TicketMate

Project Team

Maryam Khalid 22i1917

Maryam Amjad 22i1924

Munieba Hussain 22i2050

Department of Computer Science

National University of Computer and Emerging Sciences Islamabad, Pakistan

Contents

1. Introduction 1
   1. Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
   2. Problem Statement . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
   3. Proposed Solution/Method . . . . . . . . . . . . . . . . . . . . . . . . . 2
2. User Stories, Design, and Work Division 3
   1. User Stories . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
   2. Design Description and Diagrams . . . . . . . . . . . . . . . . . . . 4
      1. Use Case Diagram . . . . . . . . . . . . . . . . . . . . . . . . . 5
      2. Activity Diagrams . . . . . . . . . . . . . . . . . . . . . . . . . . 6
      3. Sequence Diagrams . . . . . . . . . . . . . . . . . . . . . . . . . 10
      4. Class Diagram . . . . . . . . . . . . . . . . . . . . . . . . . .12
      5. Scrum Board . . . . . . . . . . . . . . . . . . . . . . . . 13
      6. Architecture Diagram. . . . . . . . . . . . . . . . . . . . . . . . 17
      7. Burn down chart . . . . . . . . . . . . . . . . . . . . . . . . 18
   3. Work Division . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
   4. Boundary Value Analysis . . . . . . . . . . . . . . . . . . . . . . . . 21
   5. Lessons Learnt . . . . . . . . . . . . . . . . . . . . . . . . 22

Appendix 1: Implementation Screenshots 17

.1 System Architecture Screenshots . . . . . . . . . . . . . . . . . . . . . . 18

References

2

List of Figures

|  |  |  |
| --- | --- | --- |
| 2.1 | Use Case Diagram showing interactions between users and the system | 7 |
| 2.2 | Activity Diagram 1 – Seat Selection Flow . . . . . . . . . . . . . . . . | 7 |
| 2.3 | Activity Diagram 2 – Bus Searching Flow . . . . . . . . . . . . . . . . . | 7 |
| 2.4 | Activity Diagram 3 – Booking Confirmation Flow . . . . . . . . . . . . . | 7 |
| 2.5 | Sequence Diagram 1 –Bus search event . . . . . . . . . . . . . . . . | 7 |
| 2.6 | Sequence Diagram 2 – Seat Selection Event. . . . . . . . . . . . . . . . . | 7 |
| 2.7 | Sequence Diagram 3 – Booking confirmation Event . . . . . . . . . . . . . . . . | 7 |
| 2.8 | Class Diagram depicting core entities and their relationships . . . . . . . | 7 |
| 2.9 | Sprint 2 Scrum Board Overview . . . . . . . . . . . . . . . . . . . . . .  2.10 Architecture Diagram  2.11 Burn down Chart | 7 |
| 10 | System Architecture Overview 1: Description of major components and |  |
|  | Their interactions. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 11 |
| 11 | System Architecture Overview 2: Detailed view of the front-end compo- |  |
|  | nents and their connection to the back-end. . . . . . . . . . . . . . . . . . | 11 |
| 12 | System Architecture Overview 3: Description of the database schema and |  |
|  | API integration. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 11 |
| 13 | System Architecture Overview 4: Depiction of user interaction flow from |  |
|  | the front-end to back-end systems. . . . . . . . . . . . . . . . . . . . . . | 11 |
| 14 | System Architecture Overview 5: Final architecture diagram illustrating |  |
|  | end-to-end data flow within the application. . . . . . . . . . . . . . . . . | 11 |

3

List of Tables

Chapter 1

Introduction

TicketMate is a simple, lightweight, and user-friendly web application designed to make the process of booking event tickets smooth and efficient. The goal of the project is to eliminate the complexity often associated with current ticketing platforms by offering a clean interface and essential features without the reliance on third-party APIs. The system is particularly tailored for small to medium-sized event organizers who may not have access to enterprise-level platforms or extensive technical resources.

# 1.1 Motivation

The motivation behind TicketMate stems from the growing reliance on digital event booking systems and the realization that many existing platforms are either too complicated, overloaded with features, or require third-party integrations. This creates unnecessary barriers for smaller event organizers and everyday users. Hence, our application aims to fill this gap by providing a minimal yet functional solution that offers real-time ticket availability, basic reservation management, and easy access to event details.

# 1.2 Problem Statement

Many small to medium-sized event organizers face significant challenges in offering online ticket booking due to the complexity and cost of existing solutions. Most platforms are designed for large-scale events and rely heavily on external APIs, making them unsuitable or inaccessible for smaller events.

The core problems addressed by TicketMate include:

• Overly complex booking platforms.

1. Introduction

* High integration requirements and costs.
* Lack of simplicity and customization for smaller event setups.

By focusing on local and mid-scale events, TicketMate offers an affordable, intuitive, and reliable ticketing platform without the need for third-party tools.

# 1.3 Proposed Solution/Method

We propose TicketMate — a clean, standalone web-based event booking platform designed with minimalism and functionality in mind. It includes essential features like:

* Real-time ticket availability display.
* Basic reservation and booking management.
* A responsive and simplified user interface.

The team has adopted the Scrum framework to manage development. In Sprint 1, we focused on setting up the core architecture, initial wireframes, and user authentication. In Sprint 2, we expanded functionality by integrating the full ticket booking flow and refining the UI/UX based on feedback. Agile methodology allowed us to continuously adjust development priorities based on real-time team progress and early-stage user input.

Chapter 2

User Stories, Design, and

Work Division

# 2.1 User Stories

Following are key user stories:

* As a user, I want to view a list of upcoming events so that I can explore which ones I’d like to attend.
* As a user, I want to see detailed information about a selected event including time, location, and ticket availability.
* As a user, I want to be able to book a ticket for an event so that I can reserve my spot conveniently.
* As an admin/organizer, I want to create new events with all necessary details so they are visible on the platform.
* As a user, I want to receive a confirmation message once I book a ticket successfully.

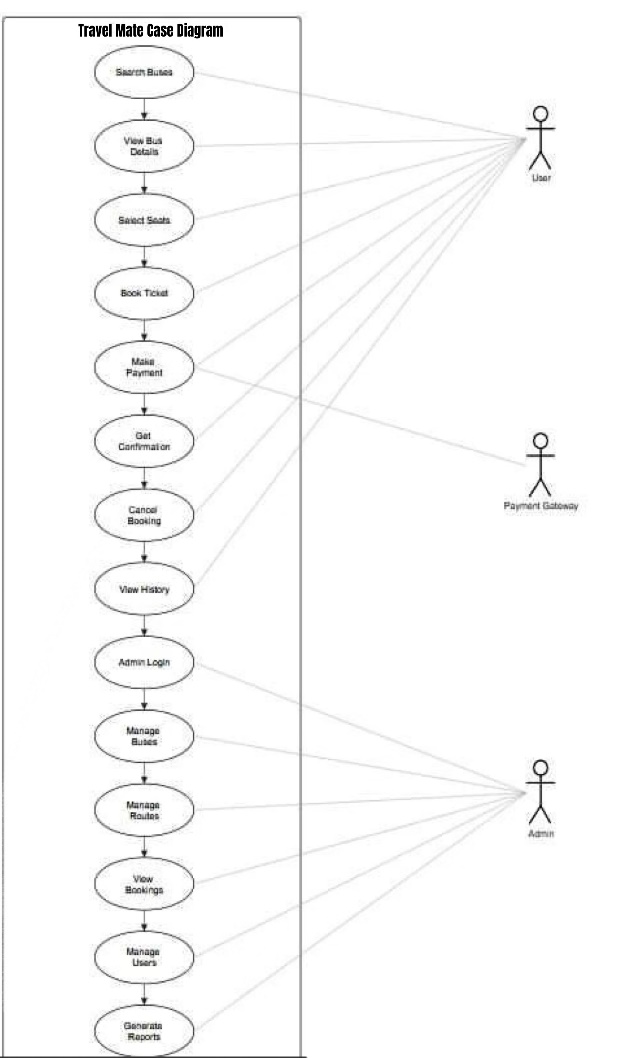
These stories helped guide development throughout the sprint and ensured that we prioritized real user needs, keeping the application lean and practical for early deployment.

2. Sprint 2: User Stories, Design, and Work Division

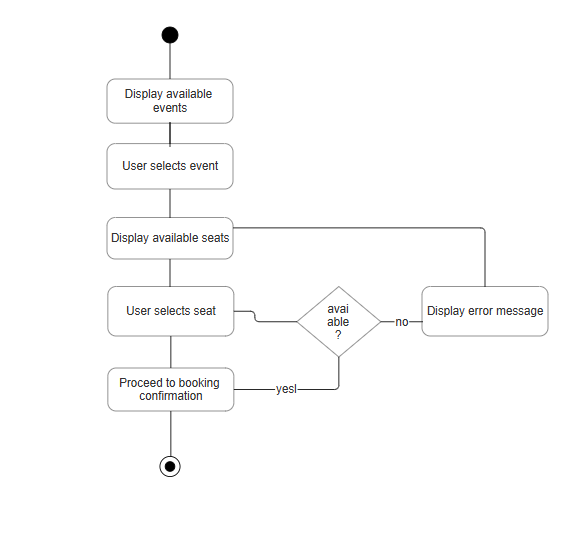
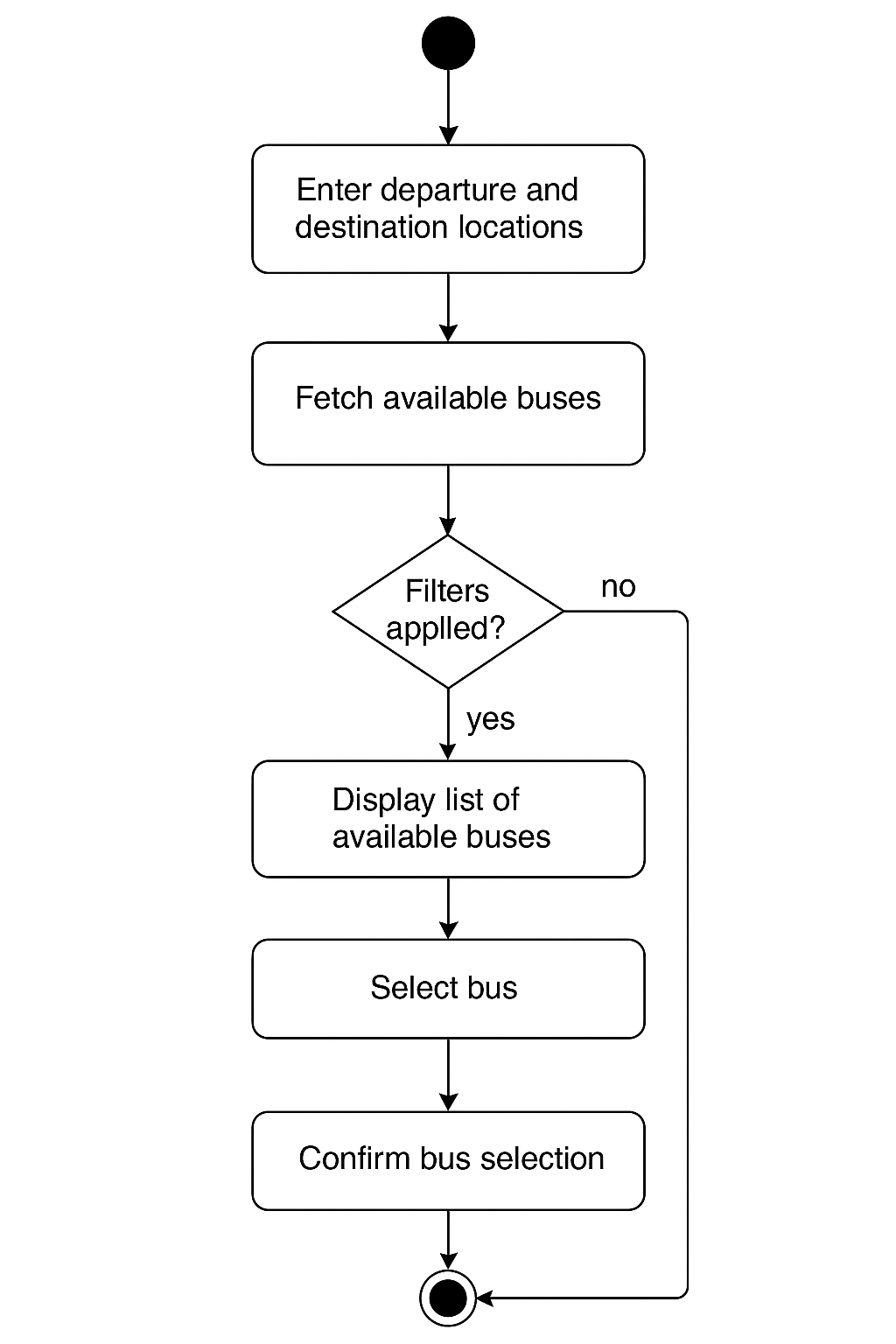
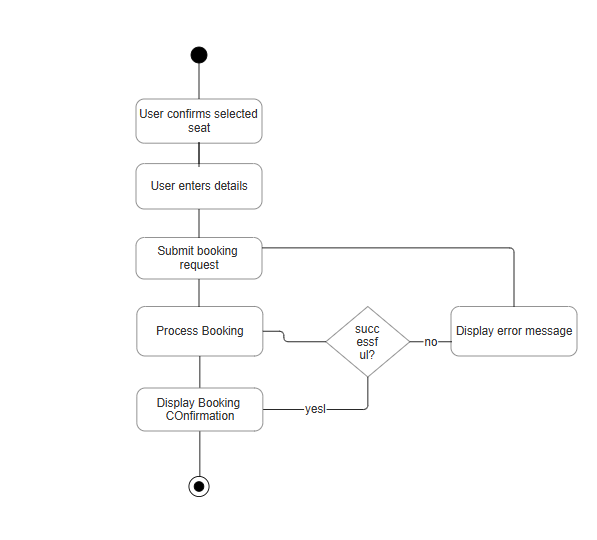
2.2 Design Description and Diagrams

During Sprint 2, we enhanced the design of the platform to support a smoother booking experience. The front-end was designed using a component-based structure for clarity and modularity. We used a clean, intuitive layout that highlights event listings and offers clear navigation. From a technical perspective, the system follows an MVC (Model-View-Controller) pattern to separate logic, UI, and data handling. The backend is structured around RESTful APIs, which makes it scalable and easier to extend in future sprints.

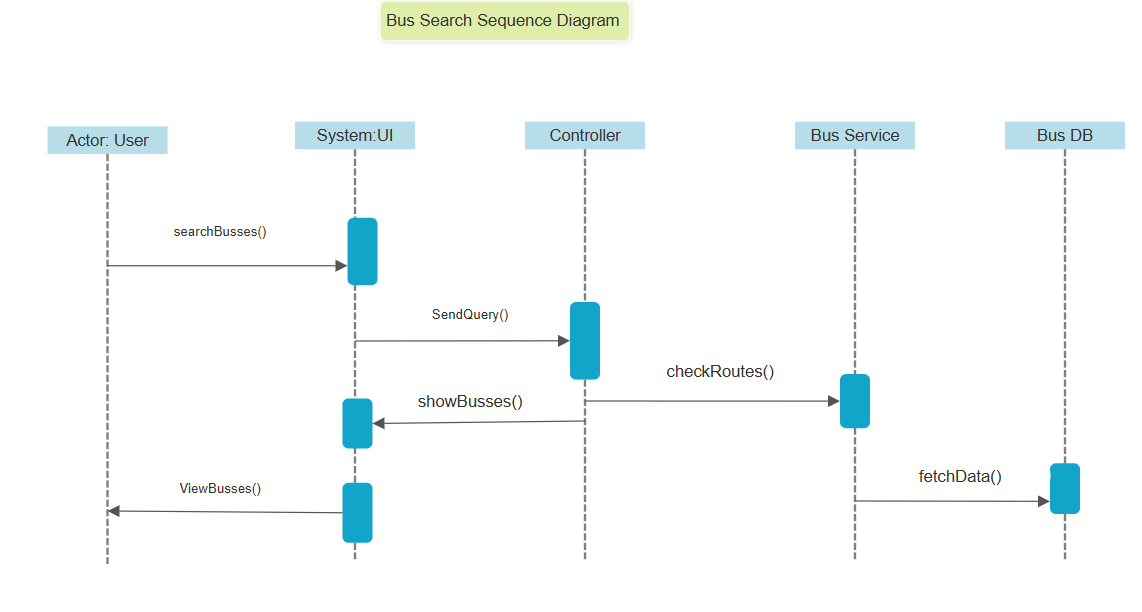
2.2.1 Use Case Diagram

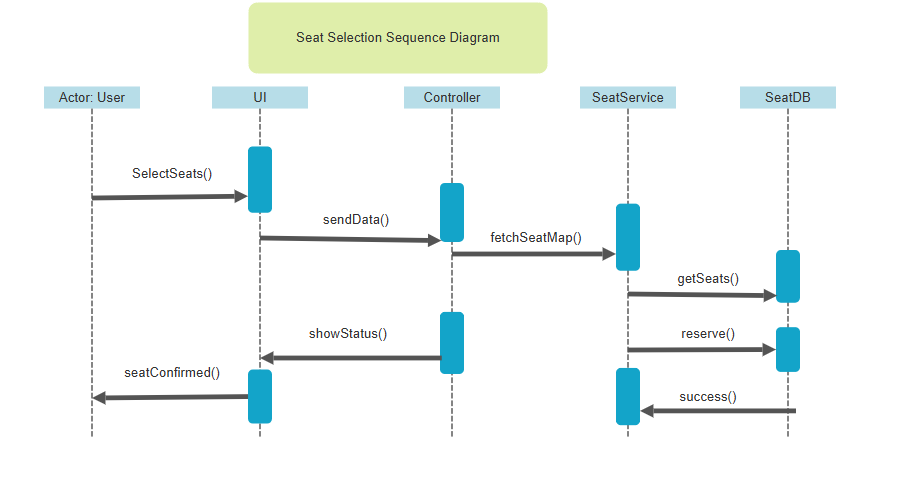


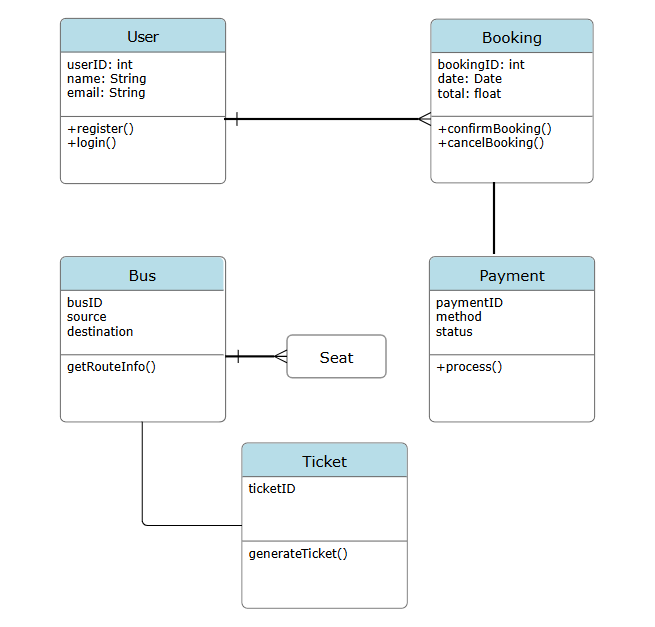
2.2.2 Activity Diagrams

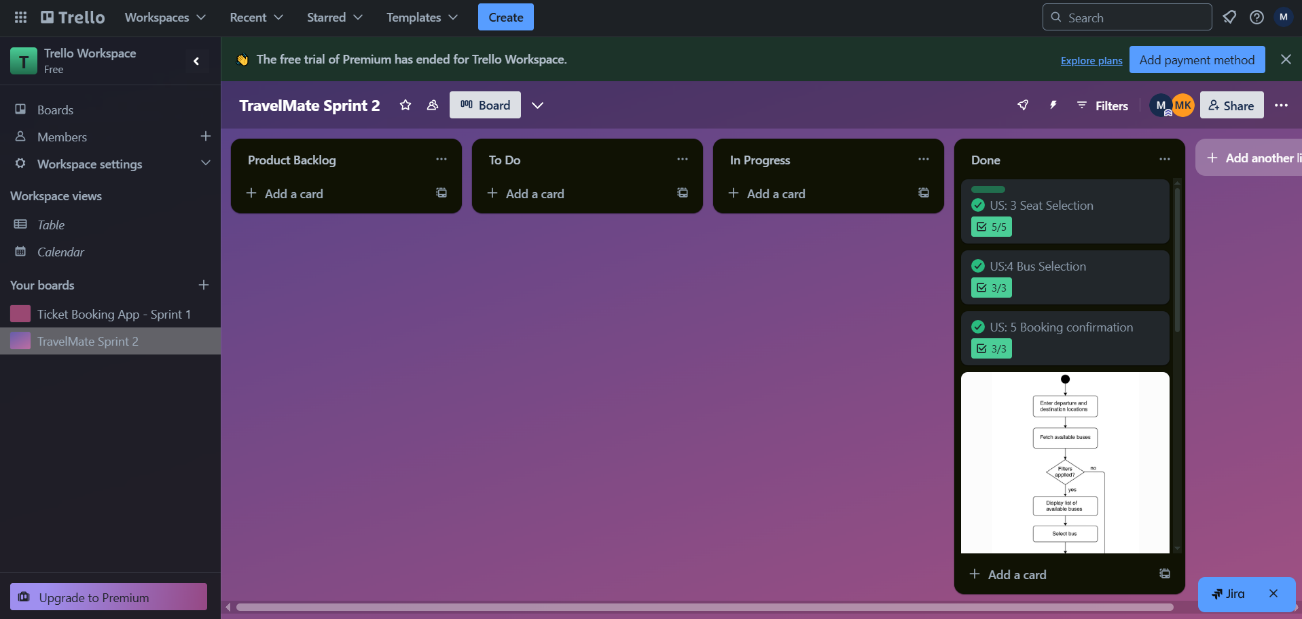
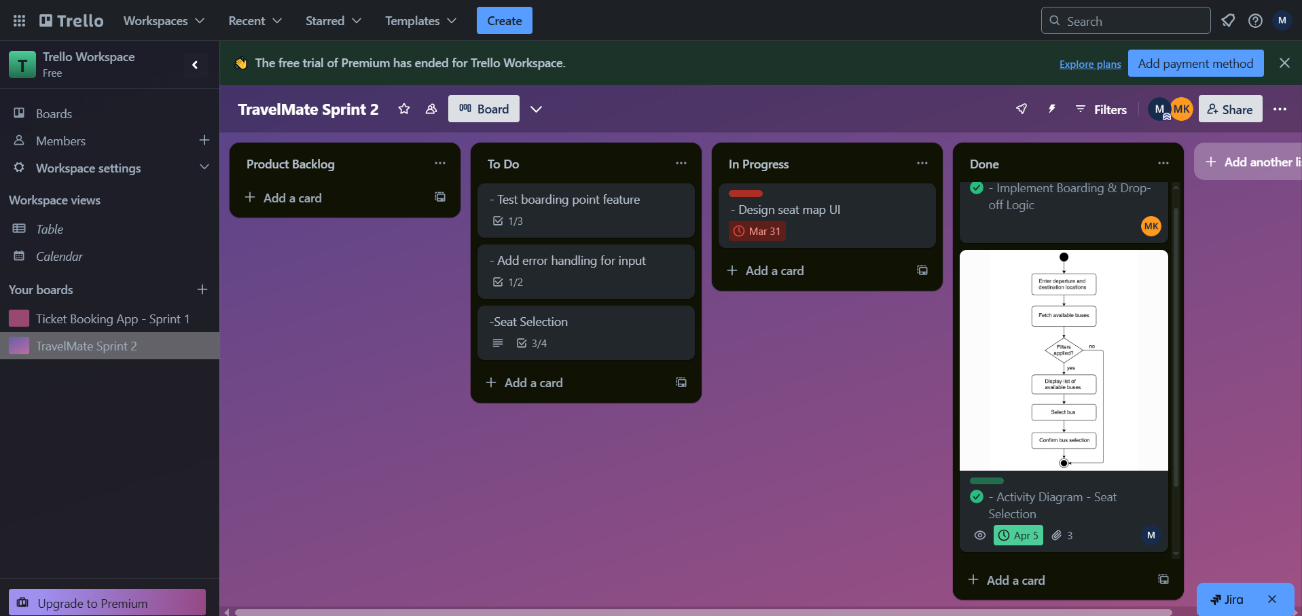
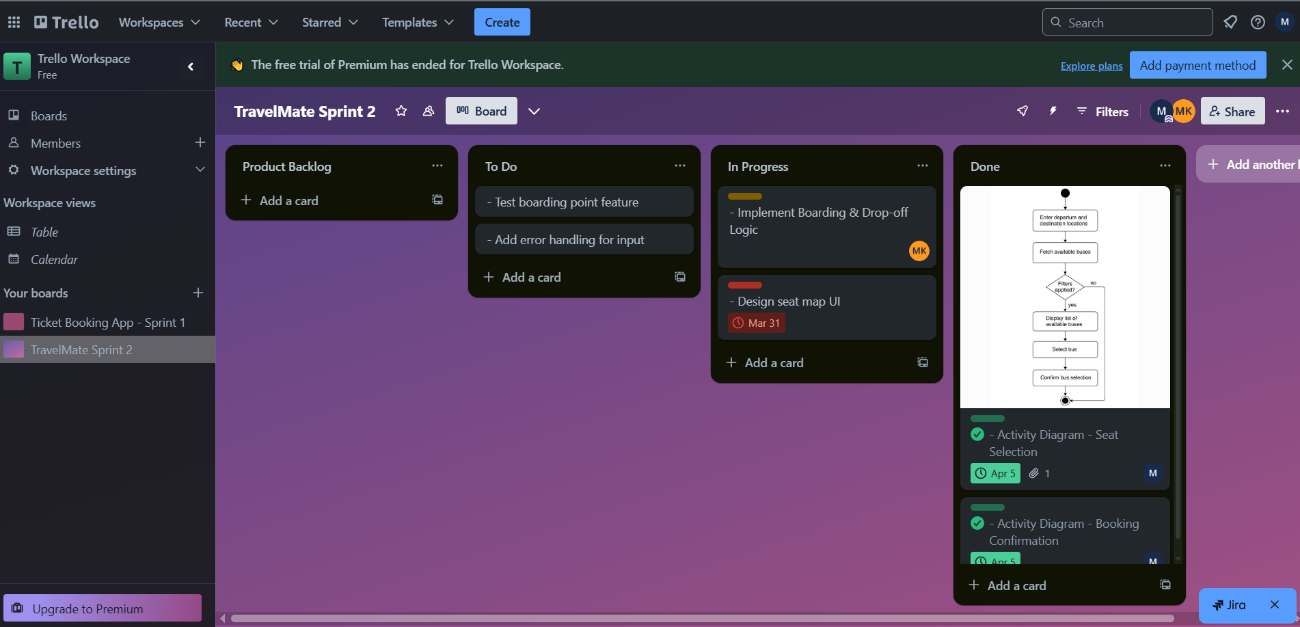


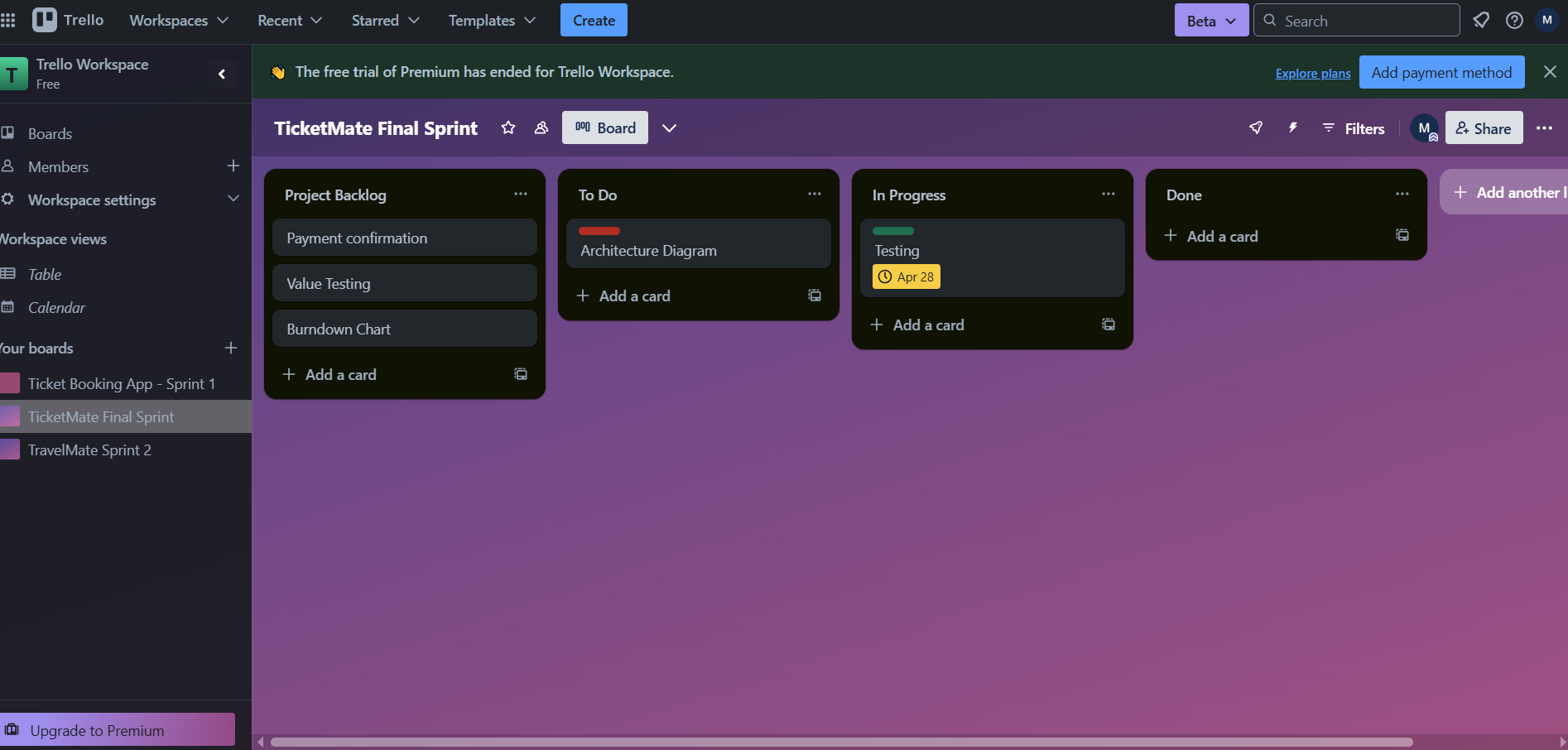
2.2.3 Sequence Diagrams

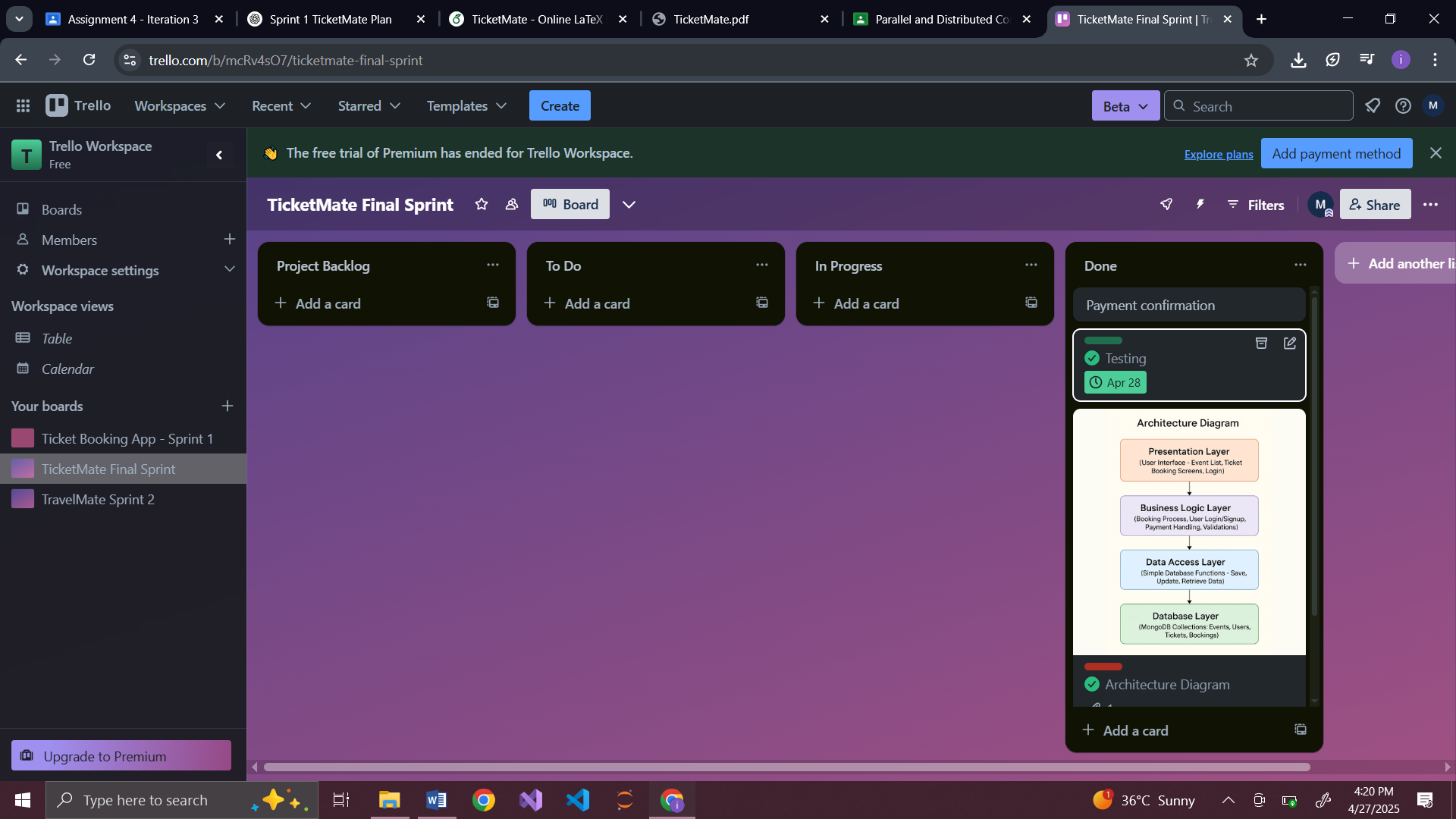




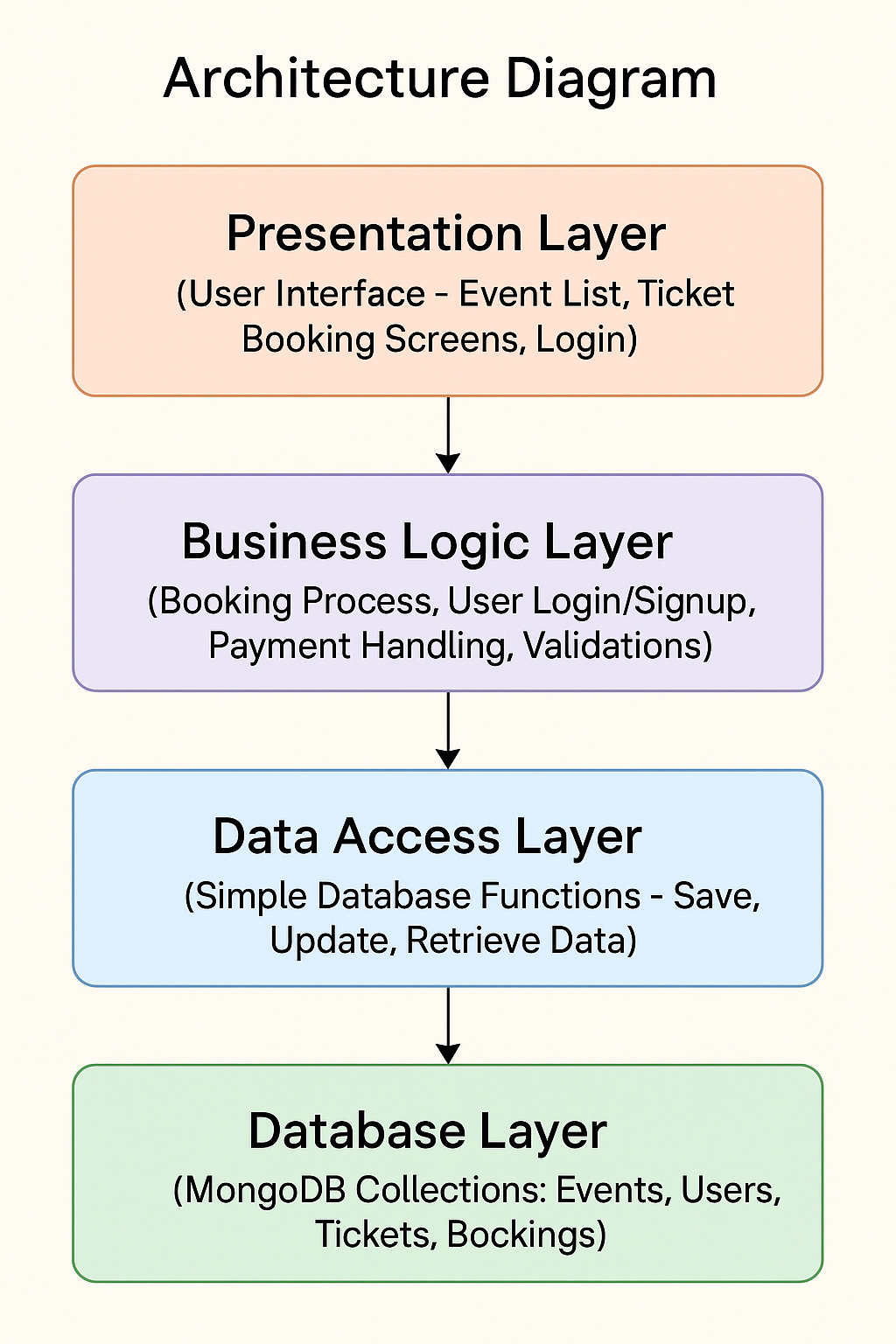
2.2.4 Class Diagram 

2.2.5 Scrum Board 

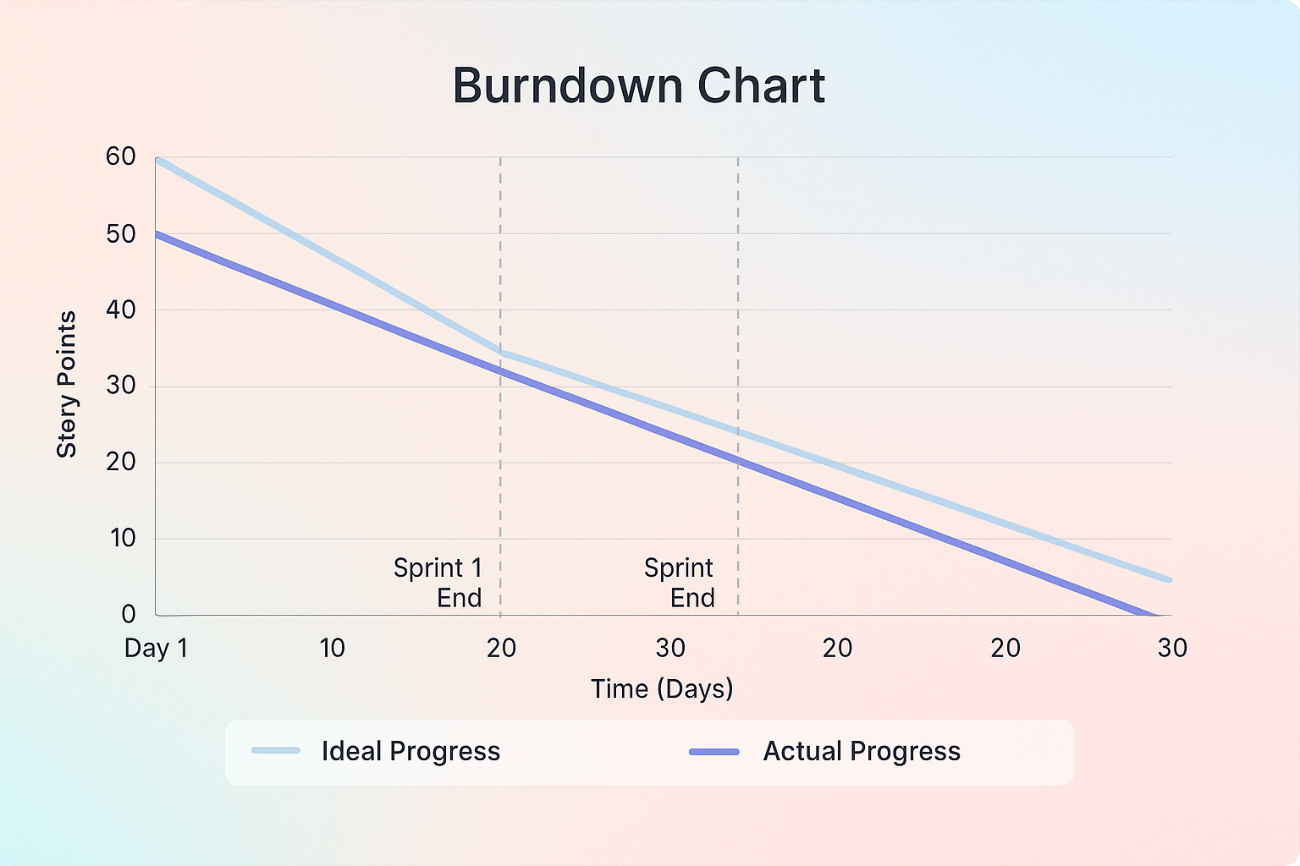




2.2.6 Architecture Diagram:



2.2.7 Burn down Chart:



# 2.3 Work Division 5

To ensure efficiency, we divided responsibilities based on each team member’s strengths and past contributions. Below is the breakdown of the work distribution for Sprint 2:

1. Sprint 2: User Stories, Design, and Work Division
   * Maryam Khalid (22i-1917) took responsibility for the front-end and making a user friendly interface
   * Maryam Amjad (22i-1924) handled coding of the implementation using C#.
   * Munieba Hussain (22i2050) focused on creating design diagrams’ styling and documentation.

All members collaborated on testing, debugging, and finalizing user stories for the sprint demo. The division of labor allowed the team to work in parallel, accelerating progress while maintaining quality across both the front-end and back-end components.

6

2.3 Work Division

Figure 2.1: Use Case Diagram showing interactions between users and the system

Figure 2.2: Activity Diagram 1 – Event Browsing Flow

Figure 2.3: Activity Diagram 2 – Ticket Booking Flow

Figure 2.4: Activity Diagram 3 – Admin Event Creation Flow

Figure 2.5: Sequence Diagram 1 – User Browses Events

Figure 2.6: Sequence Diagram 2 – User Books Ticket

Figure 2.7: Sequence Diagram 3 – Admin Creates Event

Figure 2.8: Class Diagram depicting core entities and their relationships

**2.4 Boundary Value Analysis:**

Boundary Value Analysis (BVA) testing was conducted for key input fields within the TicketMate application to ensure the system handles edge cases accurately. This method focuses on testing the values at the boundaries of input ranges, as errors are more likely to occur at these points. For TicketMate, boundary tests were designed for features such as username and password length, number of tickets per booking, payment amount, and event booking dates. For example, usernames were tested at 4, 5, 20, and 21 characters to verify that the system correctly accepts or rejects inputs based on defined limits. Similarly, the number of tickets was tested at 0, 1, 10, and 11 to validate booking constraints. This testing ensured that the application maintains robustness, enforces input validation, and provides appropriate feedback to users when boundaries are crossed, ultimately enhancing overall reliability and user experience.

**2.5 Learning From the project:**

Working on the TicketMate project provided valuable hands-on experience across the full software development cycle. It deepened our understanding of requirement gathering, breaking down features into user stories, sprint planning, and collaborative development using Scrum methodology. We learned the importance of clear communication, task division, and timely sprint reviews to keep the project on track. Technically, we strengthened our skills in implementing layered architecture, integrating MongoDB as the database, designing user-friendly interfaces, and ensuring data validation through techniques like Boundary Value Analysis. Additionally, we gained practical exposure to using project management tools like Scrum Boards and producing artifacts like burndown charts and design diagrams. Overall, this project emphasized the significance of iterative development, thorough testing, and continuous improvement, which are crucial for delivering high-quality software solutions.

Appendix 1: Implementation Screenshots

# .1 System Architecture Screenshots

The above images illustrate the system architecture and its components. These screenshots provide a visual representation of how different modules of the ticket booking system interact, showcasing the structure of the back-end, database, and front-end components.

2. Sprint 2: User Stories, Design, and Work Division

.2 Sprint 2 Progress Summary

Figure 10: System Architecture Overview 1: Description of major components and their interactions.

Figure 11: System Architecture Overview 2: Detailed view of the front-end components and their connection to the back-end.

Figure 12: System Architecture Overview 3: Description of the database schema and API integration.

Figure 13: System Architecture Overview 4: Depiction of user interaction flow from the front-end to back-end systems.

Figure 14: System Architecture Overview 5: Final architecture diagram illustrating endto-end data flow within the application.

Bibliography