Архітектурний опис системи

Мета: побудувати систему бронювання квитків, яка надає можливості переглядати доступні рейси, вибирати місця та оформляти бронювання. Користувач може створювати та переглядати власні бронювання, а адміністратор має можливість додавати, редагувати та видаляти рейси. Після створення бронювання система генерує квиток і реєструє оплату.

Тип архітектури: Система побудована за layered архітектурою: Presentation (UI), Business Logic та Data Access. UI приймає дії користувача та ініціює операції. Business Logic перевіряє доступність місць, генерує квитки та ініціює оплату. І Data Access відповідає за збереження та читання даних користувачів, рейси, квитки, платежі тощо. Такий поділ зменшує за'язанність і полегшує тестування та підтримку.

Основні компоненти:

- **UI** - це те що бачить користувач, UI відображає інтерфейс, дозволяє переглядати рейси, керувати профілем та створювати бронювання.
- **User API** - обробляє реєстрацію, авторизацію, та керування профілем.
- **Trip API** - Надає інформацію про рейси.
- **Booking API** - Обробляє створення бронювань, і повертає дані бронювання.
- **Payment Service** - взаємодіє із стороннім платіжним сервісом, та проводить оплату.
- **Email Sevice** - надсилає користувачам підтвердження бронювання, та надсилає квитки.
- **Repository Layer** - Виконує читання, створення, оновлення та видалення даних. Ізолює застосунок від роботи з базою даних.
- **Database** - зберігає інформацію.

Use Case діаграма:

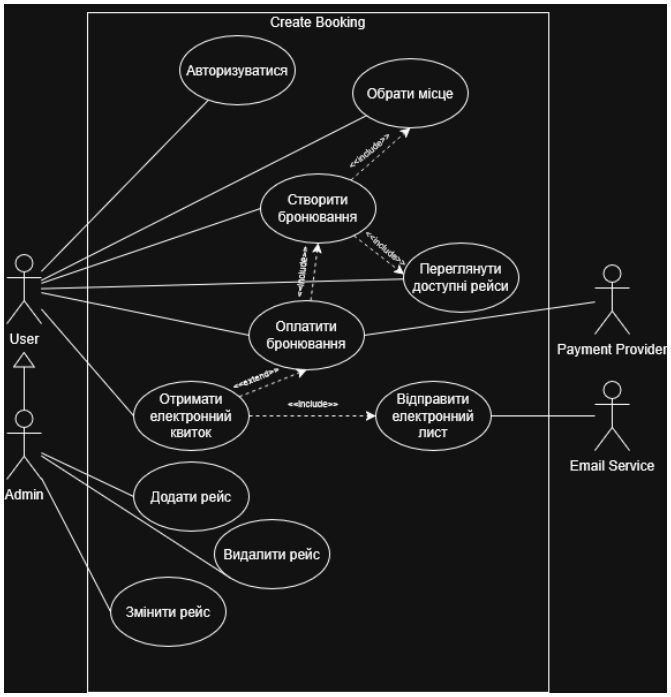

Рис. 1 – Use Case діаграма

Діаграма демонструє взаємодію користувачів, та зовнішніх сервісів між собою. А саме які можливості має звичайний користувач та адміністратор, та які сервіси обробляють ці операції.

Class діаграма:

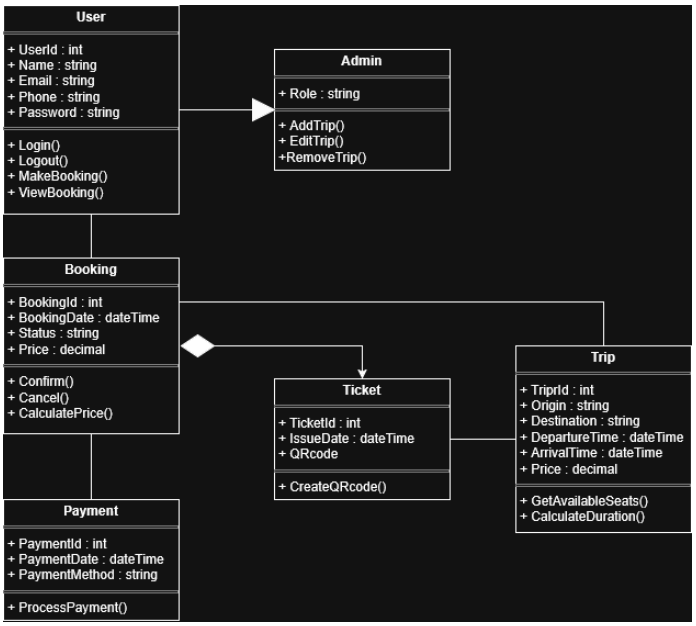

Рис. 2 – Class діаграма

Ця діаграма показує основні класи системи, їх атрибути, методи та зв’язки між ними. На діаграмі зображені основні сутності, що беруть учать у системі бронювання квитків.

Component діаграма:

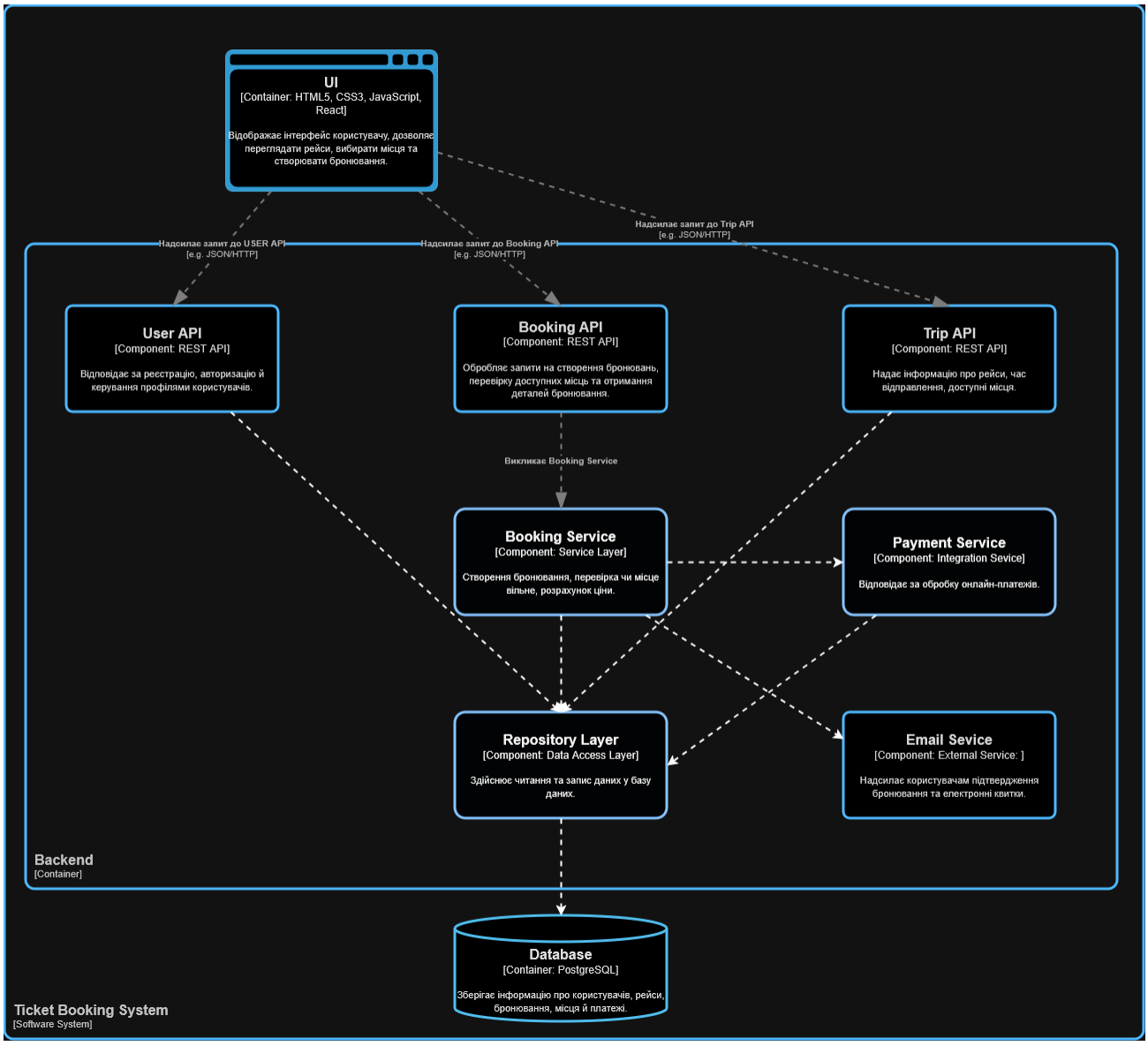

Рис. 3 – Component діаграма

Component діаграма відображає логічну структуру системи та зв’язки між програмними модулями. Система реалізована за Layered Architecture.

Sequence діаграма:

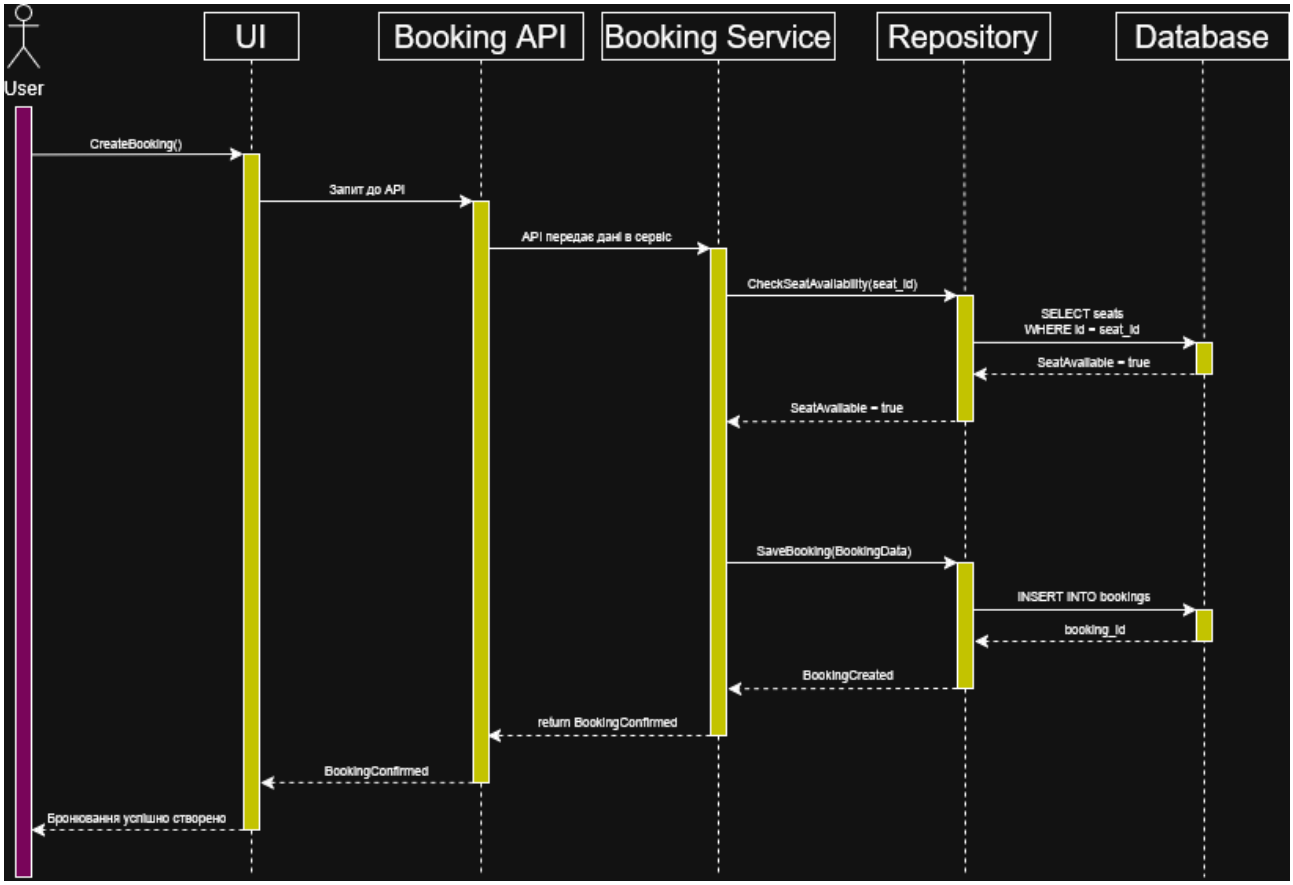

Рис. 4 – Sequence діаграма

Ця діаграма відображає послідовне виконання дій під час виконання якогось конкретного сценарію. На даній діаграмі відображений сценарій створення бронювання.

State Machine діаграма:

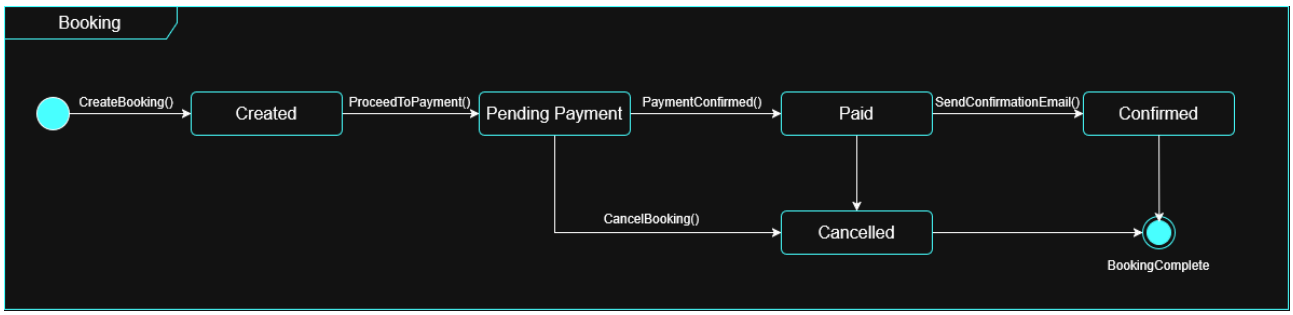

Рис. 5 – State Machine діаграма

На цій діаграмі показано зміни станів об’єкта під час його життєвого циклу, та умови після яких ці зміни відбуваються. На діаграмі показаний життєвий цикл станів Booking.

Висновок: У результаті розробки опису архітектури системи бронювання квитків було спроєктовано структуру додатку, що реалізує процес створення, оплати та підтвердження бронювання.

Система побудована за принципом багатошарової архітектури, з розділенням відповідальності між компонентами: Presentation, Business logic та Data Access.

Було розроблено п’ять UML-діаграм: Use Case, Class, Component, Sequence та State Machine, які відображають всі основні аспекти роботи системи.