicketData> ticketList;

public MyTravel()

{

InitializeComponent();

file = new SaveLoadData<RouteData>("routeData.json");

routeList = file.Load();

file1 = new SaveLoadData<string>("nicknameData.json");

nicknameList = file1.Load();

file2 = new SaveLoadData<TicketData>("ticketData.json");

ticketList = file2.Load();

}

private void MyTravel\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

Application.Exit();

}

}

private void MyTravel\_Load(object sender, EventArgs e)

{

this.FormClosing += MyTravel\_FormClosing;

List<RouteData> boughtRoute = new List<RouteData>();

for (int i = 0; i < ticketList.Count; i++)

{

if (ticketList[i].Nickname == nicknameList[0])

{

for (int j = 0; j < routeList.Count; j++)

{

if (ticketList[i].Number == routeList[j].Number)

{

string? number = routeList[j].Number;

string? fromName = routeList[j].FromName;

string? toName = routeList[j].ToName;

DateTime dateTimeFrom = routeList[j].DateTimeFrom;

string timeFrom = routeList[j].TimeFrom;

DateTime dateTimeTo = routeList[j].DateTimeTo;

string timeTo = routeList[j].TimeTo;

List<string> stops = new List<string>();

List<string> timeStops = new List<string>();

decimal freeSeats = ticketList[i].Seats;

string price = routeList[j].Price;

RouteData newRoute = new RouteData(number, fromName, toName,

dateTimeFrom, timeFrom, dateTimeTo, timeTo, stops, timeStops,

freeSeats, price);

boughtRoute.Add(newRoute);

}

}

}

}

DisplayRoute(boughtRoute);

if (boughtRoute.Count < ticketList.Count)

{

for (int i = 0; i < ticketList.Count; i++)

{

bool found = false;

for (int j = 0; j <= boughtRoute.Count; j++)

{

if(ticketList[i].Nickname == nicknameList[0] && j != boughtRoute.Count)

{

if (ticketList[i].Number == boughtRoute[j].Number)

{

found = true;

}

}

else found = true;

}

if (found == false)

{

ticketList.RemoveAt(i);

file2.Save(ticketList);

}

}

}

else if (boughtRoute.Count > ticketList.Count)

{

for (int i = 0; i < boughtRoute.Count; i++)

{

bool found = false;

for (int j = 0; j <= ticketList.Count; j++)

{

if (ticketList[j].Nickname == nicknameList[0] && j != boughtRoute.Count)

{

if (ticketList[j].Number == boughtRoute[i].Number)

{

found = true;

}

}

else found = true;

}

if (found == false)

{

boughtRoute.RemoveAt(i);

}

}

}

}

private void linkLabelBack\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

HomePage form = new HomePage();

form.Show();

this.Hide();

}

private void DisplayRoute(List<RouteData> boughtRoute)

{

foreach (RouteData route in boughtRoute)

{

TextInfo title = CultureInfo.InvariantCulture.TextInfo;

ListViewItem item = new ListViewItem(route.Number);

item.SubItems.Add(title.ToTitleCase(route.FromName ?? string.Empty));

item.SubItems.Add(title.ToTitleCase(route.ToName ?? string.Empty));

item.SubItems.Add(route.DateTimeFrom.ToString("dd/MM/yyyy"));

item.SubItems.Add(route.TimeFrom);

item.SubItems.Add(route.FreeSeats.ToString());

listTicket.Items.Add(item);

}

}

private void listTicket\_DoubleClick(object sender, EventArgs e)

{

if (listTicket.SelectedItems.Count > 0)

{

string selected = listTicket.SelectedItems[0].Text;

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected && selected != null)

{

InformationRoute informationRoute = new InformationRoute(selected);

informationRoute.ShowDialog();

}

}

}

}

private void buttonReturnTicket\_Click(object sender, EventArgs e)

{

if (listTicket.SelectedItems.Count > 0)

{

DialogResult result = MessageBox.Show("Ви впевнені, що хочете скасувати бронювання?",

"Підтвердження повернення квитка", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

string selected = listTicket.SelectedItems[0].Text;

int index = 0;

for (int i = 0; i < ticketList.Count; i++)

{

if (ticketList[i].Number == selected)

{

index = i;

}

}

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == selected)

{

routeList[i].FreeSeats += ticketList[index].Seats;

ticketList.RemoveAt(index);

}

}

file.Save(routeList);

file2.Save(ticketList);

MyTravel form = new MyTravel();

form.Show();

this.Hide();

}

}

else MessageBox.Show("Виберіть маршрут, на який хочете скасувати бронювання.",

"Повернення квитка", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

**Route.cs**

using BusStationCashDesk.Classes;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

using static System.Formats.Asn1.AsnWriter;

namespace BusStationCashDesk.Windows\_Forms

{

public partial class Route : Form

{

private SaveLoadData<RouteData> file;

private List<RouteData> routeList;

private List<string>? nameStop;

private List<string>? timeStop;

public Route()

{

InitializeComponent();

fromDateTimePicker.MinDate = DateTime.Today;

toDateTimePicker.MinDate = DateTime.Today;

file = new SaveLoadData<RouteData>("routeData.json");

routeList = file.Load();

}

private void Route\_FormClosing(object? sender, FormClosingEventArgs e)

{

if (e.CloseReason == CloseReason.UserClosing)

{

DialogResult result = MessageBox.Show("Дані не будуть збережені! " +

"Ви впевнені, що хочете закрити сторінку?", "Попередження",

MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

Application.Exit();

}

else e.Cancel = true;

}

}

private void Route\_Load(object sender, EventArgs e)

{

this.FormClosing += Route\_FormClosing;

timeFromDateTimePicker.CustomFormat = "HH:mm";

timeToDateTimePicker.CustomFormat = "HH:mm";

timeStopDateTimePicker.CustomFormat = "HH:mm";

}

private void buttonCancel\_Click\_1(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Дані не будуть збережені! Ви впевнені, " +

"що хочете скасувати створення?", "Попередження", MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

else

{

var close = e as FormClosingEventArgs;

if (close != null)

close.Cancel = true;

}

}

private void buttonSave\_Click\_1(object sender, EventArgs e)

{

if (nameStop == null)

{

nameStop = new List<string>();

}

if (timeStop == null)

{

timeStop = new List<string>();

}

foreach (ListViewItem item in stopListView.Items)

{

DateTime stopTime = DateTime.Parse(item.SubItems[1].Text);

DateTime fromTime = timeFromDateTimePicker.Value;

DateTime toTime = timeToDateTimePicker.Value;

DateTime fromDateTime = fromDateTimePicker.Value.Date;

DateTime toDateTime = toDateTimePicker.Value.Date;

if ((stopTime < fromTime || stopTime > toTime) && fromDateTime ==

toDateTime || fromDateTime != toDateTime && stopTime <

fromTime && stopTime > toTime)

{

MessageBox.Show("Час зупинок має бути після відправлення та до прибуття.",

"Помилка збереження", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

}

foreach (ListViewItem item in stopListView.Items)

{

nameStop.Add(item.SubItems[0].Text);

timeStop.Add(item.SubItems[1].Text);

}

string number = numberRouteTextBox.Text;

string fromName = fromTextBox.Text.ToLower();

string toName = toTextBox.Text.ToLower();

DateTime dateTimeFrom = fromDateTimePicker.Value.Date;

string timeFrom = timeFromDateTimePicker.Value.ToString("HH:mm");

DateTime dateTimeTo = toDateTimePicker.Value.Date;

string timeTo = timeToDateTimePicker.Value.ToString("HH:mm");

List<string> stops = nameStop;

List<string> timeStops = timeStop;

decimal freeSeats = freeSeatNumeric.Value;

string price = priceTextBox.Text;

int repeat = 0;

if (string.IsNullOrEmpty(number) || string.IsNullOrEmpty(fromName) ||

string.IsNullOrEmpty(toName) || string.IsNullOrEmpty(price))

{

MessageBox.Show("Введіть усі необхідні дані.", "Збереження",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

if (dateTimeFrom > dateTimeTo || (timeFromDateTimePicker.Value >

timeToDateTimePicker.Value && dateTimeFrom == dateTimeTo))

{

MessageBox.Show("Дата та час прибуття мають бути пізніше за відправлення.",

"Помилка збереження", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (routeList == null)

{

List<RouteData> routeList = new List<RouteData>();

RouteData newRoute = new RouteData(number, fromName, toName,

dateTimeFrom, timeFrom, dateTimeTo, timeTo, stops, timeStop,

freeSeats, price);

routeList.Add(newRoute);

file.Save(routeList);

}

else for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].Number == number)

{

repeat++;

}

if (repeat == 0 && i == routeList.Count - 1)

{

RouteData newRoute = new RouteData(number, fromName, toName,

dateTimeFrom, timeFrom, dateTimeTo, timeTo, stops, timeStop,

freeSeats, price);

routeList.Add(newRoute);

file.Save(routeList);

break;

}

else if (i == routeList.Count - 1)

{

MessageBox.Show("Вже існує маршурт з таким номером.", "Помилка збереження",

MessageBoxButtons.OK, MessageBoxIcon.Error);

repeat = 0;

return;

}

}

HomePageAdministration form = new HomePageAdministration();

form.Show();

this.Hide();

}

private void buttonAddStop\_Click\_1(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(nameStopTextBox.Text) ||

string.IsNullOrEmpty(timeStopDateTimePicker.Text))

{

MessageBox.Show("Введіть назву зупинки.", "Повідомлення",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

ListViewItem item = new ListViewItem(nameStopTextBox.Text);

item.SubItems.Add(timeStopDateTimePicker.Text);

stopListView.Items.Add(item);

nameStopTextBox.Clear();

}

private void button1\_Click(object sender, EventArgs e)

{

if (stopListView.SelectedItems.Count > 0)

{

stopListView.Items.Remove(stopListView.SelectedItems[0]);

}

}

private void textBoxPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

if (!char.IsDigit(e.KeyChar) && e.KeyChar != ',' && e.KeyChar !=

'\b' && e.KeyChar != '\u007F')

{

e.Handled = true;

}

if (e.KeyChar == ',' && ((sender as TextBox)?.Text?.Contains(",") ?? false))

{

e.Handled = true;

}

}

}

}

**SaveLoadData.cs**

using BusStationCashDesk.Windows\_Forms;

using Microsoft.VisualBasic.ApplicationServices;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.CompilerServices;

using System.Text;

using System.Text.Json;

using System.Threading.Tasks;

namespace BusStationCashDesk.Classes

{

public class SaveLoadData<T> //Клас для збереження та завантаження даних

{

private string filePath;

public SaveLoadData(string filePath)

{

this.filePath = filePath;

}

//Метод, що зберігає список у файл .json

public void Save(List<T> users)

{

string jsonData = JsonSerializer.Serialize(users);

File.WriteAllText(filePath, jsonData);

}

//Метод, що перетворює дані з файлу .json у список

public List<T> Load()

{

if (File.Exists(filePath))

{

string jsonData = File.ReadAllText(filePath);

return JsonSerializer.Deserialize<List<T>>(jsonData) ?? new List<T>();

}

else

{

return new List<T>();

}

}

}

}

**HomePages.cs**

using BusStationCashDesk.Classes;

using BusStationCashDesk.Windows\_Forms;

using System;

using System.Collections.Generic;

using System.Globalization;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BusStationCashDesk

{

// Клас містить методи, для роботи на головній сторінці користувача та адміністратора.

public class HomePages

{

private SaveLoadData<RouteData> route;

private List<RouteData> routeList;

public HomePages()

{

route = new SaveLoadData<RouteData>("routeData.json");

routeList = route.Load();

}

// Метод перевіряє чи є застарілі маршрути, якщо так, то він їх видаляє.

public void RemoveExpiredRoutes(List<RouteData> routeList)

{

for (int i = 0; i < routeList.Count; i++)

{

if (routeList[i].DateTimeFrom.Date <= DateTime.Today &&

DateTime.Parse(routeList[i].TimeFrom) <= DateTime.Now)

{

routeList.RemoveAt(i);

}

}

route.Save(routeList);

}

// Метод відповідає за розміщення маршрутів на головний сторінці.

public void DisplayRoute(List<RouteData> route, ListView listRoute)

{

listRoute.Items.Clear();

foreach (RouteData route1 in route)

{

TextInfo title = CultureInfo.InvariantCulture.TextInfo;

ListViewItem item = new ListViewItem(route1.Number);

item.SubItems.Add(title.ToTitleCase(route1.FromName ?? string.Empty));

item.SubItems.Add(title.ToTitleCase(route1.ToName ?? string.Empty));

item.SubItems.Add(route1.DateTimeFrom.ToString("dd/MM/yyyy"));

item.SubItems.Add(route1.TimeFrom);

item.SubItems.Add(route1.FreeSeats.ToString());

listRoute.Items.Add(item);

}

}

// Метод шукає маршрути за введеними даними.

public void Search(string from, string to, DateTime date, ListView listRoute)

{

if (string.IsNullOrEmpty(from) || string.IsNullOrEmpty(to))

{

MessageBox.Show("Введіть дані для пошуку.", "Повідомлення", MessageBoxButtons.OK,

MessageBoxIcon.Information);

return;

}

List<RouteData> selectedRoute = SelectedRoute(from, to, date);

if (selectedRoute.Count == 0)

{

MessageBox.Show("Жодного маршруту не знайдено.", "Повідомлення",

MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

DisplayRoute(selectedRoute, listRoute);

}

// Метод створює список зі знайденими маршрутами та розташовує їх за часом.

public List<RouteData> SelectedRoute(string from, string to, DateTime date)

{

List<RouteData> selectedRoute = new List<RouteData>();

foreach (RouteData route in routeList)

{

if (route.FromName == from && route.ToName == to && route.DateTimeFrom ==

date && route.FreeSeats > 0)

{

selectedRoute.Add(route);

}

}

if (selectedRoute.Count > 1)

{

for (int i = 0; i < selectedRoute.Count - 1; i++)

{

int min = i;

for (int j = i + 1; j < selectedRoute.Count; j++)

{

if (DateTime.ParseExact(selectedRoute[j].TimeFrom, "HH:mm",

CultureInfo.InvariantCulture) < DateTime.ParseExact(selectedRoute[min].TimeFrom,

"HH:mm", CultureInfo.InvariantCulture))

{

min = j;

}

}

RouteData k = selectedRoute[i];

selectedRoute[i] = selectedRoute[min];

selectedRoute[min] = k;

}

}

return selectedRoute;

}

}

}

**RouteData.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BusStationCashDesk.Classes

{

public class RouteData

{

public string? Number { get; set; }

public string? FromName { get; set; }

public string? ToName { get; set; }

public DateTime DateTimeFrom { get; set; }

public string TimeFrom { get; set; }

public DateTime DateTimeTo { get; set; }

public string TimeTo { get; set; }

public List<string> Stops { get; set; }

public List<string> TimeStop { get; set; }

public decimal FreeSeats { get; set; }

public string Price { get; set; }

public RouteData(string? number, string? fromName, string? toName,

DateTime dateTimeFrom, string timeFrom, DateTime dateTimeTo,

string timeTo, List<string> stops, List<string> timeStop,

decimal freeSeats, string price)

{

Number = number;

FromName = fromName;

ToName = toName;

DateTimeFrom = dateTimeFrom;

TimeFrom = timeFrom;

DateTimeTo = dateTimeTo;

TimeTo = timeTo;

Stops = stops;

TimeStop = timeStop;

FreeSeats = freeSeats;

Price = price;

}

}

}

**TicketData.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BusStationCashDesk.Models

{

public class TicketData

{

public string? Number { get; set; }

public string? Name { get; set; }

public string? Surname { get; set; }

public decimal Seats { get; set; }

public string Price { get; set; }

public string? Nickname { get; set; }

public TicketData(string? number, string? name, string? surname, decimal seats, string price, string? nickname)

{

Number = number;

Name = name;

Surname = surname;

Seats = seats;

Price = price;

Nickname = nickname;

}

}

}

**UserData.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BusStationCashDesk.Windows\_Forms

{

public class UserData

{

public string? Name { get; set; }

public string? Password { get; set; }

public bool IsAdmin { get; set; }

public UserData(string? name, string? password, bool isAdmin)

{

Name = name;

Password = password;

IsAdmin = isAdmin;

}

}

}