LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS6P05NI Final Year Project

Assessment Weightage & Type

25% Interim Report

Futsal Fusion: Futsal Scheduling & Booking App

Student Name:

London Met ID:

College ID:

Internal Supervisor:

External Supervisor:

Assignment Due Date:

Assignment Submission Date:

Word Count (Where Required): 3904

I confirm that I understand my coursework needs to be submitted online via My Second Teacher under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Sample - BIC docx

Islington College,Nepal

Document Details

Submission ID
trn:oid:::3618:73855426

Submission Date
Dec 12, 2024, 2:05 PM GMT+5:45

Download Date
Dec 13, 2024, 8:54 AM GMT+5:45

File Name
Sample - BIC.docx

File Size
35.6 KB

52 Pages

5,419 Words

30,551 Characters

8% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- **56** Not Cited or Quoted 6%
  Matches with neither in-text citation nor quotation marks
- **7** Missing Quotations 2%
  Matches that are still very similar to source material
- **1** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

8% Internet sources
4% Publications
16% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **56** Not Cited or Quoted 15%
  Matches with neither in-text citation nor quotation marks
- **7** Missing Quotations 2%
  Matches that are still very similar to source material
- **1** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

- 8% Internet sources
- 4% Publications
- 16% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	National Institute of Business Management Sri Lanka on 2024-10-31 — 3%
2	Internet	www.coursehero.com — 3%
3	Submitted works	University of Greenwich on 2010-05-29 — 1%
4	Submitted works	Kingston University on 2024-01-07 — 1%
5	Submitted works	Ghana Technology University College on 2016-10-13 — 0%
6	Submitted works	UNITEC Institute of Technology on 2021-06-29 — 0%
7	Submitted works	The University of Wolverhampton on 2024-02-17 — 0%
8	Submitted works	Queen Mary and Westfield College on 2024-12-02 — 0%
9	Internet	uir.unisa.ac.za — 0%
10	Submitted works	Jose Rizal University on 2024-09-03 — 0%

Table of Contents

List of Figures

List of Tables

1. Introduction

As the name suggests, "**Futsal Fusion: Futsal Scheduling & Booking App**" takes a fresh approach by combining human and technological advancements, for the players, by the players making futsal scheduling easier for all concerned involved users. This includes new features for requesting players and keeping track of a player's and futsal's record and transaction history. Like any other technology revolution, the digitization of Nepal's sports environment requires a lofty goal to guide the process. When developing and leading the implementation of a digital strategy, this kind of vision is essential.

In Futsal Fusion, the story starts with all players who have a deep love for the game but never seem to have enough teammates. It was discovered there were a lot of people like us who wanted to play but didn't have someone to play with or anything to help them out as we were trying to figure out how to recruit more players. As a player, I can say with confidence that Futsal Fusion is going to do its best to address the issues that players encounter.

The major mission highlighted at Futsal Fusion has always been to bring together people who love the sport of futsal and who are looking for a community of like-minded players, where they can go to play, cheer each other on, and commiserate over wins and losses. Our desire to unite gamers in their shared experience with gaming venues is the inspiration behind for the players, by the players hosting a community where players, hosts, and anybody else with a passion for the game can come together.

1.1. Problem Domain

According to the Health and Fitness Consumer Data and Industry Trends report published by the International Health, Racquet & Sports Club Association (IHRSA), health and fitness clubs lose 25% of their potential revenue due to underutilization of facilities, which includes inefficient booking systems (IHRSA, 2022). Just as the primary goal of every company is to satisfy its customers, the same holds true for any service industry.

According to a recent survey by Dimensional Research, 72% of customers have stopped purchasing from a firm owing to subpar customer service. Customer discontent is high in the sports and recreation industry due in large part to cumbersome and time-consuming booking procedures. Customers may become dissatisfied if they are unable to reserve a court during a convenient time due to a lack of information regarding available time slots (Kahwaji, 2020). The appointment scheduling with no shows and overbooking highlights the issue of a no-show rate of at least 15%, which results in lost court time and missed revenue potential for futsal facilities (Zacharias & Pinedo, 2020).

Again, the traditional system compels the proprietor to use a logbook to keep track of reservations where there is significant possibility and likelihood that the data may be lost which could be privacy concerns and service inefficiencies without a comprehensive system to manage consumer data, booking history, and preferences. Similarly, the lack of a centralized mechanism increases the likelihood of futsal court double bookings and subsequent disagreements and disputes among consumers when there are booking clashes due to mismanagement (Online Khabar, 2023).

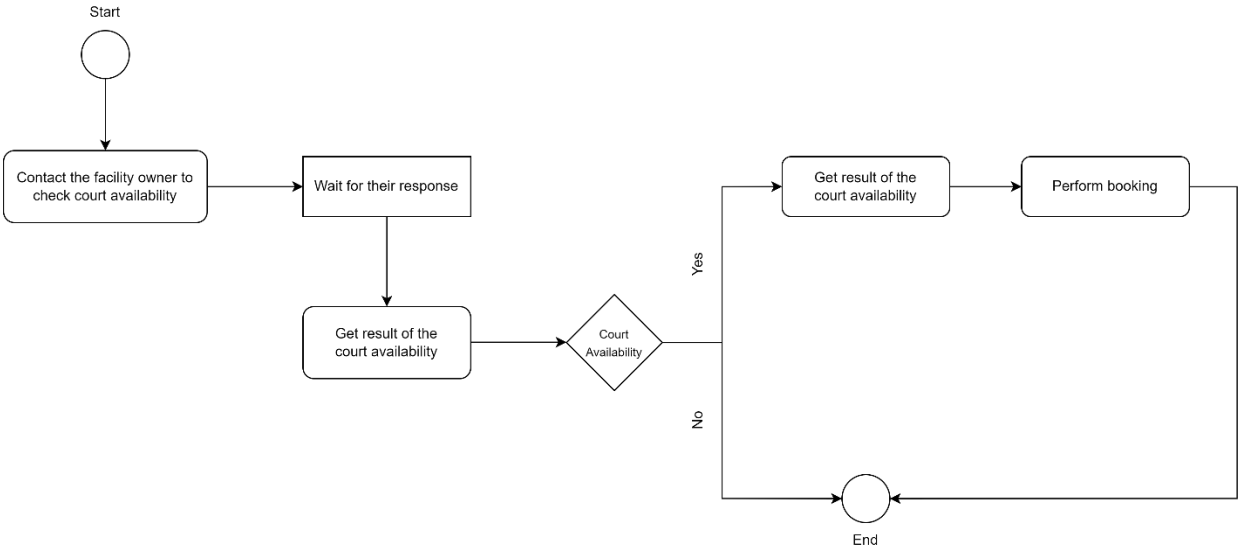


Figure 1: Problem Domain: Current Booking Scenario

As the problem domain can be differentiated based on the global and national concept, both the cases are studied and highlighted in the appendix section.

- Problem Analysis: Global Context (Link to Appendix)
- Problem Analysis: Nepal (Link to Appendix)

1.2. Project as a Solution

A comprehensive futsal management solution is proposed to improve the aforementioned areas, including the booking process, scheduling effectiveness, and facility operations, all for the benefit of the players and the owners. The system's goal is to build a simple web interface and web app where users can quickly and simply see which courts are available and reserve them. To ensure that scheduled times are used efficiently, it will also incorporate automatic reminders and notifications to cut down on no-shows. Facilitating safe online payment methods, which streamline transactions for players and guarantee timely revenue collection for facility owners, will be the next highlighted feature.

The system's intended functionality also includes allowing users to book courts for varying amounts of time, be it to host tournaments or other events. By adopting technology and establishing a user-friendly, efficient, and data-driven system, futsal facilities may revolutionize their operations, boost client satisfaction, optimize court utilization and prosper in the competitive sports sector. By connecting players and venue owners closer together, this project hopes to usher in a more streamlined era of futsal.

A descriptive detail regarding the problem-solving measure has also been mentioned in the appendix section with respect to technological advancements. Link to Appendix.

1.3. Aims and Objectives

The major aim of this project is to mitigate the problems faced by consumers and facility owners addressing a need to digitize the booking and scheduling services for a futsal promoting better interaction and efficiency through a development of a web app.

To achieve the primary aim as defined above, the core *SMART* objectives that will be strictly progressed are as follows:

1.3.1. Academic Objectives

- To learn about the .NET Core Framework and the features provided by its ecosystem.
- To learn about MVC Environment and integrate with stable backend APIs.
- To implement a structured database management system to progress the application development in a faster and reliable approach.
- To learn about REST APIs and use them for backend development purposes.
- To learn about HTTP formats for web deployment and JSON serialization to understand how data flow occurs in the client side and the server side.
- To learn about the implementation of Identity server providing both authorization and authentication features in a more secure way.
- To learn about different phases of software engineering and strongly implement the chosen methodology for development purposes throughout its lifecycle.
- To understand web application development and implement it according to the market demands.
- To learn and design implementation of external APIs such as Firebase and Payment Gateway so that the same application can be used to offer multiple source controls.
- To follow the Work Breakdown Structure and Gantt Chart to progress up the project and evaluate and monitor task progressed and track future steps.
- To implement computing knowledge and learning skills as a measure for problem solving technique to players and staff of a futsal court.

1.3.2. Project Objectives

- To interact with people and clients to understand their needs and changes required.
- To reduce the implementation and deployment cost of the app to its full limits.
- To become an alternative, preferable and reliable option for futsal systems engaging every possible activity to be performed during futsal actions.

2. Background

Futsal owners and players had a harder time communicating during the time-consuming stages of scheduling, community development, player involvement and collaboration, and gamification, which is why this app was designed. However, the problem is prevalent at several futsal courts, which made it easier to see how a digital system is needed in every part of people's lives and work. A well-known futsal facility, ***"Dhuku Sports Hub"*** was approached with the concept of creating a web app for their futsal sector's organization and management. Since the application focuses on multitenancy architecture, further set of clients will also be approached to make this project complete.

2.1. About the Client

Established in 2012 and situated in Baluwatar, Sheetal Marg, Dhuku Futsal Hub is the go-to spot for sports fans and those looking for a complete workout. Dhuku Futsal Hub offers a wide range of services and facilities to cater to the needs of those who are enthusiastic about football, swimming, fitness, and general wellness (Futsal Nepal, 2023).

Dhuku Futsal Hub stands out as a prospective important client as they are planning to build an online futsal booking and scheduling app. The hub and its customers could benefit from connecting your app with their services because of their diversified products and dedication to client involvement. The hub's facilities may be made more efficient and accessible with the help of the booking and scheduling capabilities, which can further establish it as a top destination for sports and fitness. The client approval letter and their respective concern has been attached in the appendix report: Link to Appendix.

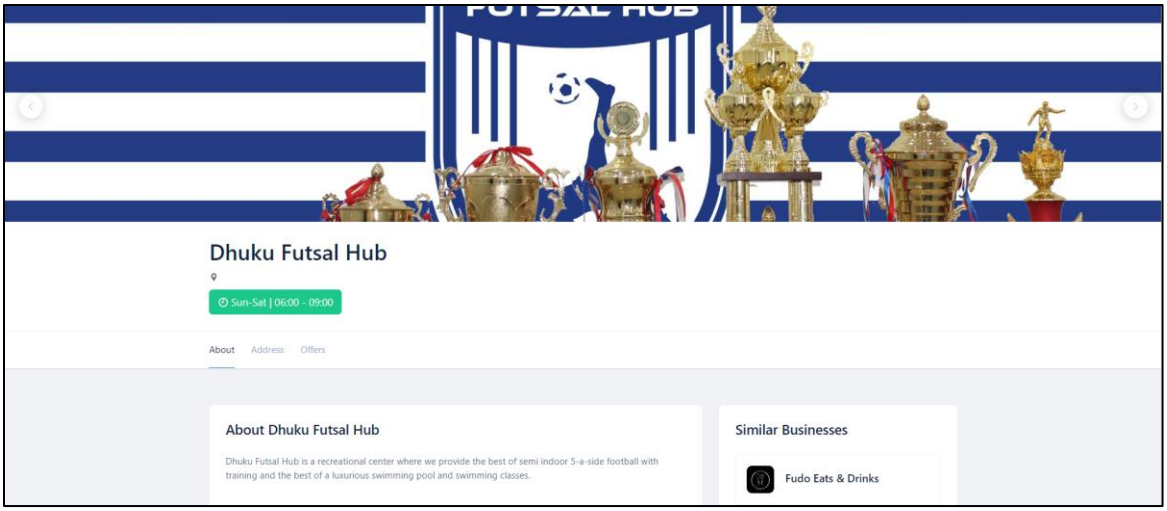


Figure 2: Dhuku Futsal Hub: About the Client (1)

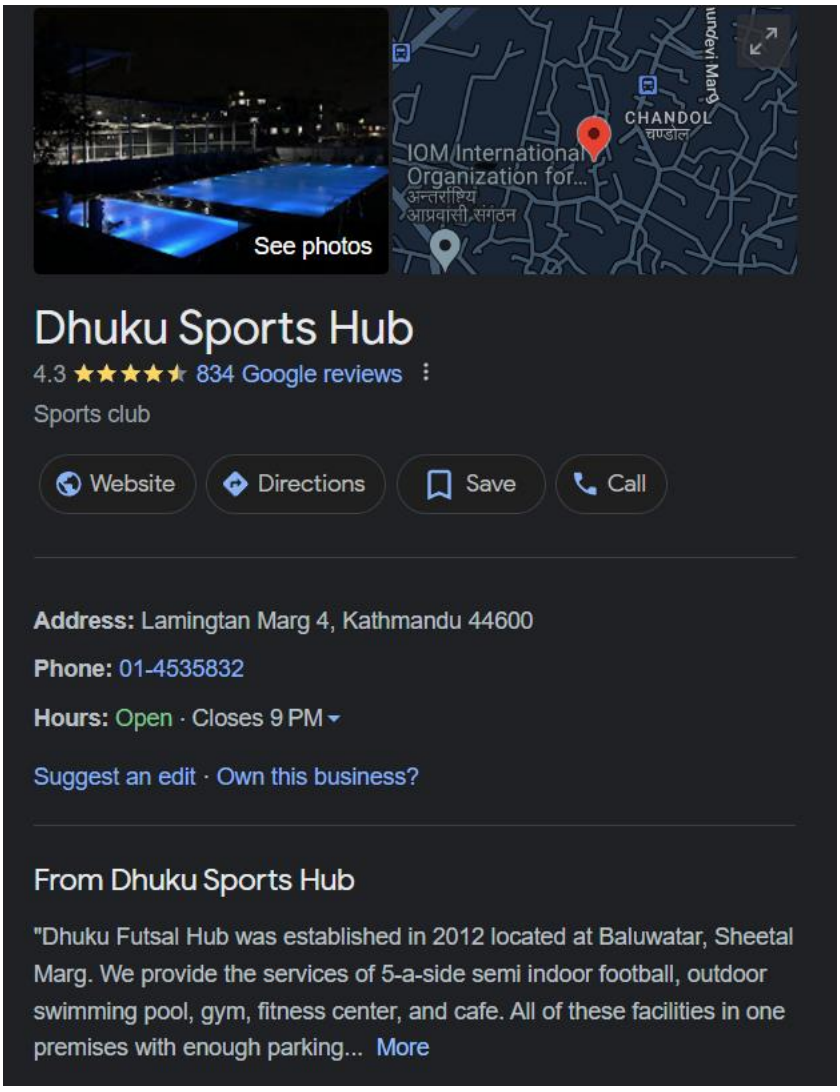


Figure 3: Dhuku Futsal Hub: About the Client (2)

2.2. Understanding the Solution

The application development for the futsal management system would be done optimizing the Scrum technique. The system development would be divided into two parts as Frontend and Backend construction segmented into several sprint cycles, each specifying the tasks to be done within the given time frame. The front-end view or the presentation component would be optimized with Razor Views and bespoke styling and dynamism. Similarly, ASP.NET Core will be optimized for the backend architecture for both database modelling and API hosting. At the end of each sprint, manual testing will be done, and client meetings will be held to find out the intended result. Finally, this project would target being a management system but with feature set achieving to address difficult issue domains in sports management sector. As the client futsal along with other futsal courts in Nepal proceed with pen-paper way of prescribing and documenting their entire records, it was vital to bring a change for the best. The technology would prove to be much of a bridge for linking the current method of communication between players and futsal owners delivering collaborative and gamification advantages.

2.3. Similar Projects

Researching and modeling one's efforts after those of experts and professionals is a smart notion if one's own ideas or concepts appear comparable to another in terms of description or feature set. In an attempt to reach new heights by altering the current sports management problem setting, futsal owners of Nepal have already adopted multiple futsal management systems. But a comparable system that is compatible with the ever-changing digital world addressing the public’s need still needs to be implemented for the players group served by the client institution.

2.3.1. Review on Similar Projects

Some comparable projects that share some of the proposed project's features are listed and described below:

2.3.1.1. WePlay Nepal

WePlay is a one-stop-shop for sports fans, letting them find and reserve futsal sites all throughout the valley. Users had to scour Google Maps for nearby sports venues, then phone each one individually to confirm availability and make a reservation before WePlay came along. Playing futsal has never been easier than with the WePlay app. Users can browse various venues, select a time and day that works for them, and even pay in advance all from the convenience of their mobile device. Launched in 2021, the WePlay app is compatible with both iOS and Android devices and features listings for around 70 locations (WePlay Nepal, 2023).

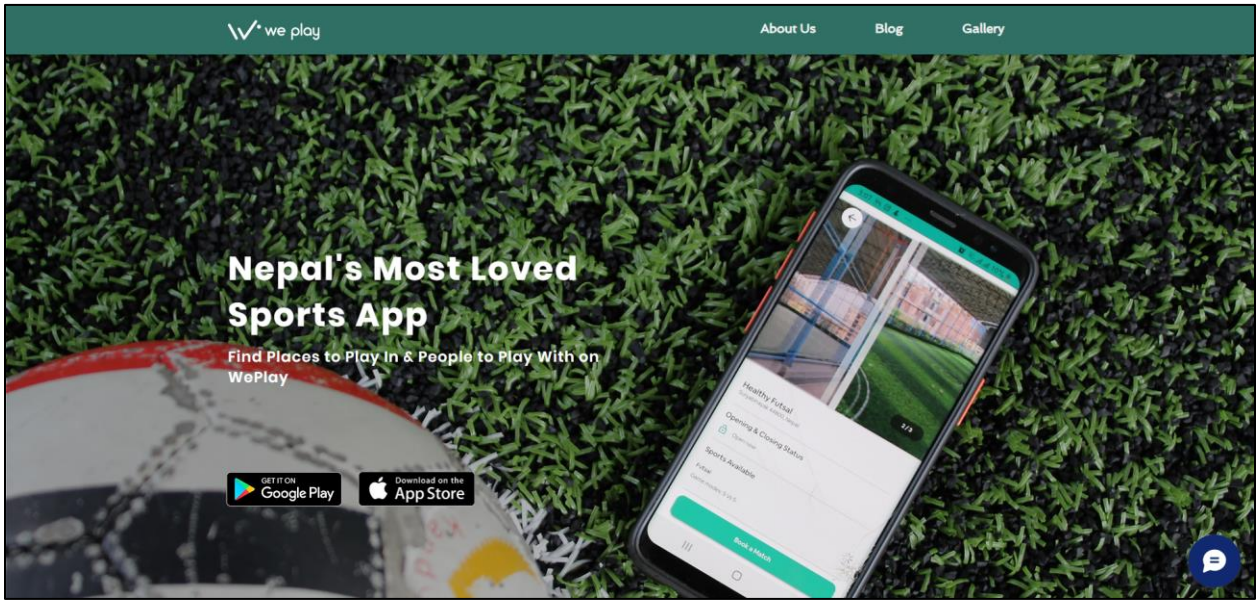


Figure 4: Similar Projects: WePlay Nepal

2.3.1.2. Vakundo App

You may book a futsal venue, search for and find numerous futsal venues around you, and challenge other teams to compete with your user's team—all on Vakundo, a platform made specifically for football fans. Not only that, but the application's dashboard also has news stories pertaining to this field. Venue owners that own futsal courts can get their names out there by registering with the Vakundo app. When you book a venue, you can pay for it using eSewa or Khalti. When users book the futsal facility, they will be able to see the booking progress in real time (Tech Sathi, 2023).

Figure 5: Similar Projects: Vakundo

2.3.1.3. Play Sports

PlaySports is a stringent sport booking app that exemplifies the desire to let people build their activity according to their own needs, rather than forcing it into rigid specifications. When a sports facility uses this system, their preferred facility management becomes up-to-date, transparent, and specialized with a simple booking system. It also provides an easy and quick way to gather contact information for use in future data gathering and analysis. Members can also use the following ways to communicate with one another and share digital club life and information. Their goal is to do away with traditional norms and establish their own digital regulations, so that people can enjoy sports whenever and whenever they like, with anyone they like (Play Sports, 2023).

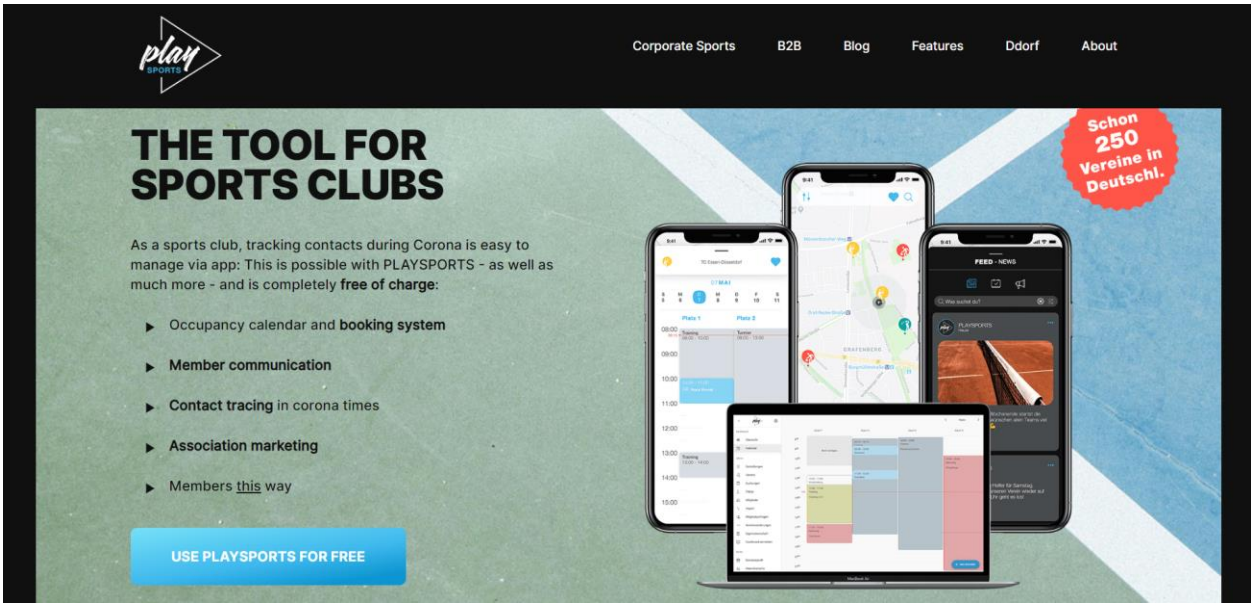


Figure 6: Similar Projects: Play Sports

2.3.2. Comparison between the Similarity of the Systems

Most competing apps shared identical functionality, making it tough to break into the same market. In addition, the idea was that these apps would cater to whatever requirement a futsal owner or player would have. A new strategy was devised to differentiate the client's futsal record system from the competitors after thorough investigation into existing alternatives. A few key characteristics are compared in the table below:

Features	WePlay Nepal	Vakundo App	Play Sports	Proposed Application
Registration and User Management	✓	✓	✓	✓
Online Futsal Slot Booking	✓	✓	✓	✓
Inventory Management and Purchases	✗	✗	✗	✓
Booking & Slots Record Tracking	✓	✗	✓	✓
Revenue Tracking and Management	✗	✗	✓	✓
Online Payment	✓	✓	✓	✓
Multitenancy Architecture	✗	✗	✗	✓
Request on Availability of Players	✓	✗	✗	✓

Table 1: System Comparison: Features and Functionalities

2.4. Software Development Methodology

According to Cambridge Dictionary, a methodology is a system of ways of doing, teaching, or studying something (Cambridge, 2023). Similarly, a software development methodology is a technique or set of techniques employed during the creation of software. Although it is a broad category, it not only includes stages such as planning and creation but every engineering structure of distinct stages from start to the end of a software development. Its purpose is to explain the mechanics behind a program's development and maintenance (Alliance Software, 2023).

2.4.1. Considered Methodology

Out of all the possible software development methodologies, a few of the reviewed and considered methodologies are as follows:

- Rational Unified Process Methodology
- Rapid Application Development Methodology
- Kanban Methodology

Each of the aforementioned models for creating software was examined in detail, right down to the smallest details. In the appendix section, a brief overview of each approach is provided and discussed on why it wasn’t put into practice.

2.4.2. Selected Methodology

Personal Scrum was chosen as the framework for the entire duration of the project, from inception to completion, as the preferred model for software development. In the appendix section, the model is defined, and its components and operation are briefly discussed. The advantages of using Personal Scrum, however, and why that methodology was selected for this project's creation, will be discussed under the next item.

Case Study Scenario	Developing a system to in iterations and following agile techniques.
Features	The development process allows continuous delivery.
Justification	Scrum permits project completion in a predetermined amount of time by delivering portions of the production in iterations, which leads to feedback for effective delivery.

Table 2: Justification Table: Scrum Model (1)

Case Study Scenario	Developing a system with innovative ideas and collaborations
Features	The development process allows assistance in boosting up fostering and designing creativity.
Justification	Scrum encourages and promotes creative approaches and allows analyzing of all those ideas into newer innovation.

Table 3: Justification Table: Scrum Model (2)

Case Study Scenario	Developing a system allowing any dynamic modifications.
Features	The development process enables addition of any new features on the system during any of its adaptation stages.
Justification	Scrum offers improved flexibility that allows enhanced adaptability for transitional development steps and modifications within any stage cycle.

Table 4: Justification Table: Scrum Model (3)

Case Study Scenario	Developing a system with continuous interaction between stakeholders to resolve any communication gap.
Features	The development process allows reciprocity between the team and the client.
Justification	Scrum incorporates meetings which helps in identifying and resolving any sorts of challenges.

Table 5: Justification Table: Scrum Model (4)

Case Study Scenario	Developing a system not in just an application level but also following all sets of guidelines for a proper documentation.
Features	The development process allows smooth transparent monitoring of work done and its task divided.
Justification	Scrum facilitates open communication and provides a standardized framework for documenting and monitoring project development which allows in achieving transparency (Quick Start, 2022).

Table 6: Justification Table: Scrum Model (5)

Thus, a Scrum model is a legitimate technique to adhere to because of its propensity to stimulate client contact, flexibility in the face of change, iterative development, and visualization to aid in the accomplishment of productivity throughout the course of application development. The second most important factor was the fact that this approach views the app service as an entity for a long-term vision, rather than just a project to complete.

A complete descriptive analysis on what scrum is, how scrum is applied in an actual long-term project and how it will be implemented in the following project has also been defined in the appendix section. Link to Appendix.

2.4.3. Methodologies Comparison

A basic justification for comparing each of the chosen approaches with each of the other methodologies that were considered have been noted down below:

Features	Methodologies			
	RAD Model	**RUP Model**	**Kanban Model**	**Scrum Model**
Agility Dynamics	✓	✗	✓	✓
Innovation Ideas	✗	✓	✓	✓
Differentiation via emerging technologies	✗	✗	✗	✓
Fast time for production	✗	✗	✓	✓
Client and End User Interaction	✓	✗	✓	✓

Table 7: Methodology Comparison: Features and Applications

3. Development to Date

Discussions regarding possible solutions and the reasoning behind the methods to be utilized in developing the project followed the specification of the problem's scope and objectives, which provided a framework for solving it. The software development lifecycle was designed with the scrum methodology in consideration. Two-week sprints were employed, and assignments concluded after a Gantt chart was created focusing on a module for each progress period. Designing a system that effectively incorporates all of the elements was the following stage after completing all of the procedures. Instead of jumping right into hard coding, it was necessary to first make design patterns, which include tracing features and functionalities, creating an entity relationship diagram and any other UML diagrams that are needed, and making wireframes. These will all be useful when it comes time to develop algorithms for coding. In terms of interim progress, three sprints have been completed, with an emphasis on management and the business case rather than technical details, as seen in the Gantt chart. The chronological progression of the system's development can be visualized in each of the headings below.

3.1. Requirement Gathering

Since the project is client-centric, most of the specifications were obtained from the provider. The client accepted the project after it was presented, and all of its features were outlined. However, it was also vital to conduct a survey to learn what individuals consider the current state of sports facilities in our country and how it may be improved in terms of user experience and to gain insights and feedback in a large scale. The primary objective of this online survey was to get feedback from a large number diverse group of individuals and audience about ways to enhance Nepal's sports management system. The survey's description has been preserved in the appendix.

3.2. Data Flow Diagram

The information in any process or system can be mapped out using a Data Flow Diagram. Inputs, outputs, storage locations, and pathways are all represented by standard icons like rectangles, circles, and arrows, along with brief descriptive captions. Data flowcharts can be as basic as a hand-drawn overview of the process or as complex as a multi-level data flow diagram, which provides progressively more specific details on the data's processing. Both existing systems and proposed ones can benefit from their utilization (Lucid Chart, 2023).

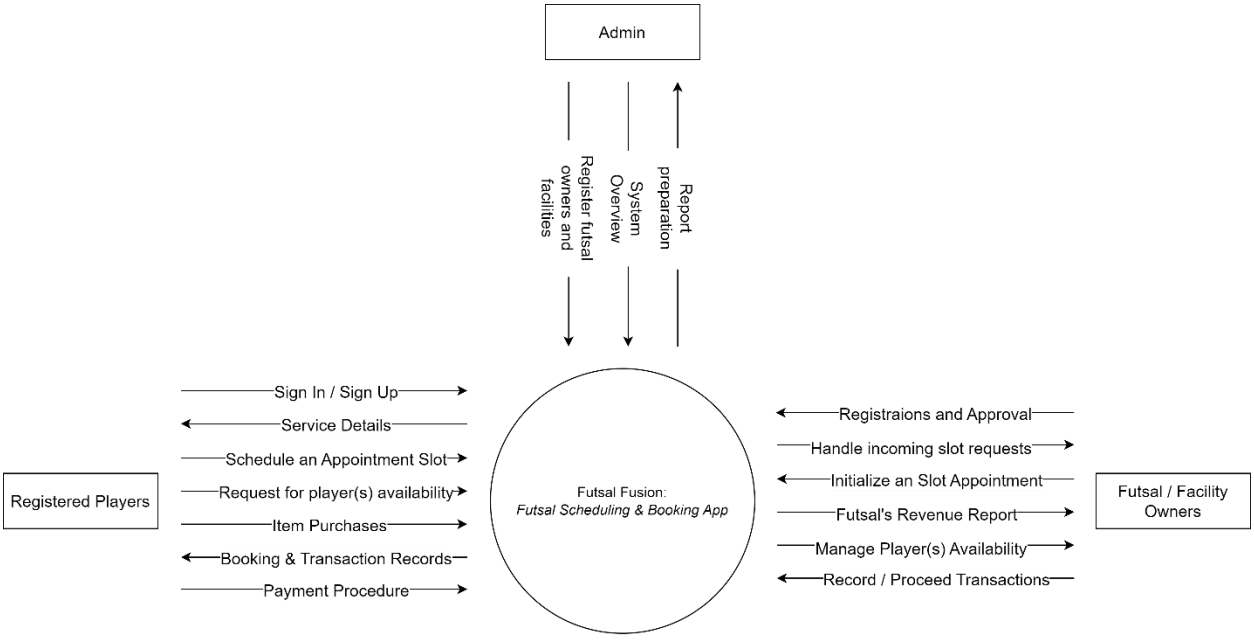


Figure 7: Data Flow Diagram: Level 0

3.3. Use Case Diagram

The Unified Modeling Language's Use Case Diagram is a graphical representation of the system's user base and their interaction with the system (Lucid Chart, 2023).

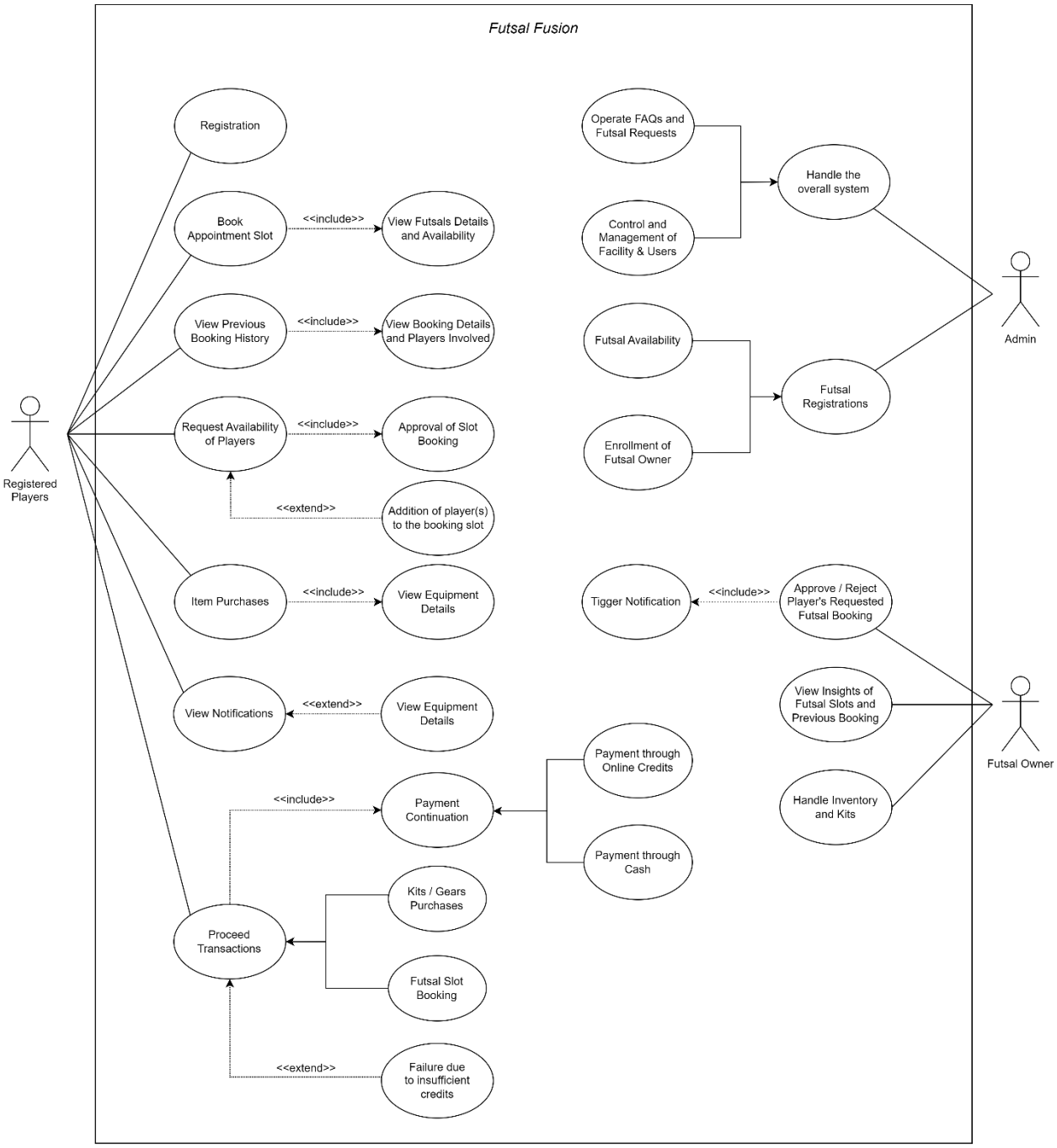


Figure 8: Use Case Diagram: Overview

3.4. High Level Use Case Diagram

Extensive application in the invention process, defines the description as brief explanation or unstructured summary of the use case at hand. The goal is to give just enough details to make its complexity clear and to help build a coherent explanation of the purpose and membership of each function. The appendixes include explanations of the various higher-level descriptions used for each of the developed use cases. Link to Appendix.

3.5. Software Requirement Specification

System Requirement Specification is the document outlining the intended functionality and performance of the software, in short also termed as the SRS Document. It goes on to detail the features that the product must have in order to satisfy everyone involved such as the company and its end users. It can be considered as a road map or blueprint for the program which lays out the goal of the product, explains what is being built, specifies each user's needs, and then submits it for review and approval.

The project's scope, features, and applications are all defined in an SRS document about the proposed system that is included in this document in the appendix. Link to Appendix.

3.6. Activity Diagram / Flowchart

A UML (Unified Modeling Language) diagram that graphically depicts the flow of operations within a system or process is an activity diagram. It shows the process flow of an application in real time, showing how various parts interact with one another to finish a job. By outlining the interdependencies and interactions between different parts, it helps capture and clarify the needs of a system or feature and provides a clear visual depiction of the interconnections and sequencing of various operations or processes. Stakeholders can better understand and participate in the early stages of a project's lifecycle when activity diagrams are used to illustrate the flow of activities. Additionally, possible bottlenecks, resource limits, or ambiguities can be discovered early on using activity and relationship mapping, enabling proactive risk reduction (Lucid Chart, 2023).

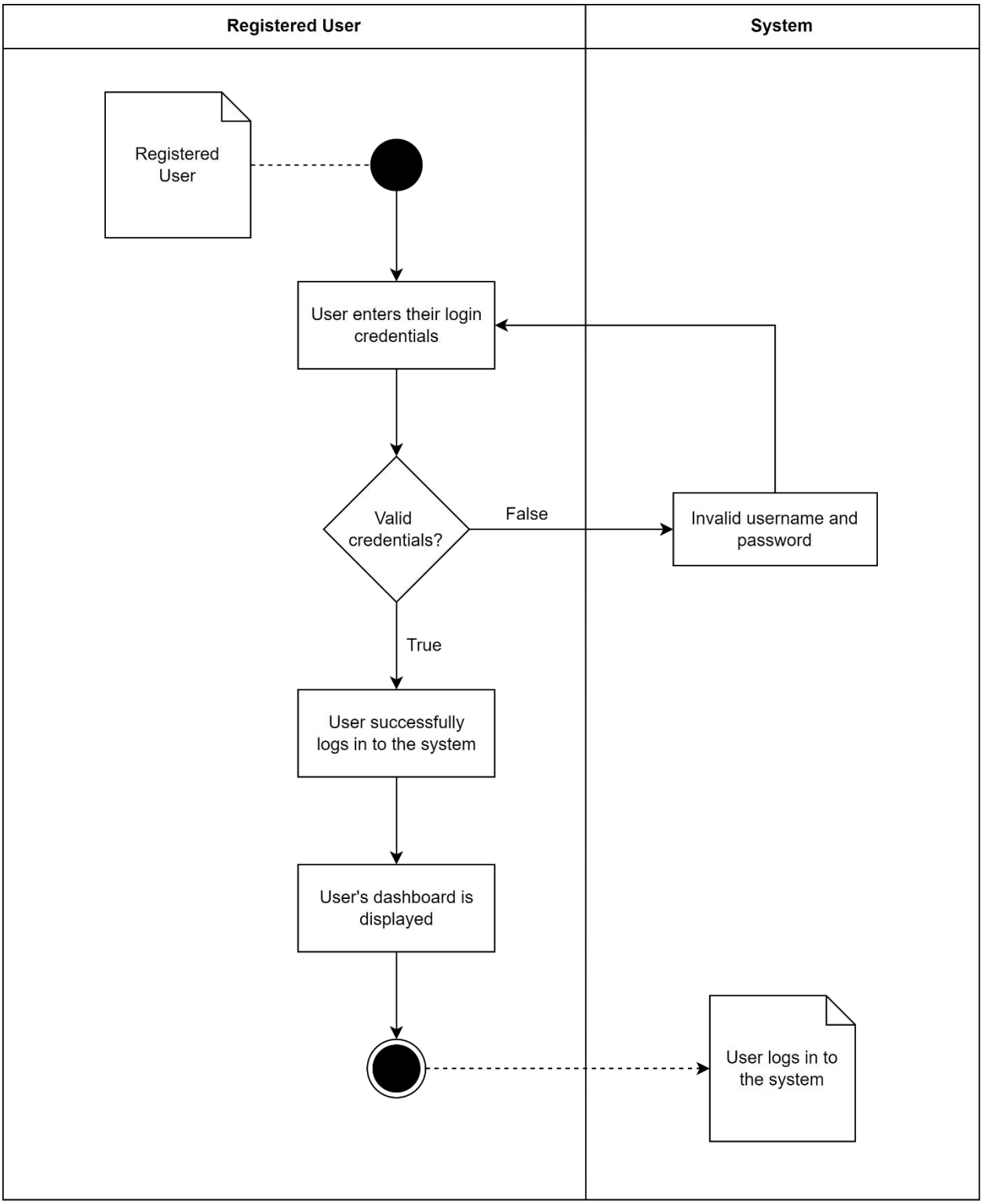


Figure 9: Activity Diagram: User Login

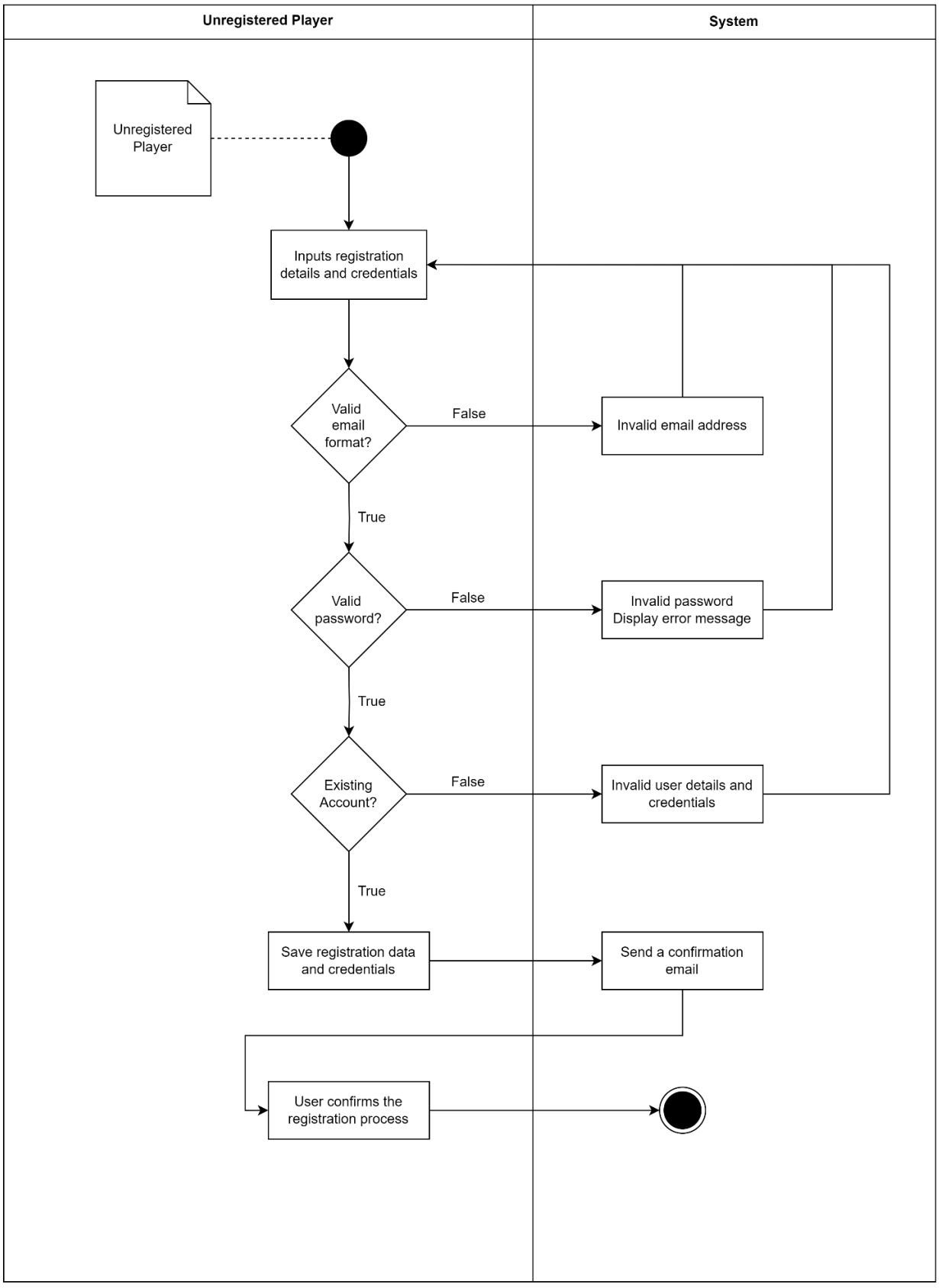


Figure 10: Activity Diagram: Player Registration

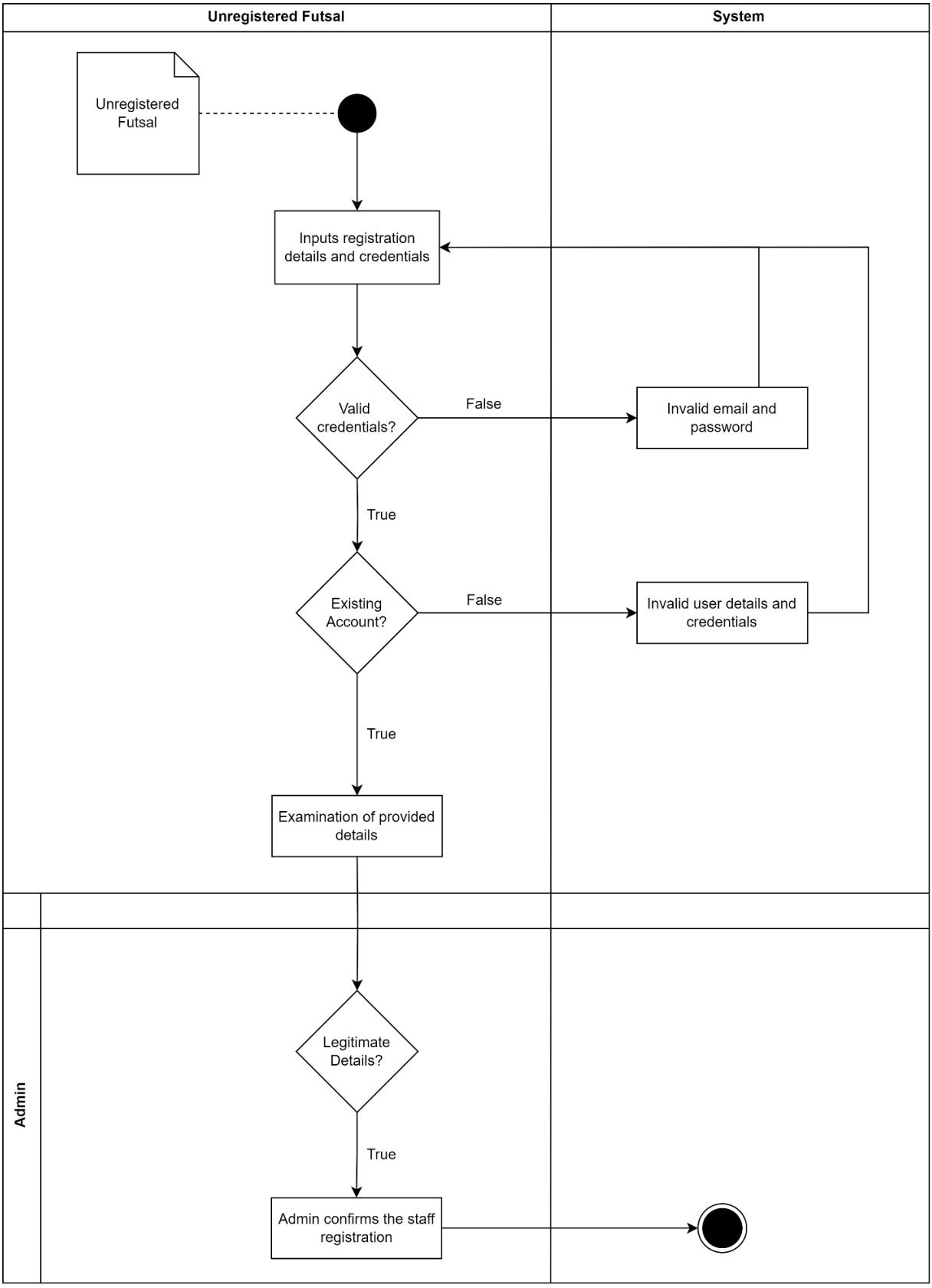


Figure 11: Activity Diagram: Futsal Registration

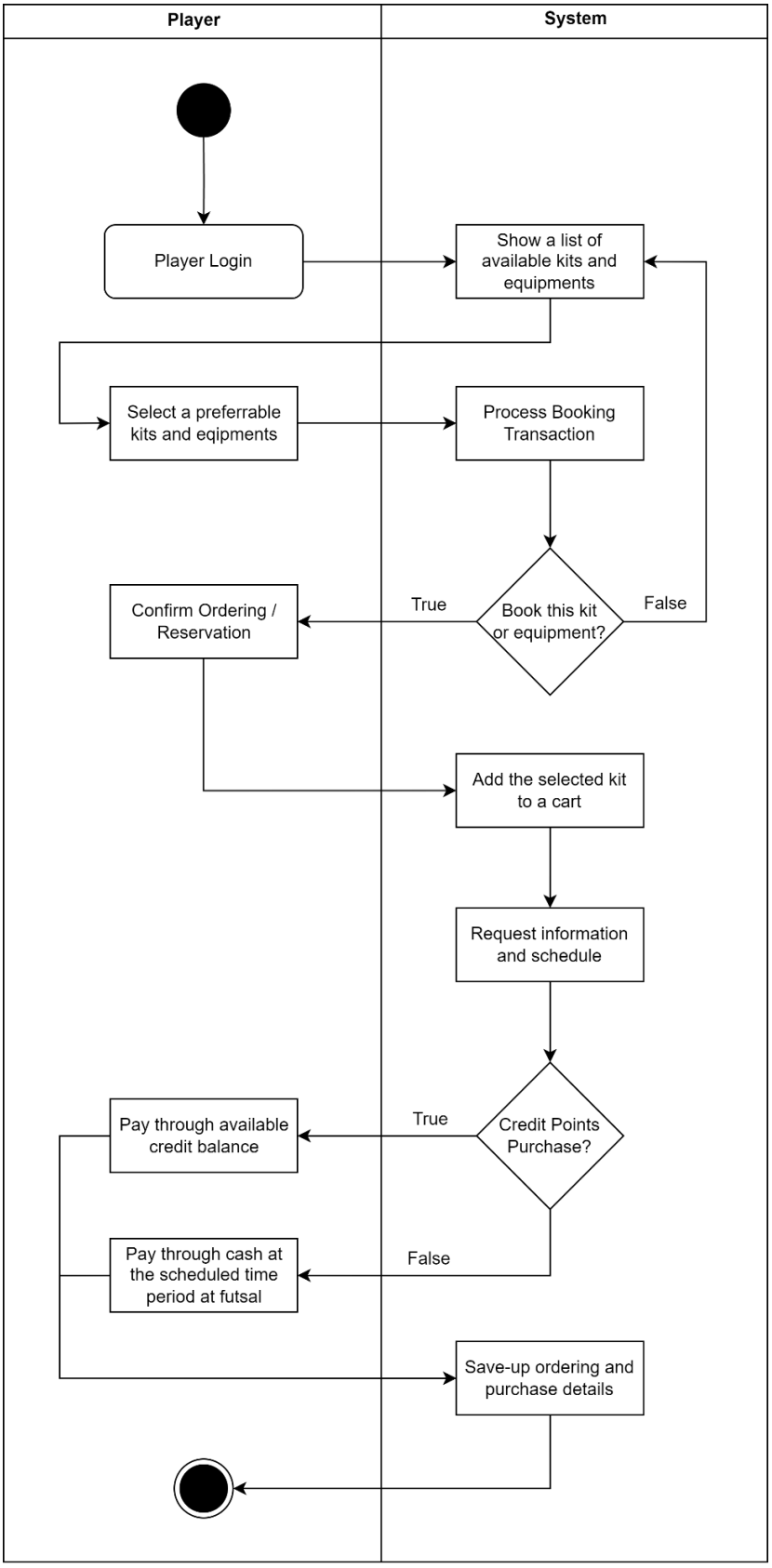


Figure 12: Activity Diagram: Kit Purchase

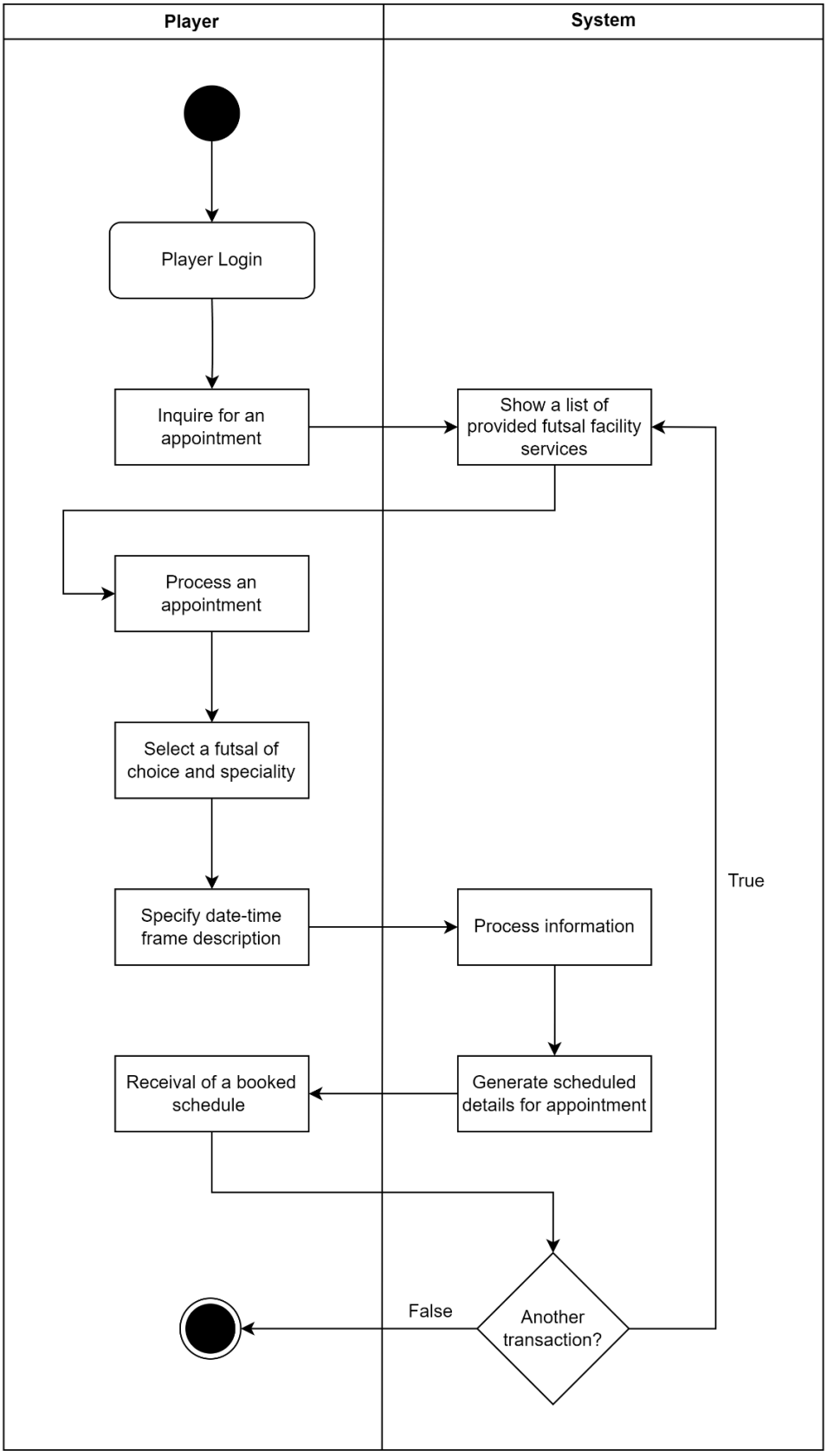


Figure 13: Activity Diagram: Appointment Initialization

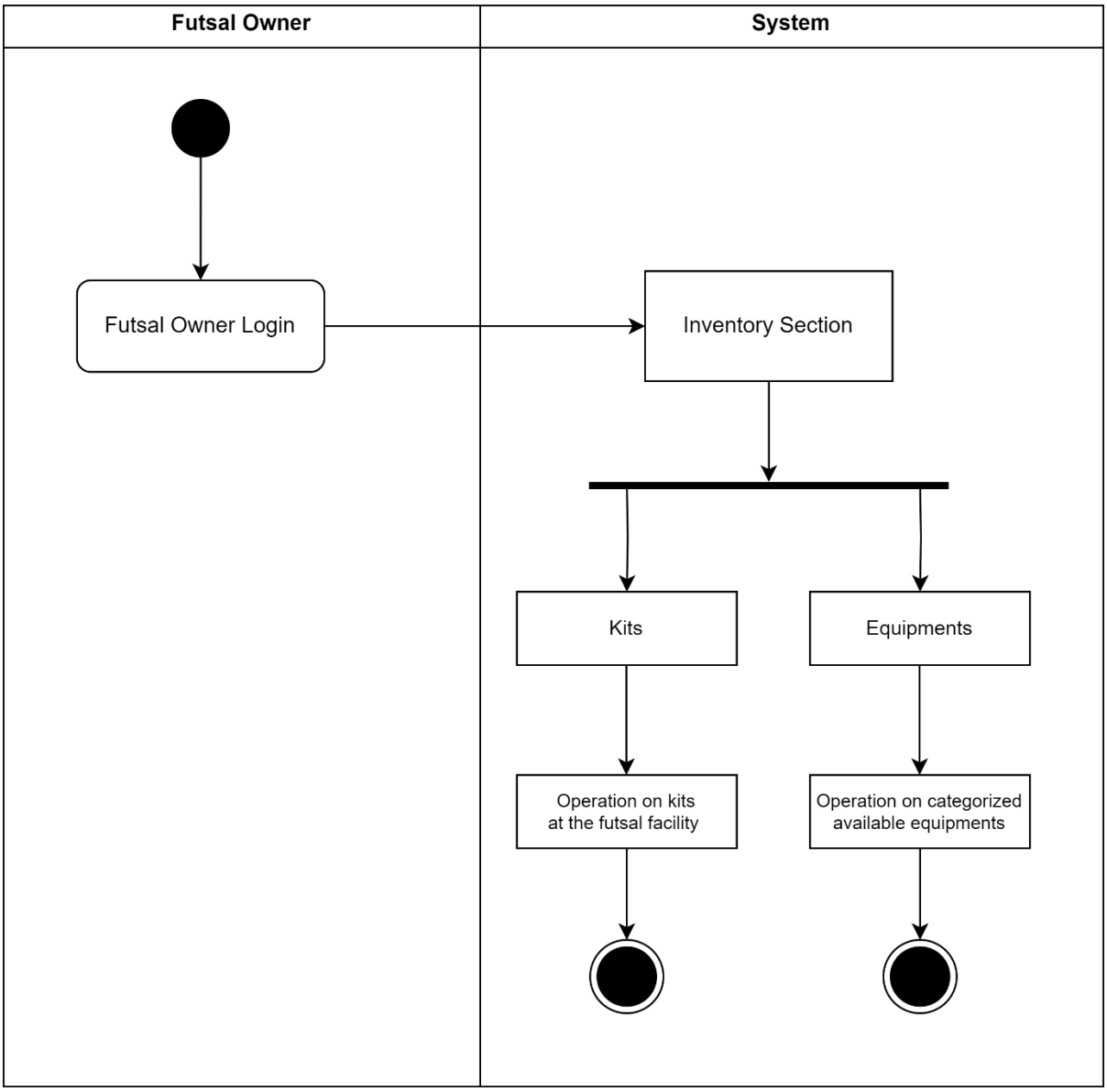


Figure 14: Activity Diagram: Inventory

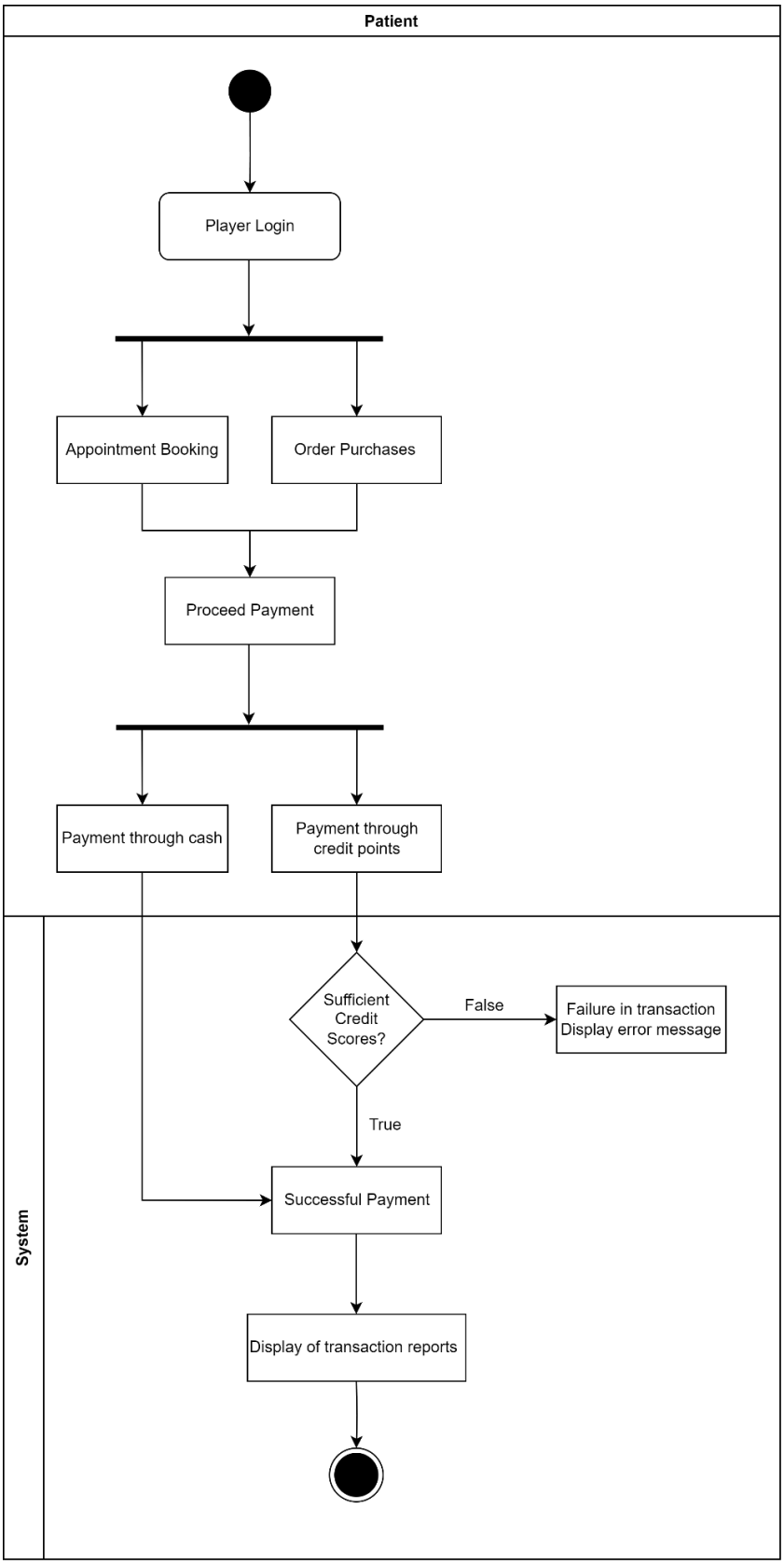


Figure 15: Activity Diagram: Payment Procedure

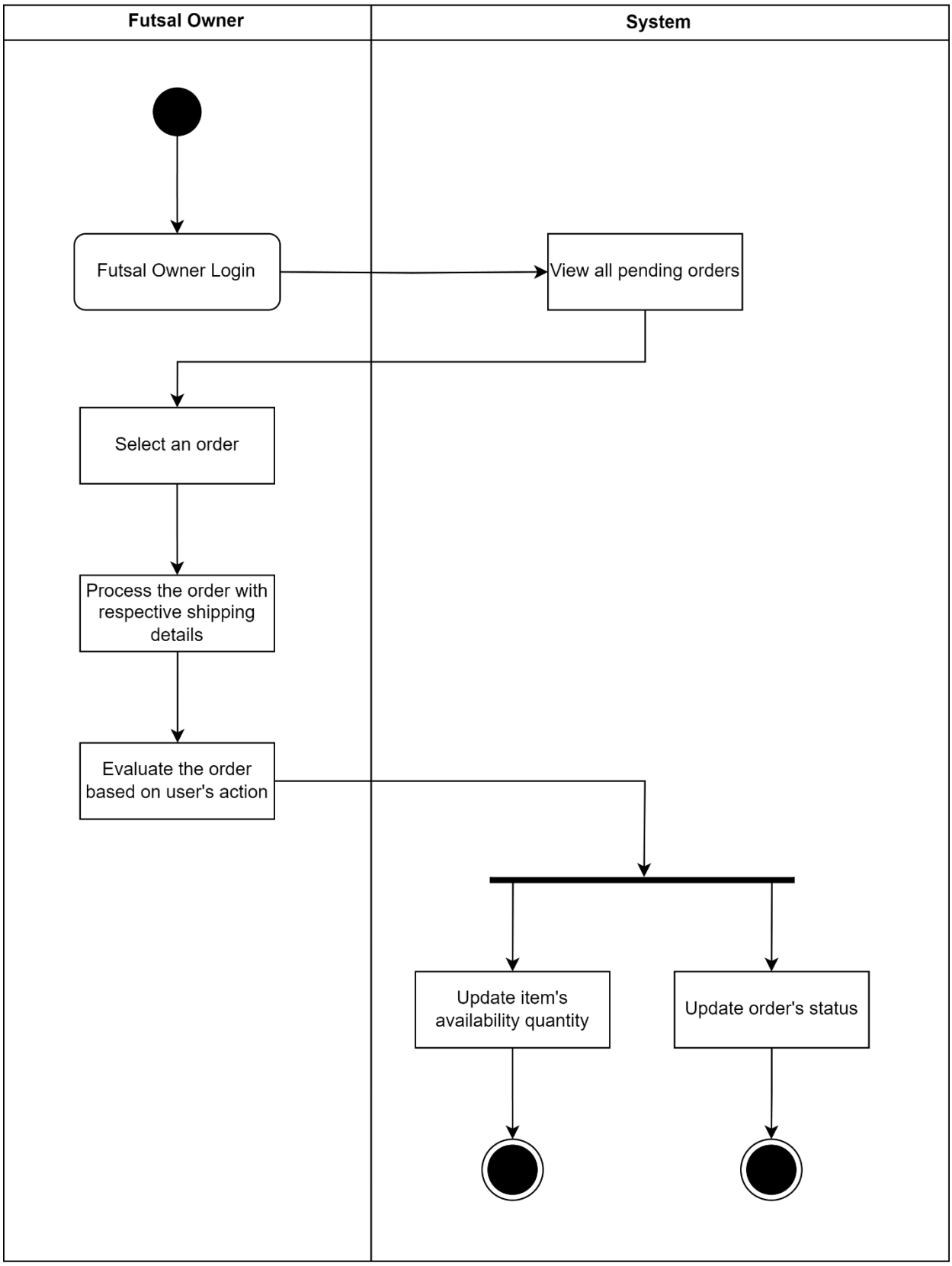


Figure 16: Activity Diagram: Order Processing

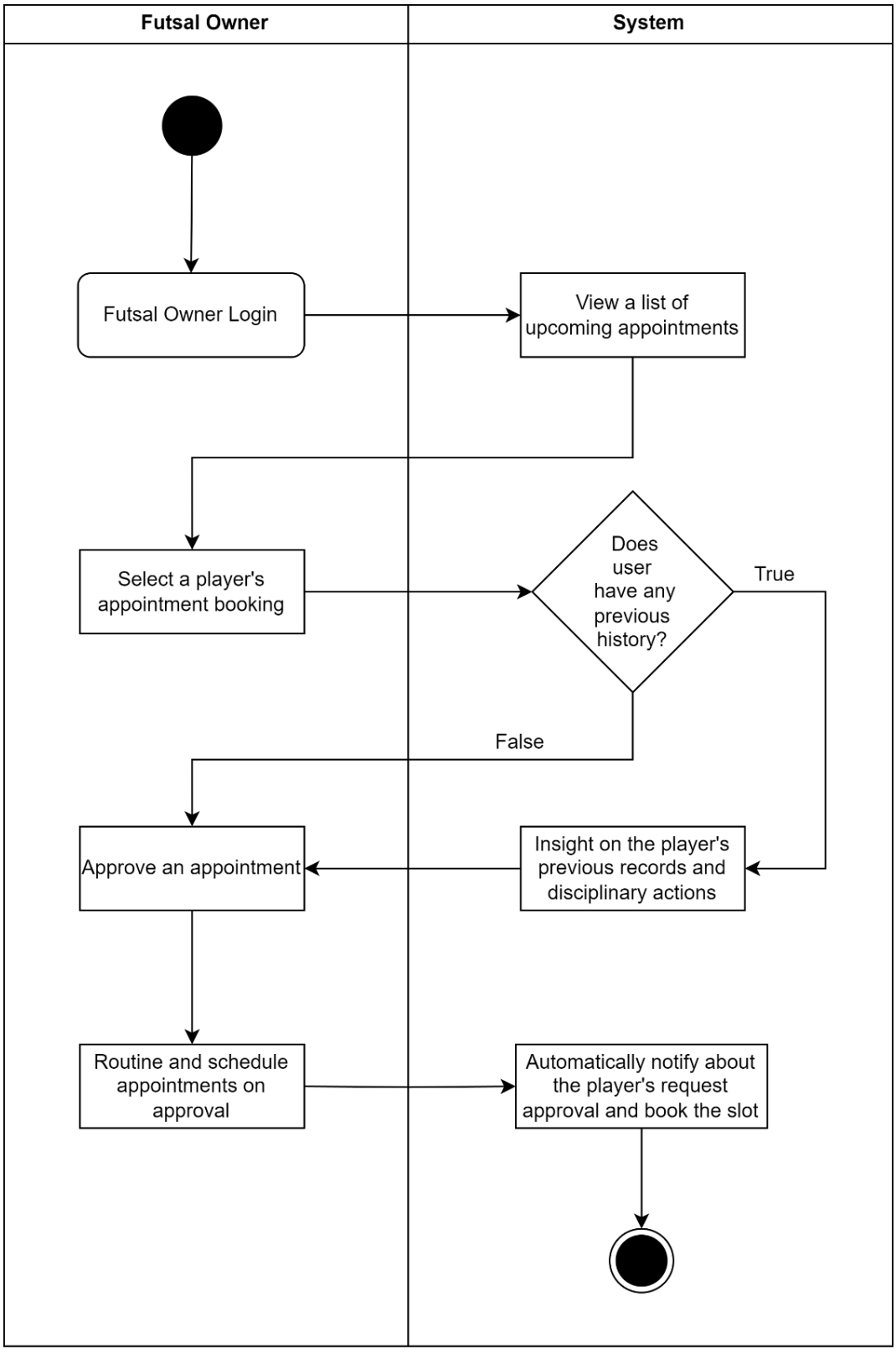


Figure 17: Activity Diagram: Appointment Processing

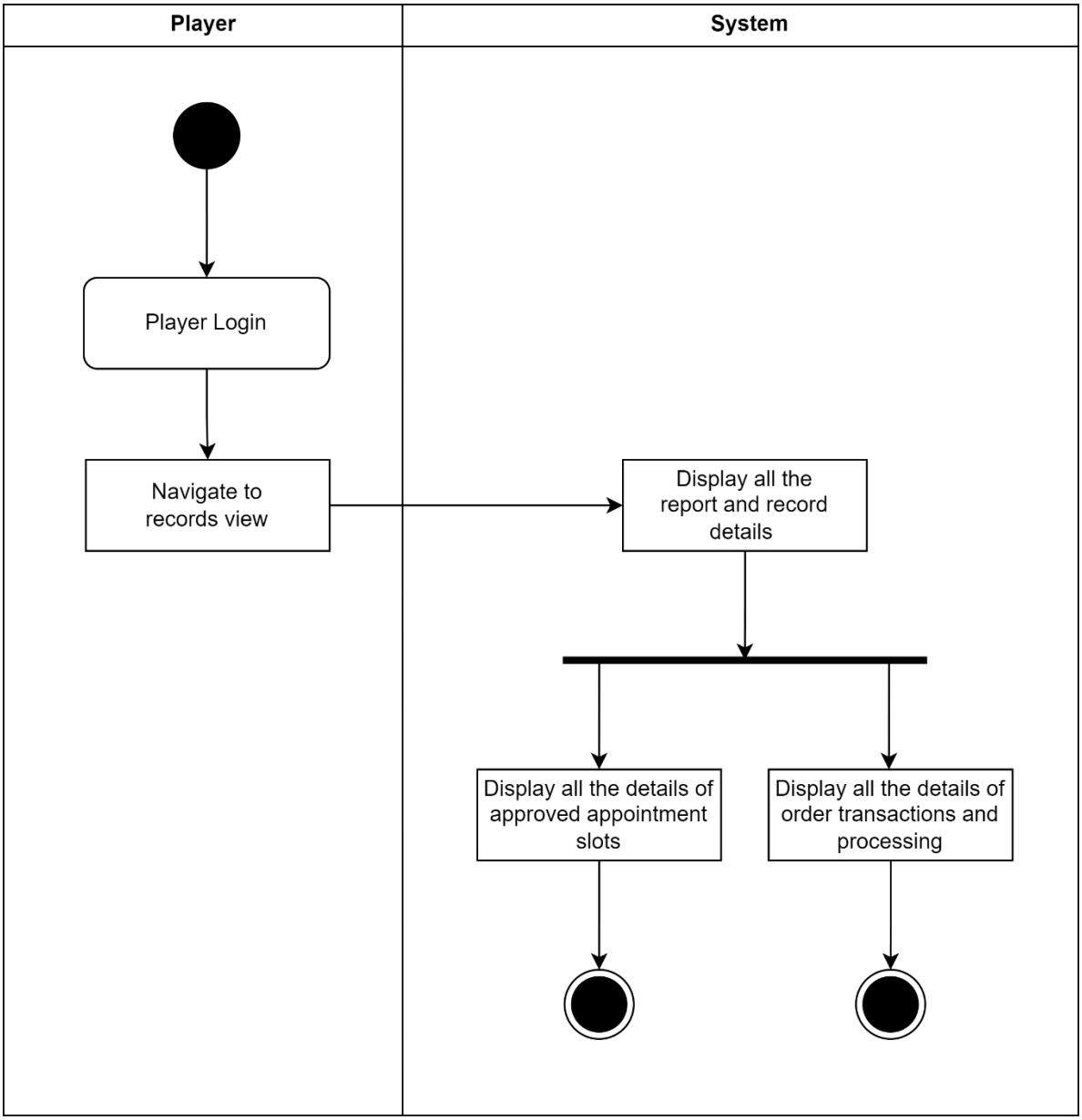


Figure 18: Activity Diagram: Report and Record Details

3.7. Sequence Diagram

Interactions and messages exchanged between objects or components within a system over time can be illustrated with a sequence diagram, a sort of UML diagram. It shows the system in real time, showing how features or scenarios are executed in chronological order. For several reasons, sequence diagrams are priceless when creating software. They guide developers by providing a visual representation of the control flow between objects and highlighting any problems or ways to enhance it. Developers and stakeholders can learn more about the system's behavior by drawing a flowchart of all the interactions. This helps them validate features before diving into technical development. A more informed and efficient development process is the result of this proactive strategy, which improves team communication, clarifies the system's dynamics, and helps refine requirements (Lucid Chart, 2023).

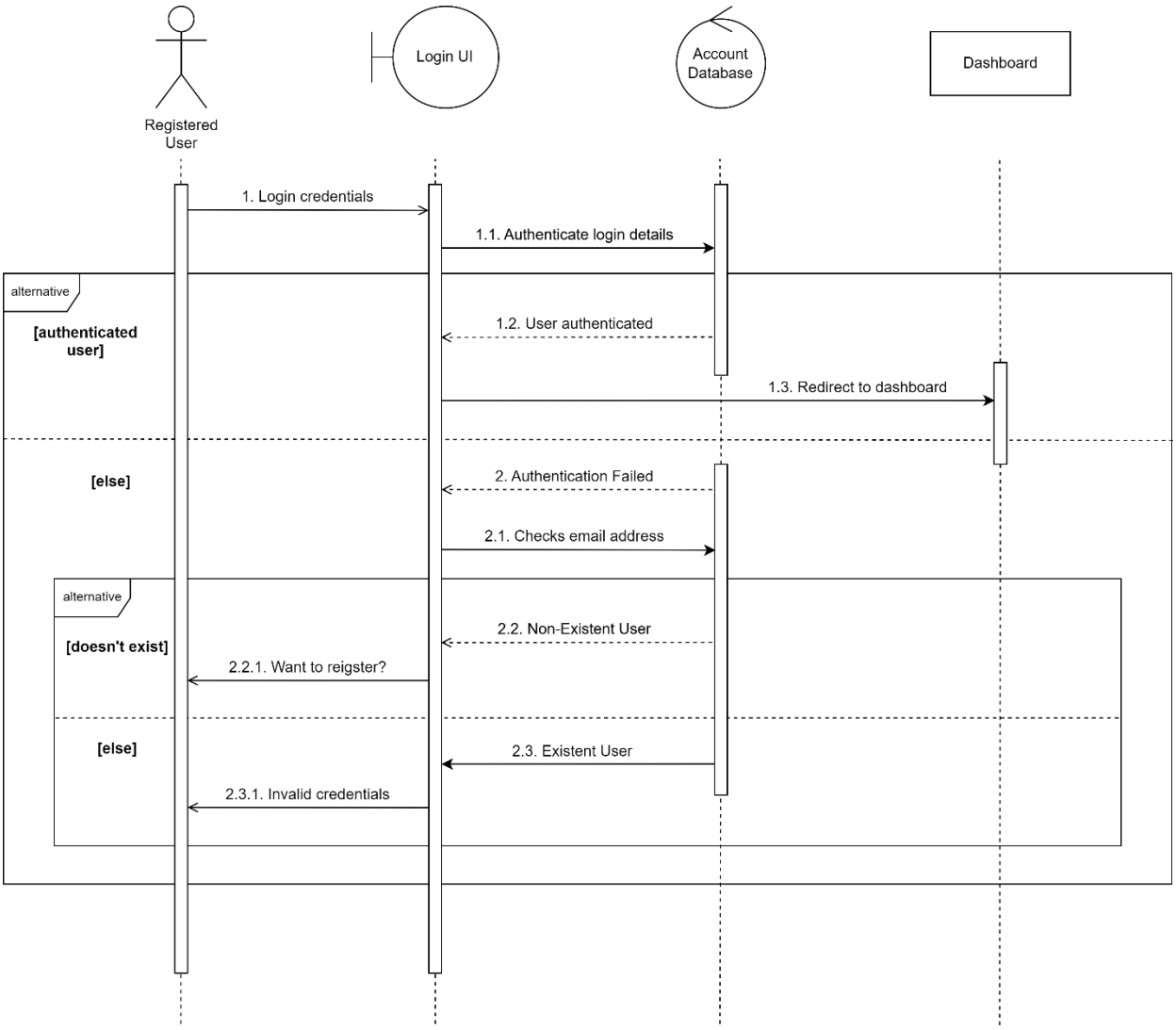


Figure 19: Sequence Diagram: Login

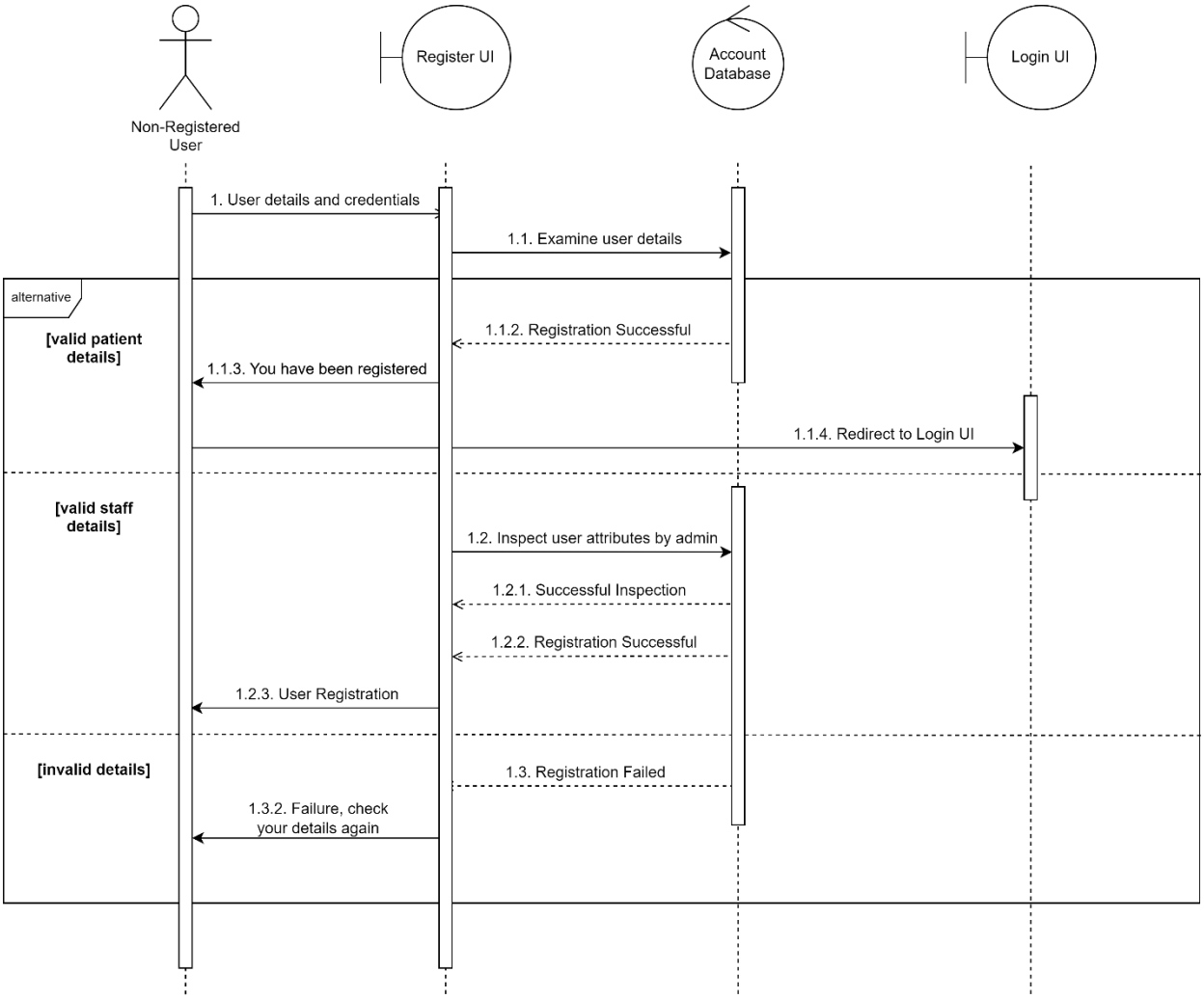


Figure 20: Sequence Diagram: Registration

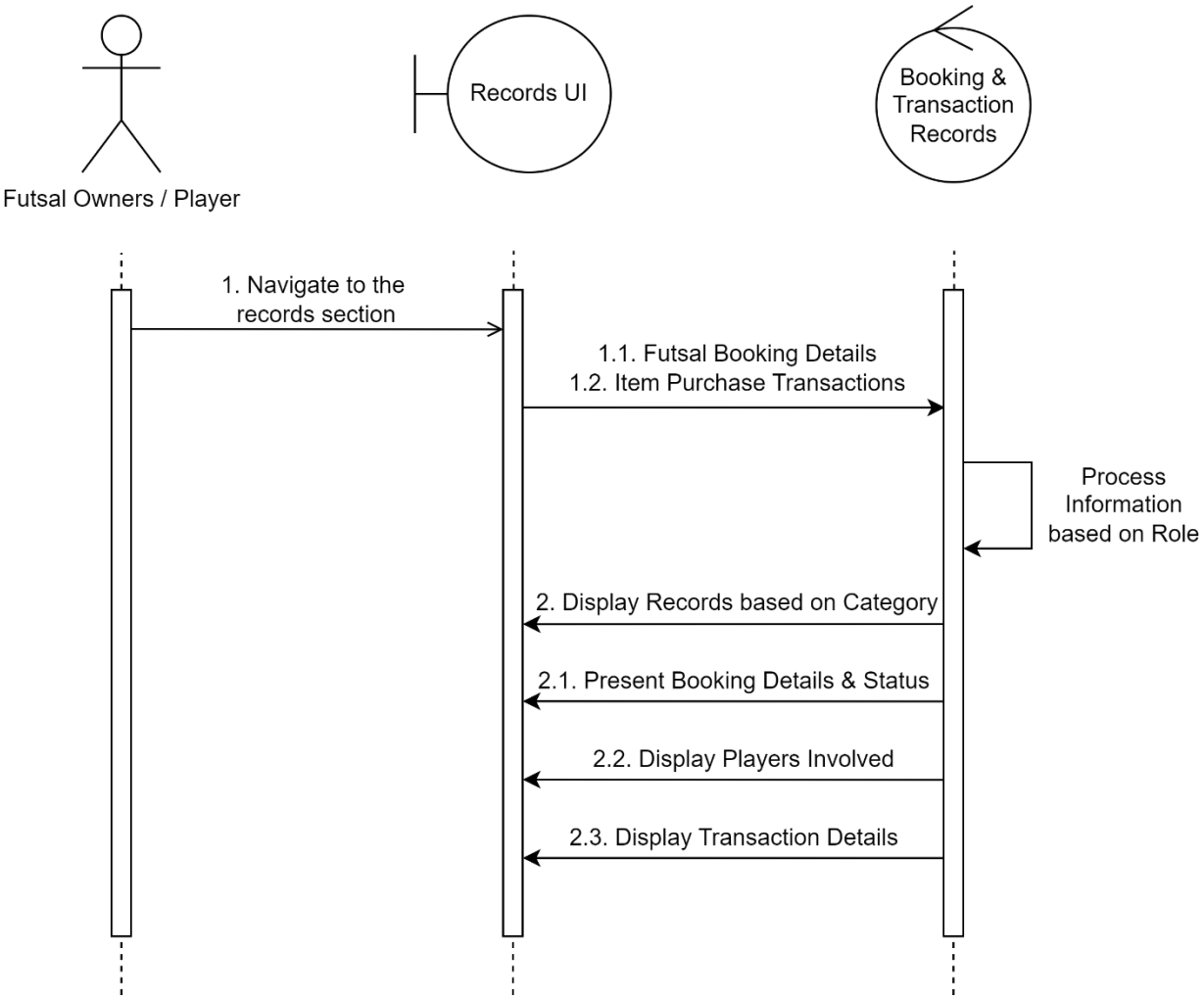


Figure 21: Sequence Diagram: Booking and Transaction Record Tracking

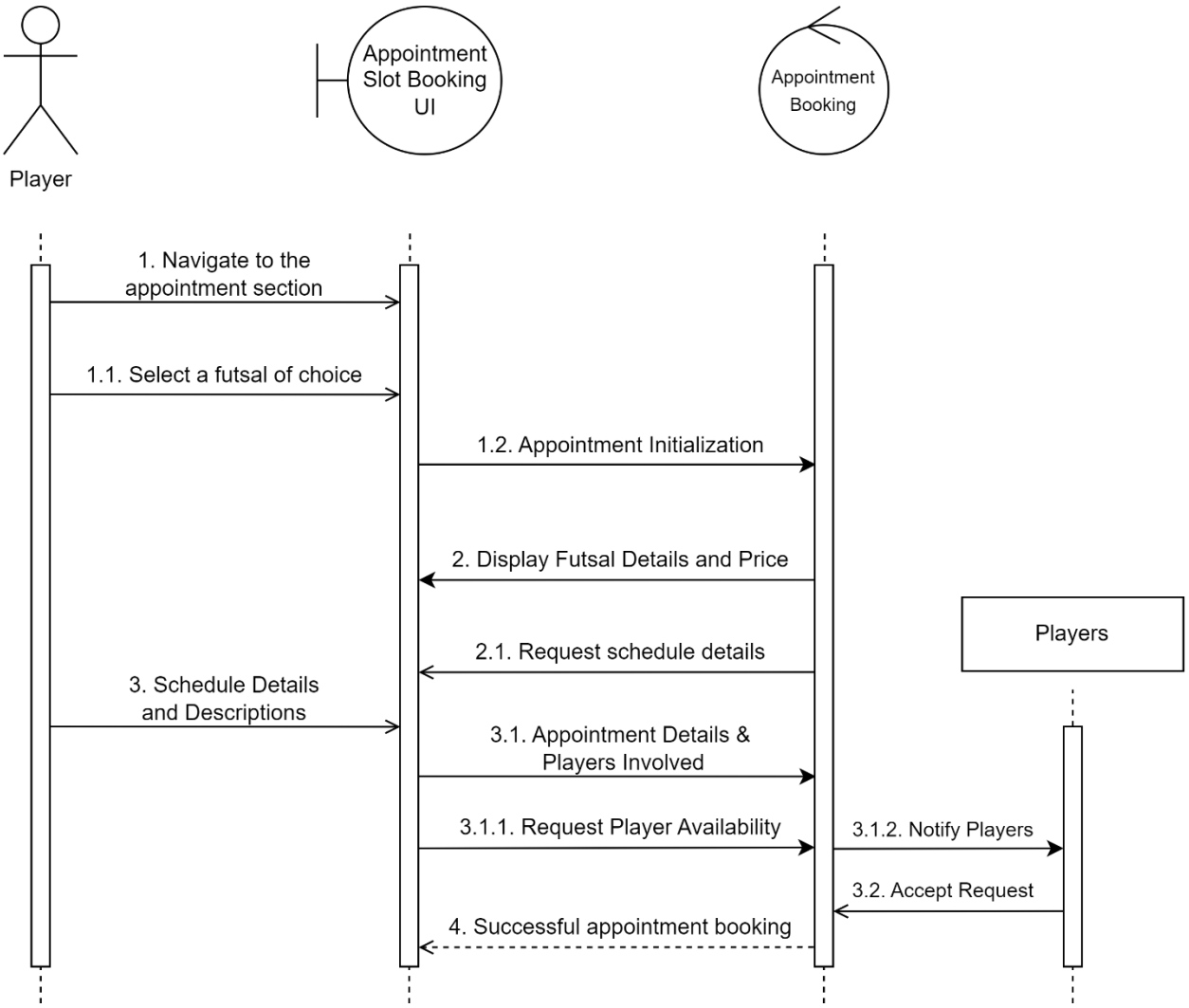


Figure 22: Sequence Diagram: Appointment Booking

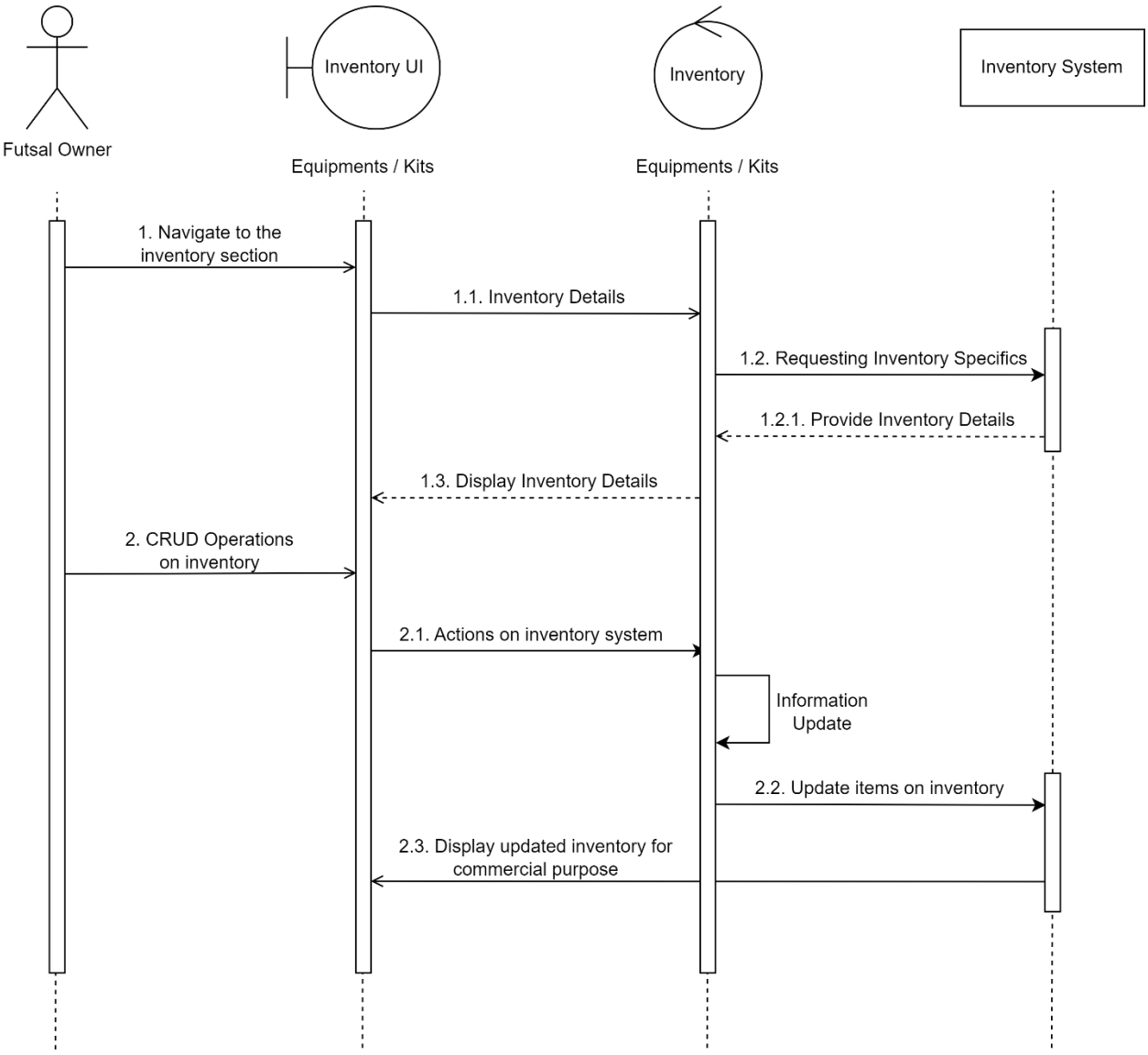


Figure 23: Sequence Diagram: Inventory Management

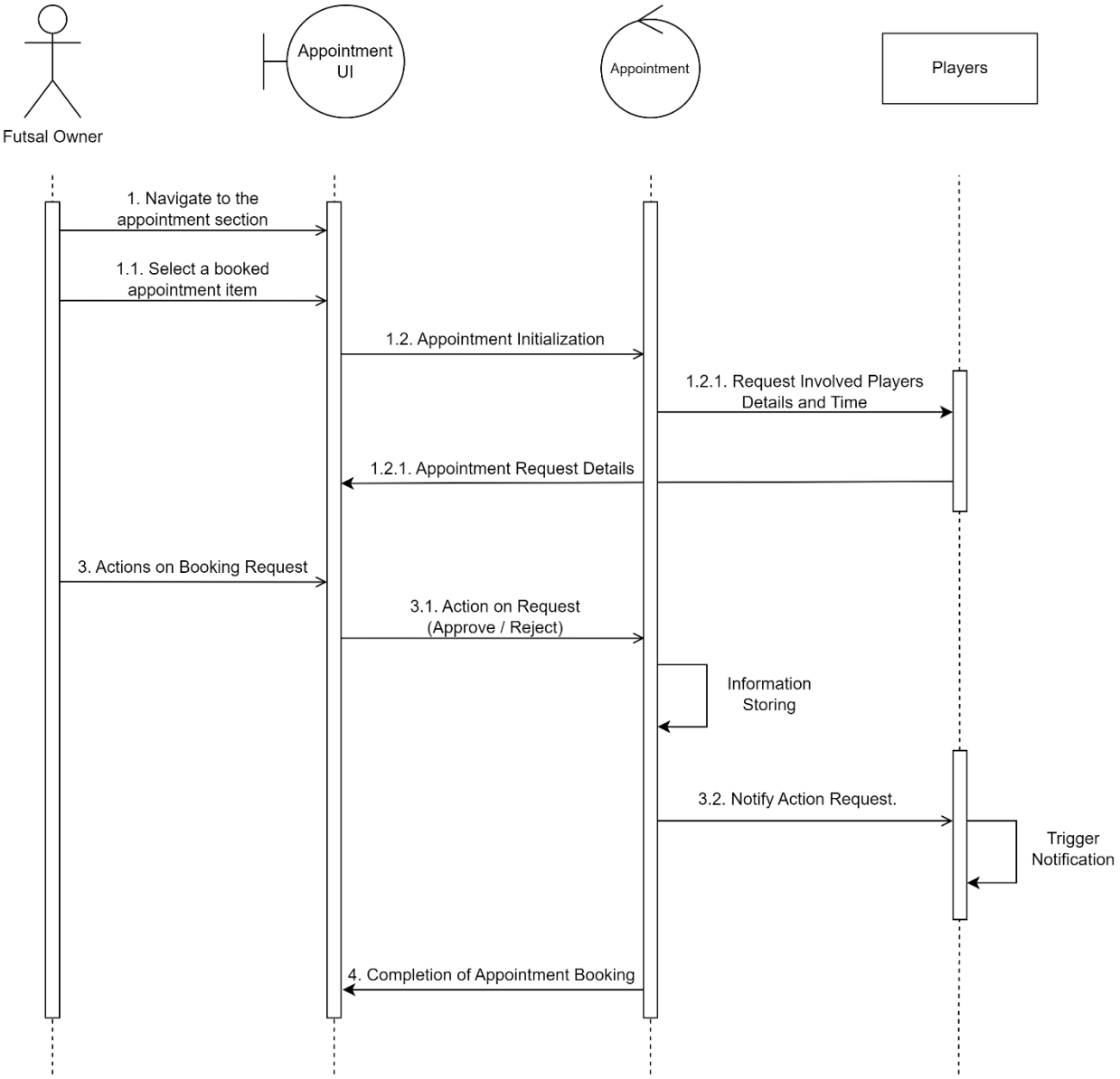


Figure 24: Sequence Diagram: Appointment Processing

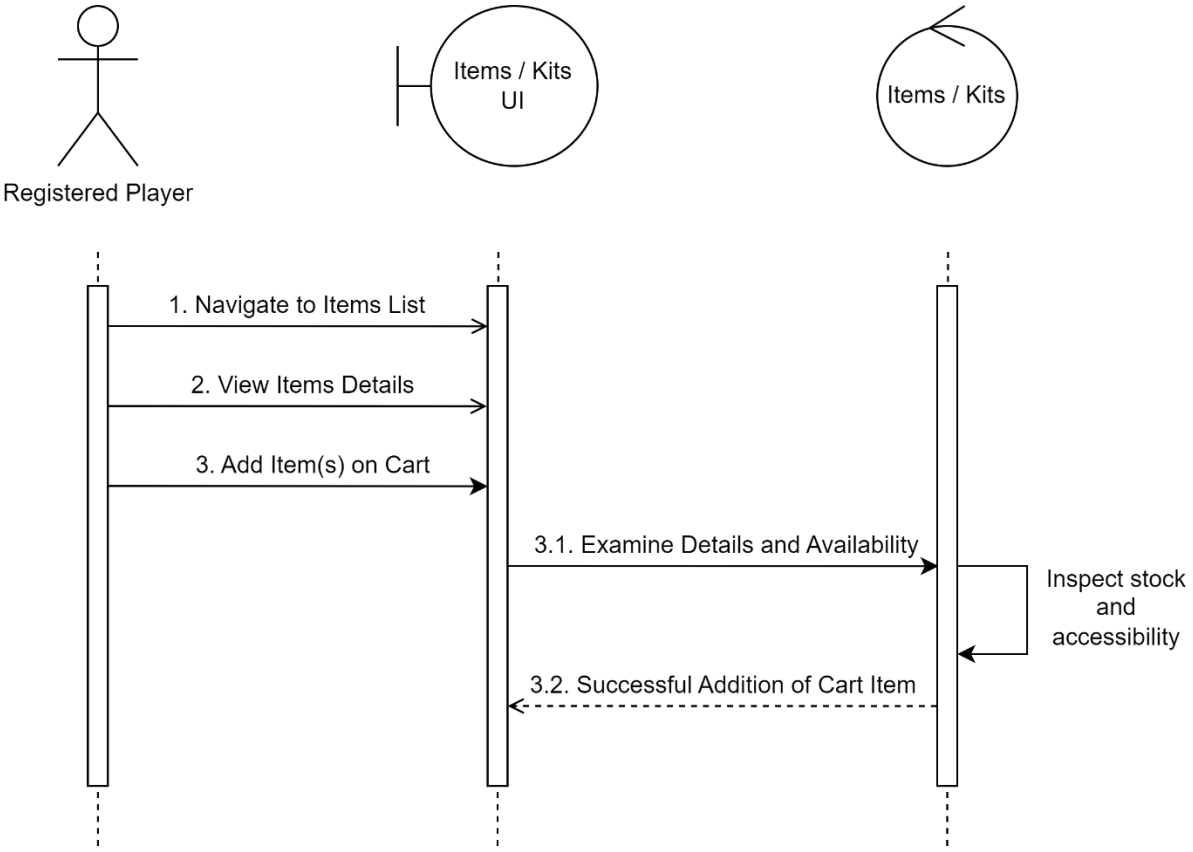


Figure 25: Sequence Diagram: Add Items to Cart

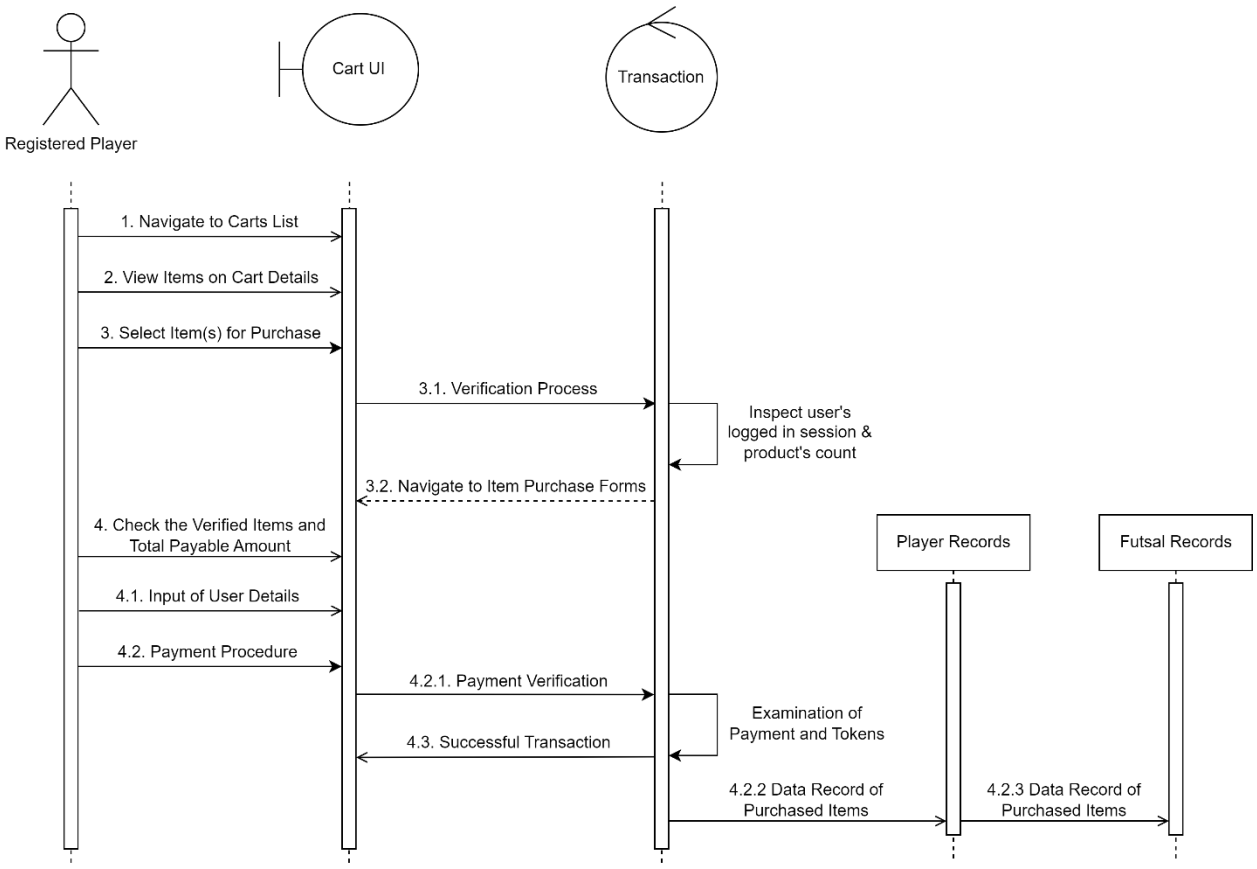


Figure 26: Sequence Diagram: Order Processing

3.8. Collaboration Diagram

The Unified Modeling Language (UML) includes a collaboration diagram that shows how different parts of a system communicate with one another. Collaboration diagrams highlight the structural structure of items and how they work together to produce specific functionality, as opposed to sequence diagrams that focus on the sequential flow of messages. Software engineers and stakeholders might benefit from collaboration diagrams while trying to visualize the system's architecture and comprehend the static relationships between items. In order to trace the aspects of a project before technical development begins, this visualization is crucial. It helps teams find and refine the partnerships that are required between various classes or components. Better communication, better design decisions, and a more streamlined development process are all benefits of using collaboration diagrams early in the project lifecycle. These diagrams outline the static structure and linkages, which helps to understand and efficiently implement the envisioned features (Lucid Chart, 2023).

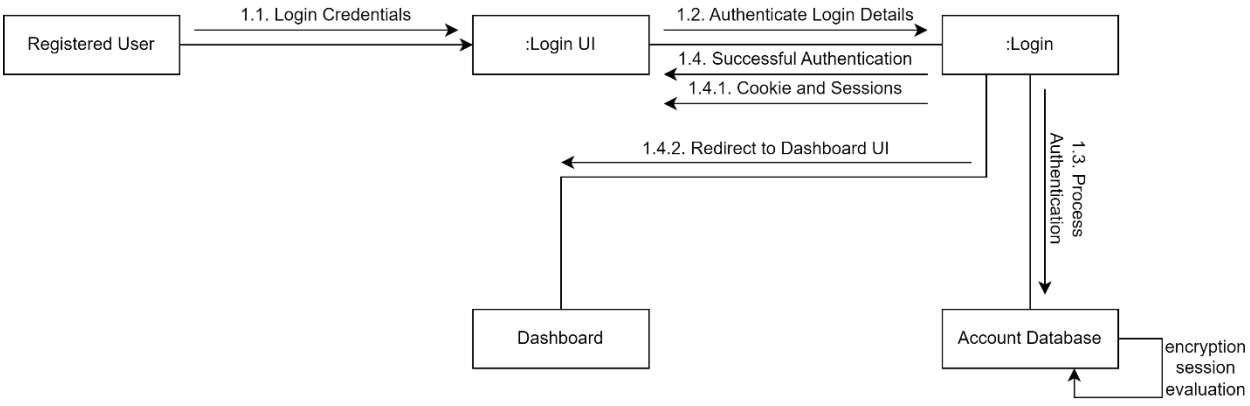


Figure 27: Collaboration Diagram: Login

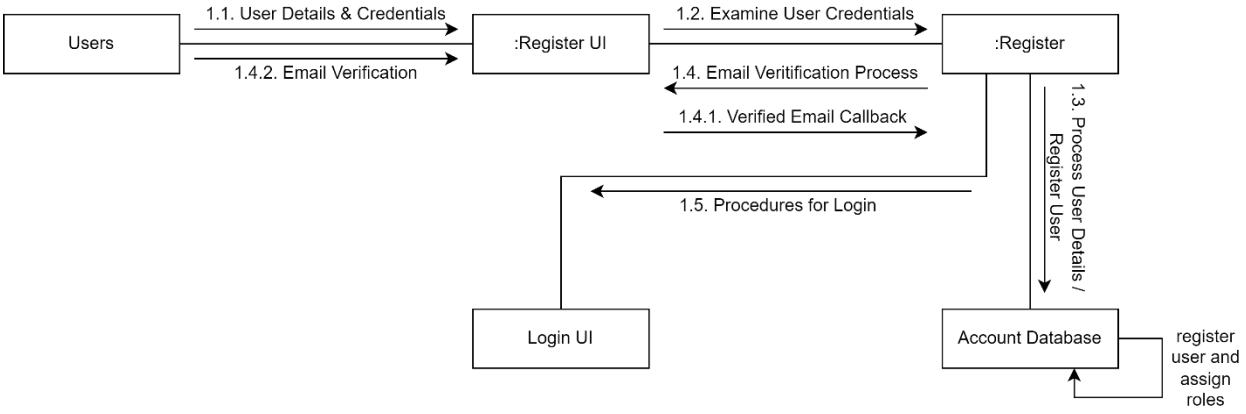


Figure 28: Collaboration Diagram: Registration

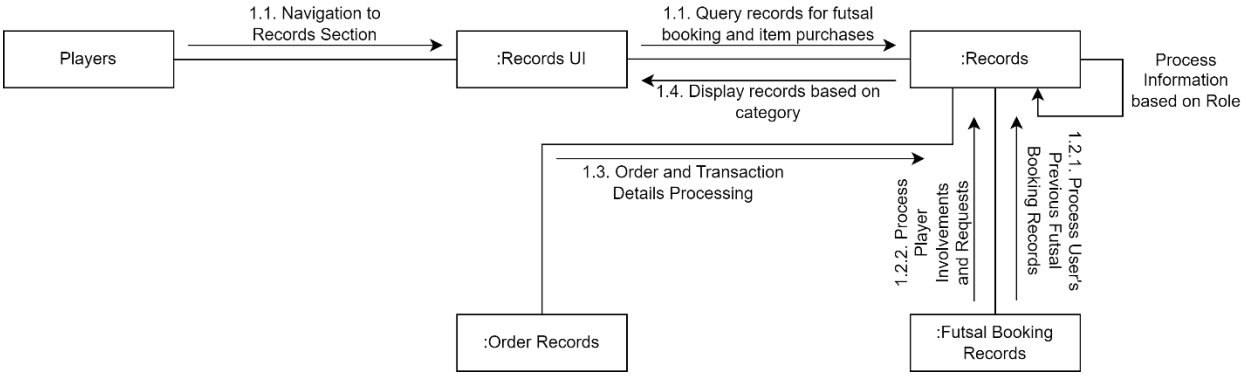


Figure 29: Collaboration Diagram: Order / Futsal Record Tracking

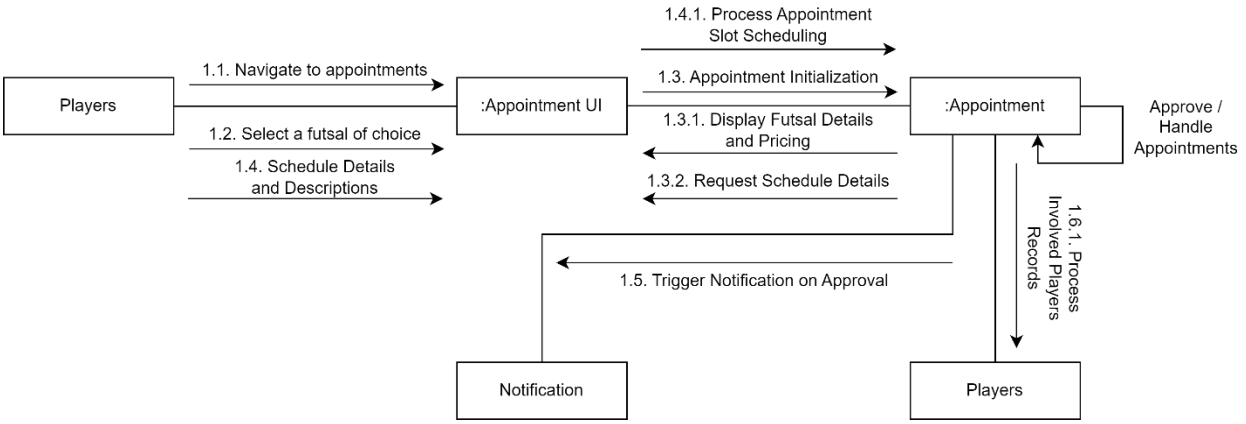


Figure 30: Collaboration Diagram: Appointment Initiation

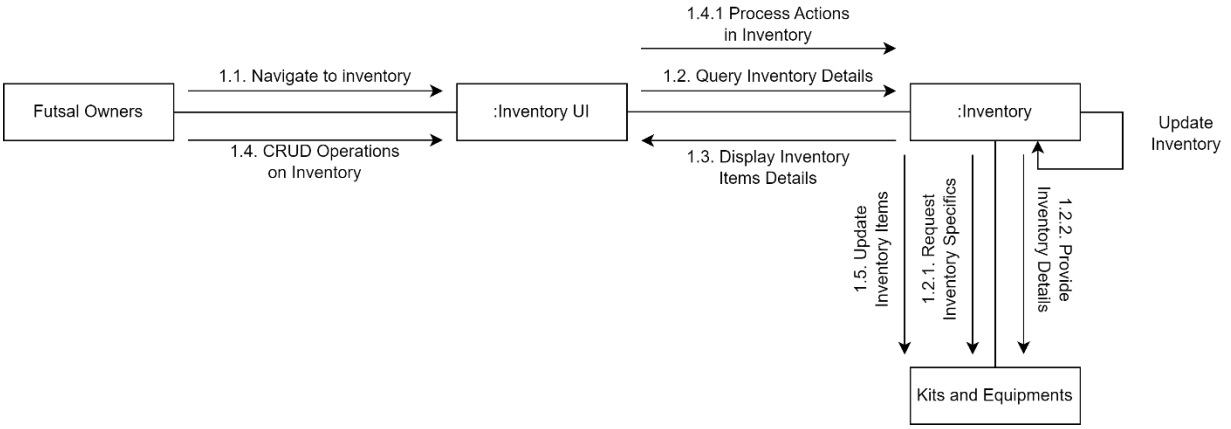


Figure 31: Collaboration Diagram: Inventory Management

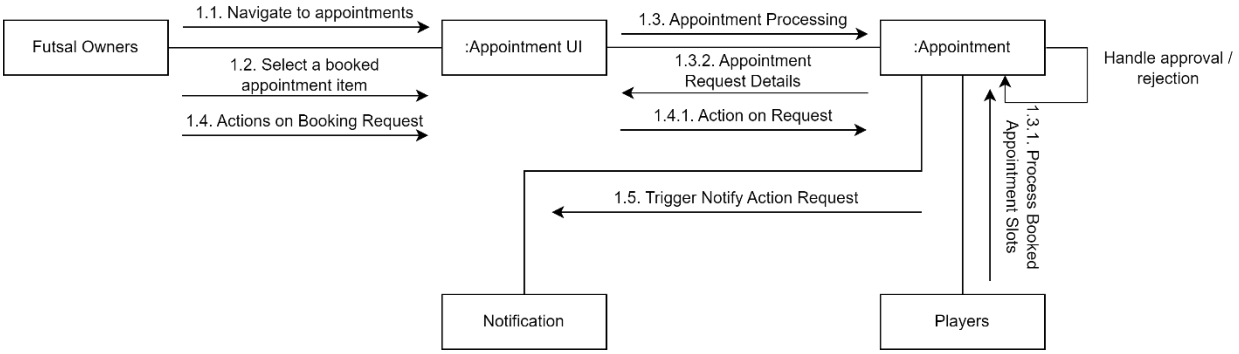


Figure 32: Collaboration Diagram: Appointment Finalization

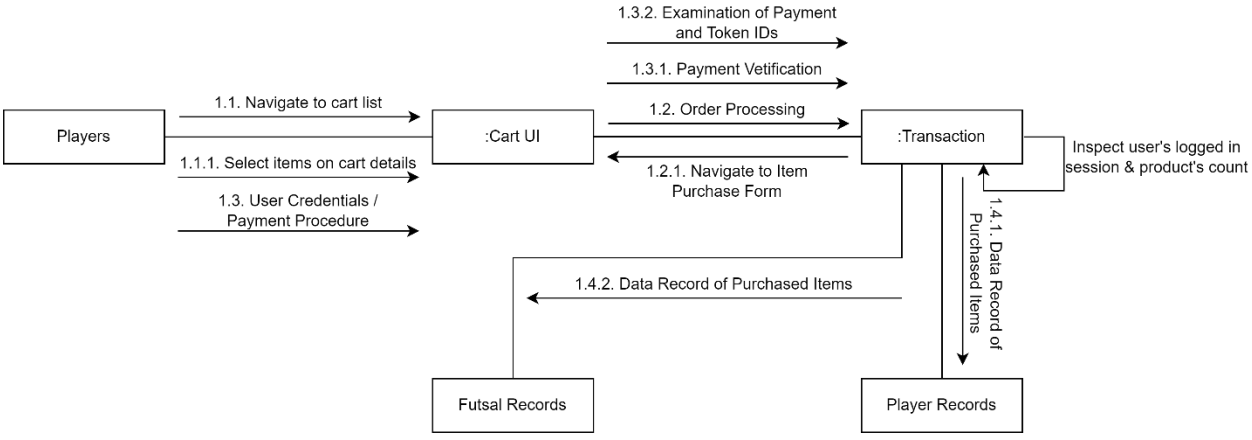


Figure 33: Collaboration Diagram: Order Processing

3.9. Block Diagram

A block diagram is a kind of schematic that uses basic geometric forms and lines to show the interconnections and functions of a system. It shows the overall structure of the system, with an emphasis on the connections and interactions between its main functional components. Software development is only one of several technical fields that greatly benefits from block diagrams. They are useful for project planning and stakeholder communication because they provide a visual abstraction that helps understand the system's structure and flow. Block diagrams are essential for tracing project aspects before technical work starts. They assist identifying critical components, dependencies, and data flows. Block diagrams let teams see and debate features holistically by providing a top-down view of the system. This helps with making informed decisions and making sure the project's architecture is in line with its requirements and objectives (Lucid Chart, 2023).

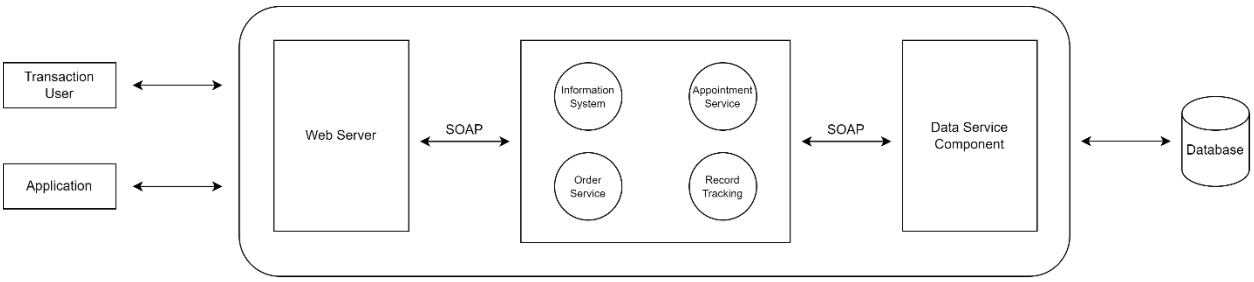


Figure 34: Block Diagram: Data Flow between Client and Server

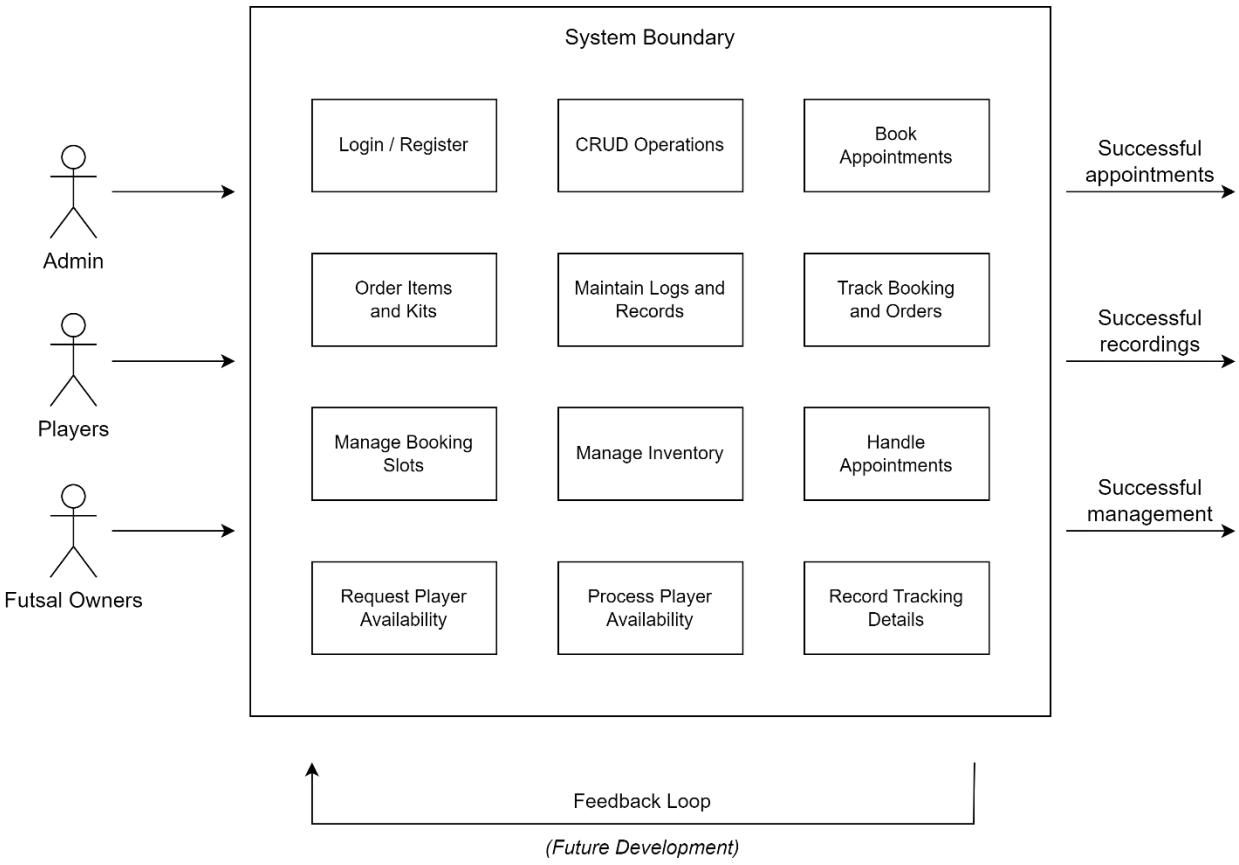


Figure 35: System Overview: Actors and Actions

3.10. Mind Map Diagram

A mind map is a graphical depiction of related thoughts, concepts, and connections organized around a main idea or subject. An overarching picture of interrelated parts is produced by its hierarchical organization of data using branches and nodes. When working on a project, mind maps are a great way to organize ideas, make plans, and generate new ideas. In the pre-technical development phase of a project, mind maps are useful for tracing the features, requirements, and functionalities of the feature, as well as any obstacles that may arise. Mind maps help in comprehending the full extent of a project by graphically linking related ideas and dividing intricate elements into more digestible parts. Essential for efficient project feature tracing, they enable team members to collaborate, boost creativity, and identify dependencies more easily. Before diving into technical development, ensure sure all stakeholders have a shared understanding of the project's goals and feature specifications with the help of mind maps, which are strong communication tools due to their simplicity and clarity (Lucid , 2023).

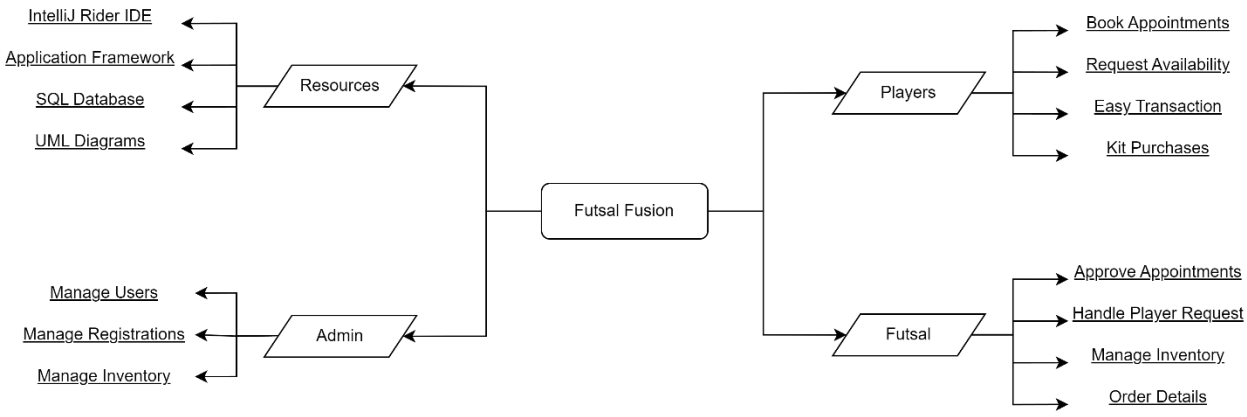


Figure 36: Mind Map Diagram: Resources and Functionalities

3.11. Wireframe Design

Wireframe design is a way to depict a website, app, or system visually. It outlines the fundamental structure and layout without going into detail about the design elements or aesthetics. Its principal goal is to depict the user interface and experience in an easy-to-understand manner. Importantly, wireframes are used to trace the features of a project prior to technical development. The ideation and refinement of the user interface and feature interactions are built upon this. The structure and functionality of a wireframe aids teams in early development in identifying critical components, navigation paths, and user interactions. Stakeholders can validate and refine design decisions prior to investing in technical development, which is critical for feature tracing. In order to ensure that the planned features are both easy to use and seamlessly incorporated into the system as a whole, wireframes help designers, developers, and other stakeholders communicate with one another (Hannah, 2023).

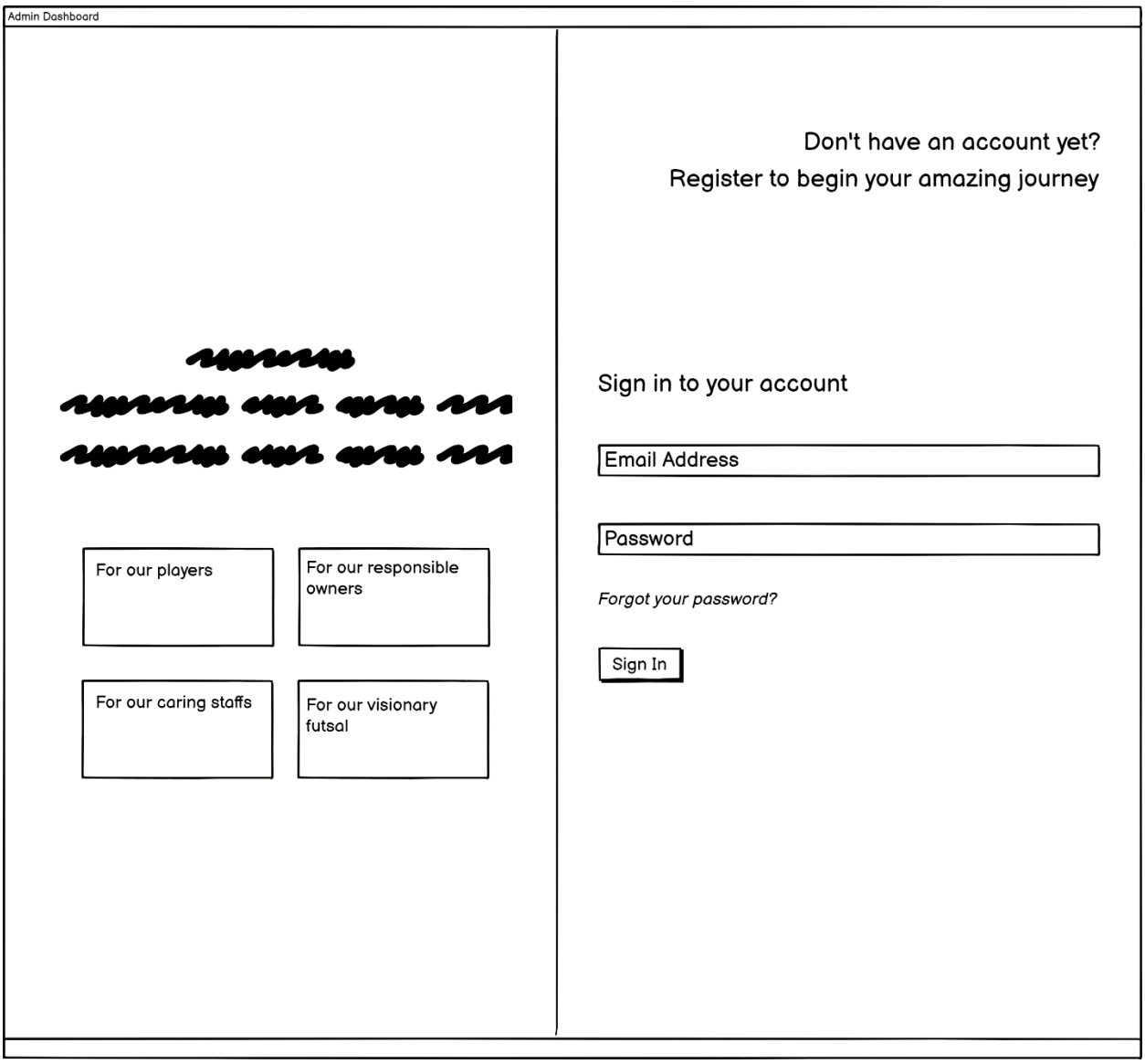


Figure 37: Wireframes: Login

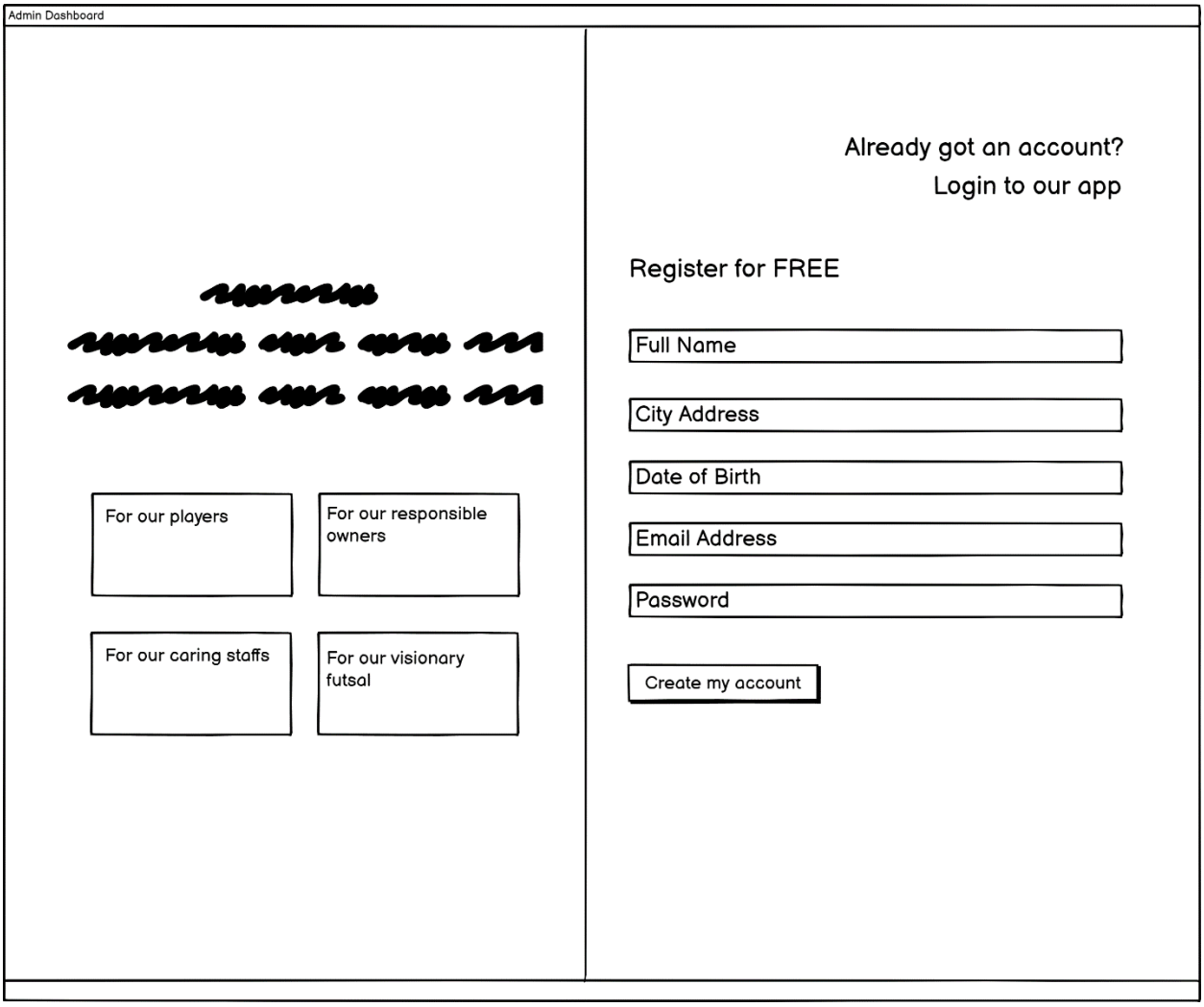


Figure 38: Wireframes: Registration

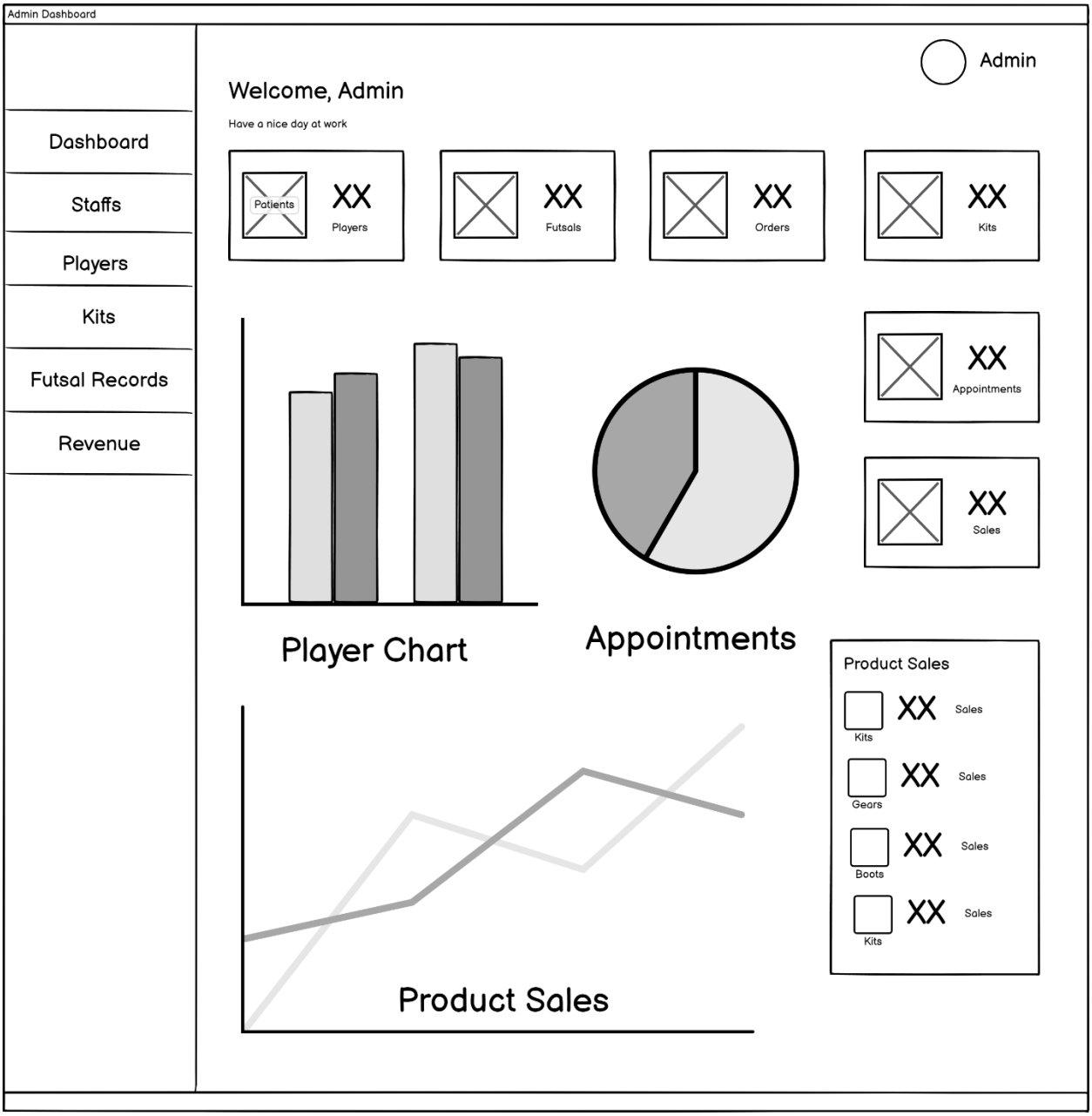


Figure 39: Wireframes: Admin Dashboard

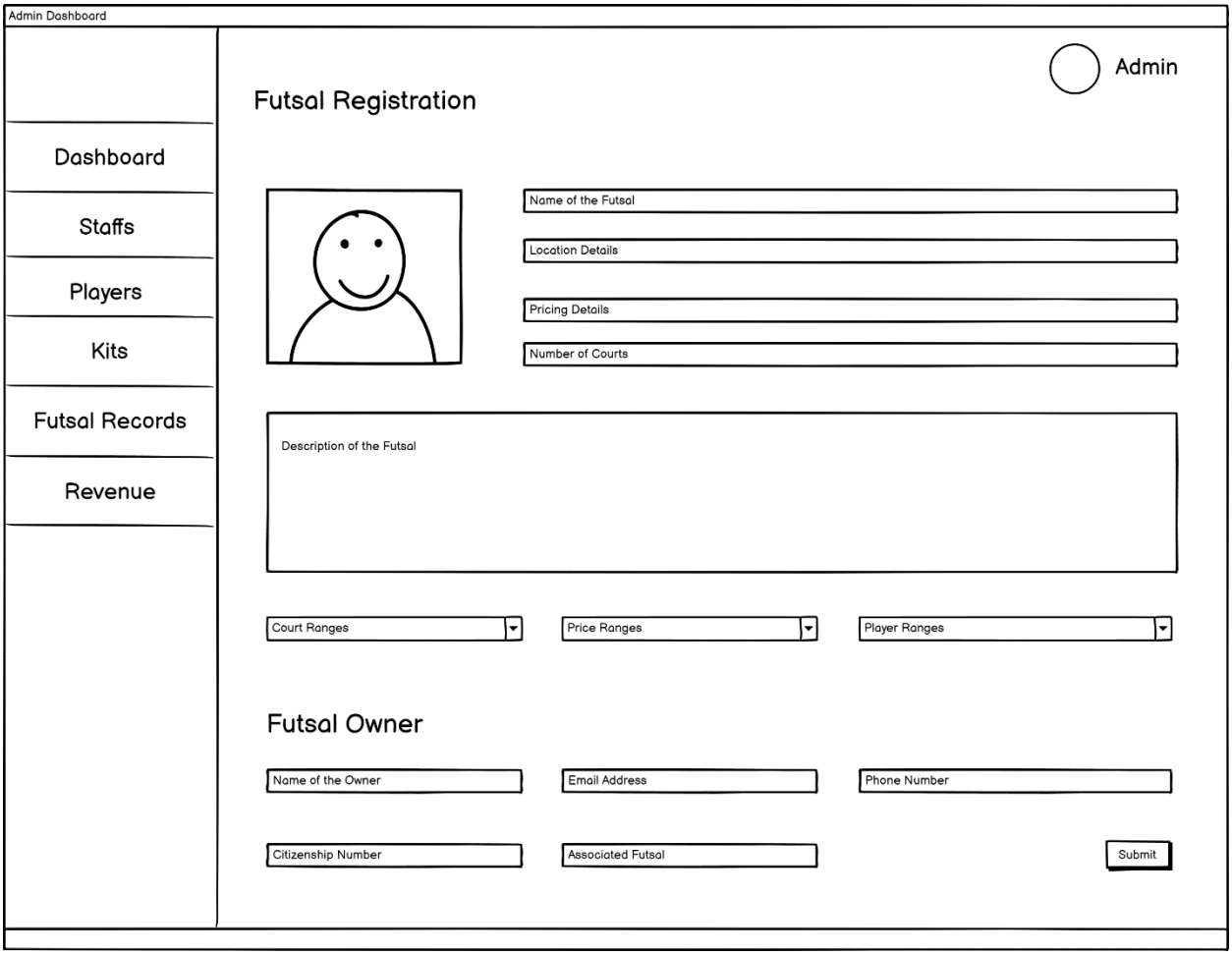


Figure 40: Wireframes: Futsal Registration

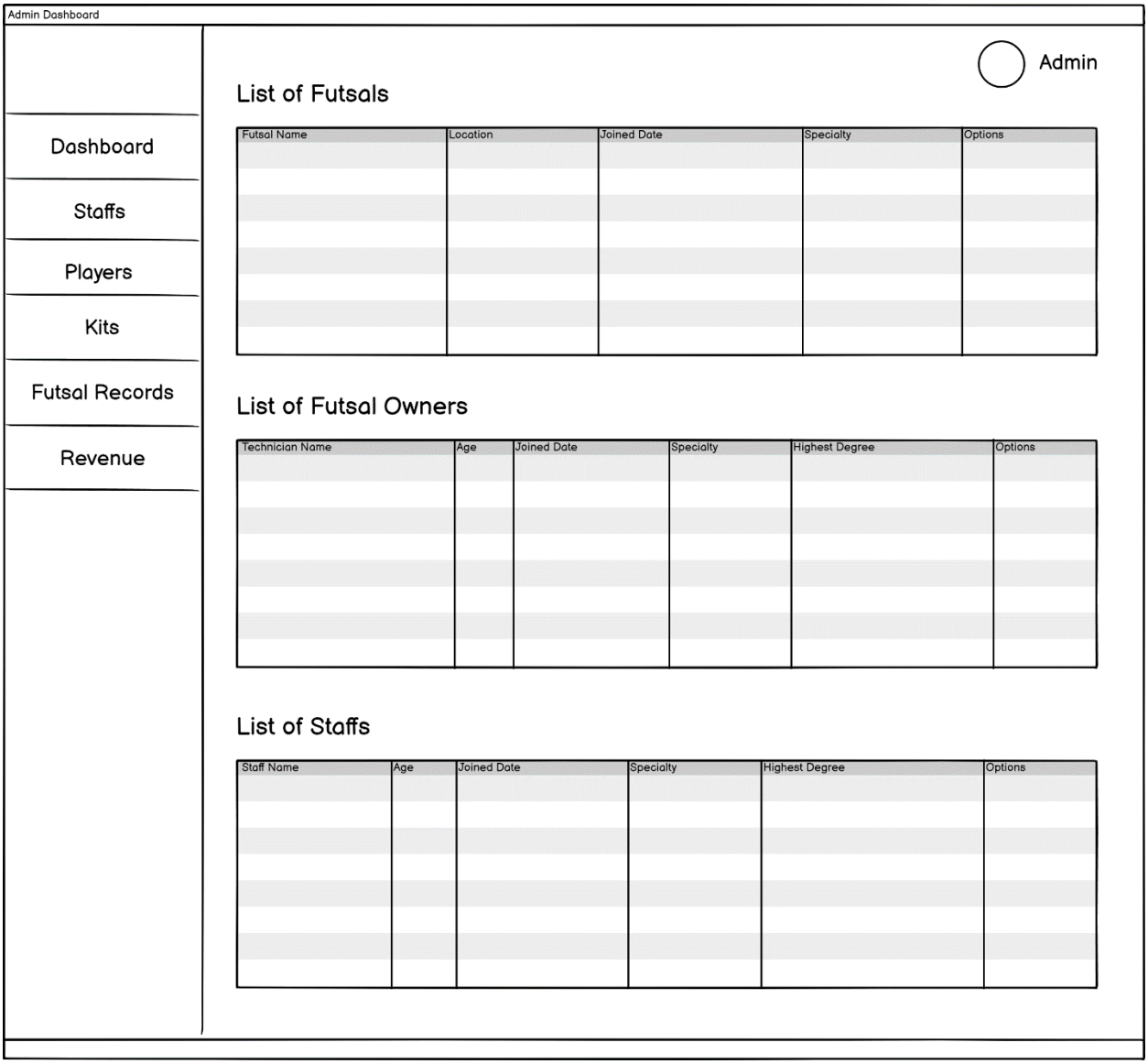


Figure 41: Wireframes: Futsal View

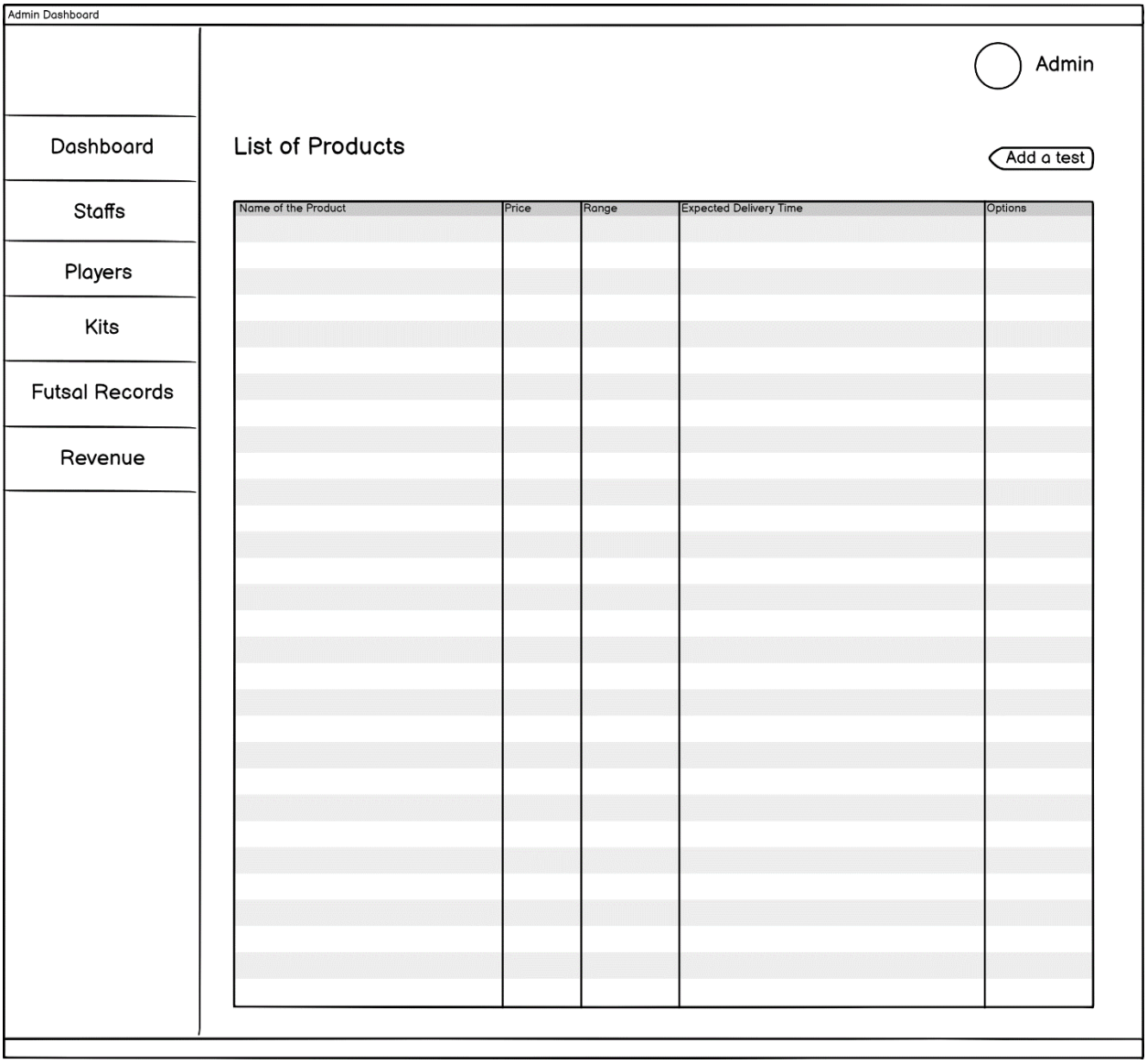


Figure 42: Wireframes: Products View

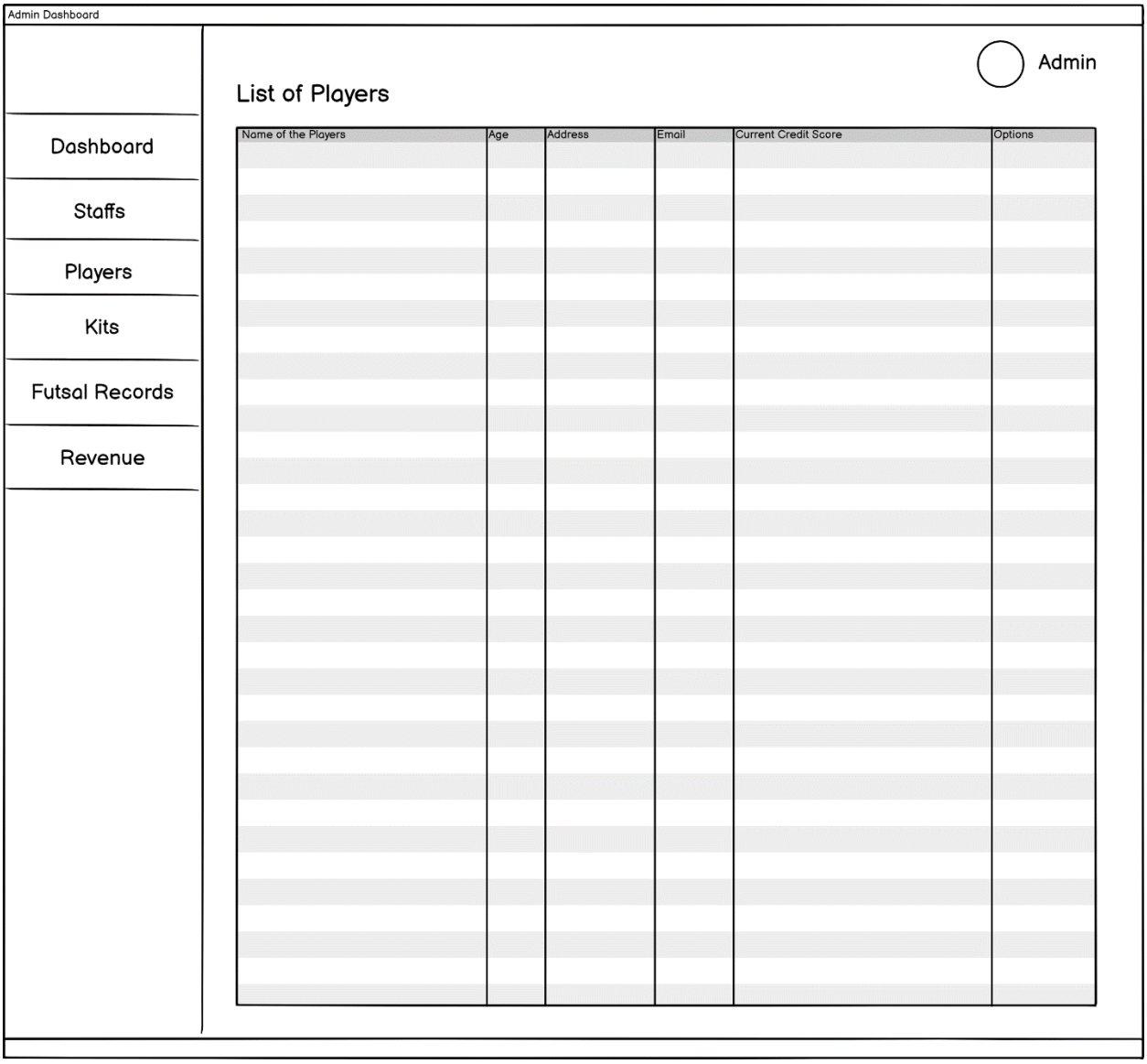


Figure 43: Wireframes: Players View

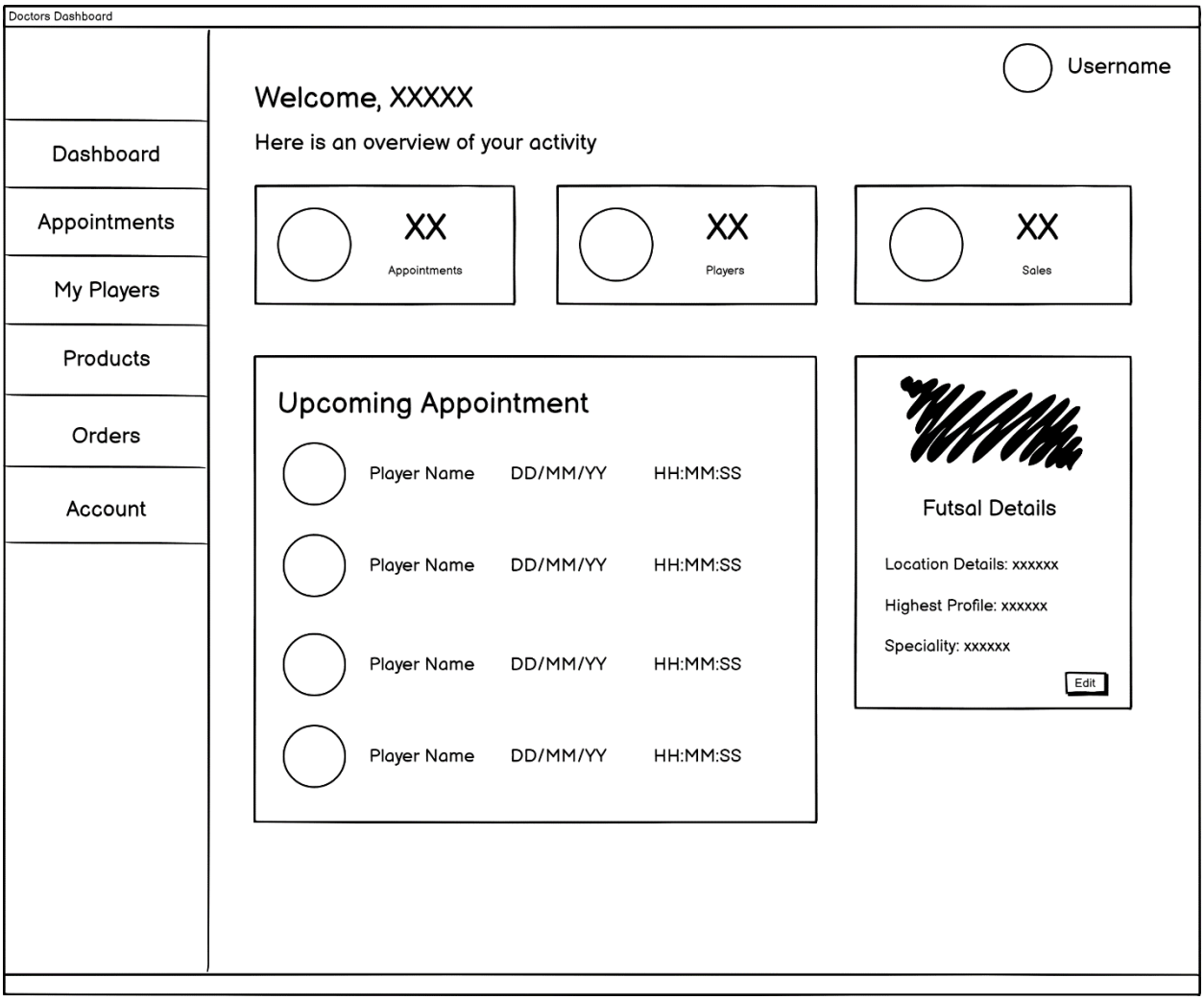


Figure 44: Wireframes: Futsal Owner Dashboard

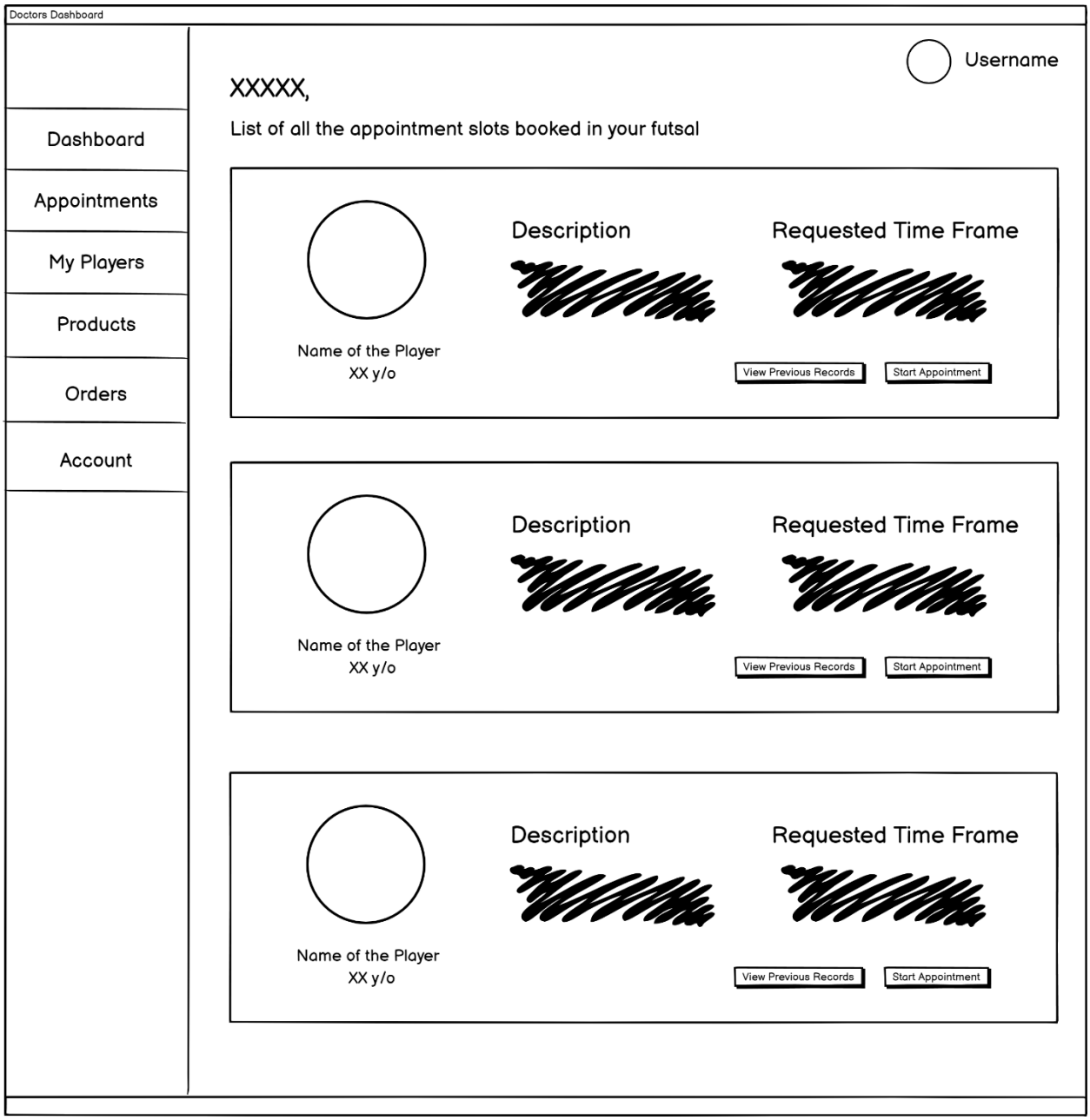


Figure 45: Wireframes: Futsal Appointment Processing

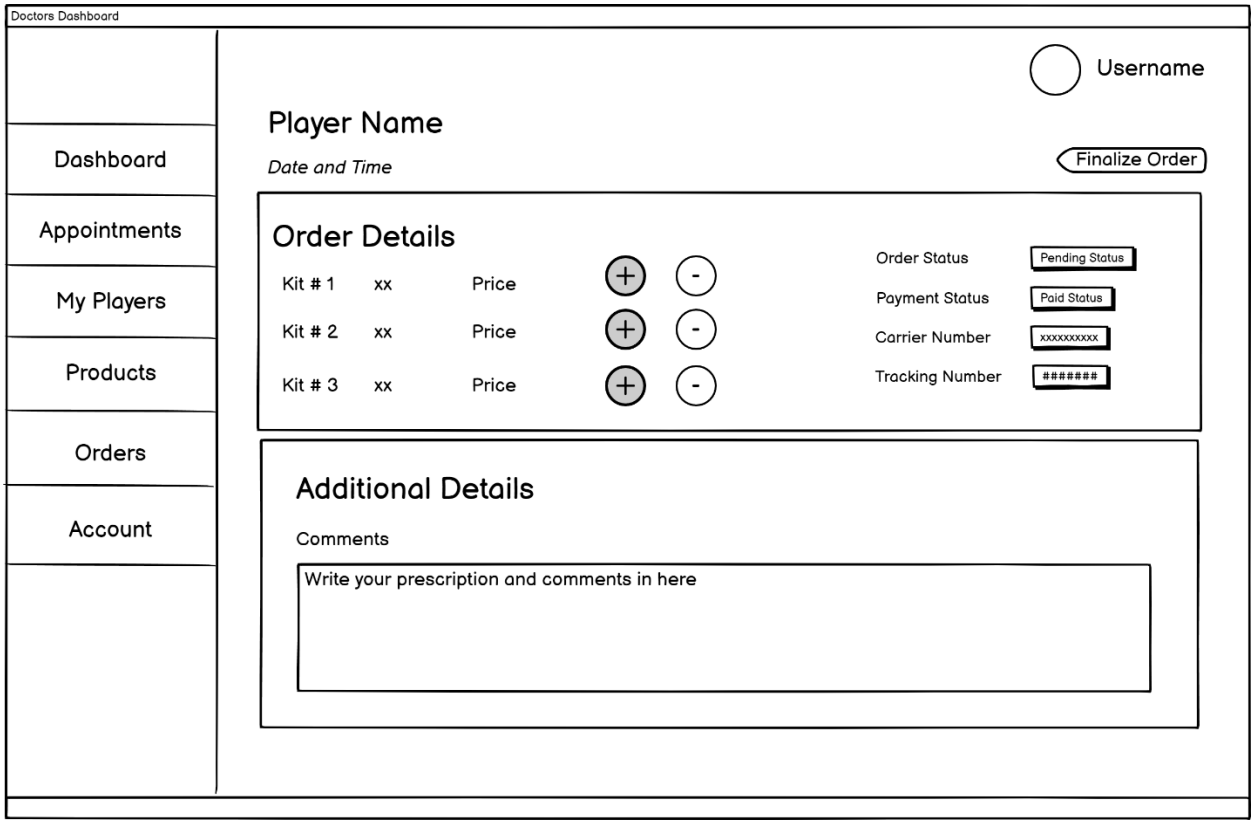


Figure 46: Wireframes: Futsal Order Processing

Figure 47: Wireframes: Players Dashboard

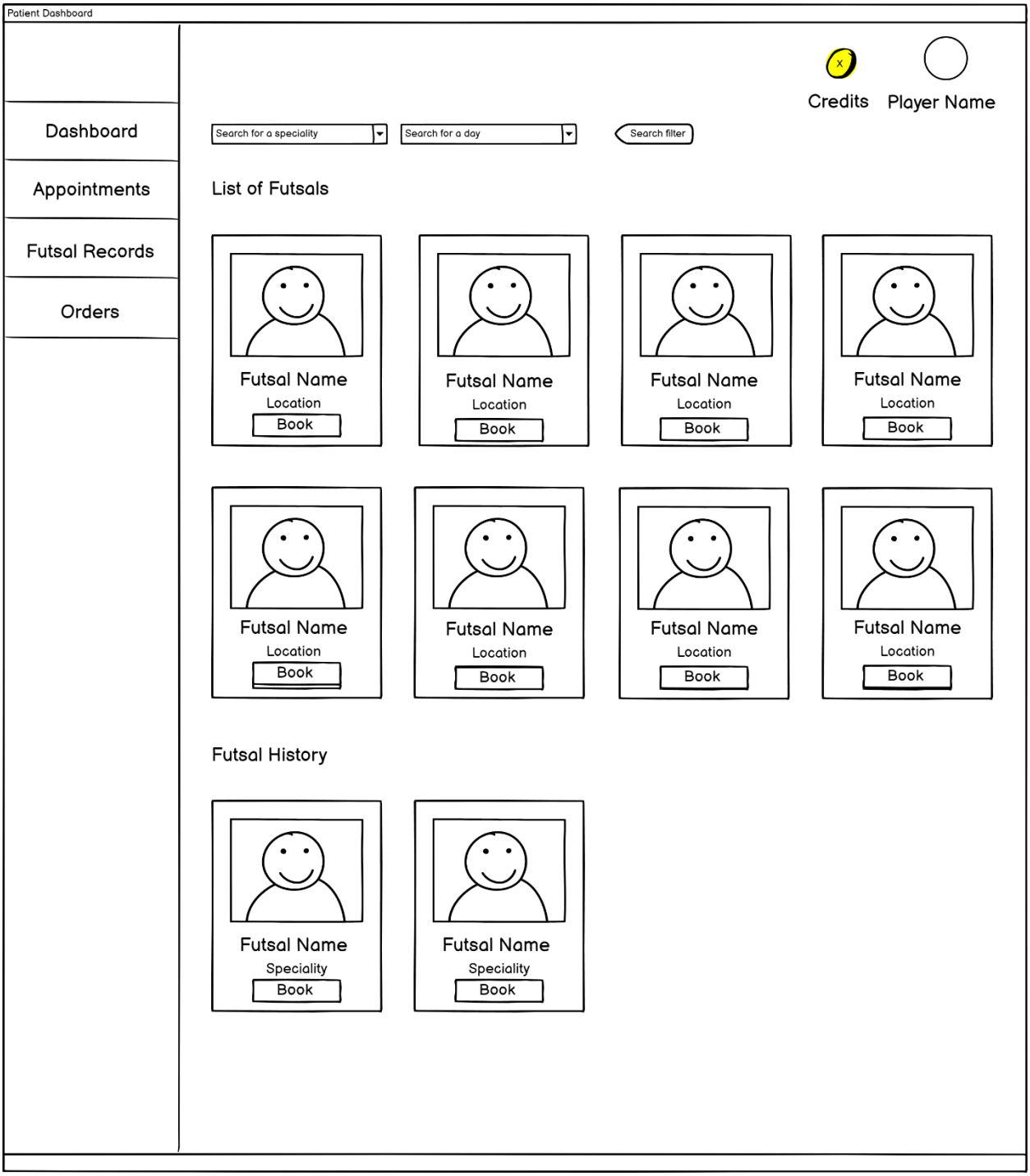


Figure 48: Wireframes: Players' Futsal Booking View

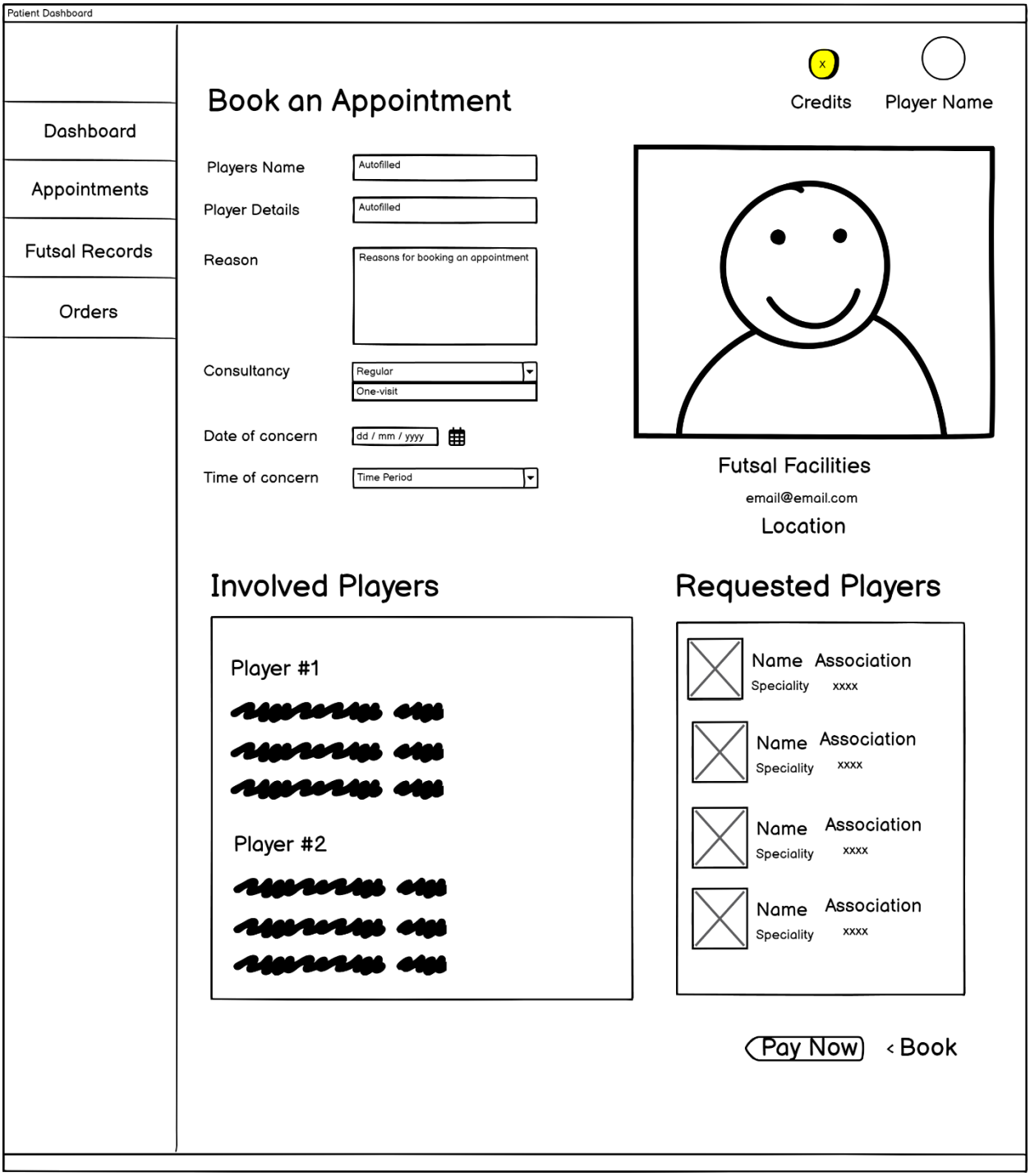


Figure 49: Wireframes: Player's Futsal Initialization

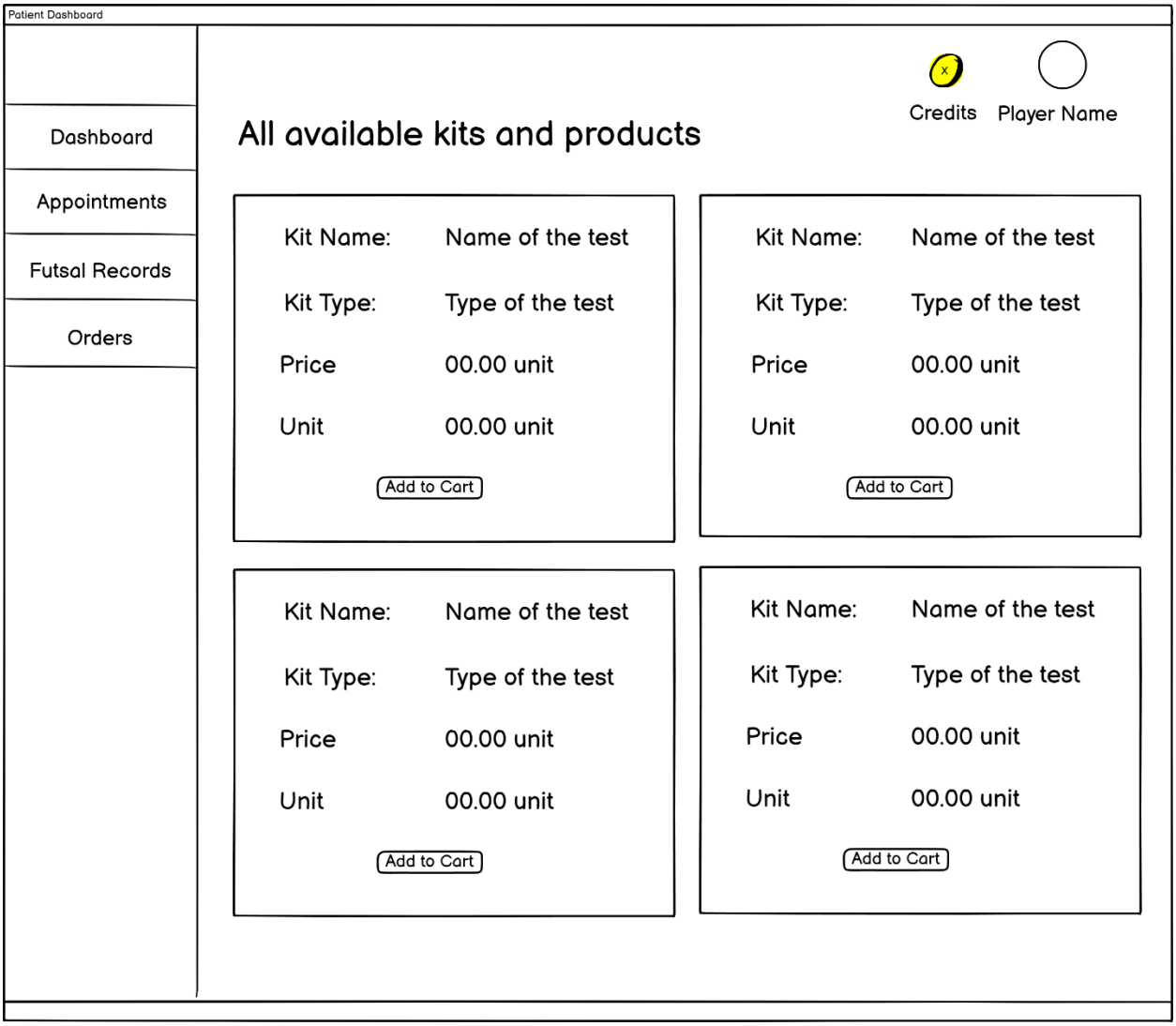


Figure 50: Wireframes: List of Products

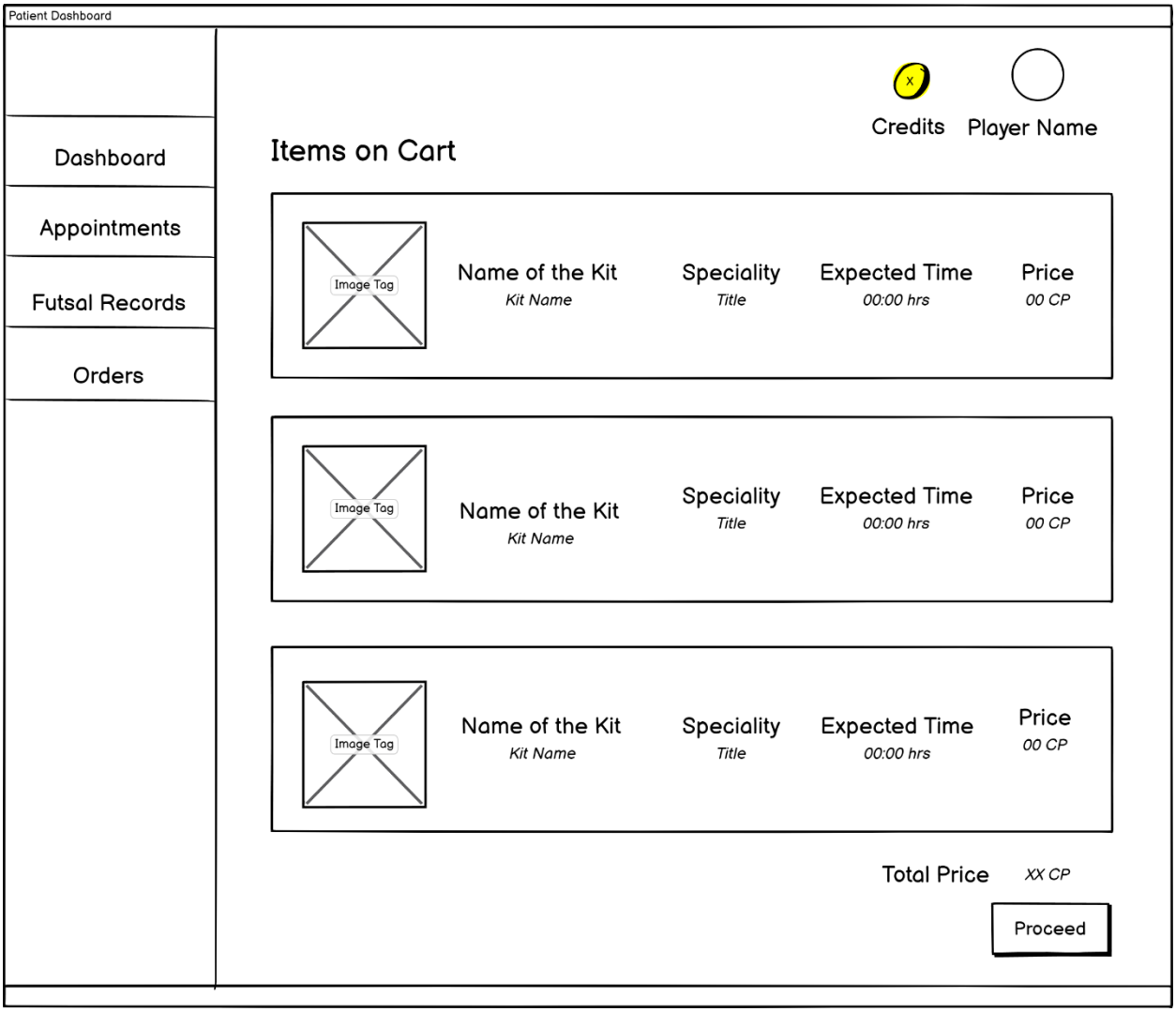


Figure 51: Wireframes: Orders on Cart

3.12. Entity Relationship Diagram

One way to graphically depict the connections between database entities is through the use of an Entity-Relationship Diagram (ERD) which illustrate the interplay between various database components by modeling their structures using entities, characteristics, and relationships. It is vital to use ERD design when tracing a project's features prior to technical development. As a result, the database architecture may be better tailored to meet the needs of the features under development, and the data model can be better defined. ERDs also provide a holistic perspective of the data flow in the system by identifying important entities, their attributes, and the links between them.

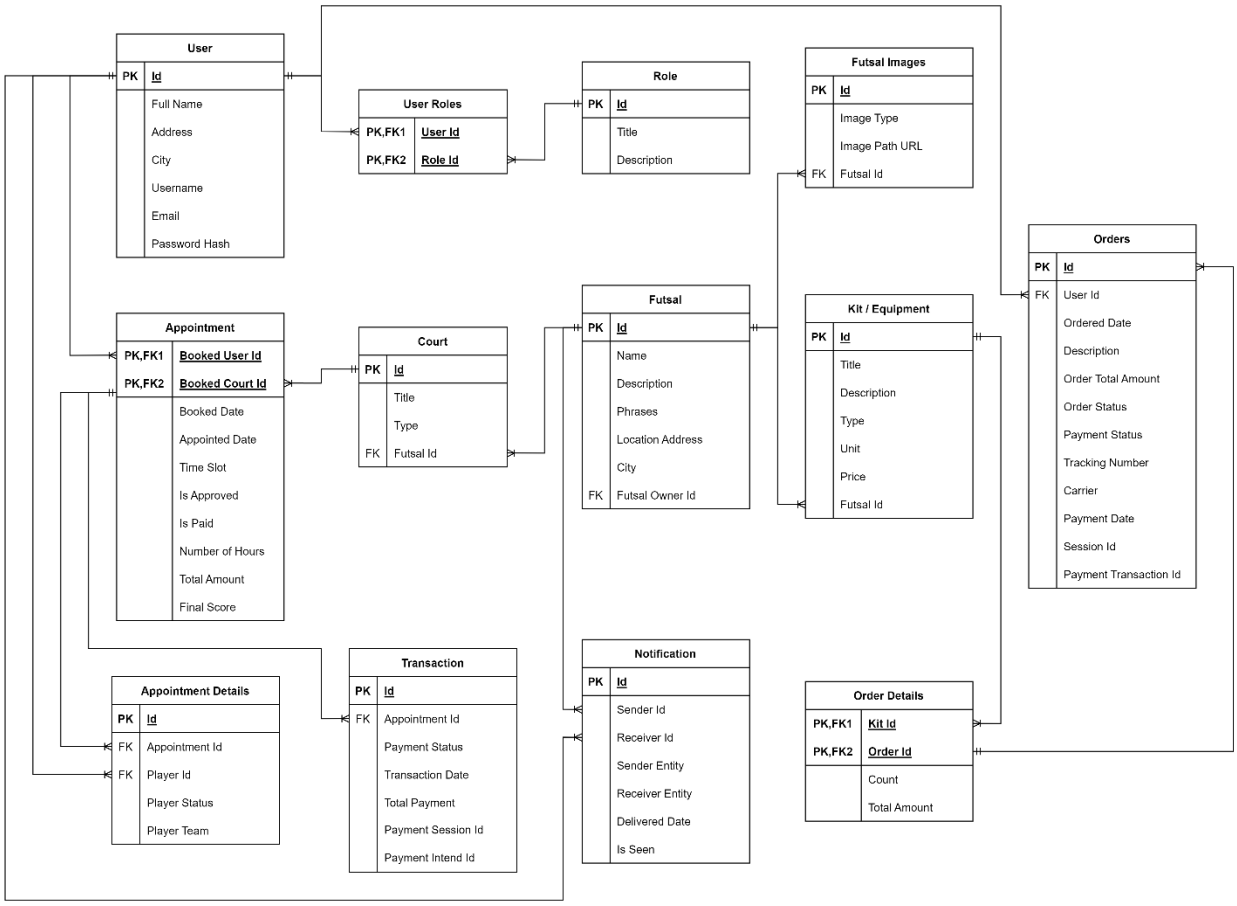


Figure 52: Entity Relationship Diagram: Initial Phase

The data structure may be better understood by stakeholders and developers with this level of clarity, which in turn allows for more efficient and accurate technical development. A more robust and well-designed system can be achieved by addressing possible difficulties with data integrity, normalization, and relationships early in the project lifecycle using ERD design, which establishes a solid foundation (Lucid Chart, 2023).

A clear cut of all the visualized entities and their respective details, relationship mapping and meta data on the data dictionary has been defined in the appendix report, which will define a clear vision on the respective entities. Link to Appendix.

3.13. Class Diagram

To visually depict a system's classes, characteristics, methods, and relationships, class diagram design is an important part of object-oriented modeling in software engineering. The class diagram is an architectural design for the software that shows how different parts work together. Class diagram design is an essential tool for tracing a project's features prior to actual technical development. To aid stakeholders and developers in comprehending the class hierarchy and their relationships, it gives a bird's-eye view of the system's architecture. Class diagrams aid in the development of technically sound projects by drawing attention to critical classes, properties, and methods at an early stage. Facilitating collaborative discussions and guaranteeing that the project's features are well-aligned with the overall software architecture, they aid in feature tracing by laying a clear basis for how various components will interact and function. Class diagram design allows for the proactive identification and resolution of any design problems, resulting in a codebase that is more robust and easier to maintain (Lucid Chart, 2023). A class diagram highlighting all the major functionalities has been defined in the following diagram.

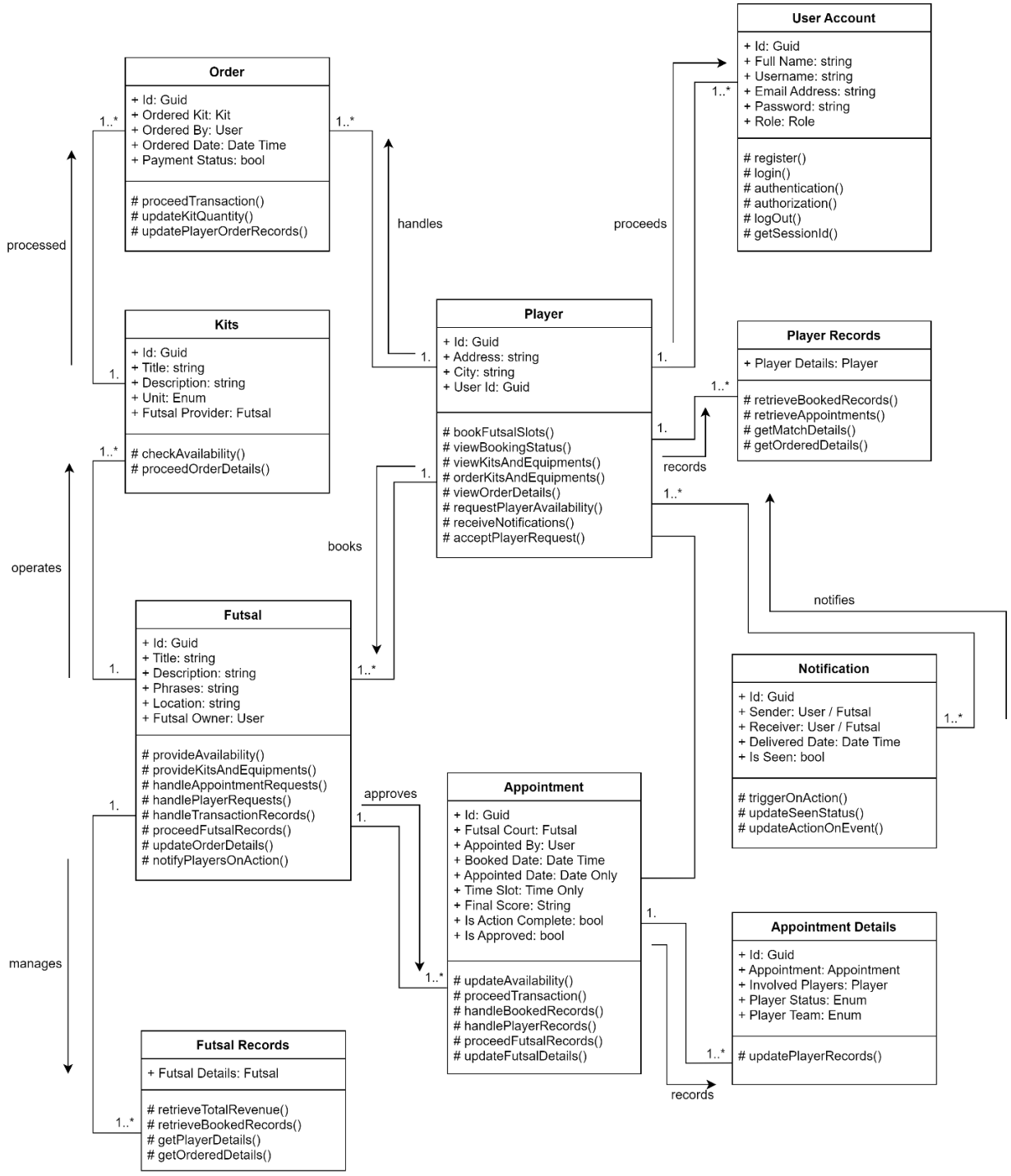


Figure 53: Class Diagram: Entity and Services

3.14. Project Management: Trello

Trello, a project management tool, provides an easier and more visually appealing way to organize projects and tasks. The user-friendly drag-and-drop interface makes it simple to assign tasks, establish due dates, and attach necessary documents. Project management becomes more efficient and transparent with Trello's ability to connect with other applications and platforms. It becomes a central center for project communication and progress tracking. The user story-derived product backlog is organized using Trello, a board-based project management tool. The activity specifications are based on the cards that represent each user story or feature capability.

Step one after starting a Trello board is to establish the different identifiers that will be used throughout the scrum development lifecycle. These identifiers include product backlog, sprint backlog (which contains the tasks for the current sprint), tasks for the next sprint, and tasks that are in progress. It is not possible to mark a job or user story as complete until it has been tested and authorized by both quality assurance and testing.

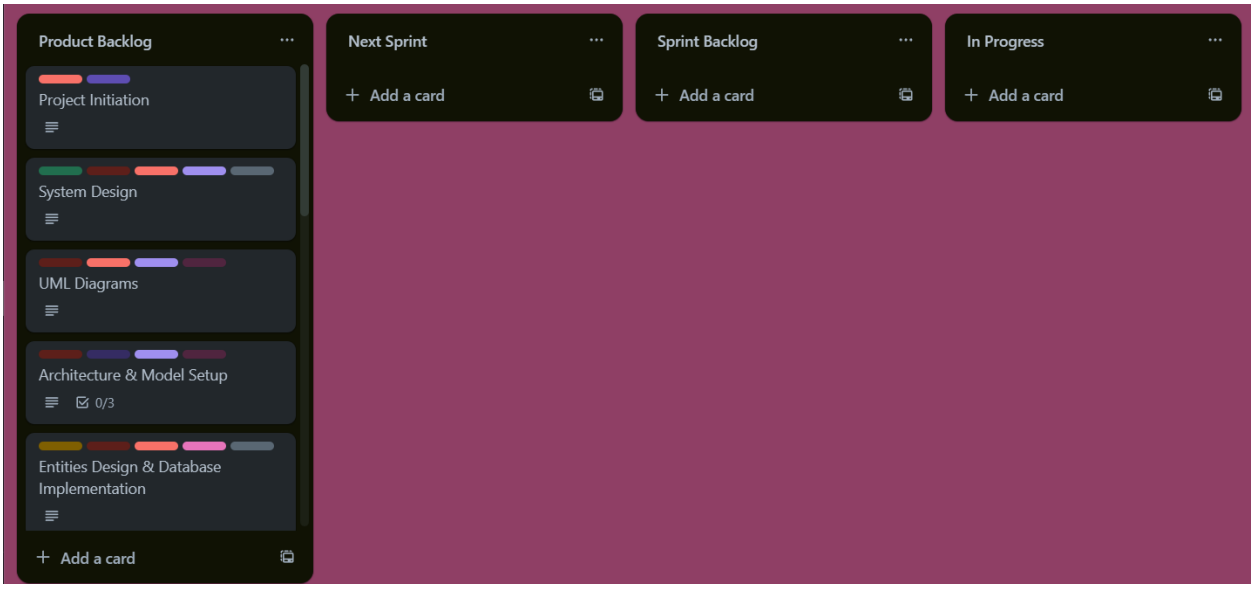


Figure 54: Trello Board: Cards (1)

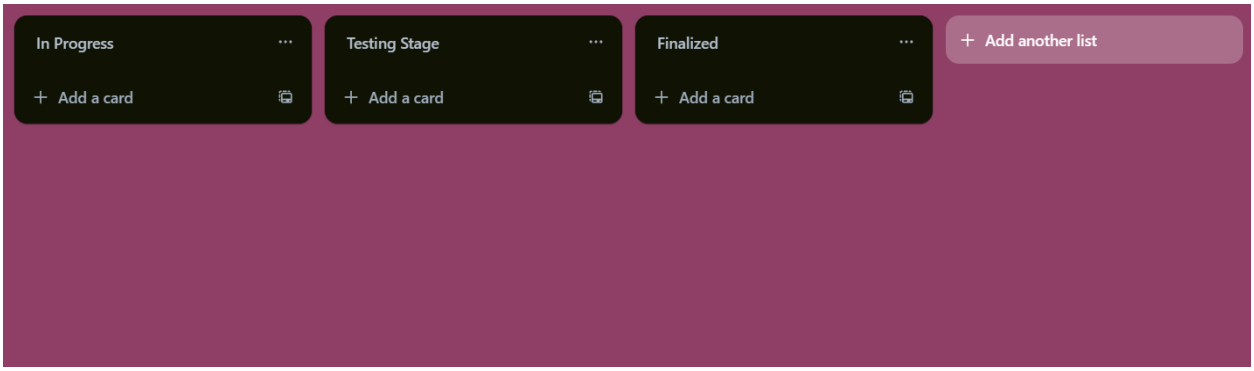


Figure 55: Trello Board: Cards (2)

Creating labels to categorize the various kinds of user stories or tasks to be completed was the first step in outlining a product backlog. For this reason, each type of task was given its own label when the corresponding card depicting a user story was produced. Here, then, are descriptions of the various labels' primary classifications:

- **Type:** Refers to the type of activity carried out in the sprint.
- **Test Case:** Refers to the activity of test case to be performed on the story.
- **Dependency:** Refers to the approximate dependency with other stories.
- **Estimate:** Refers to the time period it may account for its completion.
- **Priority:** Refers to the preference of the task based on its concern.

After all sets of labels have been created, it is now necessary to initiate the list of cards as user stories to be accomplished. As a result, in the following part of the product backlog, initialization of each group of user stories or actions is done and so on labels are assigned to them. A total of 19 cards were delivered as a result of the sprint development cycle which included pretty much every task to be completed.

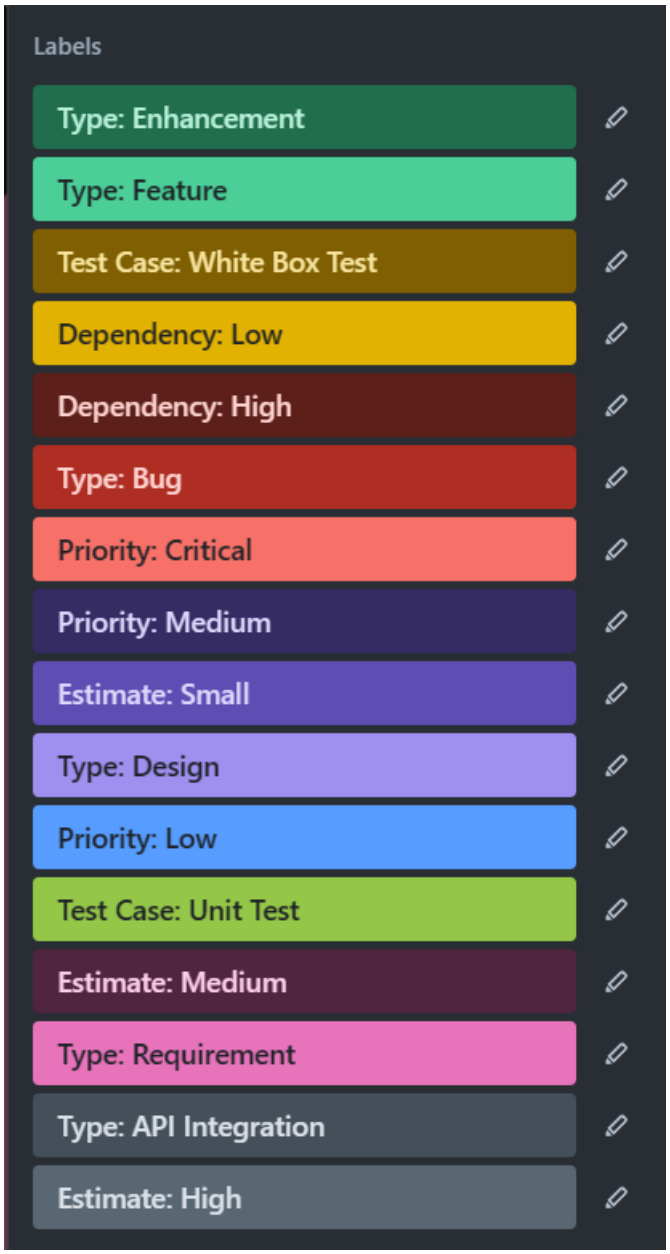


Figure 56: Trello: Assigned Labels

As adding a card to define the activity or user story wasn't enough to describe what each of the task was to be carried out, it was then necessary to add a brief description about each of them and then mend a checklist so as to find the deliverables and track progress calculated on each of them.

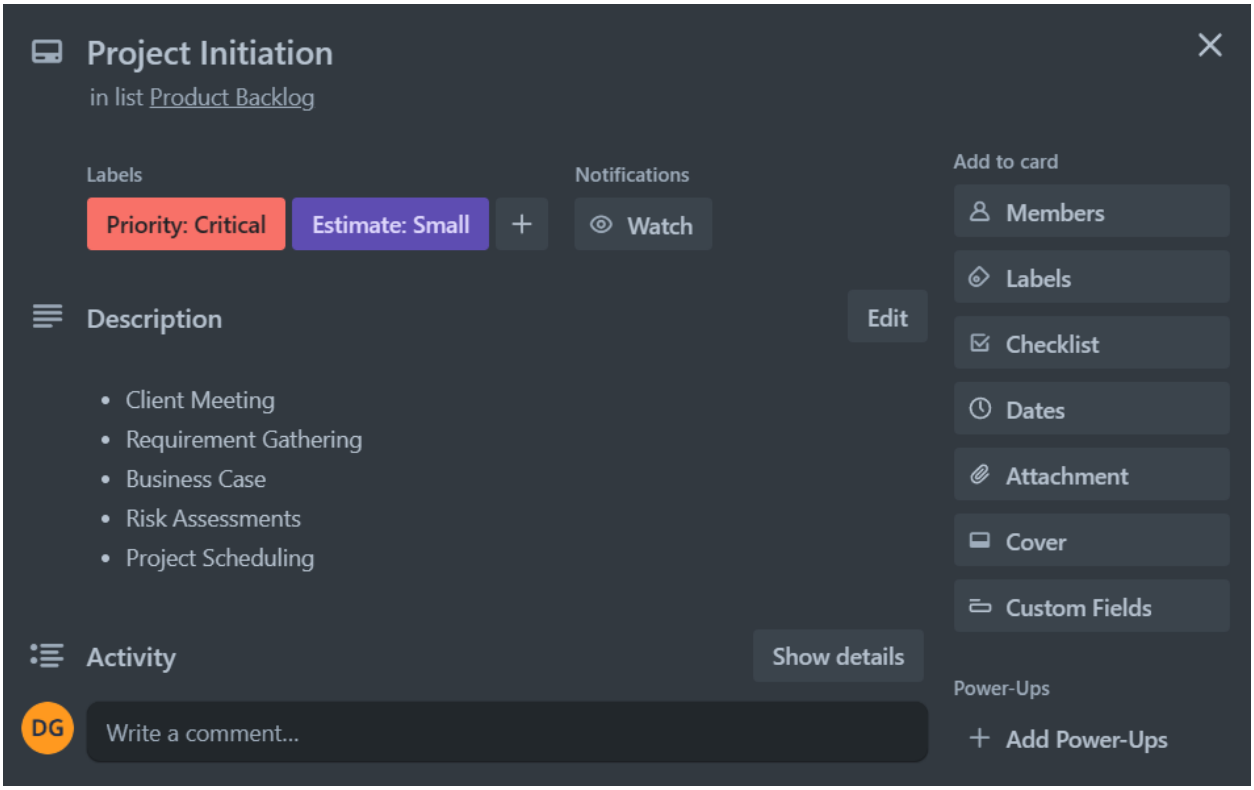


Figure 57: Trello Ticket: Project Initiation

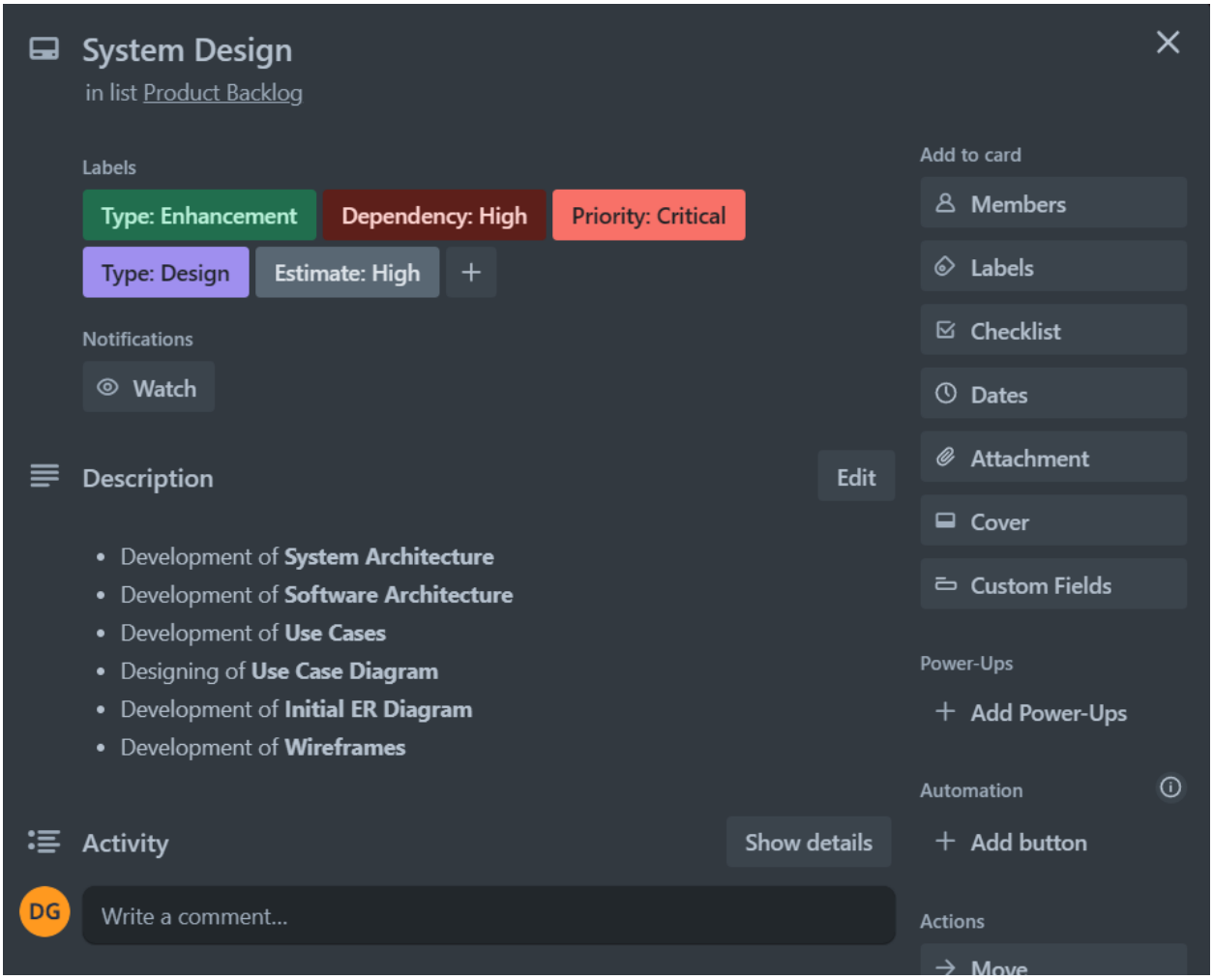


Figure 58: Trello Ticket: System Design

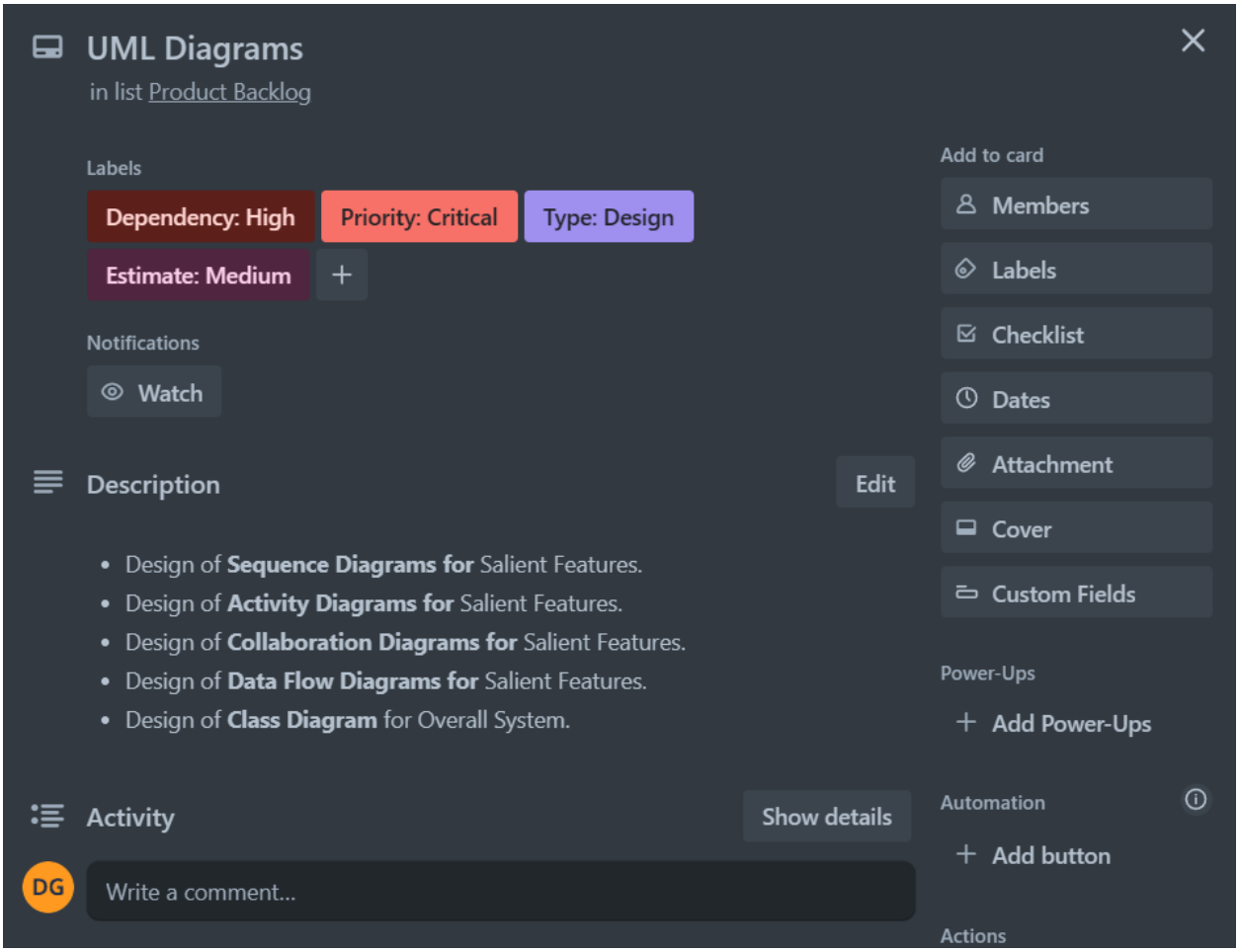


Figure 59: Trello Ticket: UML Diagrams

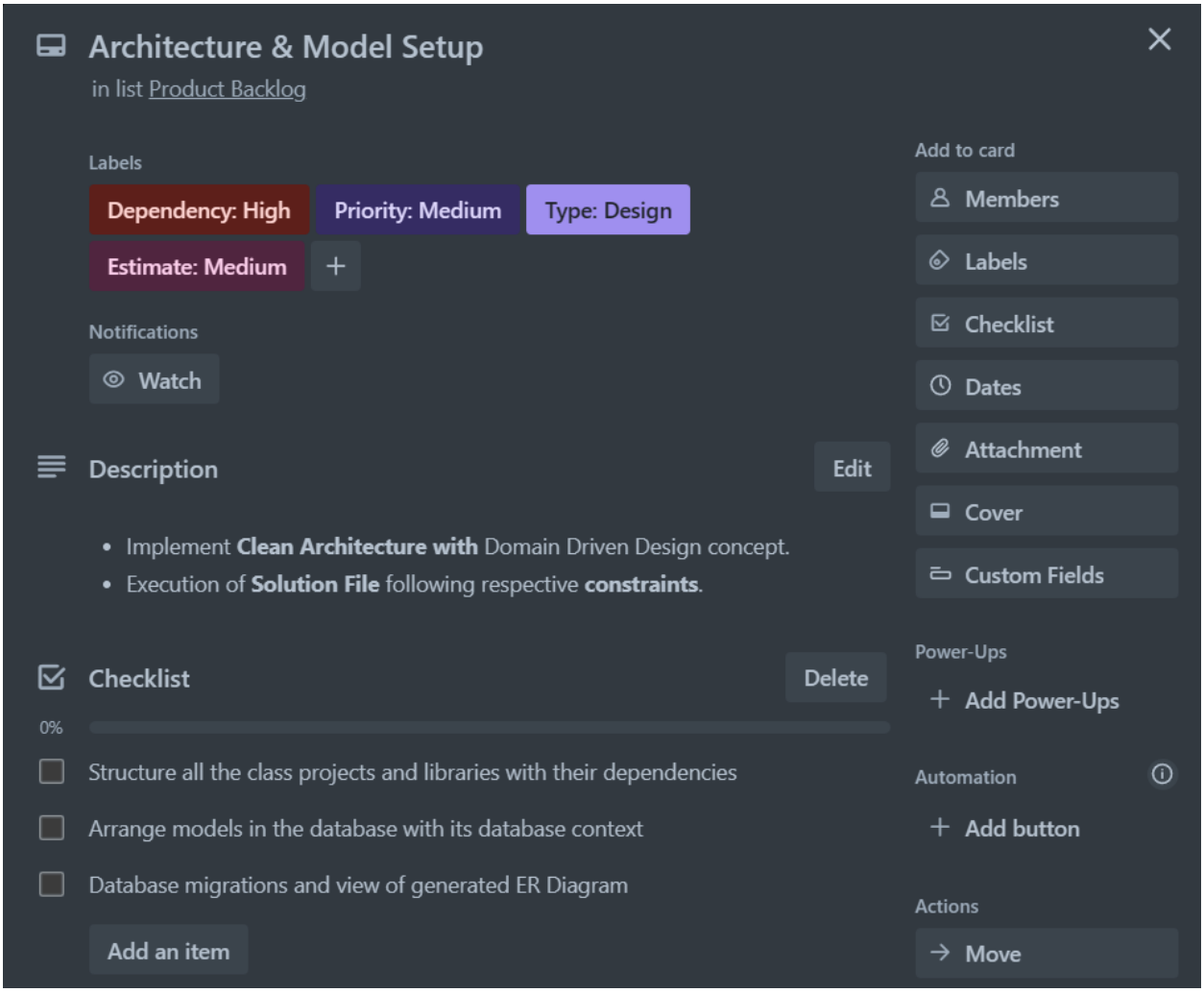


Figure 60: Trello Ticket: Architecture & Model Setup

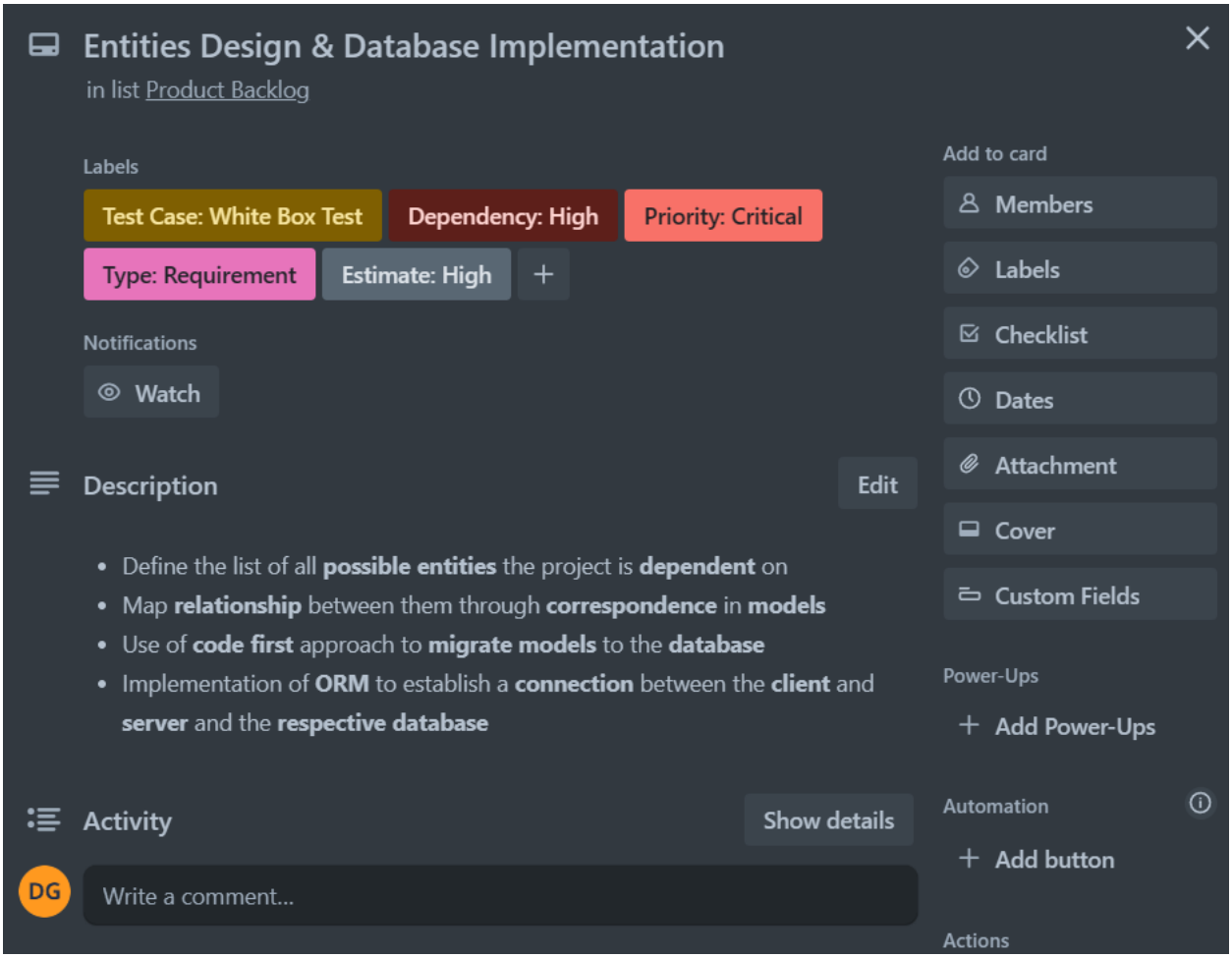


Figure 61: Trello Ticket: Entities Design & Database Implementation

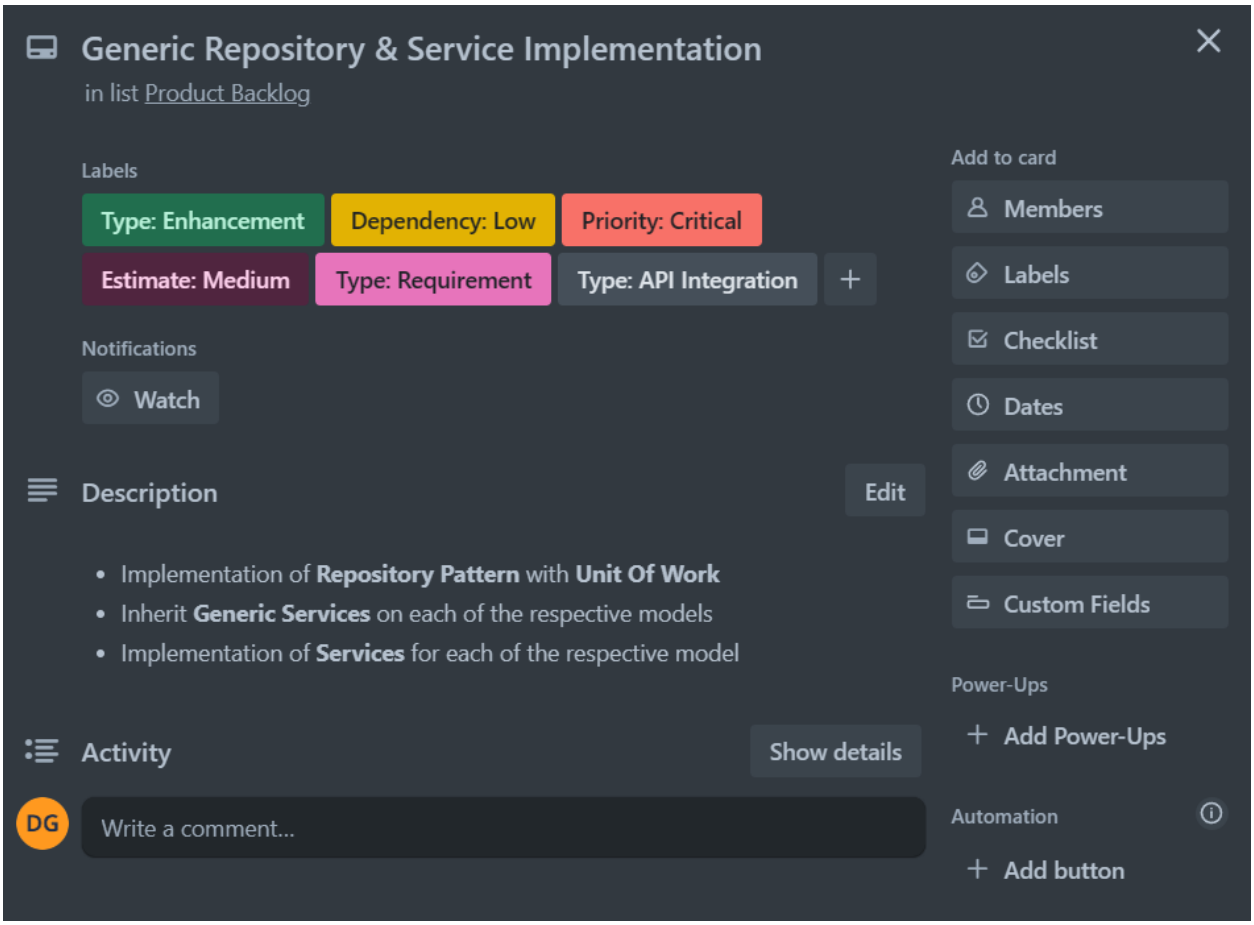


Figure 62: Trello Ticket: Generic Repository & Service Implementation

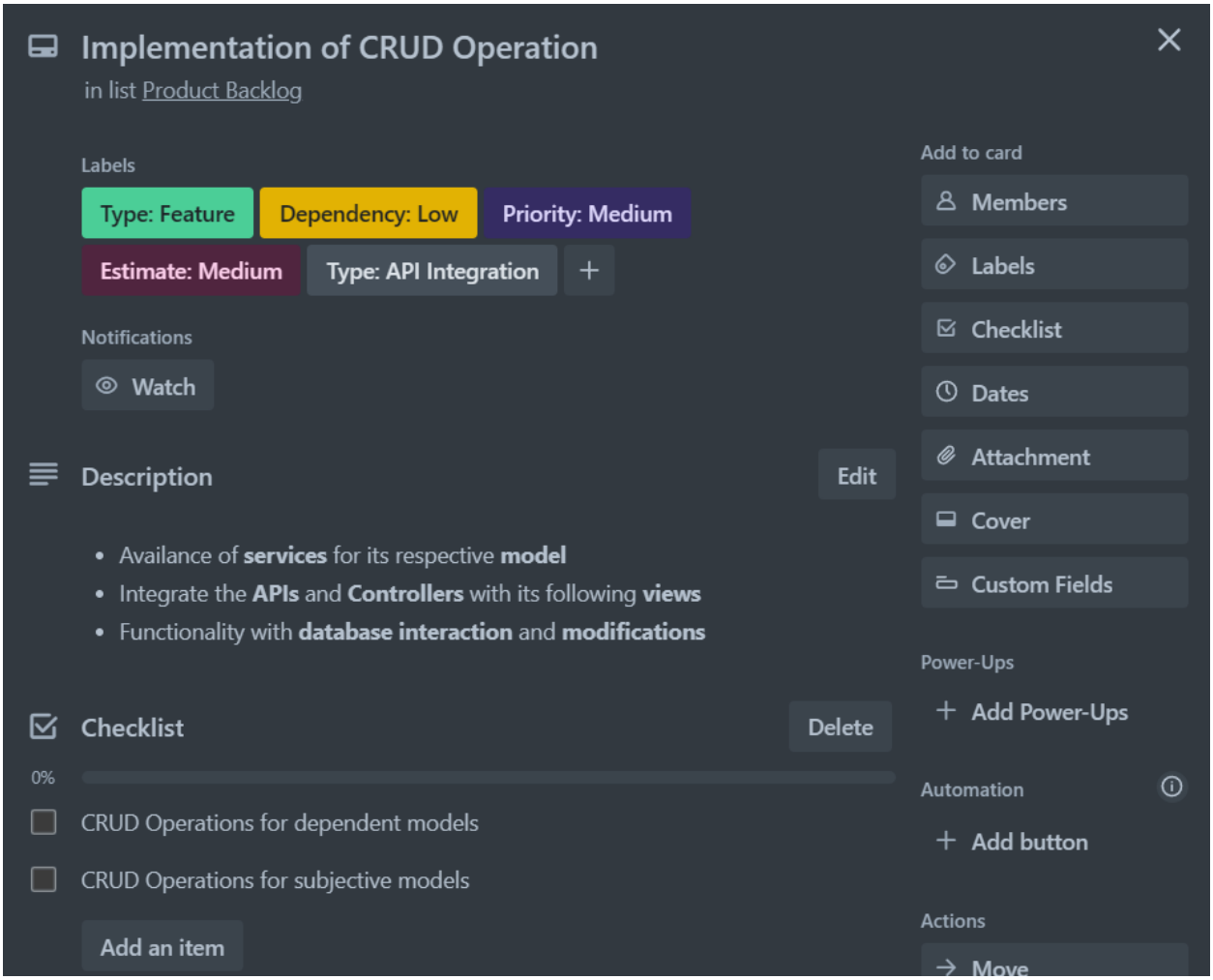


Figure 63: Trello Ticket: Implementation of CRUD Operations

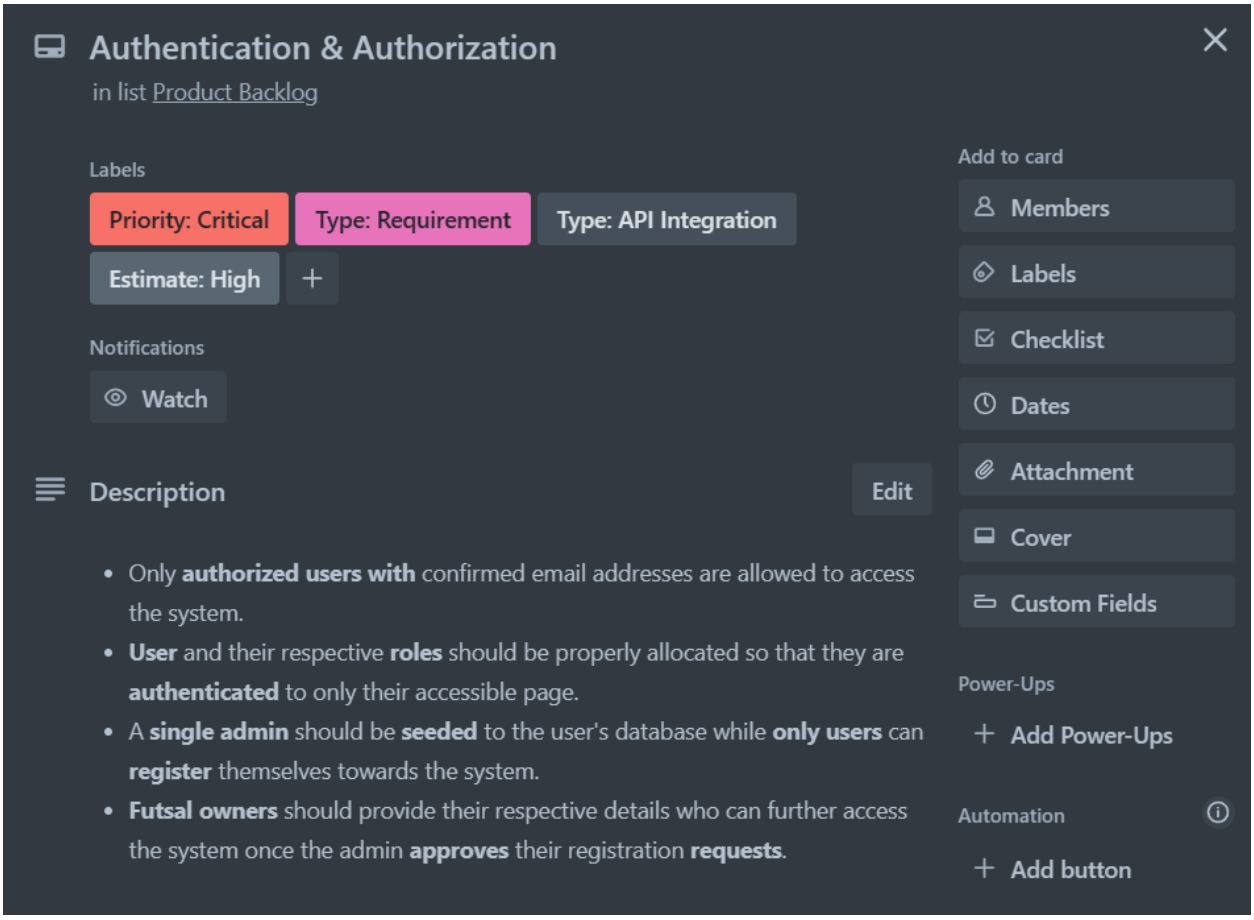


Figure 64: Trello Ticket: Authentication & Authorization (1)

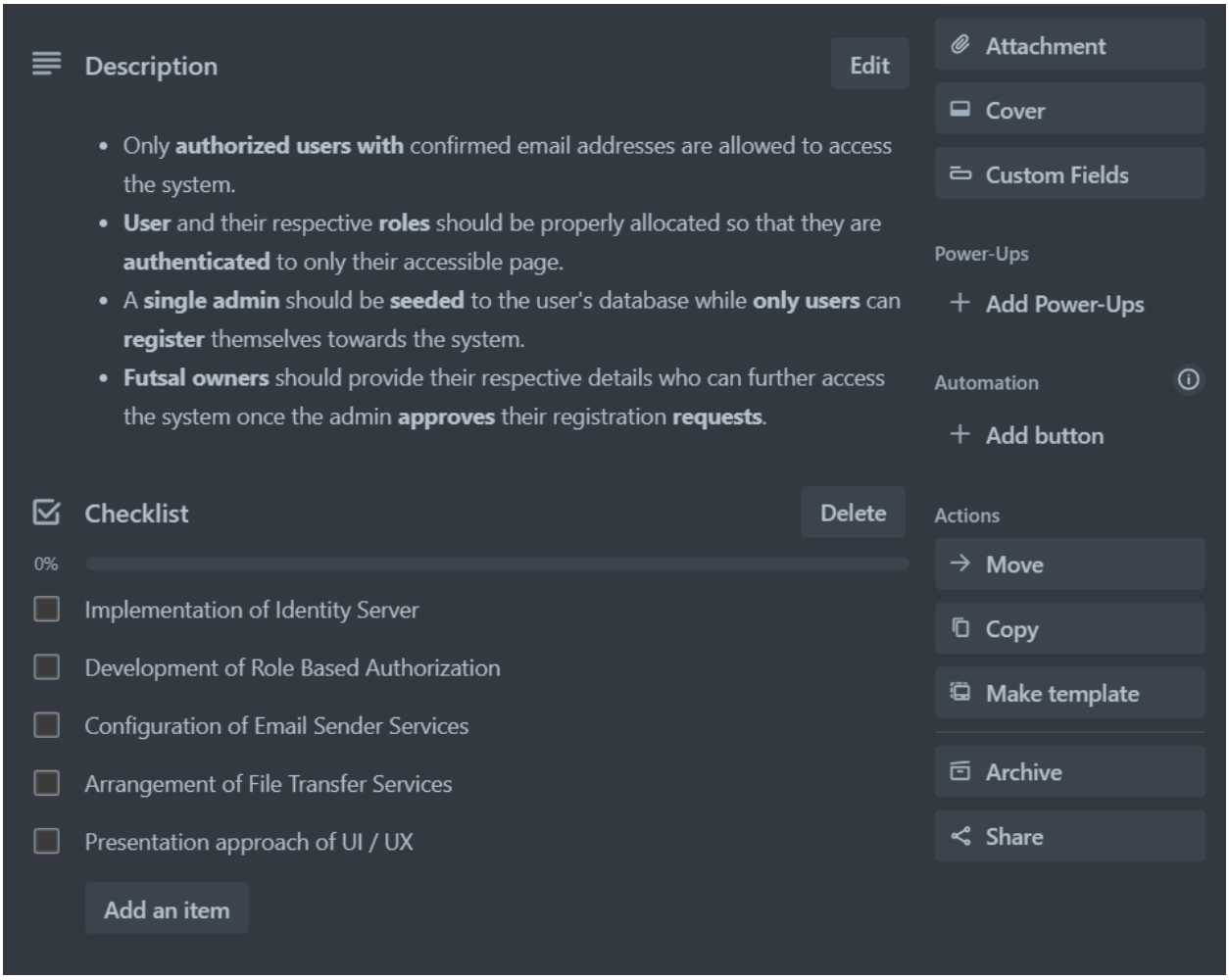


Figure 65: Trello Ticket: Authentication & Authorization (2)

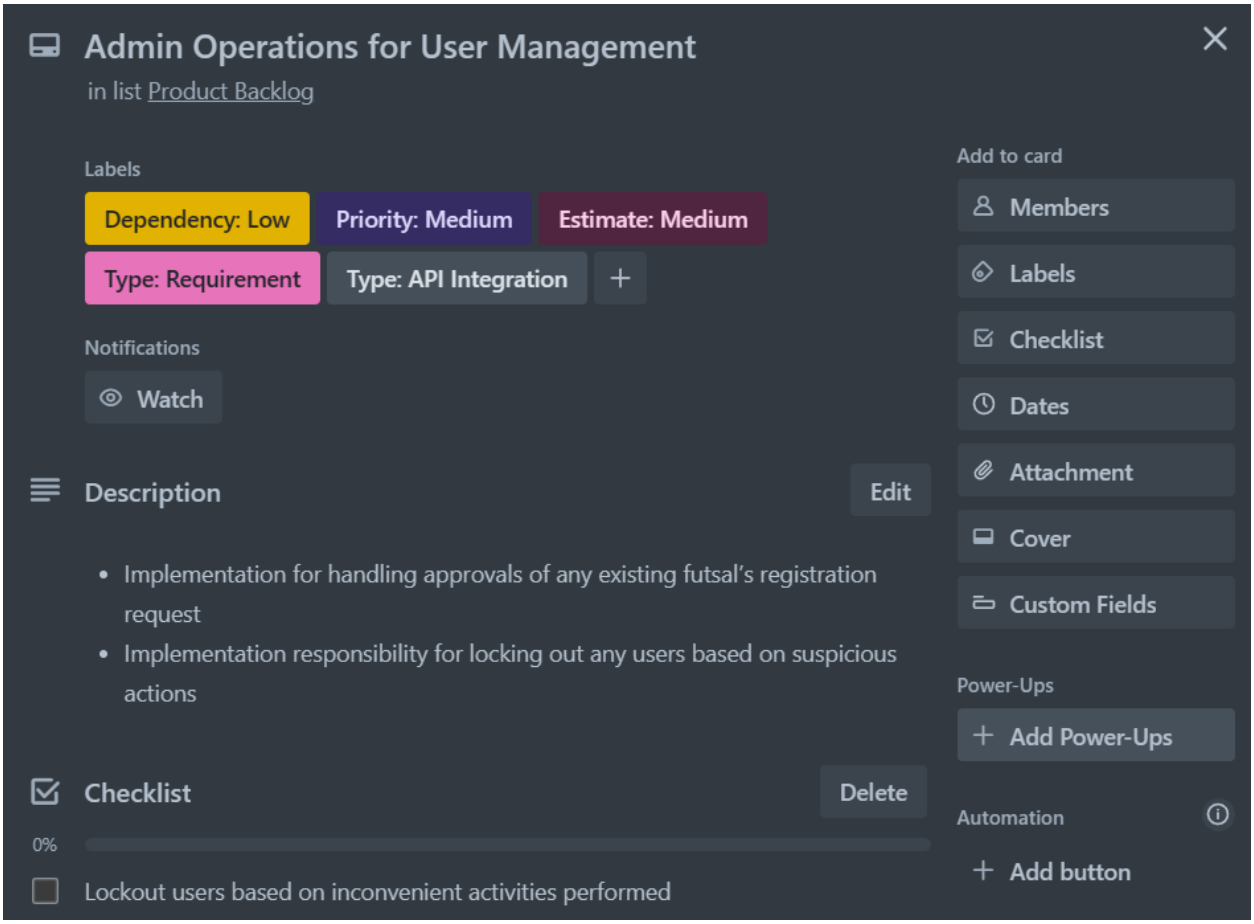


Figure 66: Trello Ticket: Admin Operations for User Management

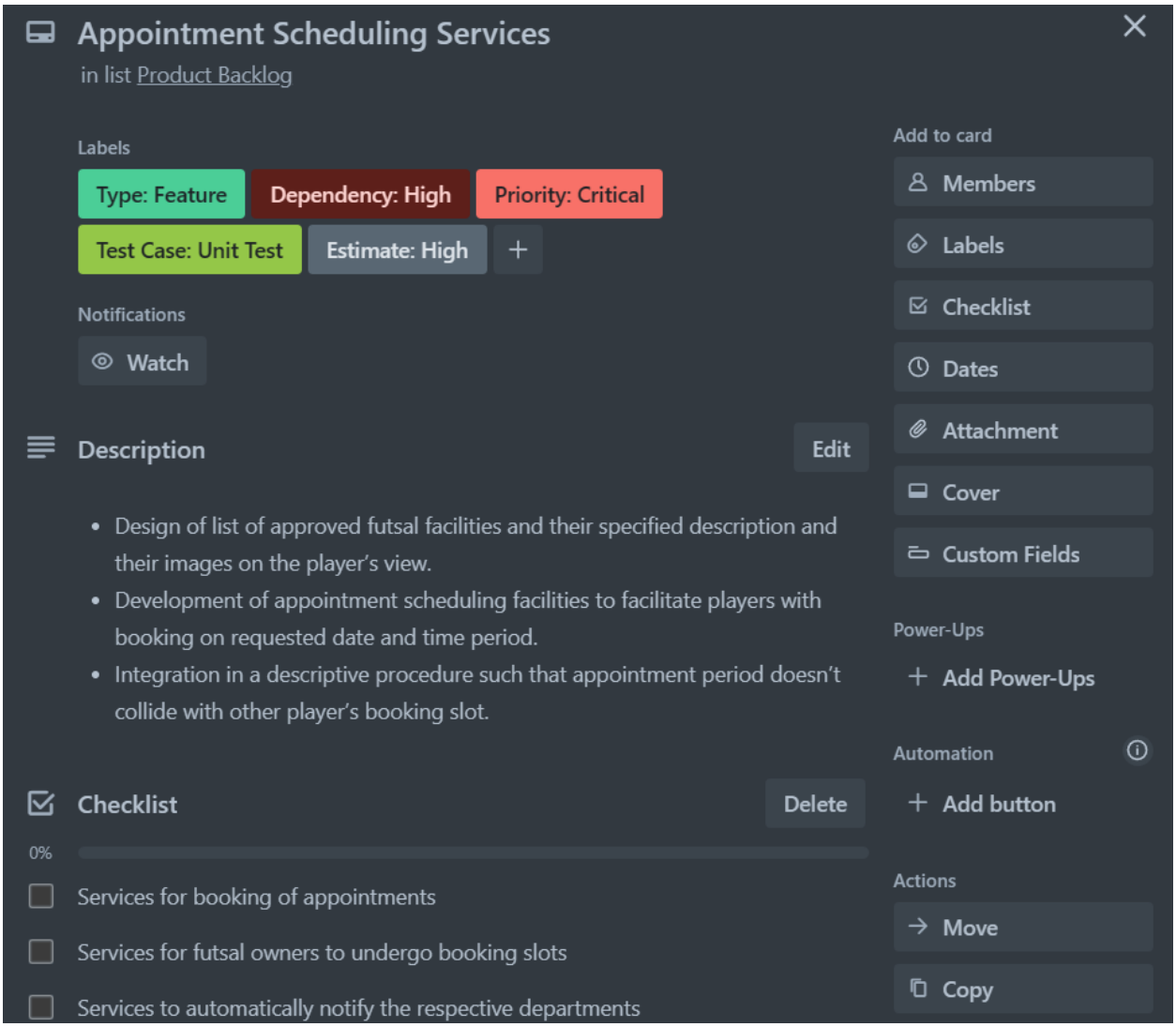


Figure 67: Trello Ticket: Appointment Scheduling Services

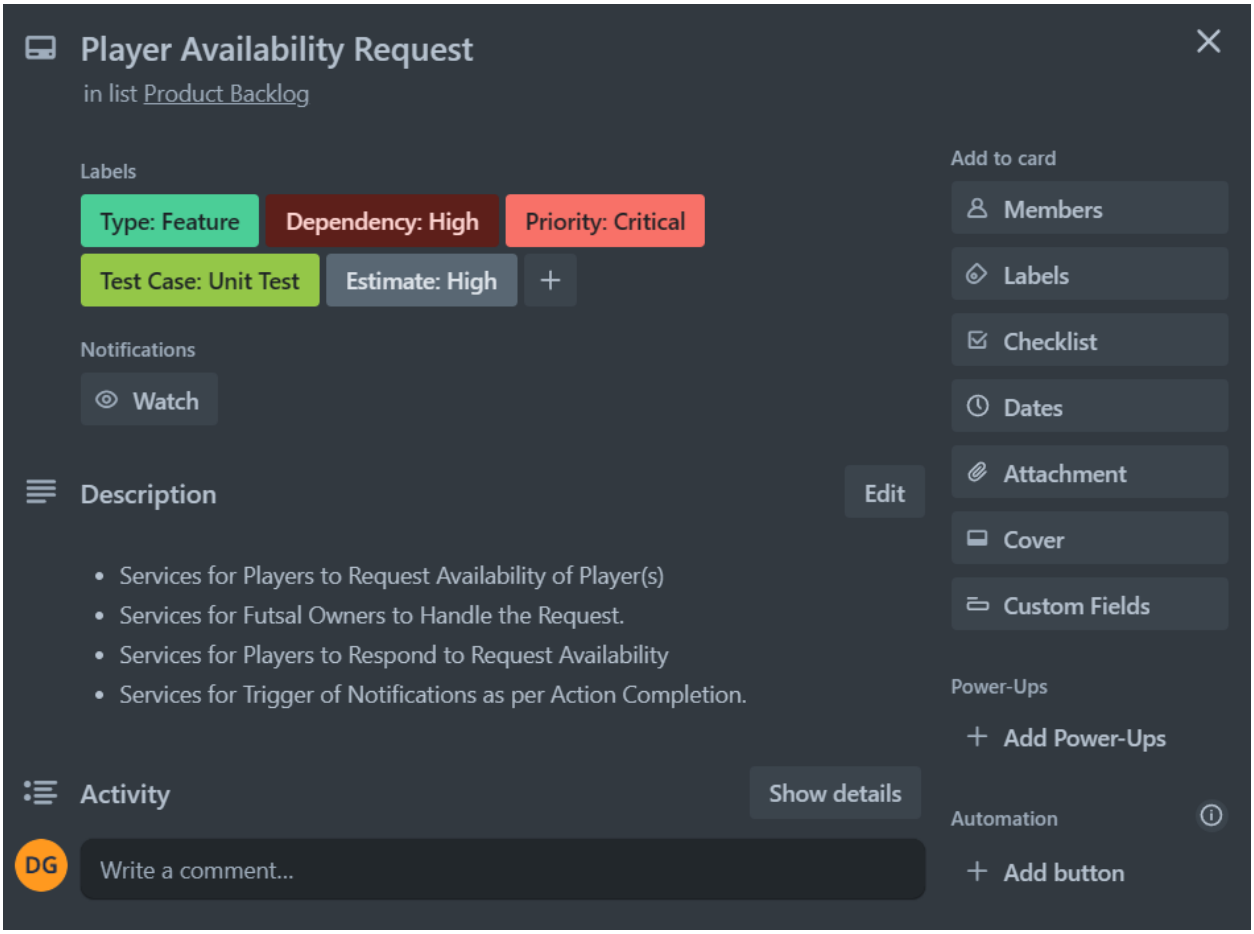


Figure 68: Trello Ticket: Player Available Request

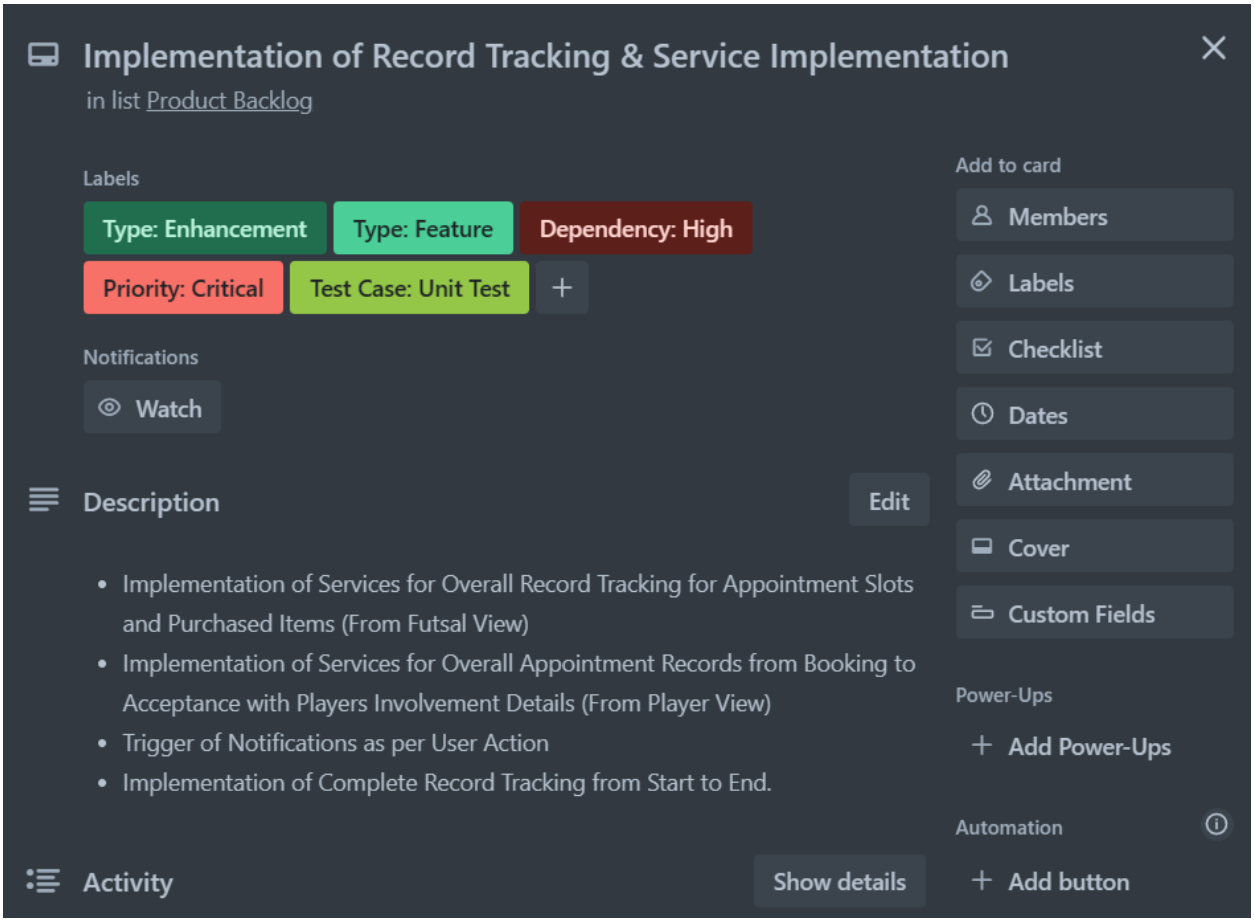


Figure 69: Trello Ticket: Implementation of Record Tracking

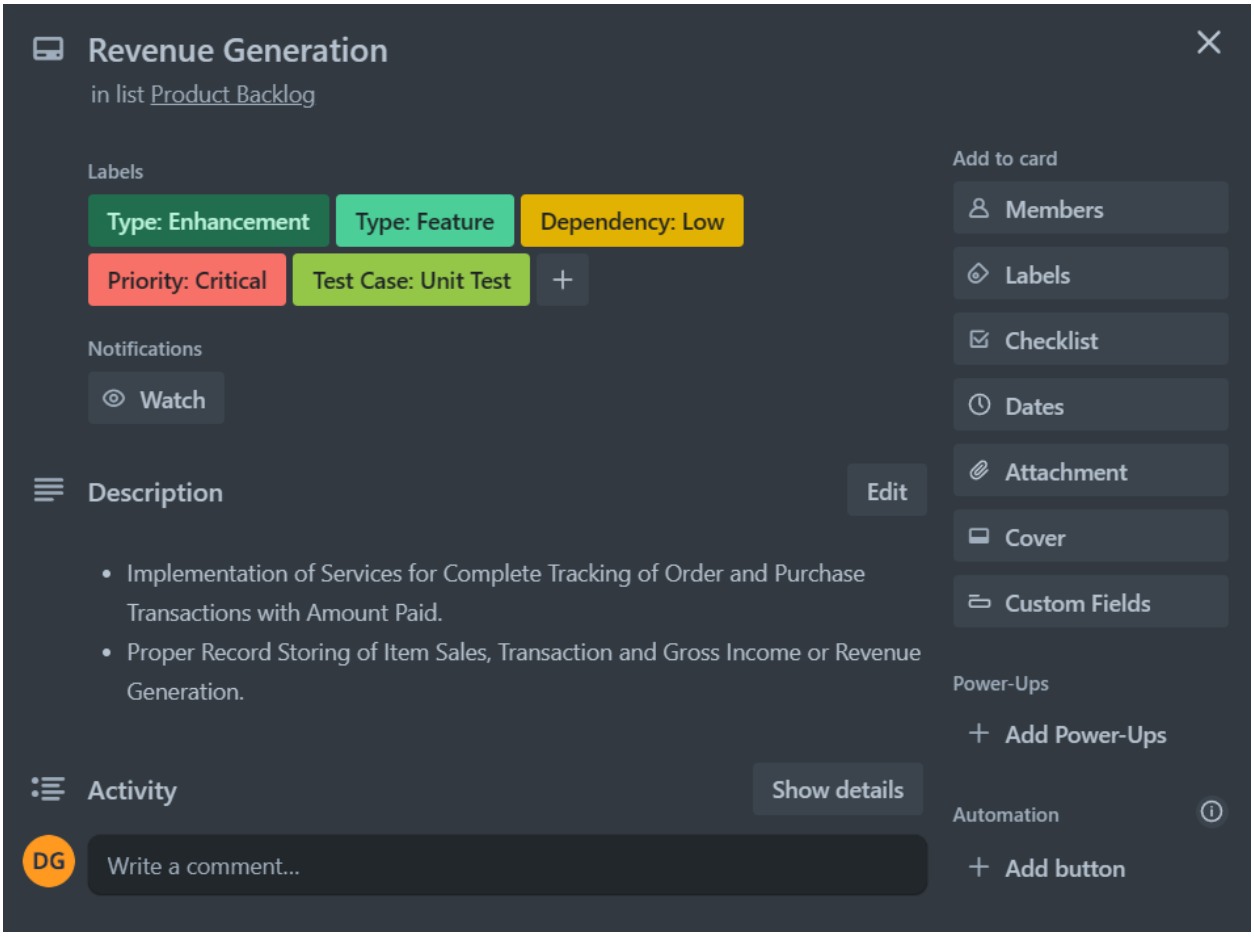


Figure 70: Trello Ticket: Revenue Generation

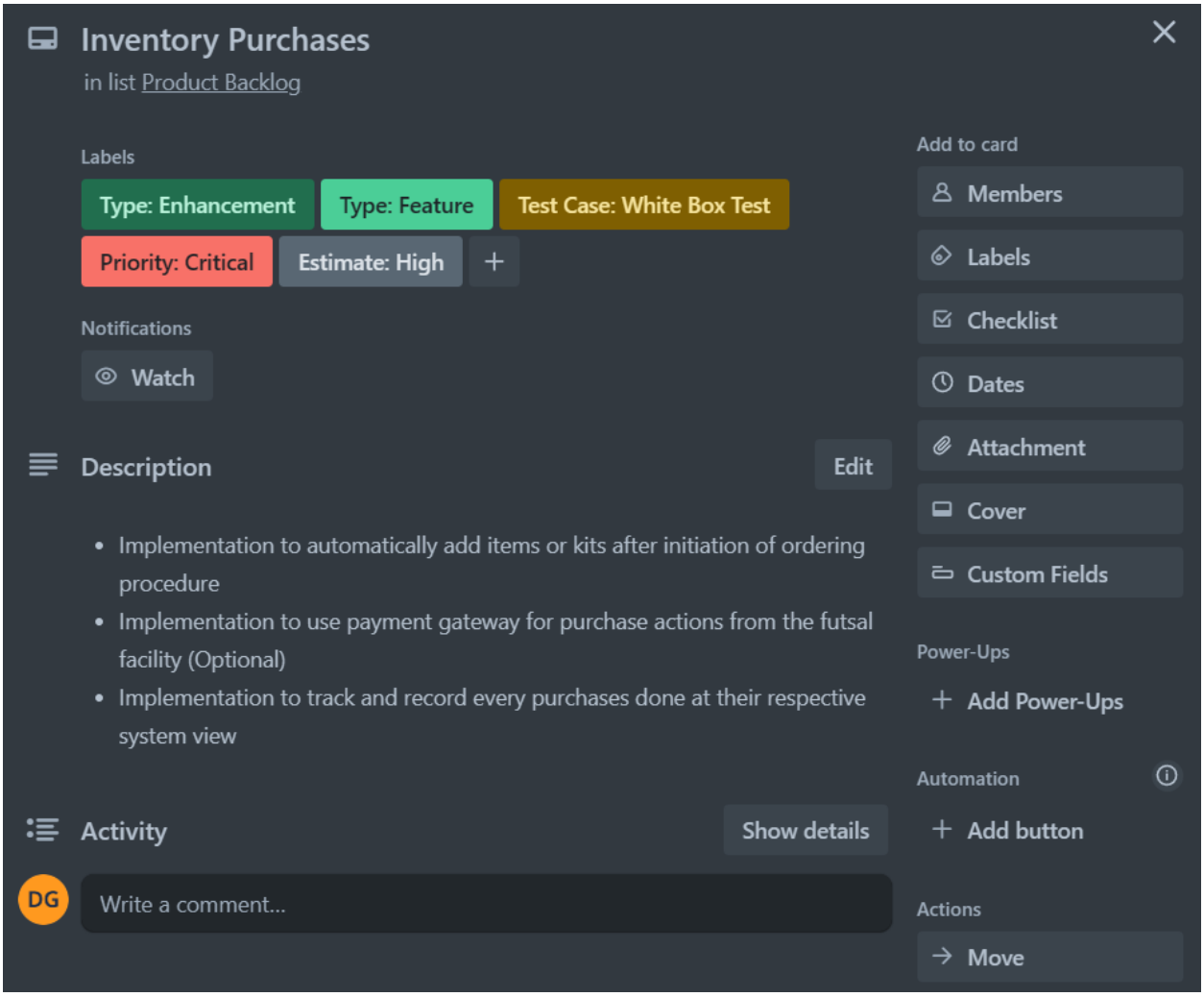


Figure 71: Trello Ticket: Inventory Purchases

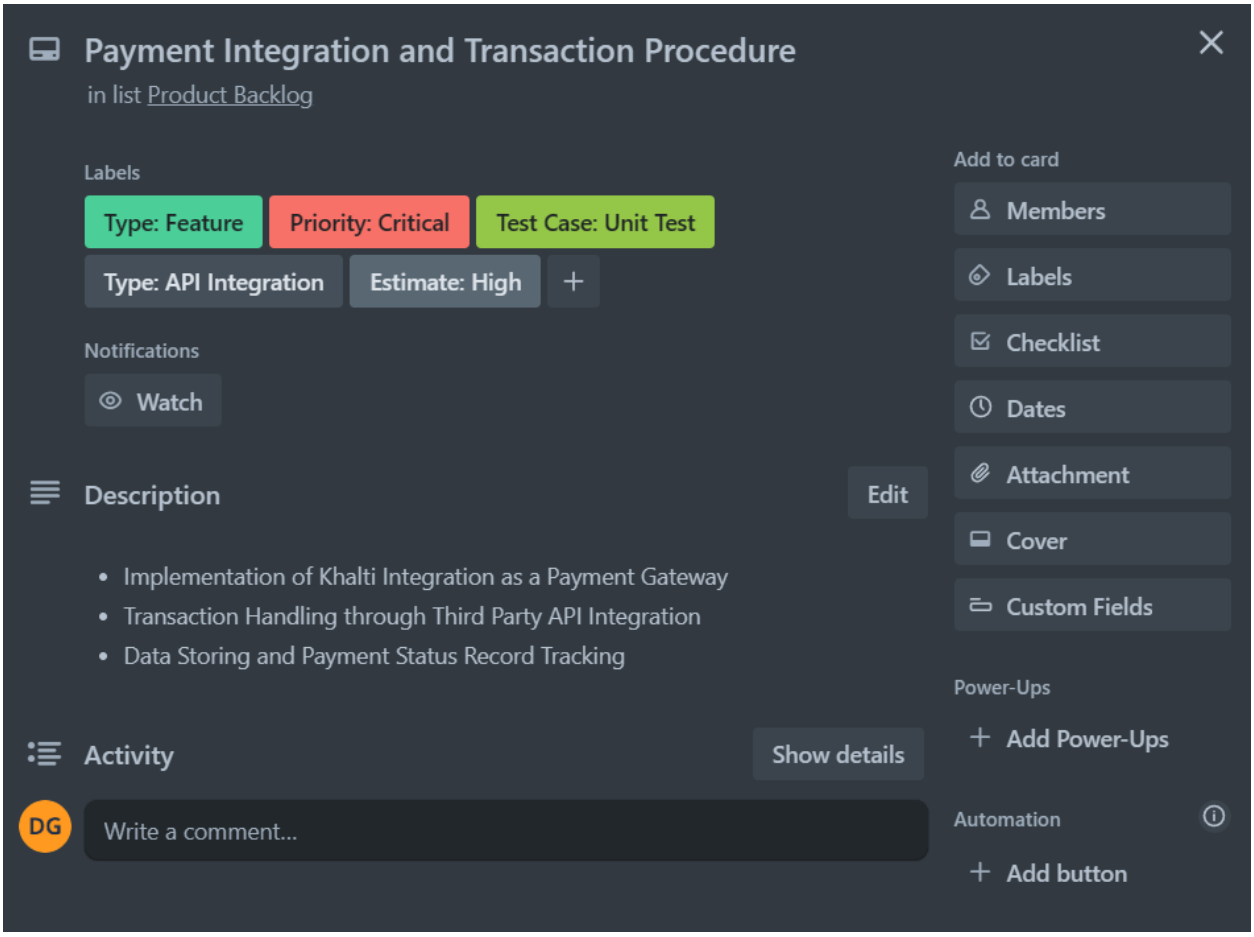


Figure 72: Trello Ticket: Payment Integration & Transaction Procedure

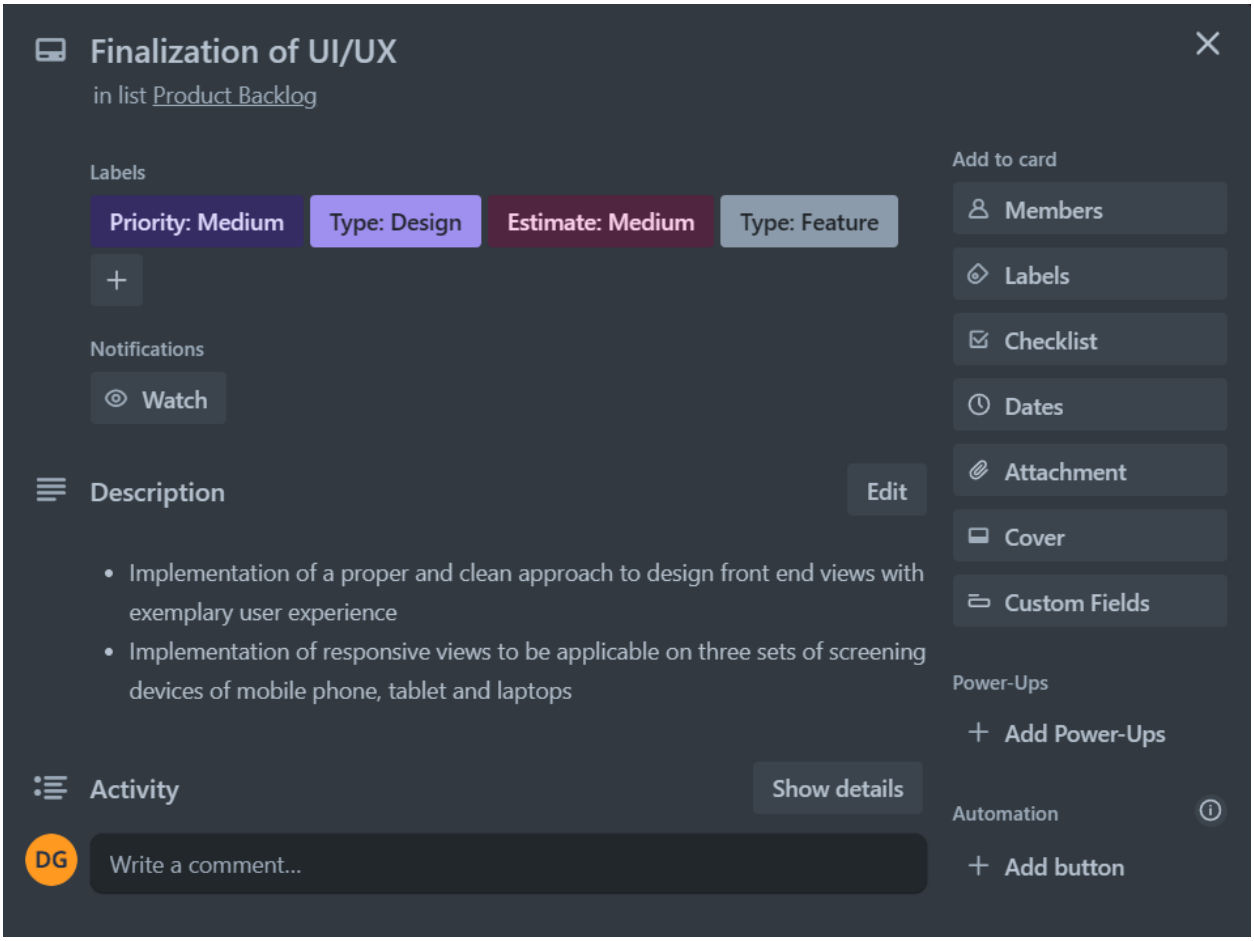


Figure 73: Trello Ticket: Finalization of UI/UX

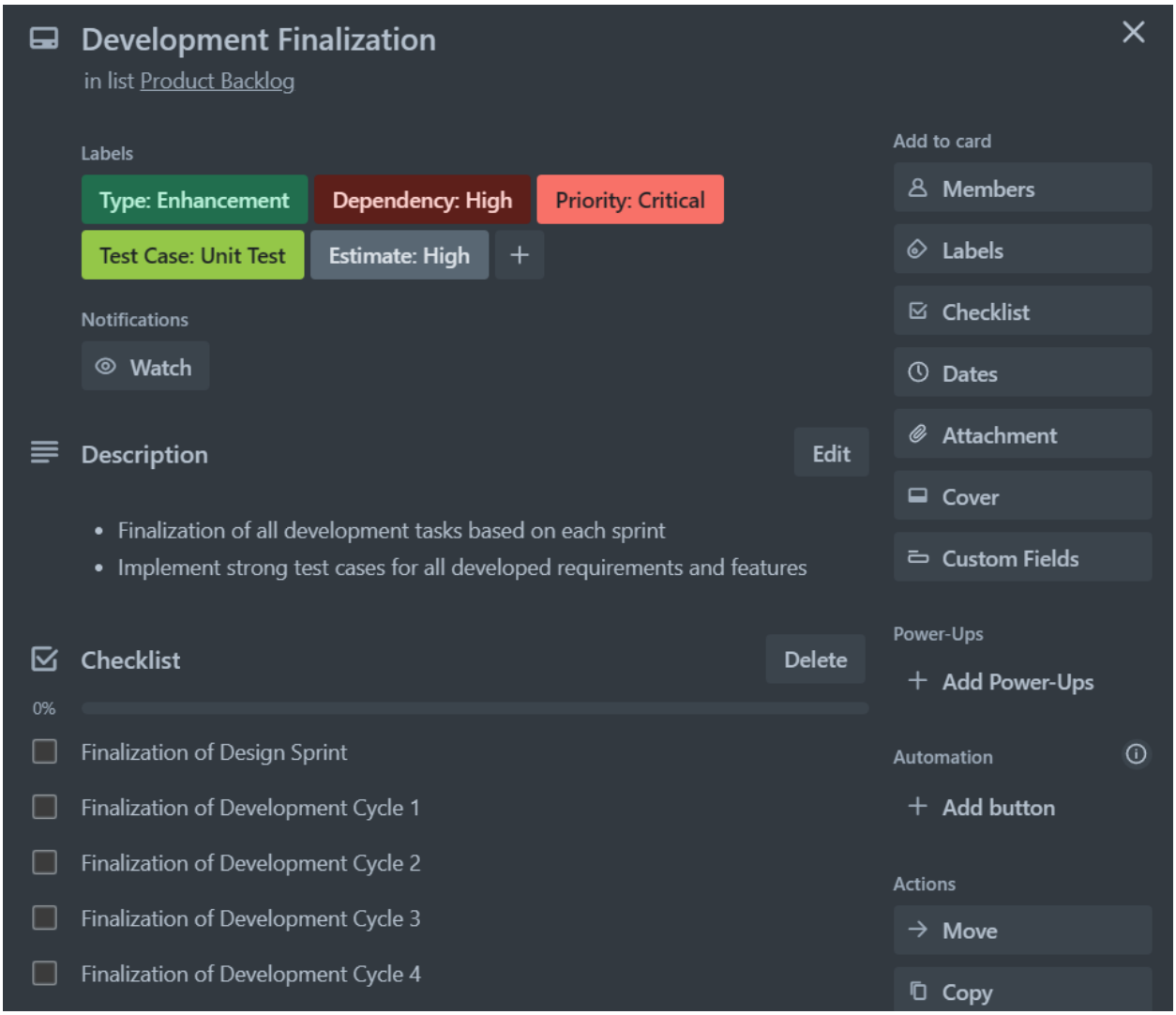


Figure 74: Trello Ticket: Development Finalization

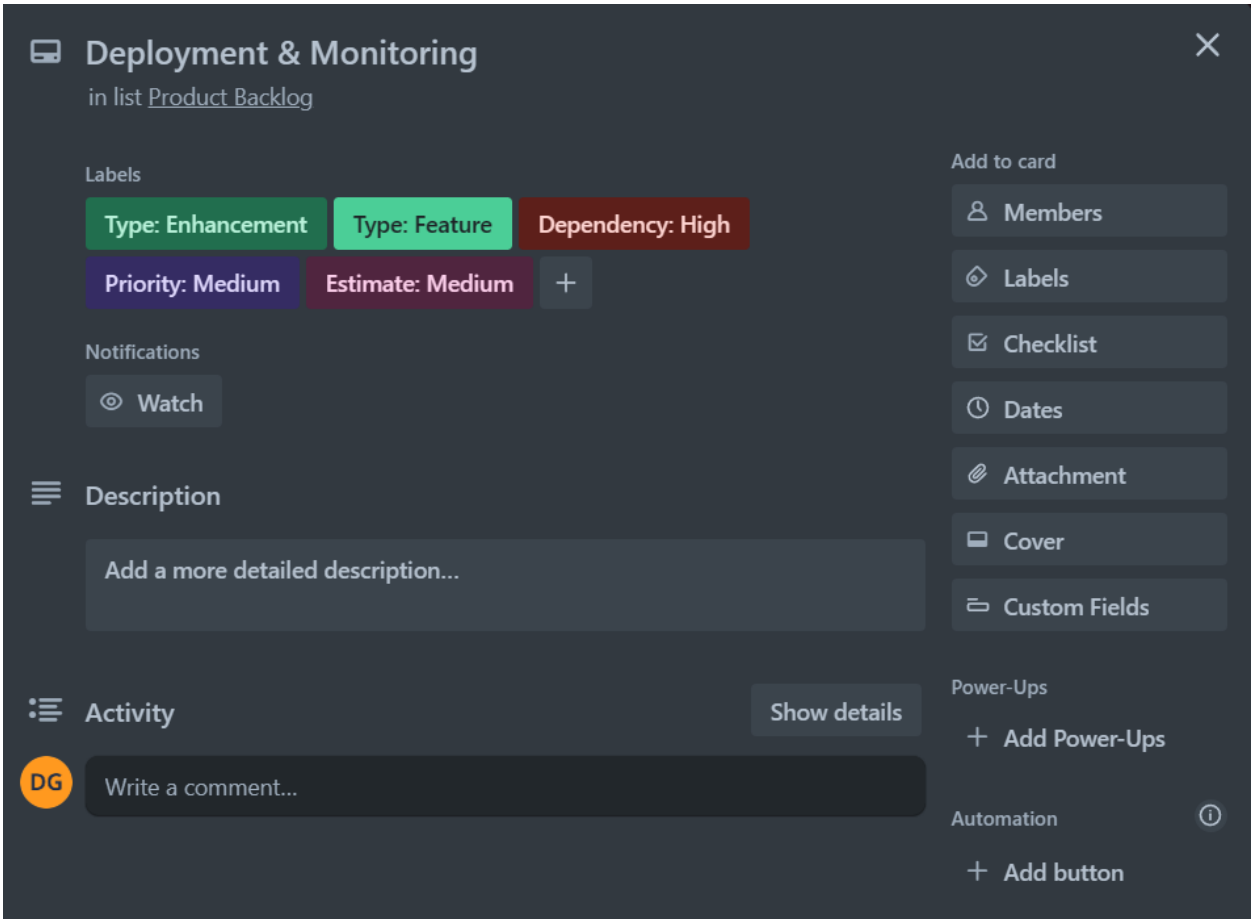


Figure 75: Trello Ticket: Deployment & Monitoring

3.15. Architecture Design

According to Britannica, architecture is the art and technique of designing and building, as distinguished from the skills associated with construction. The practice of architecture is employed to fulfill both practical and expressive requirements, and thus it serves both utilitarian and aesthetic ends (Britannica, 2022). Like its definition, software engineering and development also has its own sets of constructing and structuring any system or organization.

Software architecture is a set of guiding principles for designing and developing software which specifies the structure and organization of the software system. It also explains the interactions between components, abstraction layers, and other software system features (Interview Bit, 2022). The web architecture the system will follow is depicted in the following figure.

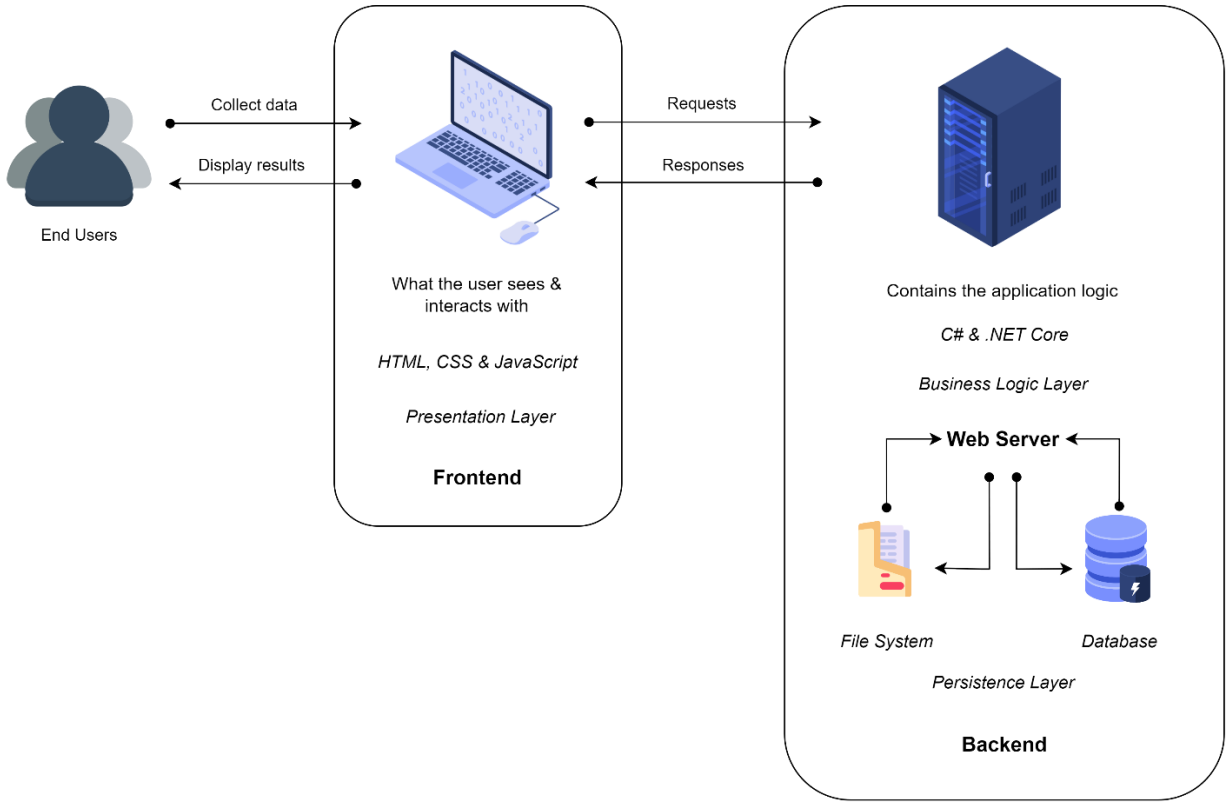


Figure 76: System Architecture: APIs and Client

Communication within a web architecture is depicted in the above graphic in a systematic manner. There are three distinct levels of abstraction in this model; they are the presentation, business logic, and persistence layers. Whenever a user makes a request to gather data through the user interface, the API or client side sends an HTTP request to the server side, where .NET follows the business logic to serialize or deserialize the data instance and responds appropriately, regardless of the success or failure of the data flow. Furthermore, to retrieve or modify data, the web server maintains communication with the persistence layer, which is often called the database. The process is a two-way process including the data communication between the client and the server which takes into account through the API requests and responses.

Since the web architecture, which specifies the rules for validating and organizing online requests and responses, has been established, it is also necessary to adhere to design patterns and development architecture of the .NET Core environment. The development patterns and architecture the application will be following are outlined below:

- Application Architecture Pattern (Link to Appendix)
- Development Architecture (Link to Appendix)

3.16. Database Migrations

Importing SQL Server and Entity Framework (EF) Core during database migrations in .NET Core applications was the approach considered for keeping up with the changing database structure. A well-liked methodology, Code-First involves creating the database schema from the application's domain model.

Setting Up Database Migrations:

Entity Framework Core (EF Core): EF Core is a lightweight and extensible Object-Relational Mapping (ORM) framework that simplifies database interactions in .NET Core applications.

SQL Server: SQL Server is a widely used relational database management system (RDBMS) that integrates seamlessly with .NET Core applications.

The approaches taken in database migration are as follows:

- The entities were defined by creating a category that stand in the domain objects called POCO classes which are the structure of the tables in the database.
- Creation of a new class called ApplicationDbContext and inheriting from DbContext. This class contains the DbSet attributes for every entity and represents the database session.
- To build a migration scaffold according to model changes, the following command was executed and the database changes were uploaded with the created migration.

```
dotnet ef migrations add DbSetup

dotnet ef database update
```

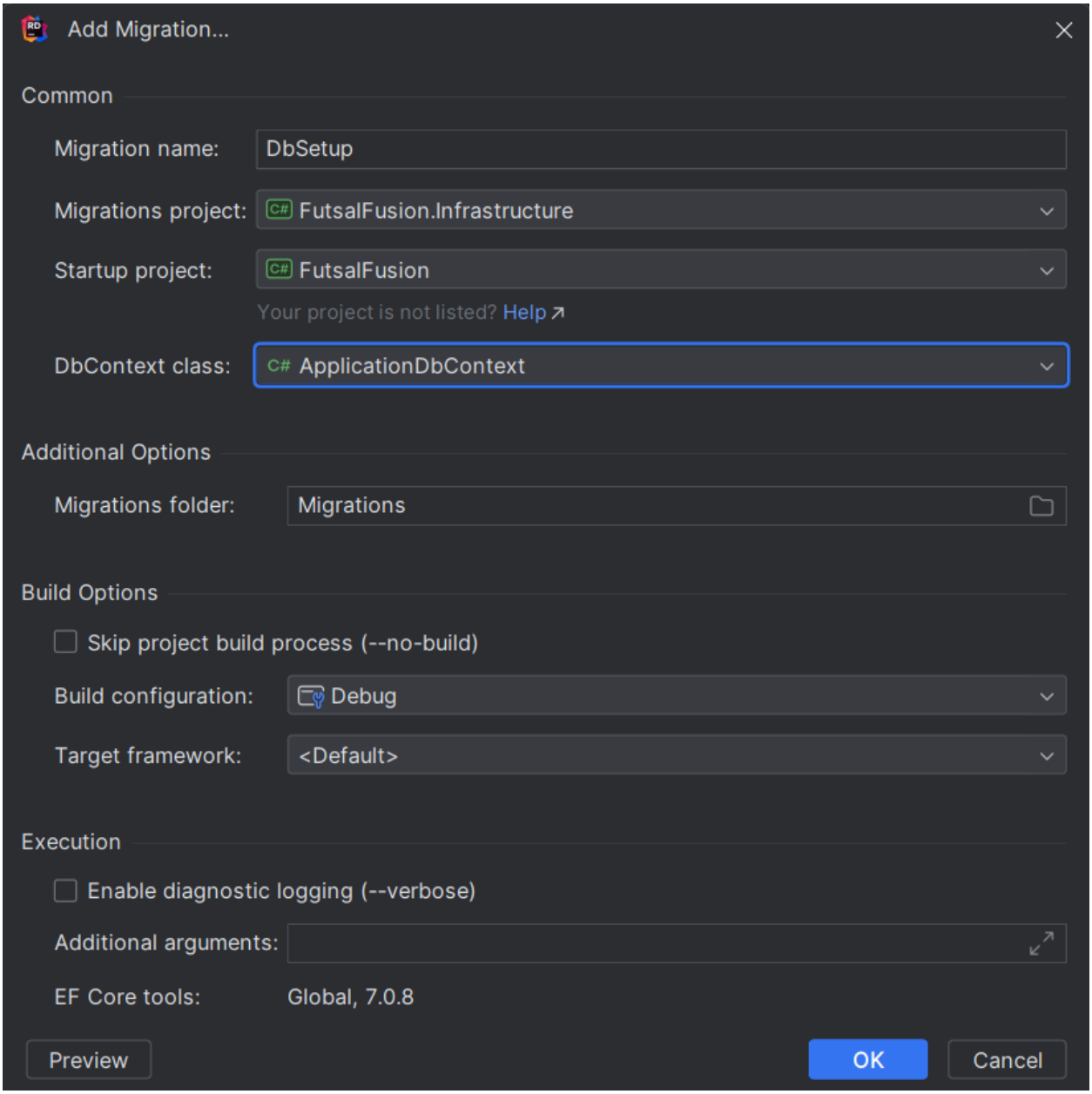


Figure 77: Migrations: Adding a Migration File

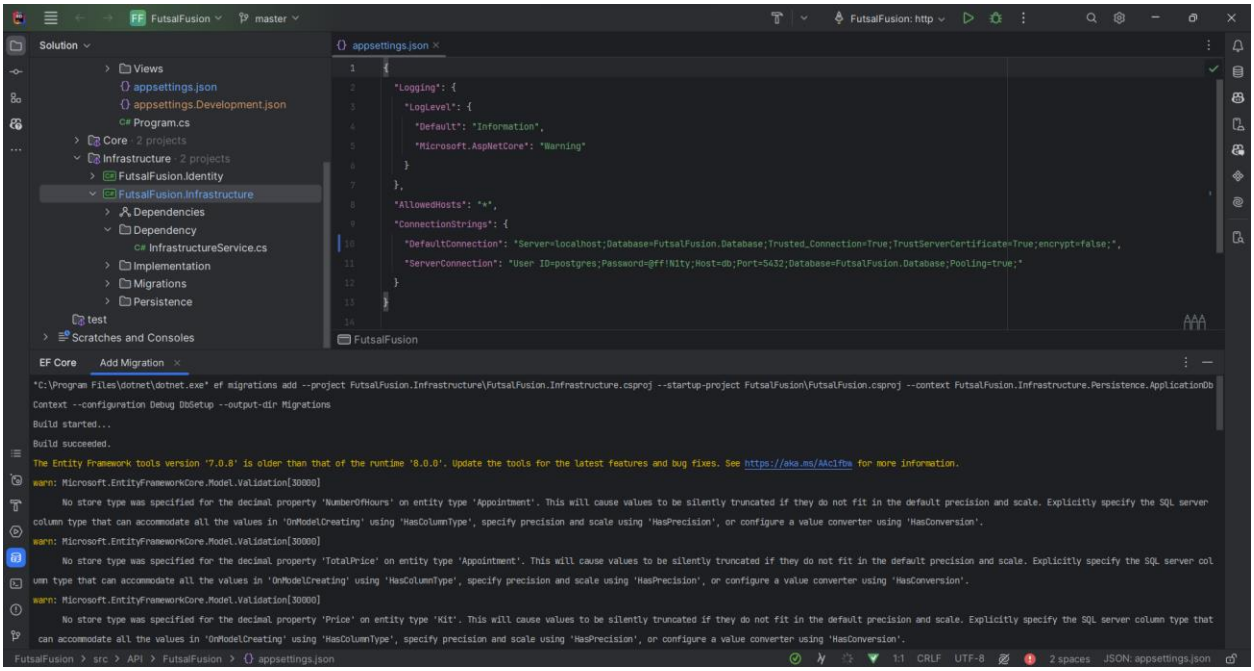


Figure 78: Migrations: Migration Script Commands

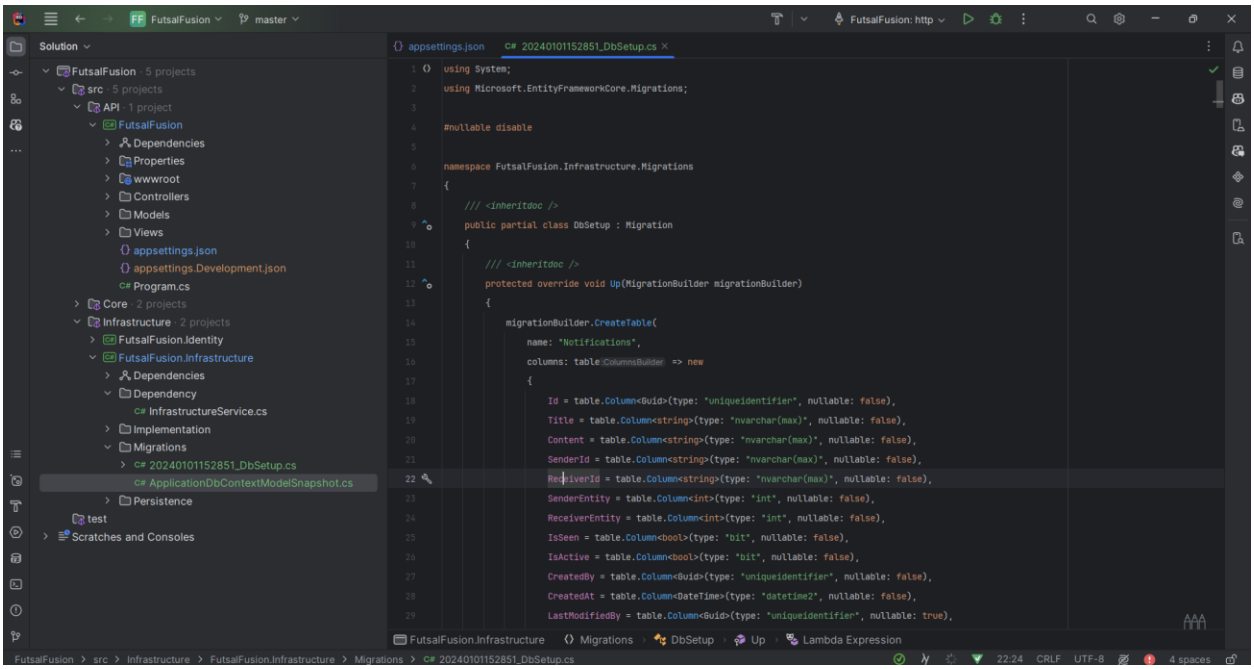


Figure 79: Migrations: Generation of Migration File

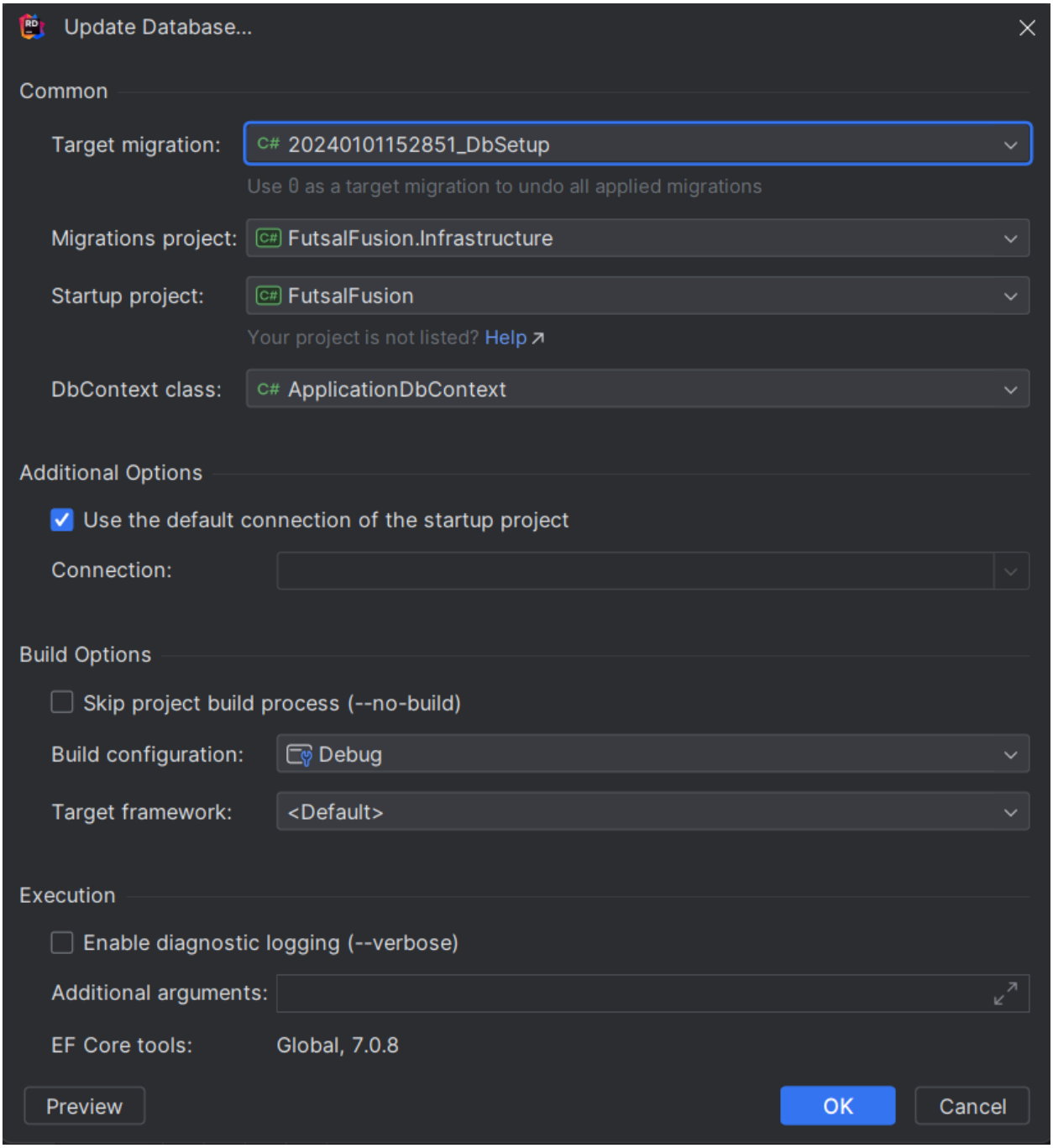


Figure 80: Migrations: Updating Migration File

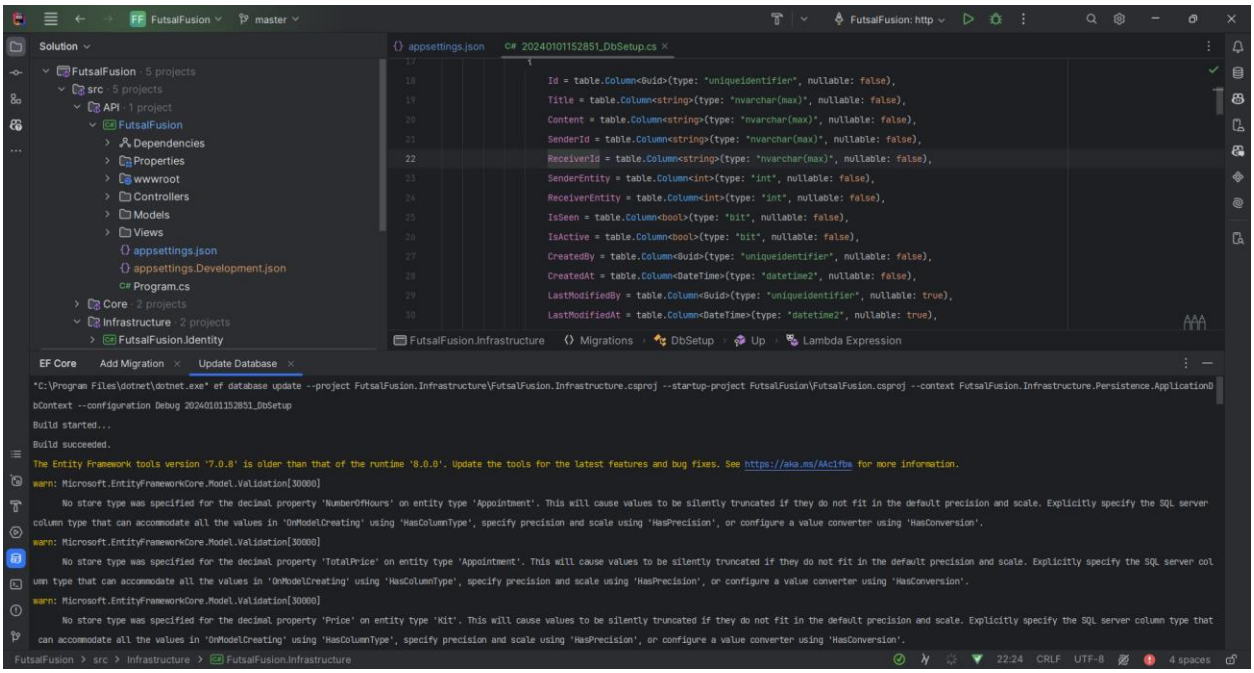


Figure 81: Migrations: Update Scripts

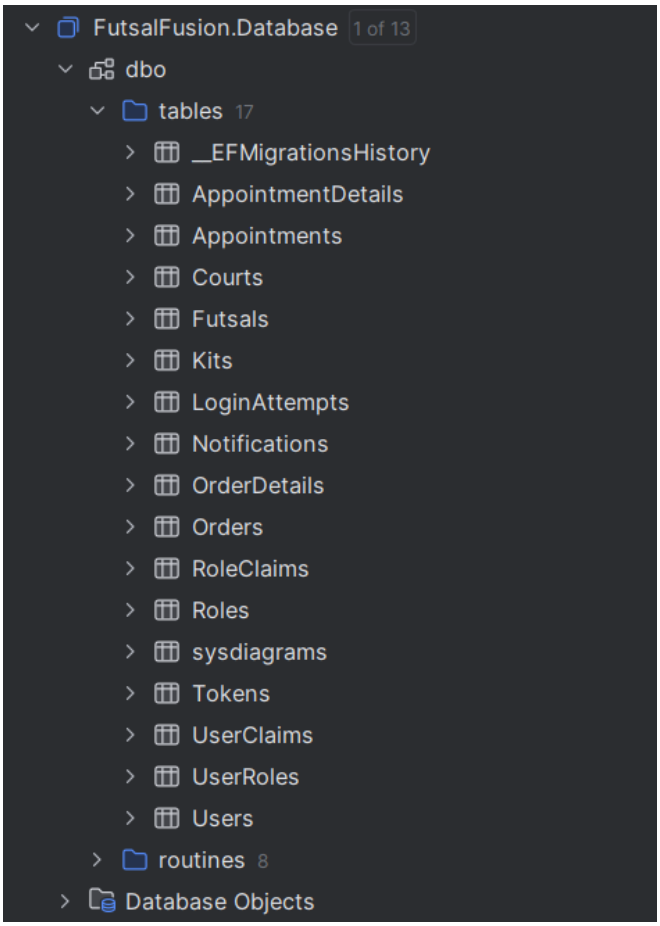


Figure 82: Migration: Updated Database Entities

3.17. System Development

The Agile development process has yielded a strong and well-structured application and the required blueprints for constructing a particular feature by the end of Sprint 3. Key architectural decisions have been solidified, and the database schema has been painstakingly constructed with user-friendly APIs. With a solid base established through user feedback, drawings, diagrams, and collaborative efforts, it was crucial to confidently go forward with system development and actual hard coding in the future sprints. This will guarantee a smooth and efficient development process.

3.17.1. Frontend Development

Since the course work mostly focused on creating and initializing the project with visual aids that would highlight important aspects of the project, a considerable amount of front-end development has been neglected. The purpose of creating wireframes is to provide a visual representation of the system's user interface and user experience (UI/UX) for the benefit of all parties concerned. With the main criteria defined and the functionality of each feature tracked back to its source, the front-end development can move forward without a hitch.

3.17.2. Backend Development

The indicated amount of back-end development is approximately 20%. The project has been built following a pattern that corresponds to the ideals of Domain-Driven Design since a stable architecture is required. We have also finished designing repositories, services, and generics, and basic CRUD operations on the API side.

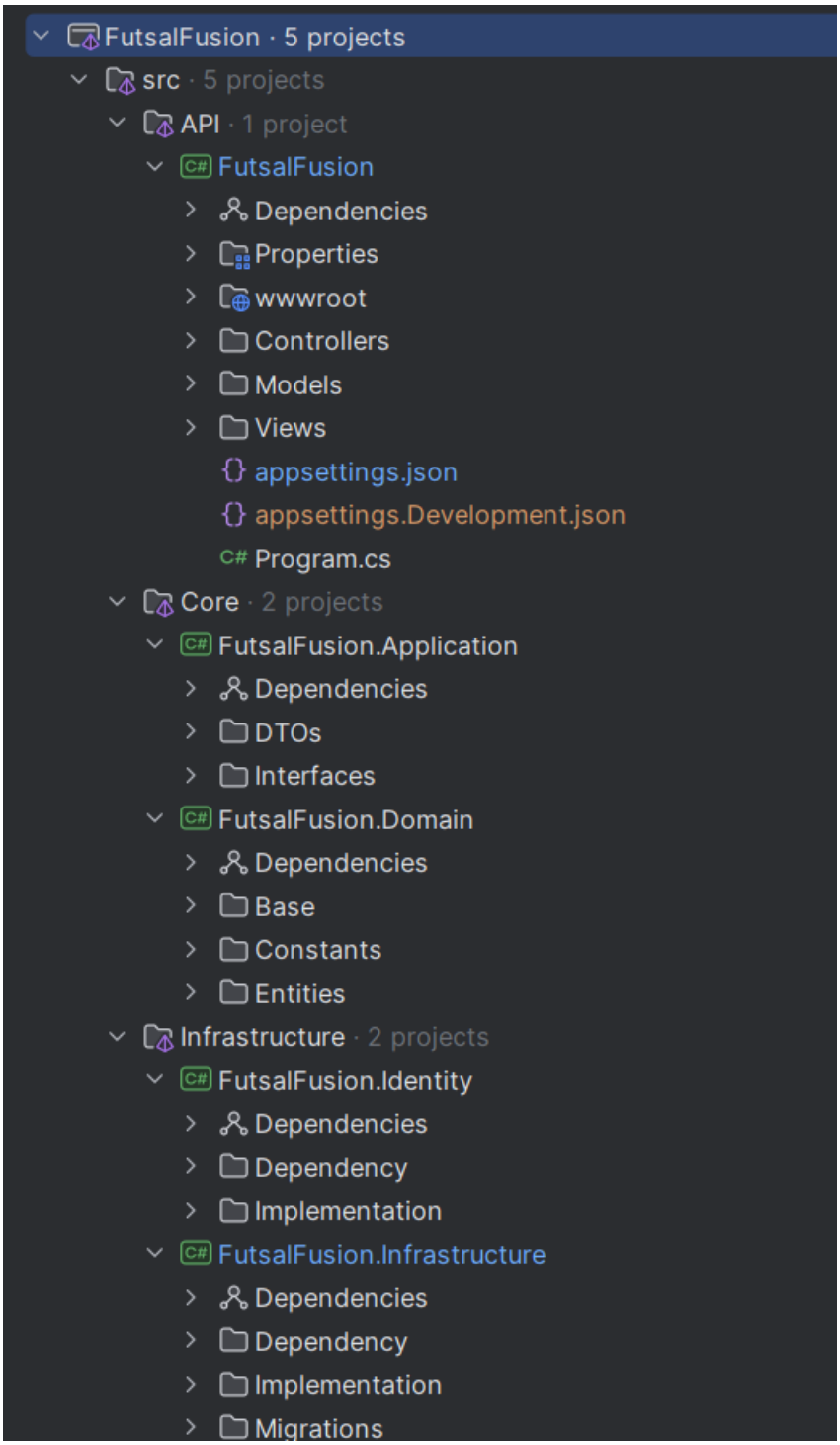


Figure 83: Development: Implementation of Clean Architecture

C# Program.cs

```
using ...

var builder = WebApplication.CreateBuilder(args);

var services = builder.Services;

var configuration = builder.Configuration;

services.AddInfrastructureService(configuration);

services.AddIdentityService(configuration);

services.AddControllersWithViews();

services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(100);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});

services.ConfigureApplicationCookie(options =>
{
    options.LogoutPath = $"/User/Account/Logout";
    options.LoginPath = $"/User/Account/Login";
    options.AccessDeniedPath = $"/User/Account/AccessDenied";
});

var app = builder.Build();
```

Figure 84: Development: Program Configuration

C# Program.cs

```
33  if (!app.Environment.IsDevelopment())
34  {
35      app.UseExceptionHandler("/User/Home/Error");
36      app.UseHsts();
37  }
38
39  app.UseHttpsRedirection();
40
41  app.UseStaticFiles();
42
43  app.UseRouting();
44
45  app.UseAuthentication();
46
47  app.UseAuthorization();
48
49  app.MapRazorPages();
50
51  app.MapControllers();
52
53  app.MapControllerRoute(
54      name: "default",
55      pattern: "{area=User}/{controller=Home}/{action=Index}/{id?}");
56
57  using (var scope = app.Services.CreateScope())
58  {
59      var dbInitializer = scope.ServiceProvider.GetRequiredService<IDbInitializer>();
60
61      dbInitializer.Initialize();
```

Figure 85: Development: Addition of Middleware and Routing

C# Program.cs C# IUserIdentityService.cs ×

```
using FutsalFusion.Application.DTOs.Identity;

namespace FutsalFusion.Application.Interfaces.Identity;

4 usages  1 inheritor  Deven Gurung
public interface IUserIdentityService
{
    1 usage  1 implementation  Deven Gurung
    Task<Tuple<string, string>> Register(RegisterDto register, string? returnUrl = null);

    1 usage  1 implementation  Deven Gurung
    Task<bool> ConfirmEmail(Guid userId, string code);

    1 usage  1 implementation  Deven Gurung
    Task<string> Login(LoginDto login, string? returnUrl = null);

    1 usage  1 implementation  Deven Gurung
    Task LogOut();

    1 usage  1 implementation  Deven Gurung
    Task<Tuple<string, string>> ForgetPassword(ForgotPasswordDto forgotPassword);

    1 usage  1 implementation  Deven Gurung
    Task<string> ResetPassword(ResetPasswordDto resetPassword);
}
```

Figure 86: Development: Interfaces for Authentication and Authorization

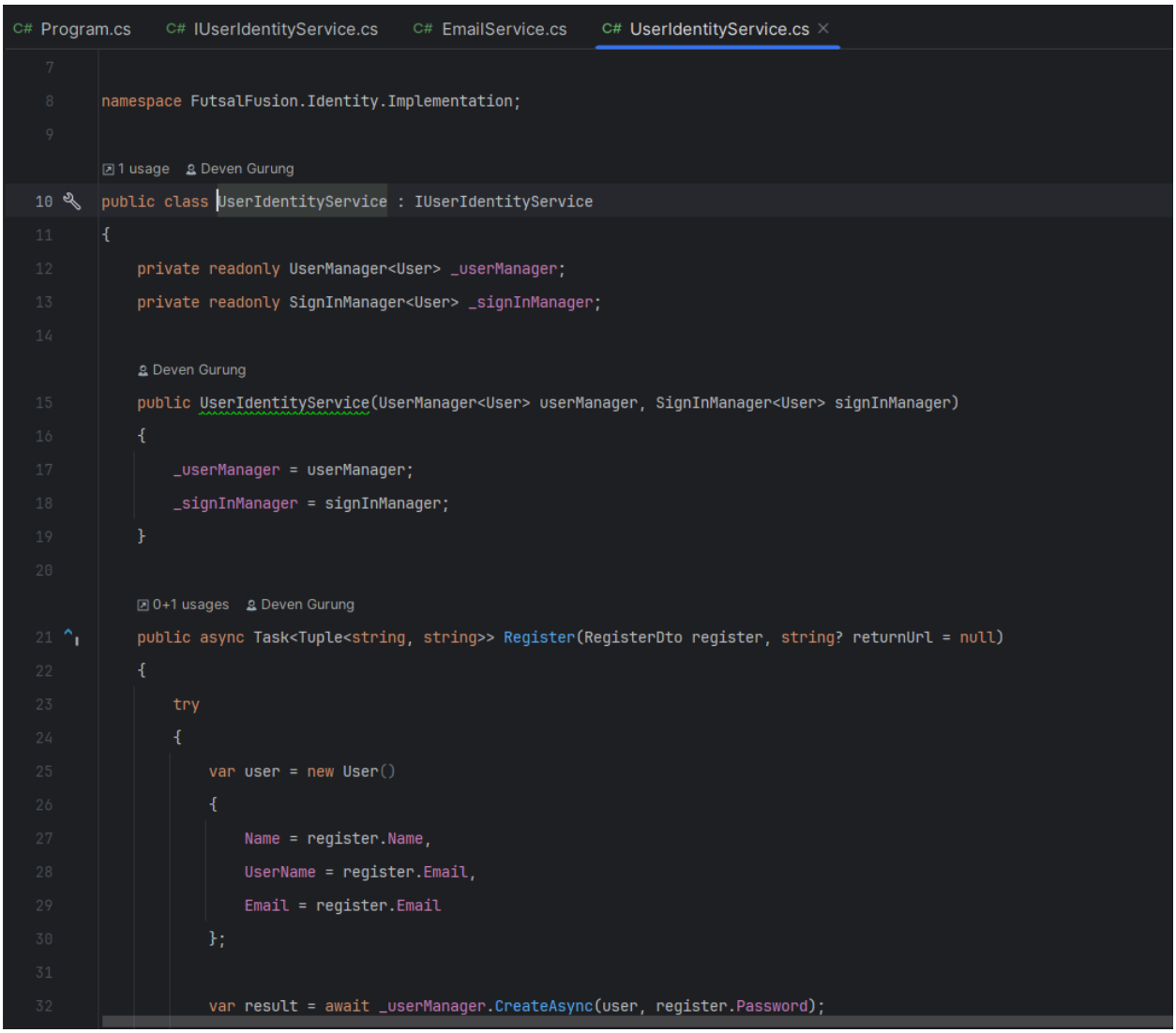


Figure 87: Development: Services for Registration

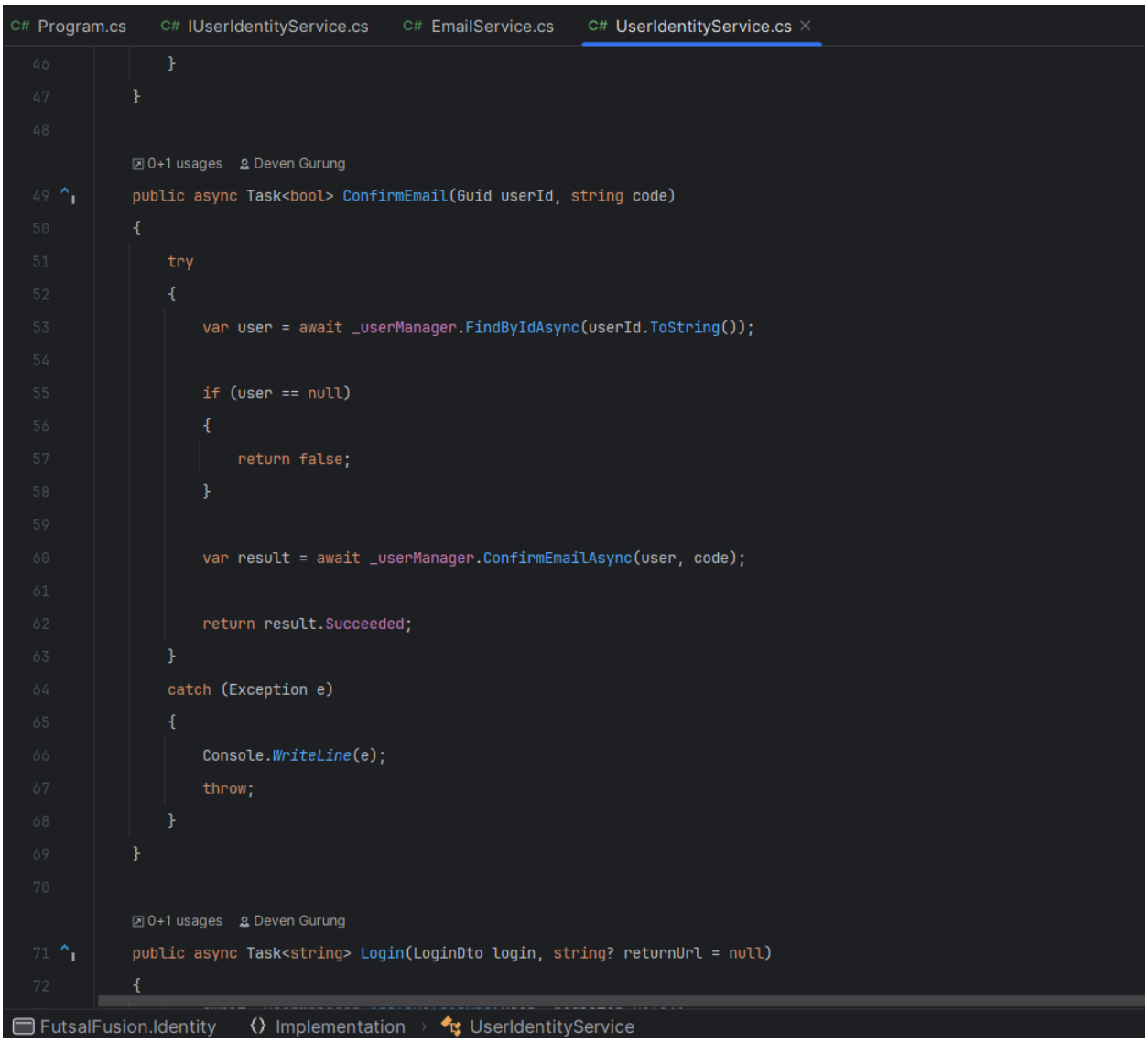


Figure 88: Development: Email Confirmation and Login

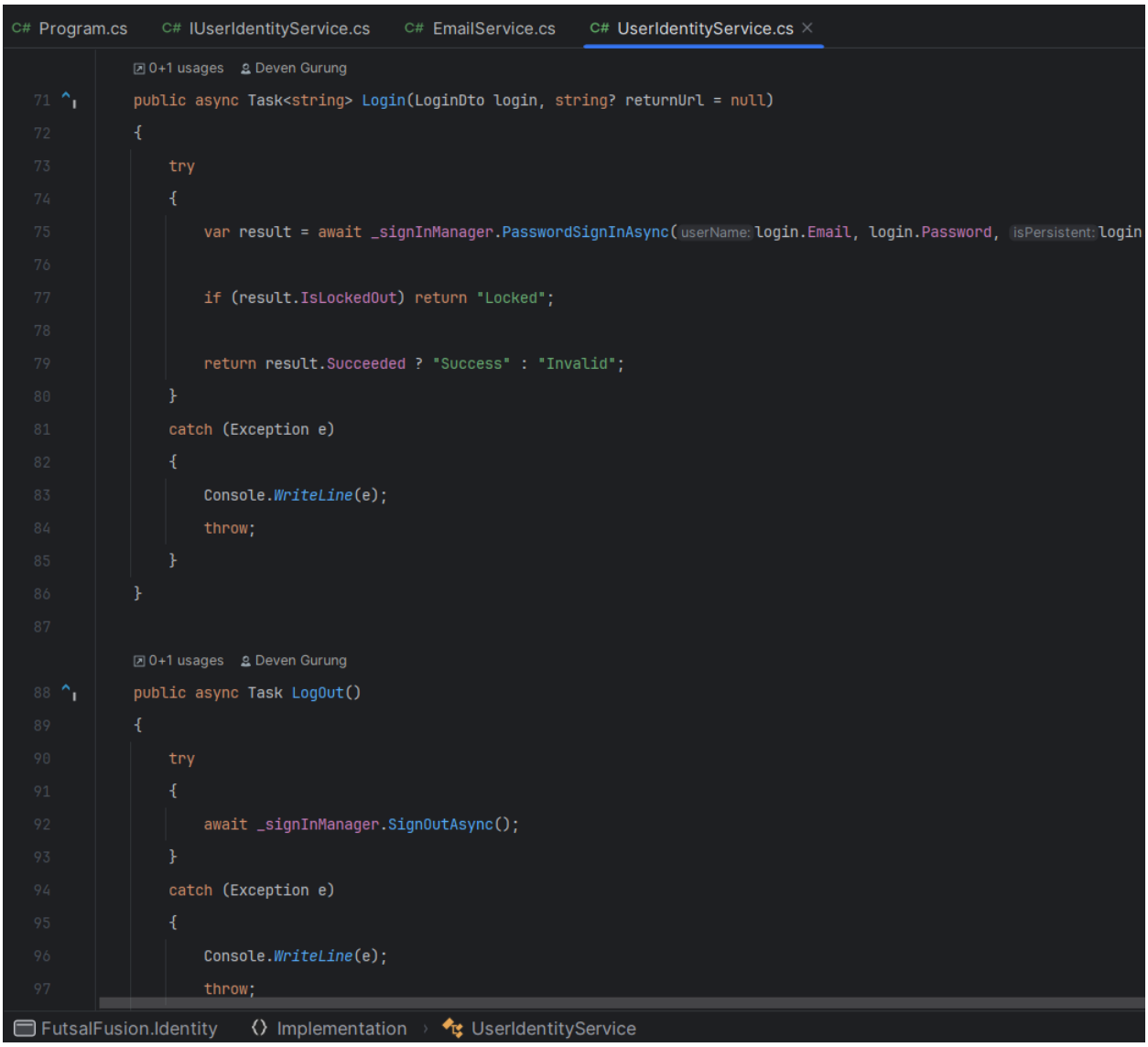


Figure 89: Development: Login and Logout

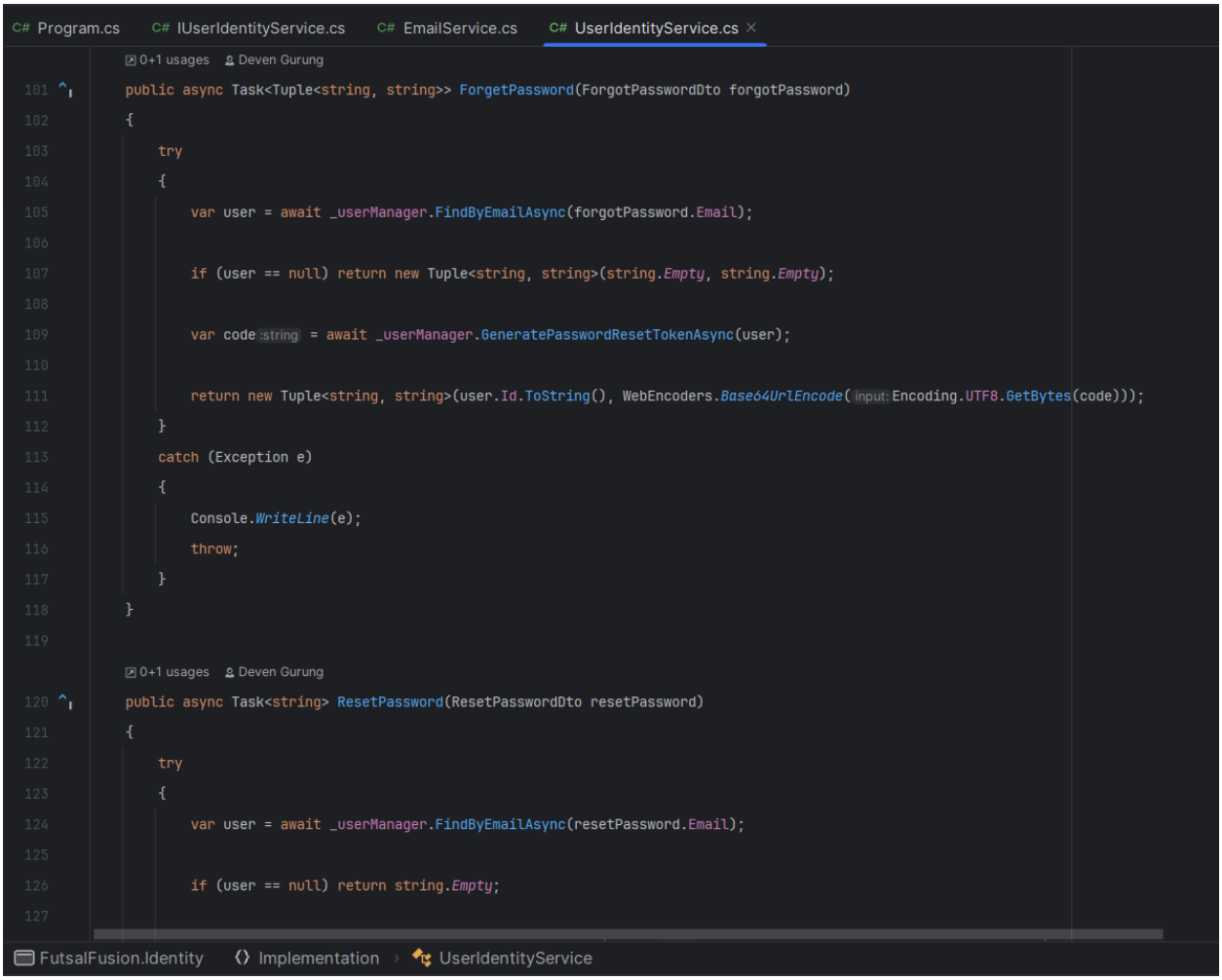


Figure 90: Development: Password Management

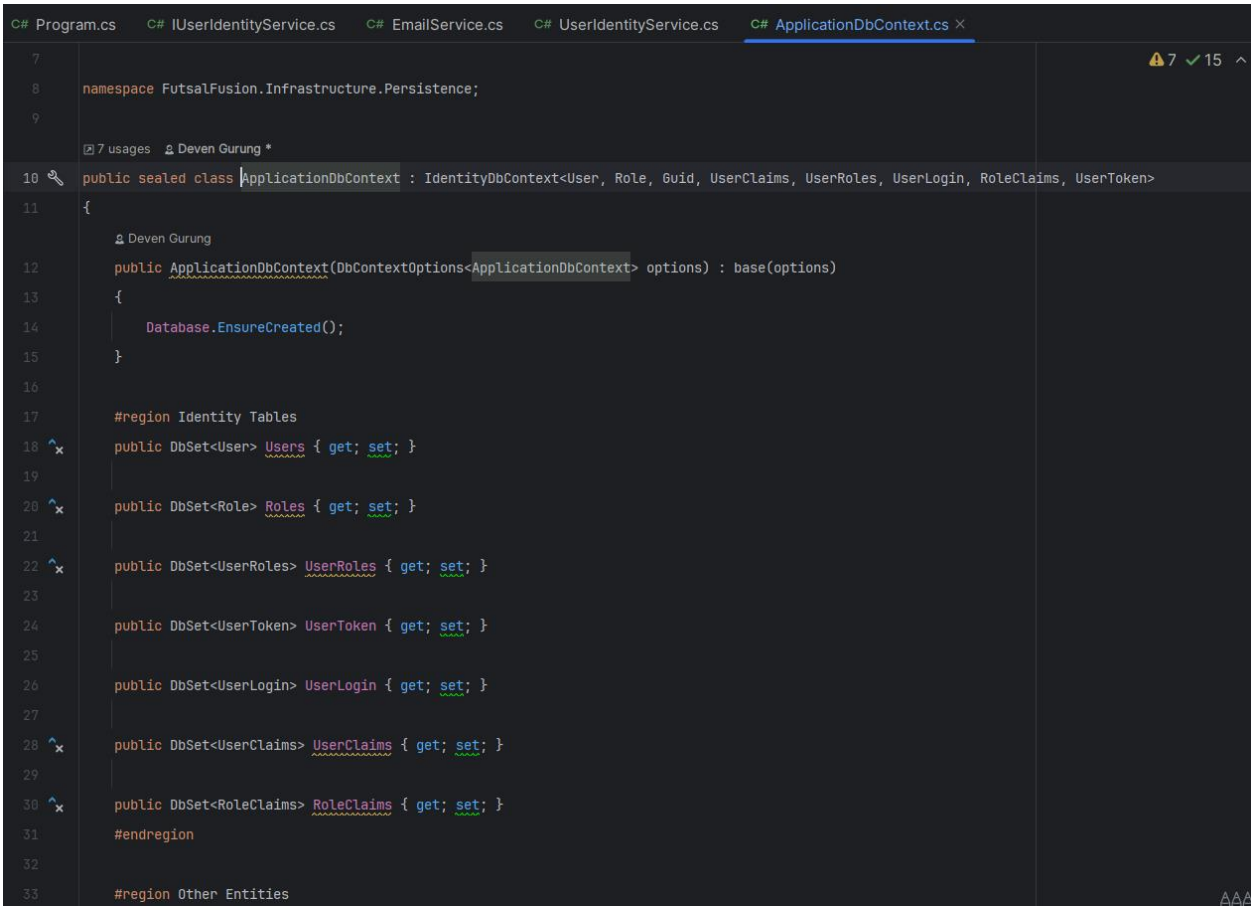


Figure 91: Development: Configuration of DbContext

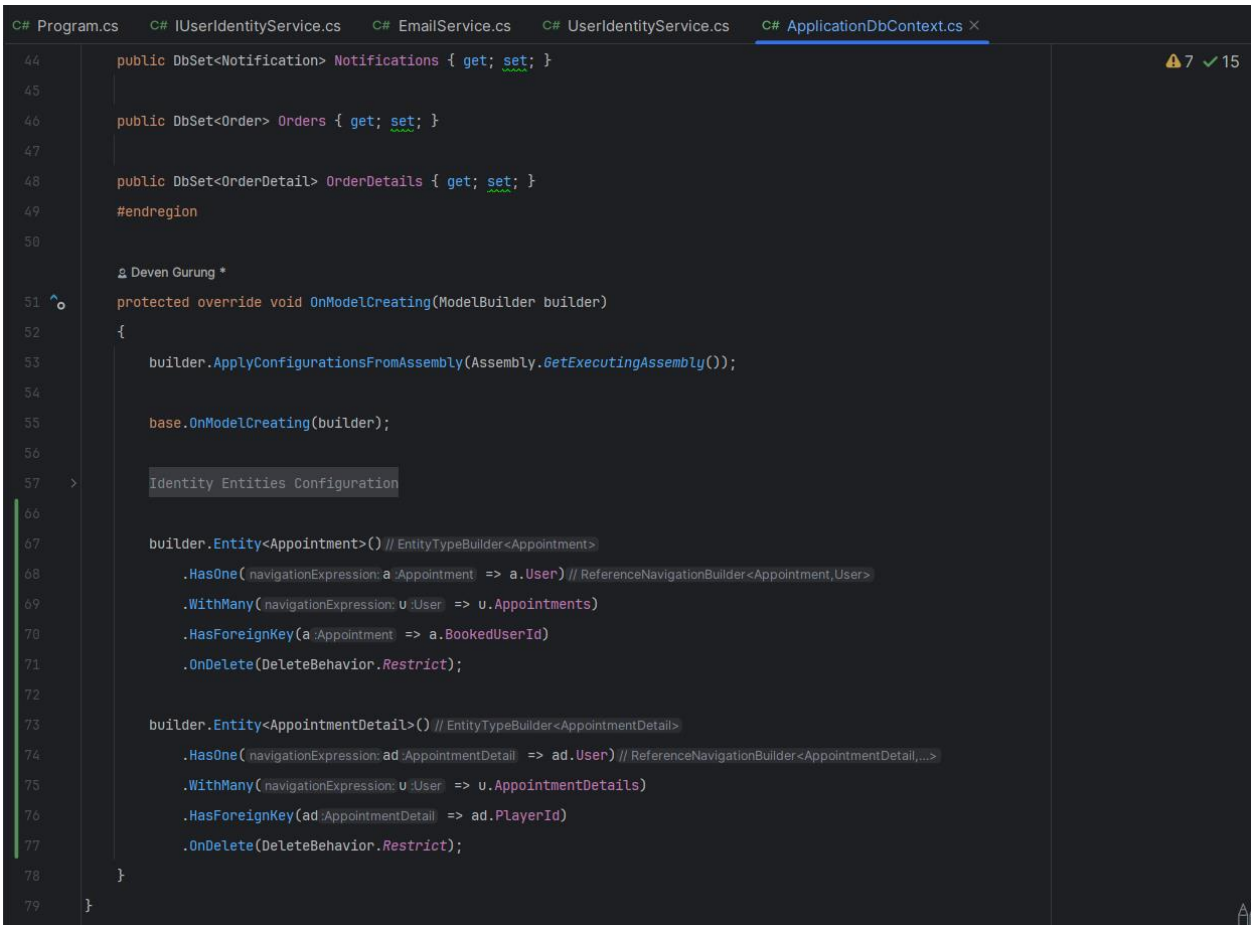


Figure 92: Development: Configuration of Model Binding for Database Setup

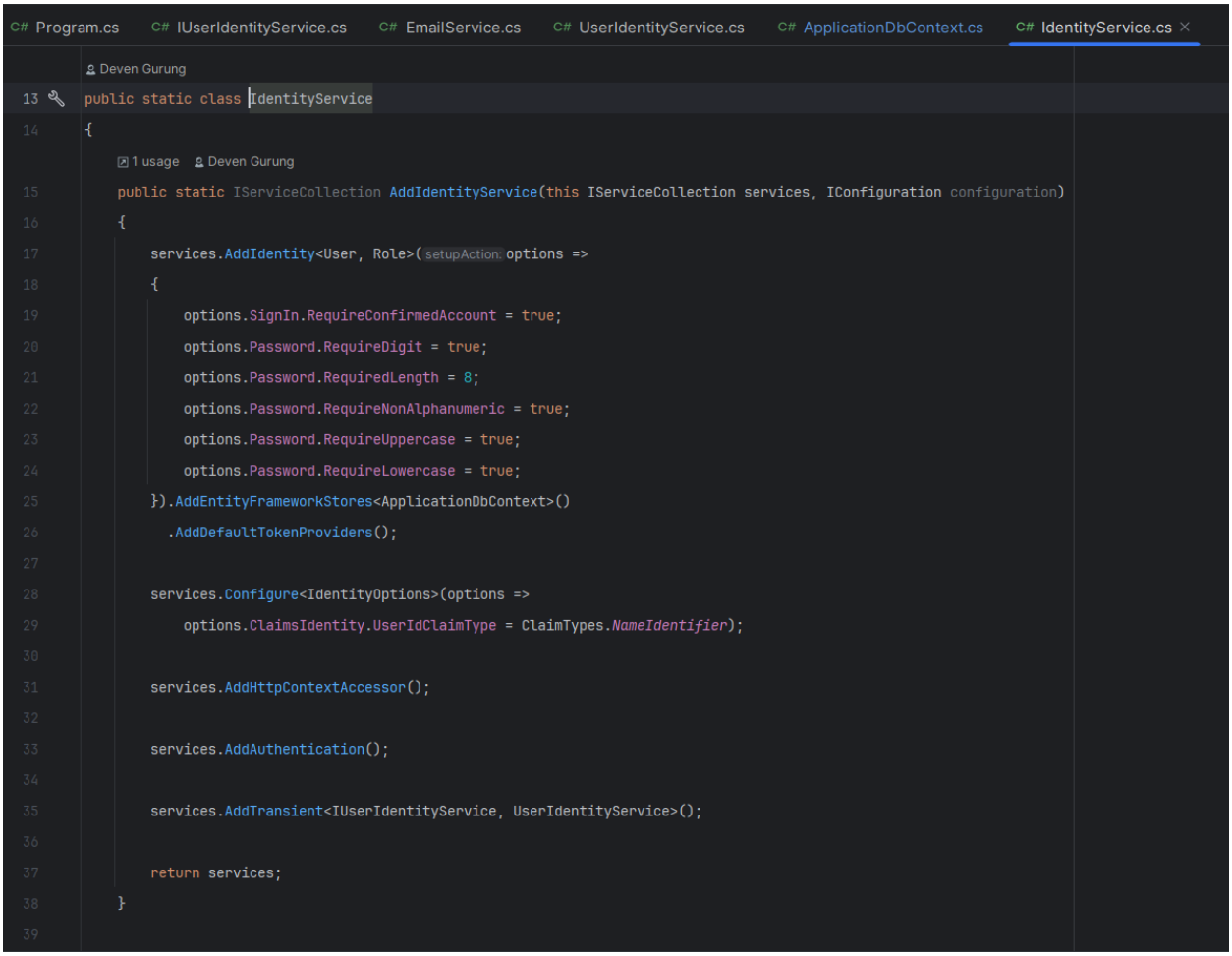


Figure 93: Development: Service Injection through DI

4. Progress Analysis

The following section includes the analysis and inspection done till the project is developed. This helps in determining the actual progress of the project based on development and design as per the Gantt Chart.

4.1. Progress Review

The development of a robust futsal app was agreed upon and the project was initiated in late October when the client settled, and due to its agile and flexible approach, Personal Scrum was selected as the Software Development Life Cycle (SDLC) methodology after deep research and approval from both the supervisors as well as the client. From its inception to the progress made to date, the project has been running smoothly and has lived up to its promise of being both illuminating and educational. In the process of developing projects in real-time, it has offered a platform for a variety of chances to learn and discover new things.

The scrum paradigm dictated that separate sprints should be conducted for the development and design phases. Every sprint is designed to help achieve a certain goal. After each sprint, it was reviewed in detail during the retrospective and client meeting to see what we learned and make any required changes based on that. Since the initial planning sprint was entirely devoted to the design phase, development was disregarded. But a new Gantt Chart was necessary to properly organize the tasks that needed to be finished in each sprint.

Sprint 1 (Project Initiation)

29th October - 10th November

During the first sprint, personal involvement helped to define the project's scope, objectives, and requirements allowing the project to get off the ground. This stage guaranteed conformity with customer requirements and set the stage for succeeding sprints. Following a consensus on a subject, a great deal of investigation was carried out, covering a wide range of topics from analyzing the approaches used in earlier projects to

developing strategies for maximizing efficiency through the application of the agile methodology. The requirements gathering process also included a poll, the findings of which informed the development of the proposed app's unique selling points.

Sprint 2 (Project Planning)

11th November - 24th November

Thorough project planning and assessment were the focus of the second sprint. A Gantt chart was made to show a progression of two-week sprints with defined goals and objectives. The team ensured a defined roadmap for the development process by estimating resources, defining roles, and setting realistic schedules for each sprint. In the next steps, the primary work of developing use cases was completed, and an overview high level description of each use case was written. The SRS Document included equivalent thorough documentation, briefing all users on the system's purpose and operation.

Sprint 3 (Desing)

25th November - 8th December

During the third sprint, the focus was on non-technical elements, such as wireframes, architecture design, Entity-Relationship Diagrams (ERD), and Unified Modelling Language (UML) diagrams. By creating a graphical depiction of the app's structure and functions, this step established the project's design. The system's flow was then followed as it executed its consecutive tasks using the activity and sequence diagrams. We followed the supervisor's instructions to make thorough wireframes, and then we used the Entity Relationship Diagram to build an entire set of entities. This gave us a bird's-eye perspective of all the tables that will have to be constructed and tallied as we go along with the development.

Sprint 4 (Development Cycle 1)

10th December - 22nd December

Having adhered to a course that outlines all of the functionalities and obligations that offered a concerned approach towards what is to be built and how to do it, the design

process has been completed without interruption. The project is now in the building phase of the development phase's coding section. The following have been finished: the architecture, database migration, establishing generic repositories and services, and the construction of the needed model and its properties. Business logic will be built upon the use case diagrams, sequence diagrams, and activity diagrams that were generated during planning stages. These diagrams will be utilized throughout the remaining development.

Sprint 5 (Development Cycle 2)

23rd December - 5th January 2024

The second development cycle has started without a hitch as of this sprint due to getting hands on dirty with codes. The development of a fundamental app framework was guided by thorough study, of the sprint focusing on user actions. This laid the foundations for future features by defining the technical activities for service and manager calls in a.NET application for user authentication and authorization, a fundamental requirement of the app. Research and implementation regarding the application's characteristics are still in the process of implementing an identity server.

Accomplishments and Efficient Implementation

- Progress is closely matching early estimates, demonstrating the effectiveness of the 2-week sprint approach as stated in the Gantt Chart.
- A thorough grasp of client requirements was established at project commencement, which encouraged teamwork in following sprints.
- Thoroughly preparing for the project helped maximize resources and minimize obstacles.
- A clear and aesthetically pleasing futsal app blueprint was produced during the third sprint, which focused on design considerations.
- The development process has started off to a good start, with the necessary features and a basic app structure now implemented.

4.2. Progress Table

A Gantt chart-based progress table was built to effectively monitor and display the status of jobs inside each sprint. In addition to the temporal perspective of project activities provided by a Gantt chart, a progress table provides granular information on the status of tasks and their percentage of completion.

	Task		Estimated Completion	Actual Completion	Status	Progress (%)
1.	**Sprint 1: Project Initiation**		10/11/2023	10/11/2023	Sprint Complete	100%
	1.1.	Client Meeting	29/10/2023	29/10/2023	Complete	100%
	1.2.	Requirement Gathering	05/11/2023	05/11/2023	Complete	100%
	1.3.	Business Case	10/11/2023	10/11/2023	Complete	100%
	1.4.	Proposal Finalization	10/11/2023	10/11/2023	Complete	100%
2.	**Sprint 2: Project Planning**		24/11/2023	24/11/2023	Sprint Complete	100%
	2.1.	Product Backlog	14/11/2023	14/11/2023	Complete	100%
	2.2.	Project Evaluation	16/11/2023	16/11/2023	Complete	100%
	2.3.	Project Scheduling	16/11/2023	16/11/2023	Complete	100%
	2.4.	Risk Assessments	20/11/2023	20/11/2023	Complete	100%
	2.5	Development of Use Cases	24/11/2023	24/11/2023	Complete	100%
3.	**Sprint 3: Design**		08/12/2023	08/12/2023	Sprint Complete	100%
	3.1.	System Architecture	27/11/2023	27/11/2023	Complete	100%
	3.2.	Entity Relationship Diagram	27/11/2023	27/11/2023	Complete	100%
	3.3.	Wireframes and Mockups	03/12/2023	03/12/2023	Complete	100%

	3.4	UML Diagrams	07/12/2023	07/12/2023	Complete	100%
	3.5.	Development Prospect	07/12/2023	07/12/2023	Complete	100%
4.	**Client Approval and Online Survey**		10/12/2023	10/12/2023	Complete	100%
5.	**Interim Progress**		03/01/2024	03/01/2024	Complete	100%
	5.1.	Finalization of Interim Report	03/01/2024	03/01/2024	Complete	100%
6.	**Sprint: 4 - 10: Development Sprint**		In Progress	15/03/2024	In Progress	20%
	6.1.	Development Cycle 1: Repository & Service Implementation	22/12/2023	22/12/2023	Complete	100%
	6.2.	Development Cycle 2: Authentication & Authorization	In Progress	05/01/2023	Partially Complete	70%
	6.3.	Development Cycle 3 - 8	-	15/03/2024	Incomplete	0%
	6.4.	Final Evaluation of Development	-	15/03/2024	Incomplete	0%
7.	**Sprint 11: Testing Sprint**		-	29/03/2024	Incomplete	0%
	7.1.	System Testing	-	26/03/2024	Incomplete	0%
	7.2.	End User Testing	-	26/03/2024	Incomplete	0%
	7.3.	Refactoring and Refining	-	29/03/2024	Incomplete	0%
	7.4.	Test Cases	-	29/03/2024	Incomplete	0%
8.	**Sprint 12: Closure**		-	19/04/2024	Incomplete	0%
	8.1.	Deploy and Monitoring	-	05/04/2024	Incomplete	0%
	8.2.	Final Client Meetup	-	08/04/2024	Incomplete	0%
	8.3.	Review and Refining	-	12/04/2024	Incomplete	0%
	8.4.	Finalized Report Structure Content	-	14/04/2024	Incomplete	0%

	8.5.	System Finalization	-	19/04/2024	Incomplete	0%

Table 8: Progress Table: Status and Completion

4.3. Progress Timeline

The Gantt Chart provided in the supplemental or appendix section has been used to manage the project's tasks and progress. The design sprint's requirements were met, and the project's design phase is now complete. The development sprint is under process with the cycle's first action completed. Also, the Gantt Chart has been revised so it can potentially reveal unanticipated avenues for growth and all the necessary steps for implementing it in detail have also been planned and precise accordingly. The revision and breakdown description of the project tools can be found in the appendix report. Link to Appendix.

5. Future Work

Examining the specifics of the project's forthcoming timetable is the purpose of this section. Approximately 40% of the total development has been completed as of the writing of this report. The remaining 60% of the project is comprised of the following critical tasks:

System Finalization

There are still a lot of moving parts in the backend and frontend code bases as the development process moves forward. Nevertheless, a visual depiction of the proposed design would offer a concise synopsis of the execution of business logic in addition to all other key features and specifications.

Task	Date Range	Actual Date
Continue Frontend and Backend API Development	Ongoing	Ongoing
Diagrammatic Representation of the application	Ongoing	Ongoing

Table 9: Future Work: System Finalization

Quality Assurance / Control and Test Cases

The last testing session occurs after all development sprints have concluded, and it is during this session that the system is tested extensively for both functionality and user experience.

Task	Date Range	Actual Date
Unit Testing	After Development Completion	Alignment with the development cycle
System Testing	After Unit Testing	17th - 26th March 2024
End User Testing	After System Testing	24th - 26th March 2024
Test Cases	After UAT	27th - 29th March 2024

Table 10: Future Work: QA and Test Cases

System Deployment and Monitoring

After the project's completion in the development and testing stages, it's necessary to state that everything was well planned and executed. To start this project as a real answer to the current sports facility system, however, deployment to a staging server is necessary so that it is set out to its live platform and can help the client reach to its respective prospects.

Task	Date Range	Actual Date
Deploy the system to the server	After Testing Sprint	1st April 2024
Monitor System Behavior and Performance	After Testing Sprint	1st - 5th April 2024
Trace API Calls & Investigate Issues	After Testing Sprint	1st - 5th April 2024
Apply Optimization and Bug Fixes	After Testing Sprint	1st - 5th April 2024

Table 11: Future Work: Deployment and Monitoring

Review and Refinement

After the system is brought out live and based on input from both the supervisor and the client, a final polish will be applied to make the project even better for everyone.

Task	Date Range	Actual Date
Collect and incorporate feedback	After Deployment	Ongoing
Make final adjustments to enhance the project	After Deployment	Ongoing

Table 12: Future Work: Review and Refinement

Finalization of Final Report

Along with a focus on development, this method follows stringent reporting standards that record every step of the development process, from brainstorming to final delivery, including any and all lessons learned, new paths investigated, and similar information. In order to ensure that the project's enquiry has tangible results, all artefacts found will be collected and preserved. In this way, the final report will complement the module's comprehension while also wrapping up the system according to the provided criteria.

Task	Date Range	Actual Date
Follow London met reporting standards	After Review and Refinement	31st March – 18th April
Document Finalization	After Review and Refinement	31st March – 18th April

Table 13: Future Work: Final Report

System Finalization

The system will be finally brought to its final foundation after much research, consideration, struggle, and work have gone into analyzing all of its investigations and the challenges it has overcome.

Task	Date Range	Actual Date
Conclude the system based on findings	After Final Report Finalization	Ongoing - 15th April
Ensure alignment with specified criteria	After Final Report Finalization	Ongoing - 15th April
Enhance understanding of the module	After Final Report Finalization	Ongoing - 15th April

Table 14: Future Work: System Finalization

6. Bibliography

Alliance Software, 2023. *An Introduction To Software Development Methodologies.* [Online]
Available at: https://www.alliancesoftware.com.au/introduction-software-development-methodologies/
[Accessed 18 December 2023].

Atlassian, 2023. *Product Backlog - What Is It & How to Create One.* [Online]
Available at: https://www.atlassian.com/agile/scrum/backlogs
[Accessed 20 December 2023].

Britannica, 2022. *Architecture.* [Online]
Available at: https://www.britannica.com/topic/architecture
[Accessed 10 December 2022].

Cambridge, 2023. *Methodology | English Meaning.* [Online]
Available at: https://dictionary.cambridge.org/dictionary/english/methodology
[Accessed 18 December 2023].

Crystalloids, 2022. *Working in the Scrum.* [Online]
Available at: https://www.crystalloids.com/about-us/how-we-work
[Accessed 13 November 2022].

Digite, 2022. *What is Kanban?.* [Online]
Available at: https://www.digite.com/kanban/what-is-kanban/
[Accessed 17 November 2022].

Drumond, C., 2022. *Scrum - what it is, how it works & why it's awesome.* [Online]
Available at: https://www.atlassian.com/agile/scrum
[Accessed 10 November 2022].

Forbes, 2023. *What is Work Breakdown Structure?.* [Online]
Available at: https://www.forbes.com/advisor/business/what-is-work-breakdown-structure/
[Accessed 20 December 2023].

Geeks for Geeks, 2022. *Software Engineering: Rapid Application Development Model (RAD).* [Online]
Available at: https://www.geeksforgeeks.org/software-engineering-rapid-application-

development-model-rad/
[Accessed 15 November 2022].
Geeks for Geeks, 2023. *Software Engineering: Rapid application development model (RAD).* [Online]
Available at: https://www.geeksforgeeks.org/software-engineering-rapid-application-development-model-rad/
[Accessed 8 November 2023].
Hannah, J., 2023. *What Exactly Is Wireframing?.* [Online]
Available at: https://careerfoundry.com/en/blog/ux-design/what-is-a-wireframe-guide/
[Accessed 20 December 2023].
IHRSA, 2022. *IHRSA Media Report: Health and Fitness Consumer Data & Industry Trends,* Boston, MA: IHRSA.
Interview Bit, 2022. *System Architecture.* [Online]
Available at: https://www.interviewbit.com/blog/system-architecture/
[Accessed 10 December 2022].
Janse, B., 2022. *The Rational Unified Proces Methodology (RUP).* [Online]
Available at: https://www.toolshero.com/information-technology/rational-unified-process-rup/
[Accessed 6 November 2022].
Kahwaji, A., 2020. A Study of Customer Satisfaction Dimensions and Impact on Customer Loyalty. 6(1), pp. 1042-1054.
Kissflow, 2022. *Rapid Application Development (RAD) Model: An Ultimate Guide For App Developers in 2022.* [Online]
Available at: https://kissflow.com/application-development/rad/rapid-application-development/
[Accessed 6 November 2022].
Kissflow, 2023. *Rapid Application Development (RAD) Model.* [Online]
Available at: https://kissflow.com/application-development/rad/rapid-application-development/
[Accessed 8 November 2023].
Lucid , 2023. *Mind Map Diagram.* [Online]

Available at: https://www.lucidchart.com/pages/examples/mind_mapping_software
[Accessed 20 December 2023].
Lucid Chart, 2023. *Block Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/examples/block-diagram-maker
[Accessed 20 December 2023].
Lucid Chart, 2023. *Collaboration Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/uml-communication-diagram
[Accessed 20 December 2023].
Lucid Chart, 2023. *UML Activity Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/uml-activity-diagram
[Accessed 20 December 2023].
Lucid Chart, 2023. *UML Class Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/uml-class-diagram
[Accessed 20 December 2023].
Lucid Chart, 2023. *UML Sequence Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/uml-sequence-diagram
[Accessed 20 December 2023].
Lucid Chart, 2023. *UML Use Case Diagram.* [Online]
Available at: https://www.lucidchart.com/pages/uml-use-case-diagram
[Accessed 20 December 2023].
Lucid Chart, 2023. *What is a data flow diagram?.* [Online]
Available at: https://www.lucidchart.com/pages/data-flow-diagram
[Accessed 20 December 2023].
Obergfell, Y., 2023. *The Scrum Product Backlog - International Scrum Institute.* [Online]
Available at: https://www.scrum-institute.org/The_Scrum_Product_Backlog.php
[Accessed 20 December 2023].
Online Khabar, 2023. *Futsal in Nepal: Amid hurdles, growing up slow and steady as a business.* [Online]
Available at: https://english.onlinekhabar.com/futsal-in-nepal-growth-challenges.html
[Accessed 5 November 2023].
Quick Start, 2022. *Pros and Cons of Scrum Methodology.* [Online]

Available at: https://www.quickstart.com/blog/pros-and-cons-of-scrum-methodology/
[Accessed 8 December 2022].
Scrum, 2023. *What is a Product Backlog?.* [Online]
Available at: https://www.scrum.org/resources/what-is-a-product-backlog
[Accessed 20 December 2023].
Shore, J., 2021. *The Art of Agile Development.* 2nd ed. s.l.:O'Reilly Media, Inc..
Siderova, S., 2022. *The Kanban Method: The Ultimate Beginner's Guide.* [Online]
Available at: https://getnave.com/blog/what-is-the-kanban-method/
[Accessed 6 November 2022].
Study, 2021. *What is the Rational Unified Process? Methodology, Tools & Examples.* [Online]
Available at: https://study.com/academy/lesson/what-is-the-rational-unified-process-methodology-tools-examples.html
[Accessed 16 November 2022].
Sutherland, J. & Schwaber , K., 2020. *The Scrum Guide.* 1st ed. s.l.:Creative Commons.
Tech Sathi, 2023. *Vakundo App: Book Futsal Venue, Join Team or Challenge Opponent.* [Online]
Available at: https://techsathi.com/vakundo-app
[Accessed 18 December 2023].
UC Merced Library, 2023. *What Is a Data Dictionary?.* [Online]
Available at: https://library.ucmerced.edu/data-dictionaries
[Accessed 18 December 2023].
WePlay Nepal, 2023. *WePlay Nepal | Online Futsal Booking App in Nepal.* [Online]
Available at: https://www.weplaynepal.com/
[Accessed 18 December 2023].
Zacharias, C. & Pinedo, M., 2020. Appointment Scheduling with No-Shows and Overbooking. *Production and Operations Management,* 23(1), pp. 54-61.

7. Appendix

A report would not be complete without an appendix, a supplementary part that can hold

supplementary materials such as further data, information, or documents. Its significance is emphasized by giving readers access to detailed information without overwhelming the report's primary body. To keep the main narrative of the report from getting overwhelmed, the following appendix section contains extra material that stays focused and brief.

7.1. Problem Context

Issues with futsal infrastructure and appointment slot booking do arise in the technical world of sports, reflecting broader concerns about accessibility, the state of facilities, and the necessity for inclusive, tech-driven solutions.

7.1.1. Nepal

There are problems with the sports infrastructure in general, and the growing popularity of futsal in Nepal is no exception. The lack of accessible and well-kept futsal courts is a continuing worry in an industry where the demand for such facilities is on the rise. The necessity for effective booking systems is heightened in metropolitan settings due to the high demand and limited availability. The entire potential of futsal in the country is unfortunately retarded by outmoded managerial approaches, inadequate promotion, and a lack of financial investment.

A growing number of Nepalese are taking an interest in sports like futsal, but current polls show that just a small percentage of the country's sports facilities are up to scratch. Furthermore, many sports fans are unhappy with the way things work when they book, so there's a pressing need for more efficient alternatives that make use of technology. To overcome these obstacles, the government, facility management, and sports authorities must work together to improve infrastructure, create more accessible booking systems, and encourage a more inclusive and sports-minded society. Nepal can fully develop its sports environment and tap into the potential of futsal by allocating resources to these sectors.

7.1.2. Global Context

Challenges in futsal and sports infrastructure in Nepal are representative of bigger

difficulties in many nations around the world. A frequent topic is accessibility concerns, which arise when the demand for high-quality sporting facilities exceeds the available resources. The exponential growth of futsal's popularity in many major cities around the world has put pressure on already overburdened infrastructure and highlighted the urgent need for efficient scheduling and administration tools.

It is statistically evident that many sporting facilities do not exceed international standards, highlighting the need for improvements and modernization on a global scale. Because of inequalities in resources, the international sports community has a hard time fostering equality. The incorporation of user-friendly technologies for the booking of sports facilities is a challenge for many nations in view of recent technology developments.

Governments, sport’s governing organizations, and facility administrators must work together to tackle these concerns on a worldwide level. Challenges such as investing in modern infrastructure, raising sports awareness, and streamlining booking processes are not unique to Nepal; they affect people all over the world. The global community may work together to improve people's health, fitness, and happiness by recognizing and resolving these common issues, creating a more welcoming and accessible sports environment.

7.2. Problem Solving Measures

Innovations in technology have become powerful tools for addressing problems in the dynamic field of sports management. This is especially true when it comes to scheduling appointments. There has been a sea change in the ease of making reservations for sports fans due to the proliferation of intuitive booking apps. There has been a marked improvement in the number of successful bookings, with fewer cases of misunderstanding and obstacles, according to recent statistics.

By giving users access to availability data and real-time updates, these apps do more than just make scheduling easier; they also increase transparency. Both players and facility administrators have benefited greatly from the increased accessibility and

organization brought about by the widespread use of technology in sports administration, which has reduced logistical hurdles and made the whole process much smoother.

7.3. Technology and Stack Implementation

Once the components of a project's stack have been brought together, the first question raised by the team is why they've chosen to use that stack rather than its alternative technologies. Accordingly, this section defines the stacks that have been selected for the construction of the application.

Why web application over a mobile app?

Considering that this system is meant to depict a structured management overview, a web app would be preferable to a mobile app because of its superior layout, user experience, and distribution system. Users prefer a non-clustered app, such as those provided by online apps, to book appointments and keep track of their entire booking history and details. Further on, no matter what operating system a user has, either a desktops, laptops, tablets, and smartphones, web apps may be accessed from any device with internet connectivity. This broadens its appeal and increases its adaptability and also provides the ability to centralize data storage, guaranteeing that all users may get the most current information instantly, as the system also focuses on data security and tracking as an important feature. Given the dynamic nature of sports scheduling, player profiles, and match outcomes, this is of paramount importance for any sports management app. Also, web applications may make it easier for consumers to navigate and engage with the app by providing a consistent user experience across various devices increasing user engagement and happiness.

Why do you prefer .NET than its respective counterparts?

To move forward with the web app's construction, I've decided to create a cross-platform web app in a.NET environment due to its advanced programming features, such as memory management, asynchronous programming, and deployment flexibility that optimizes cloud-based services; these features are what convinced me to go with this approach. The framework's track record of success with large-scale, resource-intensive applications, as well as its performance and community support, make it a top contender for use in creating the suggested app. The databases that need to be synchronized with the program are Microsoft SQL Server Studio and PostgreSQL.

Both offer a lightweight, easily integrated data storage platform that is flammable. Since both Entity Framework Core on ORM and Dapper are incredibly fast and time-efficient, and ultimately offer the best balance between built-in capability and read/write performance, the well-documented study is useful for accessing the database with either.

7.4. SRS Documentation

A software requirement specification, also termed as an SRS, is a document that outlines the features and functionality that a software is expected to provide. In addition, it specifies the features the product must have to satisfy the requirements of all relevant parties for the business and the users. In accordance with the SRS protocol document, the following document will provide a synopsis of the proposed software's four D's which are outlined below

- Define the product’s purpose
- Describe what is being built
- Detail each and every requirement
- Deliver all of them for approval

Revolving around all these conventions, each of them will be thoroughly explained to describe exactly what the system is about and what it aims in achieving at its completion.

7.4.1. Purpose

Project Title: Futsal Fusion: Futsal Scheduling & Booking App

Project Category: Web Application

The purpose of this document is to provide a thorough explanation of the system's functionality, including the definition of every characteristic, as well as its non-functional aspects, such as the user experience, system capabilities and constraints, the interfaces it supports, and other technical dependencies.

7.4.1.1. Intended Audience

This document serves as a guide for the system's developers as they build out the various features called for by the system. QA engineers can use this document to have their perspective highlighted into the product's vision and test accordingly.

Anyone from the project's stakeholders to the end users is welcome to read this document to gain insight into the project's goals and objectives.

7.4.1.2. Project Scope

It is obvious that there is an enormous need for a futsal management system in our county, and to satisfy it, we will need to completely revamp the current system. The main objective of the software is to allow users to book appointments on their preferred futsal court using an online platform. This platform also includes digital means of making payments, calculating revenue, and keeping track of everyone's records. The system is built with state-of-the-art security mechanisms, guaranteeing user privacy and compliance with all requirements. Eliminating the need for vulnerable paper records, the system digitizes all transactions and records and makes them securely accessible through a cloud-based system. Using an iterative approach that respects the client's specifications and gets their approval at each step, the project will be completed within the specified time.

7.4.1.3. Existing System

The current approach is entirely manual and entails a great deal of paperwork in the form of a schedule to keep track of all appointments. Keeping records of each daily transaction can be a tedious and pricey chore, not to mention it being an imprecise and inefficient approach. Here are some drawbacks of the old-school pen-and-paper method:

- In addition to being a time-consuming operation, the manual system's limitations and it being non-user friendly makes it difficult to utilize.
- Payment and scheduling records under the current system must be stored in a record file, which can be lost or damaged.
- Due to the inherent insecurity of a manually operated system, it is possible that important information could be lost if not handled properly.

7.4.1.4. Proposed System

The primary goal of the proposed system is to provide a user-friendly web application for the futsal group owners, staff and end users or the players involved to replace the present traditional pen-and-paper technique way of prescribing and appointing services on a futsal arena. The following are some of the potential benefits that a digital record system could bring to each of the associated actors:

Advantages for Futsal Courts / Facility Owners

- Futsal courts can easily provide different kinds of kits and supplies to be purchased by the players.
- Futsal courts can easily view all the current booking slots by the players and handle to either approve or reject it, and process on cancelation of the booked slot.
- Futsal courts can easily view over all the players' details, their booking history on the respective futsal and total revenue generated.
- Futsal courts can easily proceed with the request to fill in missing player(s) during an appointment by notifying all the registered users with the details of the appointment.

Advantages for Players

- Players can easily view the list of registered futsal and book their appointments accordingly.
- Players can easily request availability of players during an absence of a (some) players to fill the void of the missing player, which will later be processed by the respective futsal facility owners.
- Players can perform purchases and transactions based on online payment or offline medium with all the records stored in their respective accounts.

Advantages for Admin

- Admin can easily perform addition of any futsal courts along their respective details and the facility owner.
- Admin can easily handle any help queries raised by the facility owner of the futsal.

7.4.2. System Perspective

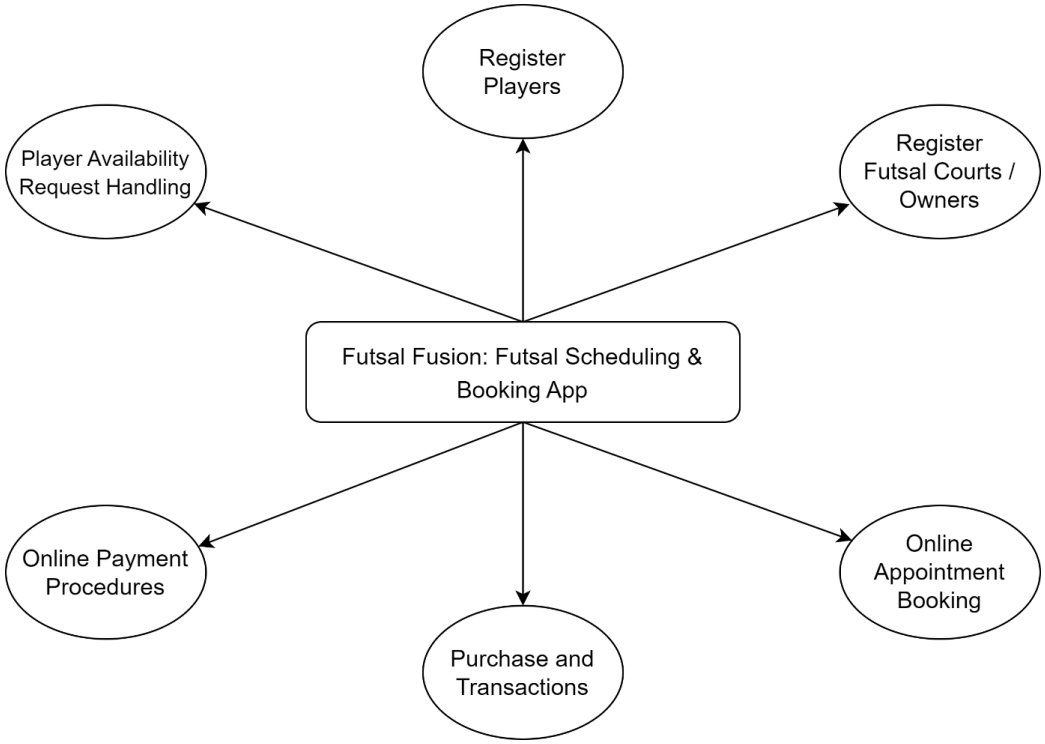


Figure 94: SRS: System Perspective

Brief descriptions of the features of the system are provided below, with reference to the figure describing the system perspective that depicts them:

Register Players

A user can sign up for the service by giving the application their personal information and any credentials they plan to use for making appointments and making transactions.

Register Futsal Courts / Owners

Admins are responsible for registering new futsal courts to be made available to the end users or the players while shortlisting their details such as images, names, descriptions, prices and so on. Further on, while registering at a futsal court, a spokesperson or the owner will also be registered so as they can handle all the other procedures for booking appointments and purchase handling.

Online Appointment Booking

Once logged in, players are shown a list of available futsal courts, organized by their location details, and given the option to schedule an appointment with one of them at a time and day of their choosing. The player is then required to register all the players to be involved in the following booking slot for further information tracking.

Purchase and Transactions

As the system also provides the futsal owners to provide their own list of items such as gears, kits, boots, and other items for purchase during the futsal matches, the registered players can browse through all the item of choice and proceed with the transaction and purchases which will all be tracked down by the system along with the necessary details such as ordered and payment status and so on.

Online Payment Procedures

Furthermore, during purchase of items and kits, the players will also have access to an online payment medium through which they can pay for the items on an online basis through payment gateways.

Request Player Availability Handling

One of the major features to be handled in the following app is to pre-notify a list of

available registered players to fill in a void of a missing player during a slot booking. The feature is made available to the players who have booked a particular slot on a futsal ground and will thus provoke the registered user's notification area triggered by the request which will be invoked by the players but ultimately handled by the futsal owners themselves.

7.4.3. User Class and Characteristics

The following application will be used by the major three actors: players, owners and admin. These users' designated functions are outlined below.

UC 1: Players

UC 1.1. Can register themselves to the system

UC 1.2. Can log in to the system via valid credentials

UC 1.3. Can view the list of available futsal and book an appointment

UC 1.4. Can proceed to pay through credit points or an offline medium

UC 1.5. Can request for availability of players during an absence of a (some) players

UC 1.6. Can view all the futsal bookings based on the appointment

UC 2: Futsal Courts / Facility Owners

UC 2.1. Can provide different kinds of kits and supplies for commercial purposes

UC 2.2. Can log in to the system once the administrator approves their request

UC 2.3. Can view the list of the current booking slots being referred by the players

UC 2.4. Can handle the bookings by either approving or rejecting the application

UC 2.5. Can view over all the registered player's details and total revenue generated

UC 3: Admin

UC 3.1. Can approve any incoming requests for the post of technical staffs

UC 3.2. Must handle the queries raised by the facility owners

7.4.4. Operating System

OE 1. The proposed system application is intended to reach its end users through a web application which is supported by most of the web browsers

7.4.5. Assumptions and Dependencies

AS 1. All of the user's personal data are to be accessible by the client organization

7.4.6. Design and Implementation Constraints

CO 1. The application can not be executed on a platform without an internet connection

CO 2. The application will not be reliable on a device with a small screen size

7.4.7. Functional Requirements

The service that the program must provide is defined in what is called a Functional Requirement. It provides information about a piece of software or an entire system. Simply described, a function is the combination of the system's inputs, its behavior, and the results it produces. A system's expected functionality can be defined by a calculation, data manipulation, business process, human interaction, or any other unique functionality. An outline regarding each and every function of the system along with its descriptions and requirements are thus explained below:

Players Registrations

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 1.</td><td>A general player can sign up to access the system’s functionalities.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>The user views the registration form and enters all their personal details and credentials.</td></tr>
<tr><td></td><td>The system then validates whether all the required areas are filled with data or not.</td></tr>
<tr><td></td><td>The system then validates all the user input and then sends a confirmation email to the user to confirm their registration.</td></tr>
<tr><td></td><td>Once the registration link is clicked, the system then takes the user to a successful registration page, else displays an area of error.</td></tr>
</table></td><td>High</td><td>High</td></tr>
</table>

Table 15: Functional Requirement: Player Registrations

Admin Roles for Futsal Grounds Activities

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 2.</td><td>It is the sole responsibility of the system’s admin to look after all the futsal grounds and perform any superior actions towards any of the futsal grounds.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>As there is no provision of requests processing for registration of a futsal ground, admin directly registers the futsal court via the registration form and enters all of required personal data.</td></tr>
<tr><td></td><td>As the process is handled, a futsal court is included in the active session of available slots for booking for players and similarly a main futsal owner is also registered simultaneously who have all the rights towards their respective futsal.</td></tr>
<tr><td></td><td>The super admin of the system or the admin of the futsal court now has the responsibility to alter any changes to the description of the futsal, alter prices on booking slots, modify the rules and regulations, add or remove any items for purchases and so on handle any incoming booking requests or provision for any player availability.</td></tr>
<tr><td></td><td>The super admin has access to all of</td></tr>
</table></td><td>High</td><td>High</td></tr>
</table>

		the registered futsal grounds and courts, however the admin or the spokesperson of a futsal court can access only their respective futsal court's features and availability at a time.		

Table 16: Functional Requirement: Futsal Registrations

Login (Admin / Futsal Owners / Players)

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 3.</td><td>The feature includes as one of the major functionalities for each set of users to access the system.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>User can view the login form from which they input all their necessary credentials.</td></tr>
<tr><td></td><td>The system shall validate on each value being entered. It should manipulate the username and password in a secured way.</td></tr>
<tr><td></td><td>For an invalid credential, the system generates an unsuccessful login.</td></tr>
<tr><td></td><td>However, for a successful login, the system then takes the user to their desired home page described by their role.</td></tr>
<tr><td></td><td>Each set of users have their own privileges granted to them, so the logged in user can only access the pages to which they have been authorized with.</td></tr>
</table></td><td>High</td><td>High</td></tr>
</table>

Table 17: Functional Requirement: Login

Profile and Account

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 4.</td><td>All the logged in users can view their profile and account and manipulate it accordingly.<table><tr><th colspan="2">System Requirement</th></tr><tr><td></td><td>Once the user is logged in to the system, they view a profile item on the navbar.</td></tr><tr><td></td><td>Once navigated to that page, they view all of their account details and their respective information.</td></tr><tr><td></td><td>The user may change their password or enable any sorts of two factor authentication through the profile tab</td></tr><tr><td></td><td>Similarly, the user based on the roles can thus access their records of either slots booking, item purchases, futsal appointments or revenue generation, all distinguished by their account settings and role defined.</td></tr><tr><td></td><td>However, the super admin can overview all of these details for an individual player or respective futsal owners.</td></tr></table></td><td>High</td><td>High</td></tr>
</table>

Table 18: Functional Requirement: Profile and Account

Online Futsal Slot Booking

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 5.</td><td>Registered players can schedule an appointment with the futsal court of preference.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>Once a user is registered and logged in to the system, it views over a slot booking tab on the navbar.</td></tr>
<tr><td></td><td>Once navigated, the user views the list of all available futsal courts and their enlarged details.</td></tr>
<tr><td></td><td>The user may filter the futsal courts accordingly such as by their specialty.</td></tr>
<tr><td></td><td>Once a futsal court is chosen for the booking, the user then schedules a slot with the chosen court on a given time frame at a particular date.</td></tr>
<tr><td></td><td>However, the approved appointment date and time should not collide with any other user’s booking slots.</td></tr>
<tr><td></td><td>When the appointment is scheduled, the user then requires selecting all the registered players and fill in details for number of players viable for the slot.</td></tr>
<tr><td></td><td>The user can also request for an availability of registered players and then choose to pay for the appointment now through credits or later after the appointment by cash. However, all these status will be fully</td></tr>
</table></td><td>High</td><td>High</td></tr>
</table>

		tracked by the system.		

Table 19: Functional Requirement: Online Futsal Slot Booking

Request for Availability of Registered Players

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 6.</td><td>As a futsal court’s slot is appointed, it is also possible for the players to not have a complete count for players to fill in the slot, which will be addressed by the following feature.
<table><tr><th colspan="2">System Requirement</th></tr><tr><td></td><td>Once the booking is confirmed and slot is made available to the registered player’s records, the player can thus then also request for an availability of player(s) to fill in the missing void on the absence of complete players.</td></tr><tr><td></td><td>The request includes processing of notifications to the registered players involved in that futsal court.</td></tr><tr><td></td><td>Further on, once the notification has been triggered and any other player has responded to the request, it is then the responsibility of the futsal owner to process the request and inform the respective registered player who initially booked the slot.</td></tr></table></td><td>High</td><td>High</td></tr>
</table>

Table 20: Functional Requirement: Player Availability Request

Item Purchases

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 7.</td><td>A player can order items and kits through an online medium or purchase it directly from the futsal’s premises, with complete record and status tracking.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>The futsal’s or facility owner can preview all the items such as futsal kits and gears to be made available for commercial purposes.</td></tr>
<tr><td></td><td>The players can then view these details and proceed on with the purchase of their selected items of choice.</td></tr>
<tr><td></td><td>The ordered and transaction details are also stored on the system’s administrative and record tracking procedure such that the facility owner’s can view their product sales and users can filter out their purchases.</td></tr>
<tr><td></td><td>All of the records will be completely tracked along with their respective details and statuses.</td></tr>
</table></td><td>High</td><td>High</td></tr>
</table>

Table 21: Functional Requirement: Item Purchases

Booking Slots and Transaction Records

<table>
<tr><th>Req. ID</th><th>Requirement Description</th><th>Priority</th><th>Complexity</th></tr>
<tr><td>FR 8.</td><td>The system also focuses on complete record tracking for closure and clarifications, including either court’s booking slots or transaction process.
<table>
<tr><th colspan="2">System Requirement</th></tr>
<tr><td></td><td>Once a futsal has been booked over, the respective details of the players and addition of available players are also well handled, and all these details are completely tracked by the system.</td></tr>
<tr><td></td><td>The system notes down every action from start of booking till the closure of the appointment ensuring clear and concise data flow between futsal owners and the players as well.</td></tr>
</table></td><td></td><td></td></tr>
</table>

Table 22: Functional Requirement: Booking Slots and Transaction Records

7.4.8. External Interfaces Required

Once all the features and requirements have been described, the development should be based on the following.

7.4.8.1. User Interface

The application's user interface (UI) will be designed with the end user in mind to eliminate any confusion over page selection and navigation. Each page will have an appropriate color scheme and a name that reflects its purpose. Images will also be preserved in the program in accordance with sites to give consumers the most relevant data possible. In order to build the presentation's front-end views, wireframes will first be instantiated to provide the presentation's structure and layout.

7.4.8.2. Hardware

- The system is dependent upon a web application.
- The system is applicable on any device with an internet connection.

7.4.8.3. Software

- **Frontend (Client) Programming Language:** Razor Pages, CSS, JavaScript
- **Frontend (Client) Frameworks:** Bootstrap, AJAX, External Libraries
- **Frontend (Client) Development Tools:** Visual Studio Code, IntelliJ Rider

- **Backend (Server) Programming Language:** C#
- **Backend (Server) Frameworks:** ASP.NET Core 6
- **Backend (Server) Development Tools:** IntelliJ Rider, Postman

- **Database Server:** Microsoft SQL Server Management Studio, PostgreSQL

7.4.8.4. Communication Protocol

The necessary interaction between the database and the web app is achieved through

HTTP requests and their corresponding responses.

7.4.9. Non-Functional Requirements

The quality attribute of a software system is defined by the Non-Functional Requirement. They evaluate the software based on criteria that aren't directly related to its functionality but are nonetheless essential to its success. Systems that fall short of users' expectations may have been built without sufficient attention to non-functional criteria. A few of the system attributes to be justified are outlined below

Reliability

Attribute ID	Requirement Description	Priority	Complexity
R.1.	Opening the application shouldn't result in a crash.	Maximum	High
R.2.	It's unacceptable for the application to respond slowly.	Maximum	Normal

Table 23: Non-Functional Requirement: Reliability

Security

Attribute ID	Requirement Description	Priority	Complexity
S.1.	There should be no disruptions in the transmission of information.	Maximum	High
S.2.	Only the authorized and verified users should be able to access the system.	Maximum	Normal

Table 24: Non-Functional Requirement: Security

Maintainability

Attribute ID	Requirement Description	Priority	Complexity
M.1.	The program shouldn't pose any sort of upkeep risk in times for maintenance.	Maximum	High
M.2.	Future programmers should be able to understand the code and use the application with such little difficulty.	Maximum	Normal

Table 25: Non-Functional Requirement: Maintainability

Performances

Attribute ID	Requirement Description	Priority	Complexity
P.1.	The application should result to boosted up performance in terms of UI/UX	Maximum	High

Table 26: Non-Functional Requirement: Performances

Availability

Attribute ID	Requirement Description	Priority	Complexity
A.1.	The application should be readily available and accessible to any device connected with an internet connection.	Maximum	High

Table 27: Non-Functional Requirement: Availability

7.5. Product Backlog

Based on the needs and requirements outlined in the roadmap, the development team creates a prioritized list of tasks called a product backlog. The product backlog is organized with the most critical items at the top, so the team understands what to deliver first. The product backlog is built upon the requirements and team's roadmap (Atlassian, 2023).

At the sprint planning event, the team is ready to choose which items from the backlog can be completed in a single sprint. It is common for them to achieve this level of openness following the refining process. Refining entails reducing the items in the list to more manageable, specific components. Further on, the product goal is emphasized, which outlines the desired product state in the future, as a benchmark for their planning efforts. Finally, after the product goal is defined, the remaining items in the product backlog come into play (Scrum, 2023).

Scrum Product Backlog, in its most basic form, is just a list of everything that has to be done as part of the project. It is meant to serve as a replacement for the standard artifacts used to specify requirements. These things can be purely technical or user-focused, like user stories (Obergfell, 2023).

Given the different circumstances indicated by the scrum product backlog, the following table defines a set of attributes by which the product backlog can be characterized, following all the scrum standards and methodologies that should be considered in a scrum model. In order to specify the user's activity, we will develop user stories and assign features to them. This will allow us to track the functionality of the action. The feature size-based estimation will also be defined, along with its priority and the estimated effort, which will be computed according to the number of days.

Product Backlog					
User Story ID	**User Story**	**List of Features**	**Estimate (size)**	**Priority**	**Estimate Effort**
US. 1.	As a new user, I need to be able to register myself as a player online	Register as an end user in the form of a player	Small	High	2 days
US. 2.	As an admin of the system, I need to be able to register any futsal organization into the system	Approve and register futsal administrations	Small	High	2 days
US. 3.	As an end user or a staff of the futsal, I need to be able to log into the application	Log in to the application based on appropriate roles	Small	High	5 days
US. 4.	As a futsal owner, I need to be able to login to the application after completion of successful registration approval	Log in to the application proceeding successful futsal registrations	Small	High	3 days
US. 5.	As a player, I need to be able to view the list of all futsal courts and already booked futsal facilities.	View all the list of futsal facilities either an already visited court on a non-appointed one	Medium	High	2 days

US. 6.	As a player, I need to be able to initialize an appointment with a futsal of choice based on all the requested descriptions	Book an appointment with a futsal facility on a scheduled date and time	High	High	12 days
US. 7.	As a futsal owner, I need to overview all the list of incoming requests regarding the appointment.	Insights of all the request descriptions and records of the player who has booked an appointment	High	High	10 days
US. 8.	As a futsal owner, I need to be able to process the appointment request, either approve or reject.	Proceed an appointment, approve or disapprove altering futsal availability condition	High	High	10 days
US. 9.	As a futsal owner, I need to be able to handle any incoming player availability request and trigger notifications for sets of players involved in the facility	Handle any requests for player availability	High	High	10 days

US. 10.	As a futsal owner, I need to approve or reject any response from players to player available request and allocate the newly formed team management	Process any responses for player availability request	High	High	5 days
US. 11.	As a player, I need to be able to view all the list of booked appointments with their status	Review any currently booked appointments	High	High	2 days
US. 12.	As an end user in the form of a futsal owner, I need to be able to operate all the sets of available kits and equipment to be purchased by any player	Allow kit purchases to the players with proper order tracking	Medium	High	5 days
US. 13.	As a player, I need to be able to process the transactions for ordering kits and items from the facility's inventory.	Proceed any transaction based on item orders	High	High	10 days
US. 14.	As a player, I need to be able to overview all the list of appointment details along with the list of approval status, involved players and other respective results	View the list of futsal slot descriptions and its respective players involvement	High	High	5 days
US.	As a facility owner, I will	Manage end	Small	Small	5 days

15.	be able to access details of all the players, defining their playing time and involvement and ordered items	users and track their involvements and purchases			
US. 16.	As a facility owner, I will be able to overview all the system information along with all of the details regarding the sales, appointments and further informational details of the futsal	Overview the whole system and define the existing sales and descriptions of the system	Medium	Medium	8 days

Table 28: Product Backlog: User Actions and Activities

7.6. High Level Use Case Description

Based on all of the use cases generated in the preceding use case diagram, a brief summary of each use case will be provided below.

Use Case: Registration

Actor(s): Players

Description: Players will register in this application by filling out an enrollment form with their personal information and credentials, which, if validated, will grant them access to the appointment scheduling, order purchases and record tracking program's functionalities.

Use Case: Login

Actor(s): Players, Futsal Owners, Admin

Description: Through the sign in portal, only authorized users with valid credentials are permitted to access the system. Once logged into the system, each set of roles is granted access to specific sites based on their unique attributes.

Use Case: View Profile

Actor(s): Players, Futsal Owners, Admin

Description: A logged in individual may access the profile section where they can view all of their respective details in the form of name, email and so on. Also, this use case allows them to change passwords and enable any two-factor authentication mode.

Use Case: Futsal Registration

Actor(s): Admin

Description: As mentioned in the registration procedure for technical staffs for futsal, the administrator is responsible for reviewing all applicant and then registering an active futsal along with their respective owners. Their respective credentials and information will be accessible to the administrator, who will then evaluate whether or not to approve the registration request. Once approved, the futsal registration will conclude adding a new card of available futsal facilities in the player's view along with a user creation for the respective futsal's in charge or owner.

Use Case: Handle the Inventory System

Actor(s): Futsal Owner

Description: Futsal Owners or in charges are directly involved in the inventory system of the items or kits or equipment available at their respective futsal facility for purchase and thus can update any relevant information when required.

Use Case: Book Appointment Slots

Actor(s): Player

Description: A player can book an appointment with the futsal territory or facility of choice. Initially the system allows the player to view all the active futsal courts filtered on the basis of their specialty along with their specified date time period when they are allocated for booking. For the procedure of booking an appointment, the player should select a particular date and time frame which will then prevent collision with any other existing appointments or new booking appointments. The respective futsal owner or staff will automatically be notified regarding it, and it is their responsibility to accept the particular request of booking or not.

Use Case: Request Player(s) Availability

Actor(s): Player

Description: Once the futsal owner has approved their booking slot, the respective player who booked the futsal slot will have the option to select all of the players from their group and for the case of unavailability or missing of a few players resulting in lack of total players, the player can also choose to request an availability of players to the futsal facility who will handle all of these requests and notify the respective players. Furthermore, once the request has been validated, verified and responded and approved by the new player, the system will automatically match the requested player to the booked team.

Use Case: View Booking History / View insights of transaction record

Actor(s): Player / Futsal Owner

Description: For each appointment, the player's model will be updated with the appointment details, scheduled timing and thus their booking details and scores. This information can be easily retrieved in the future by both the player and the futsal owner in charge of appointing the player so that they are informed of the player's past history and current disciplinary actions. Similarly, the player can also retrieve all of their past appointments with their respective date and detailed information regarding the meet up. Furthermore, as the system also provides services for order purchases, all the necessary transaction details, purchases, and information will be available.

Use Case: Item Purchases / Proceed Transaction

Actor(s): Player

Description: Since every action a player performs whether it be a booking slot, or kit purchases, they offer a transaction procedure. The payment can be conducted through credit points or payment gateway established in their profile and account or through cash when they physically visit the futsal for their action.

7.7. Entity Relationship Diagram

Due to the ongoing nature of the project's development, this Entity Relationship Diagram merely offers a cursory description of the system and its core domains. The following graphic contains all the essential major database tables; however, it is important to note that these are not the final database state for the system, as changes may be necessary as development progresses. In the future, the initial collection of databases will not hinder system development; further tables will be added as needed, regardless of whether it is the ultimate state or not.

7.7.1. Possible Entities

A list of possible entities of the system are listed and summarized below:

User: To store details regarding each possible user of the system, further on, other actors will inherit the properties defined by the user’s table.

Role: To store details regarding every available role of the system to grant privileges to each user accordingly.

Futsal: To store details with respect to all the active futsal facilities and stations defining all of their other respective information.

Futsal Image: To store all the allocated image paths of a particular futsal to identify the sets of images referring to a particular futsal.

Court: To store all the available or presentable futsal courts in a specified futsal accounting to other sets of records and details.

Appointment: To store all the records of appointments once they have been initialized and booked by the players capturing all the respective booked and appointed date and time frames along with flag values for action status and booking status.

Appointment Detail: To store all records of futsal appointment slots that have been granted, including details of each player's participation and any additional players that have been requested.

Transaction: To store all records of transaction details for any sets of online payment procedures for a successfully approved appointment during the booking phase.

Notification: To store all information needed to notify trigger activities, such as exchanging information during the futsal request approval process or requesting actions for player availability and their responses following action from the futsal facility, also accounting to the entities involved, either a futsal or a player themself.

Kit: To store all the records accounting to the different sets of available kits and equipment for a respective futsal academy or facility.

Order Header: To store all the necessary details of the player’s order in its header form which then integrates with the item or kit purchases.

Order Details: To store all the necessary details for the kits to be bought by the player along with their respective order tracking details and payment actions.

7.7.2. Relationship between Entities

A simple brief explanation regarding the major entities of the system and how they are interrelated will be described in the following heading.

User and Role:

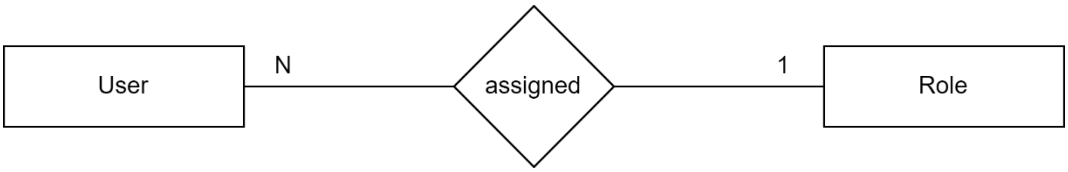


Figure 95: Entity Relationship: User and Role

A user will be assigned to just a single role; however, a particular role can be allocated to multiple users of the system.

Futsal, Courts and Images

Figure 96: Entity Relationship: Futsal, Court and Images

A futsal facility can have provision of multiple courts available for booking and each set of courts can have multiple sets of images to define their look and feel.

User and Appointment

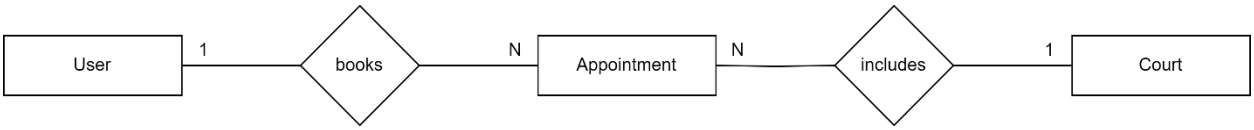


Figure 97: Entity Relationship: User and Appointments

A user once when booking an appointment includes a set of just a single court being registered or appointed in the booking slot. Thus, this can be a multiple set of processes when the user books multiple appointments for a single court at each appointment.

Appointment and Appointment Details

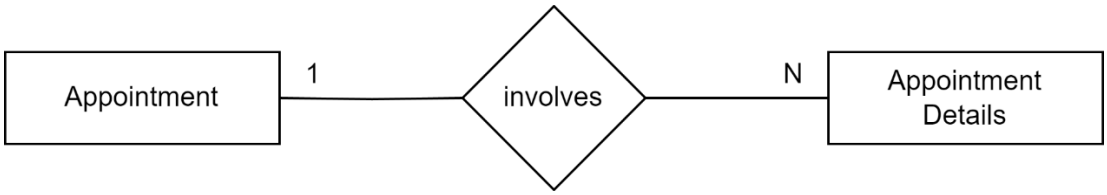


Figure 98: Entity Relationship: Appointment and Appointment Details

A single appointment can define details of the appointment in a one-to-many relationship as the details of the appointment include all other necessary information of all the players involved, their statuses and team involved in.

Appointment and Transaction Details

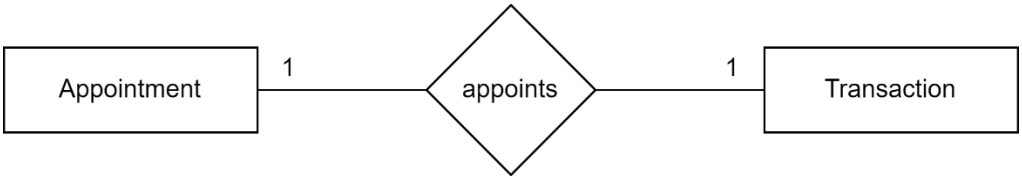


Figure 99: Entity Relationship: Appointment and Transactions

As an appointment once approved must have transaction process, either cash payment or online score payment, it is only done once as an appointment concludes, defining a one-to-one relationship in here.

User, Futsal and Notification

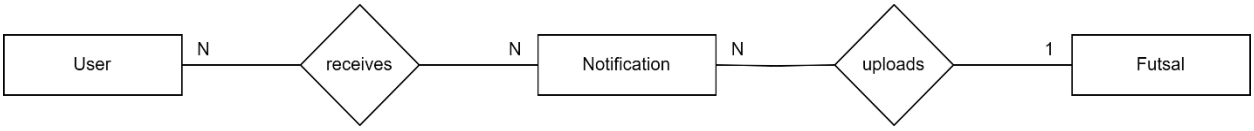


Figure 100: Entity Relationship: User, Futsal and Notification

During user actions such as order placing and purchasing or actions for player requests and responses, it is necessary to trigger necessary notification to the user, which is activated by the diagram that a single notification can be transmitted to multiple users. Similarly, a single futsal uploads multiple notification belonging to the same futsal which will be generated and produced to multiple sets of users.

Order Header and Order Details:

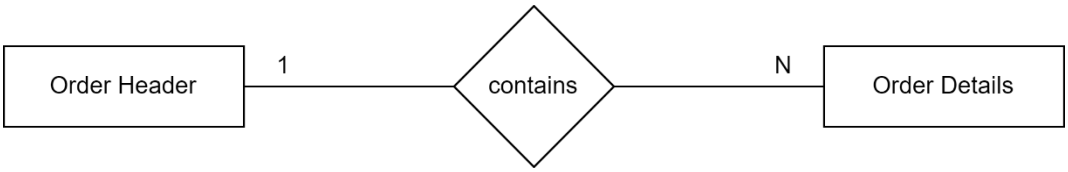


Figure 101: Entity Relationship: Order Header and Details

Due to the fact that a player may have more than one set of kit purchases, and that each set of kits or equipment may be ordered by more than one player, bridge entities must be introduced to handle the orders, payments, and examination information associated with each player's set of item uses.

7.7.3. Data Dictionary

Information systems, databases, and research projects often make use of dictionaries that list the names, meanings, and qualities of data components. Project objectives, data components, and standards for the interpretation, representation, and use of said data items are all laid out in detail in this document (UC Merced Library, 2023). Following this, you will see a description of a basic dictionary that, for each of the initially designed database tables, defines the metadata regarding data elements and their attributes.

Data Dictionary for Users Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Username	NVARCHAR (450)	
3.	Full Name	NVARCHAR (450)	
4.	Address	NVARCHAR (450)	
5.	Email Address	NVARCHAR (450)	
6.	Email Confirmed	BIT (Boolean)	
7.	Password Hash	NVARCHAR (450)	
8.	Password Salt	NVARCHAR (450)	

Table 29: Data Dictionary: Users

Data Dictionary for Roles Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Title	NVARCHAR (450)	
3.	Description	NVARCHAR (450)	

Table 30: Data Dictionary: Roles

Data Dictionary for User Roles Table

	Attributes	Data Type	Properties
1.	User Id	NVARCHAR (450)	COMPOSITE PRIMARY KEY “User (Id)”, NOT NULL
2.	Role Id	NVARCHAR (450)	COMPOSITE PRIMARY KEY, “Role (Id)” NOT NULL

Table 31: Data Dictionary: User Roles

Data Dictionary for Futsal Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Name	NVARCHAR (450)	UNIQUE
3.	Description	NVARCHAR (450)	
4.	Phrase	NVARCHAR (450)	UNIQUE
5.	Location Address	NVARCHAR (450)	
6.	City	BIT (Boolean)	
7.	Futsal Owner Id	NVARCHAR (450)	FOREIGN KEY “User (Id)”

Table 32: Data Dictionary: Futsal

Data Dictionary for Futsal Images

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Image Type	NVARCHAR (450)	
3.	Image Path URL	NVARCHAR (450)	UNIQUE
4.	Futsal Id	NVARCHAR (450)	FOREIGN KEY “Futsal (Id)”

Table 33: Data Dictionary: Futsal Images

Data Dictionary for Futsal Courts Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Title	NVARCHAR (450)	
3.	Type	NVARCHAR (450)	
4.	Futsal Id	NVARCHAR (450)	FOREIGN KEY “Futsal (Id)”

Table 34: Data Dictionary: Futsal Courts

Data Dictionary for Appointments Table

	Attributes	Data Type	Properties
1.	Player Id	NVARCHAR (450)	COMPOSITE KEY “User (Id)”, NOT NULL
2.	Futsal Id	NVARCHAR (450)	COMPOSITE KEY “Futsal (Id)”, NOT NULL
3.	Booked Date	DATETIME2	
4.	Appointed Date	DATE ONLY	
5.	Time Slot	DATETIME2	
6.	Is Action Complete	BIT (Boolean)	
7.	Is Approved	BIT (Boolean)	
8.	Number of Hours	DECIMAL	
9.	Total Amount	DECIMAL	
10.	Final Score	NVARCHAR (10)	

Table 35: Data Dictionary: Appointments

Data Dictionary for Appointment Details Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Appointment Id	NVARCHAR (450)	FOREIGN KEY "Appointment (Id)"
3.	Player Id	NVARCHAR (450)	FOREIGN KEY "User (Id)"
4.	Player Status	NVARCHAR (450)	Enums
5.	Player Team	NVARCHAR (450)	

Table 36: Data Dictionary: Appointment Details

Data Dictionary for Transactions Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Appointment Id	NVARCHAR (450)	FOREIGN KEY "Appointment (Id)"
3.	Payment Status	NVARCHAR (50)	
4.	Transaction Date	DATETIME2	
5.	Total Payment	DECIMAL	
6.	Payment Session Id	NVARCHAR (100)	UNIQUE
7.	Payment Intend Id	NVARCHAR (100)	UNIQUE

Table 37: Data Dictionary: Transaction

Data Dictionary for Notifications Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Sender Id	NVARCHAR (450)	FOREIGN KEY "Futsal (Id)"
3.	Receiver Id	NVARCHAR (450)	FOREIGN KEY "User (Id)"

4.	Sender Entity	NVARCHAR (50)	
5.	Receiver Entity	NVARCHAR (50)	
6.	Delivered Date	DATETIME2	
7.	Is Seen	BIT (Boolean)	

Table 38: Data Dictionary: Notifications

Data Dictionary for Kits Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Title	NVARCHAR (450)	
3.	Description	NVARCHAR (450)	
4.	Unit	INT	Enum
5.	Price	DECIMAL	
6.	Type	NVARCHAR (50)	
7.	Futsal Id	NVARCHAR (450)	FOREIGN KEY “Futsal (Id)”

Table 39: Data Dictionary: Kits

Data Dictionary for Order Table

	Attributes	Data Type	Properties
1.	Id	NVARCHAR (450)	PRIMARY KEY, UNIQUE, NOT NULL
2.	Player Id	NVARCHAR (450)	FOREIGN KEY “User (Id)”
3.	Ordered Date	DATETIME2	
4.	Description	NVARCHAR (450)	
5.	Order Total Amount	DECIMAL	
6.	Order Status	NVARCHAR (15)	
7.	Payment Status	NVARCHAR (15)	
8.	Payment Date	DATETIME2	
9.	Tracking Number	NVARCHAR (100)	

10.	Carrier	NVARCHAR (100)	
11.	Payment Session Id	NVARCHAR (100)	UNIQUE
12.	Payment Intend Id	NVARCHAR (100)	UNIQUE

Table 40: Data Dictionary: Order Details

Data Dictionary for Order Details Table

	Attributes	Data Type	Properties
1.	Order Id	NVARCHAR (450)	COMPOSITE PRIMARY KEY “Order (Id)”, NOT NULL
2.	Kit Id	NVARCHAR (450)	COMPOSITE PRIMARY KEY, “Kit (Id)” NOT NULL
3.	Count	INT	
4.	Total Amount	DECIMAL	

Table 41: Data Dictionary: Order Details

7.8. Architecture Design

Since the development of the application had begun, the first order of business was to establish the project's architecture, which comprised the following components: the MVC Pattern, Domain Driven Design, and Clean Architecture.

7.8.1. Application Architecture (MVC Pattern)

When it comes to development, the application must also adhere to a set of guidelines describing its structure and organization. As the subsequent.NET Core framework includes an MVC architectural pattern, it facilitates the definition of a holistic solution based on logically connected and consistent principles, concepts, and attributes. In this instance, MVC stands for Model View Controller, and their corresponding explanations are provided below.

Model: It corresponds to all the data-related logic that the user works with.

View: It corresponds to all the UI logic and front-end view of the application.

Controller: It corresponds as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output.

The list of features provided by ASP.NET Core’s MVC pattern are as follows:

- It offers an extensible and modular framework that is easily replaceable and modifiable.
- It employs the component-based design of the program by logically splitting it into Model, View, and Controller components, hence facilitating the management of the complexity of large-scale projects and the development of individual components.
- It improves test-driven development and testability of the program because all components can be interface-based built and tested with mock objects.

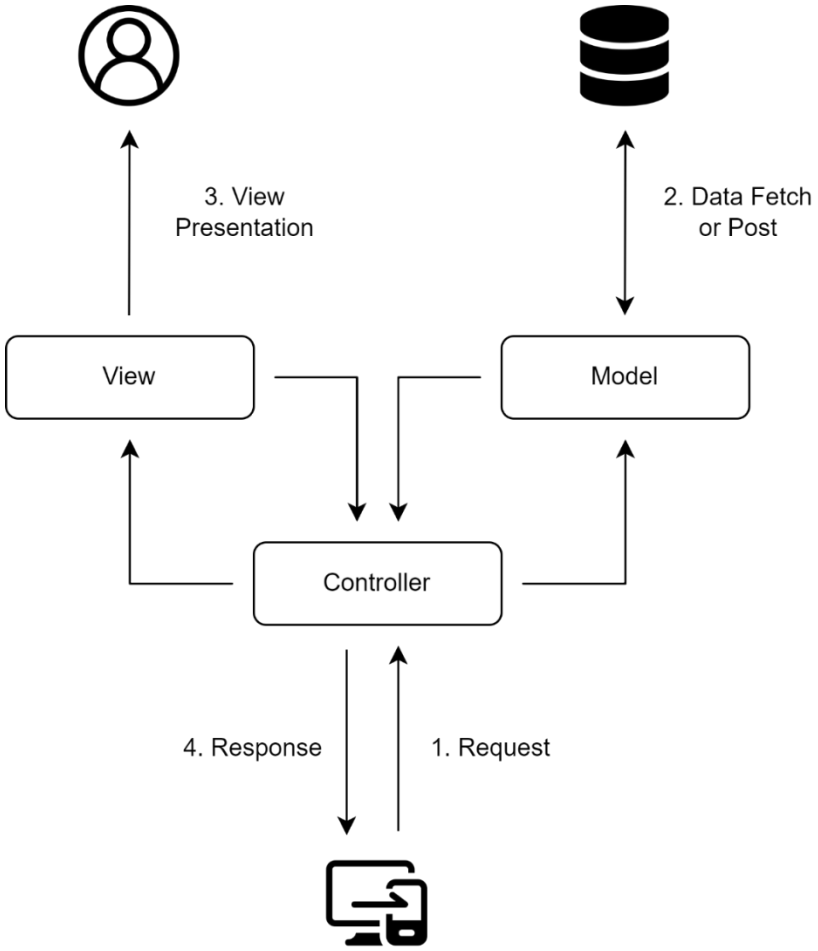


Figure 102: MVC Pattern: Data Flow

7.8.2. Development Architecture (Clean Architecture)

Implementing use cases with minimal coupling is a primary goal of clean architecture. Use cases provide a high-level organizing structure that is abstracted from the specifics of frameworks and technologies. Domains, in this sense, are the topics on which a currently-under-development application focuses. The entities or models around which an application is created remain fundamental to clean architecture, as does the business logic, its implementation, and subsequent practices.

In Clean Architecture, the Core refers to the intersection of the Domain and Application layers. These layers include the logic and rules that govern the business. Because of this, the Presentation and Infrastructure layers are mere details, and the business layers should not rely on them. The traditional dependency of business logic on data access and other infrastructure issues has been inverted, with infrastructure and implementation details now being dependent on the Application layer. To do this, abstractions (or interfaces) are specified in the Application layer, while types are created in the infrastructure layer to provide their implementation. Core does not rely on any higher-level abstractions and all dependencies are inverted there. Both Infrastructure and Presentation rely on Core, although none is necessary for the other.

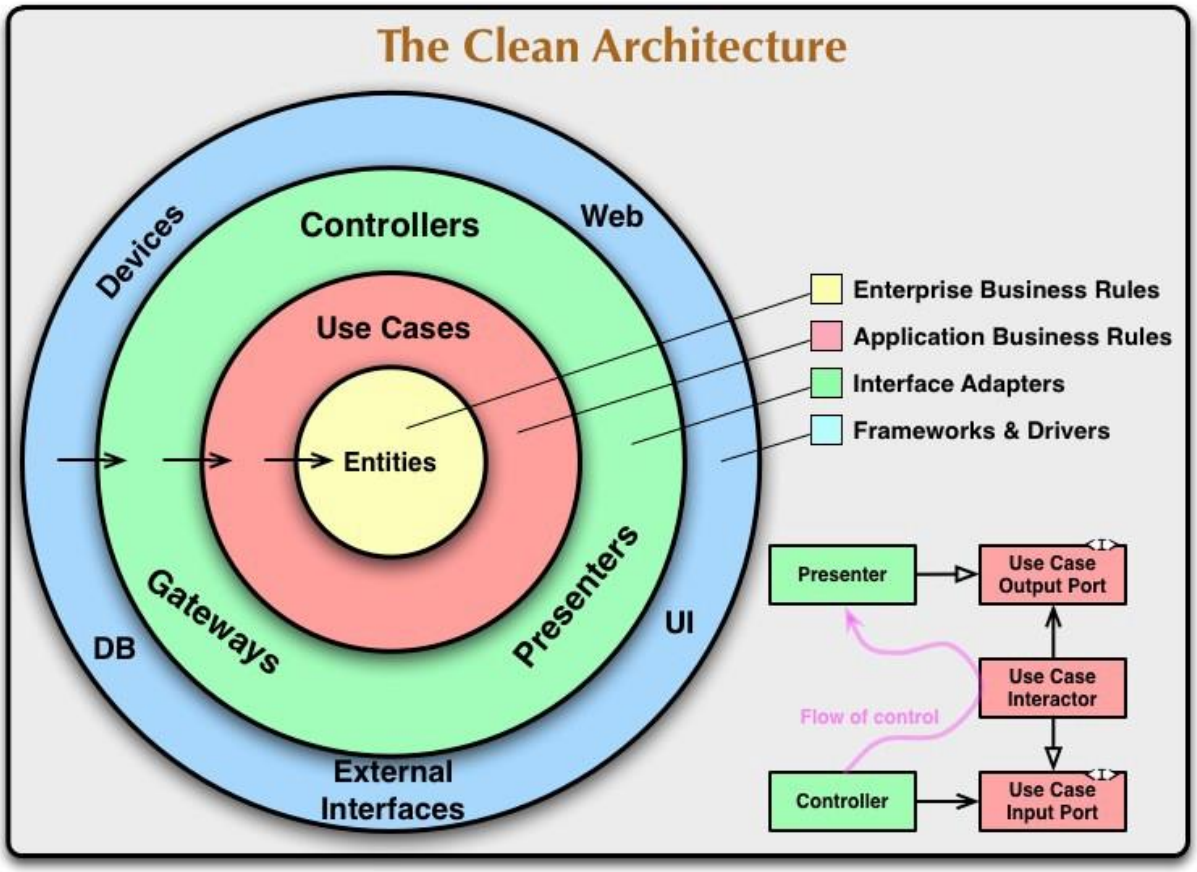


Figure 103: Clean Architecture: Data Flow

The components to be included in each of the layers are listed below:

	Project Layer	Components
	Domain Layer	Entities / Models, Aggregates, Value Objects, Enums, Constants
	Application Layer	Abstractions, Interfaces, Application Services, Commands and Queries, Exceptions, View Models (DTOs), Mappers, Validators, Behaviors, Specifications
	Infrastructure Layer	Persistence Layer, Identity Services, Payment Services, Third Party Services, Notification Services (Email and SMS Extension)
	Persistence Layer	Data Context, Repositories, Data Seeding, Data Migrations
	Presentation Layer	Controllers, Views, Middleware, Filters, Web API Utilities

Table 42: Clean Architecture: Layers and Components

7.9. Revision of Work Breakdown Structure

A work breakdown structure (WBS) is a project management technique that uses a step-by-step approach to finish complex, multifaceted tasks. A WBS can integrate scope, cost, and deliverables into a single tool by decomposing the project into its constituent parts. The 100% rule is an essential component of a work breakdown structure. This indicates that the WBS incorporates all parts of the project as well as the individual or team responsible for each component. Another important aspect of WBS is its tiered structure. When the 100% rule is applied, Level 1 of the WBS will represent the entirety of the project (Forbes, 2023).

The following WBS outlines the high-level frameworks of the proposed work, as described in the proposal report. The offered figure is ambiguous because the plan simply defined the total work to be done and achieved and not what work was to be done in each of the sprint cycle.

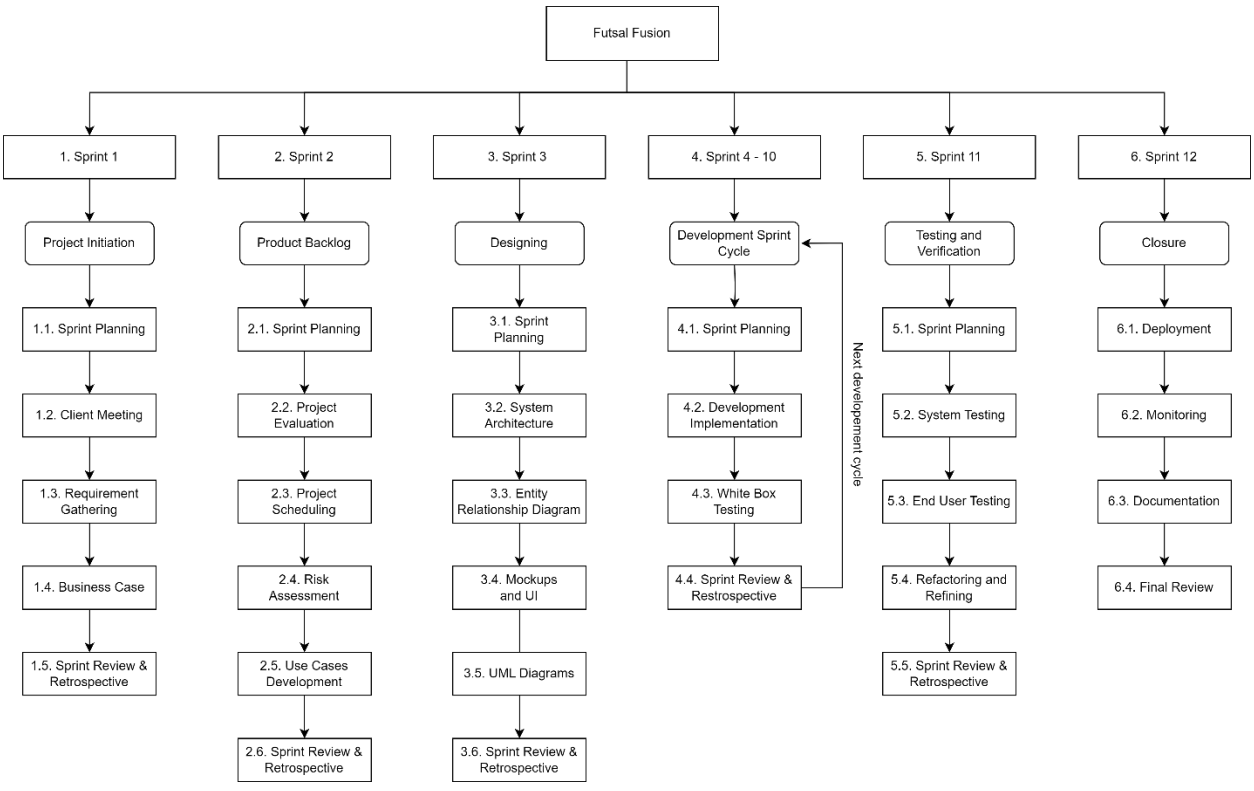


Figure 104: Work Breakdown Structure: Scrum Model

An even more extensive breakdown structure has to be instantiated now that the quantity of work done has been tracked successfully and a complete estimate of what needs to be done and how work can be done has been made. The following WBS specifies a sprint approach for both the beginning and the end of the development process; however, the development sprint will be refined based on the specific tasks that need to be completed in each time period. Besides the development sprint cycle, other major task work will be instantiated as the same following its own set of time frames.

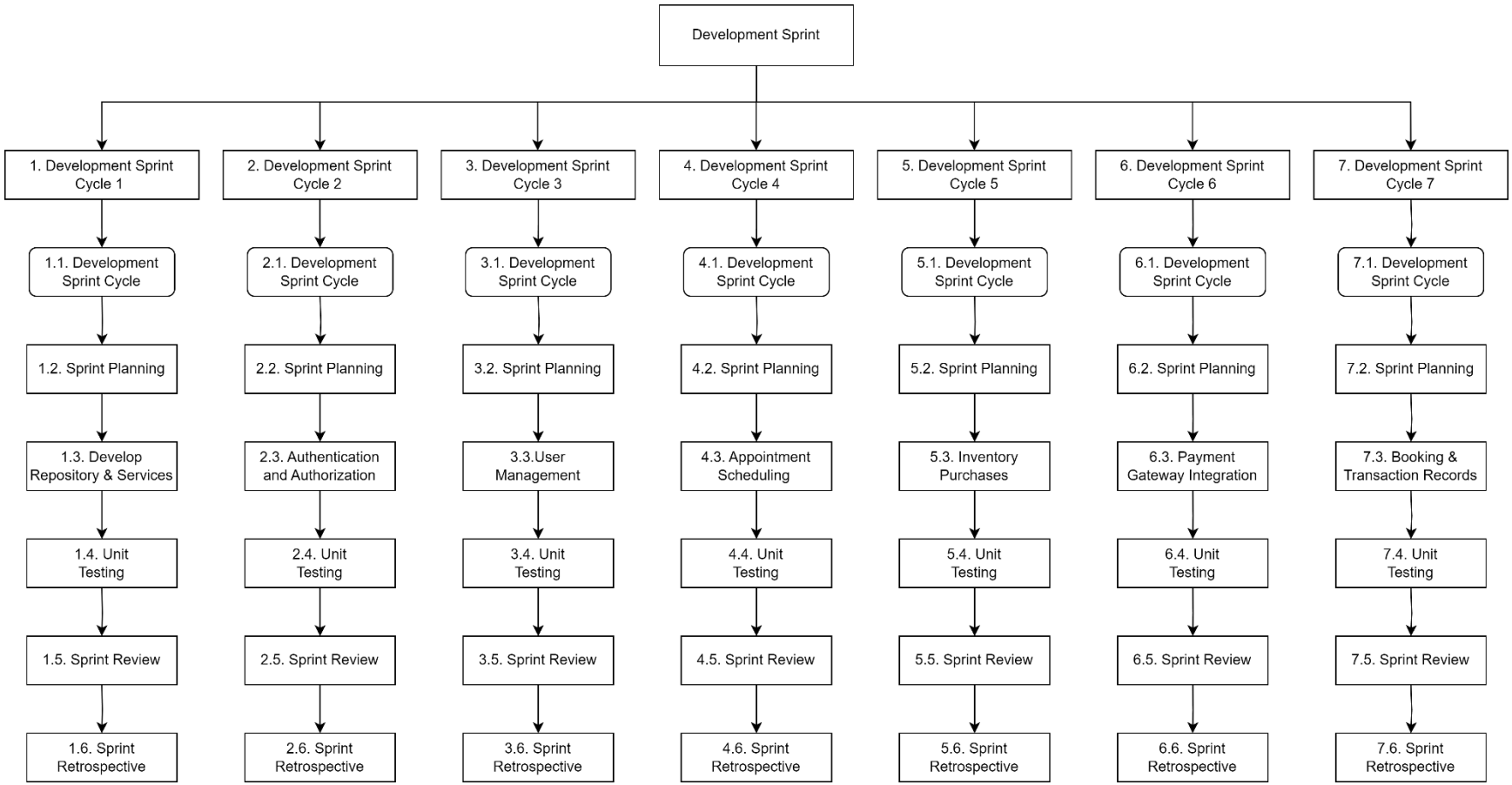


Figure 105: WBS Model: Development Sprints

7.10. Revision of Milestone

Milestones, also known as scheduling techniques, are used to mark important points in a project's timeline, such as the beginning and end of the project or the conclusion of a significant phase of the work. A structured figure decapitating the major highlights of the overall project has been labeled in the following figure with representation of each of the task to be done on its particular time frame.

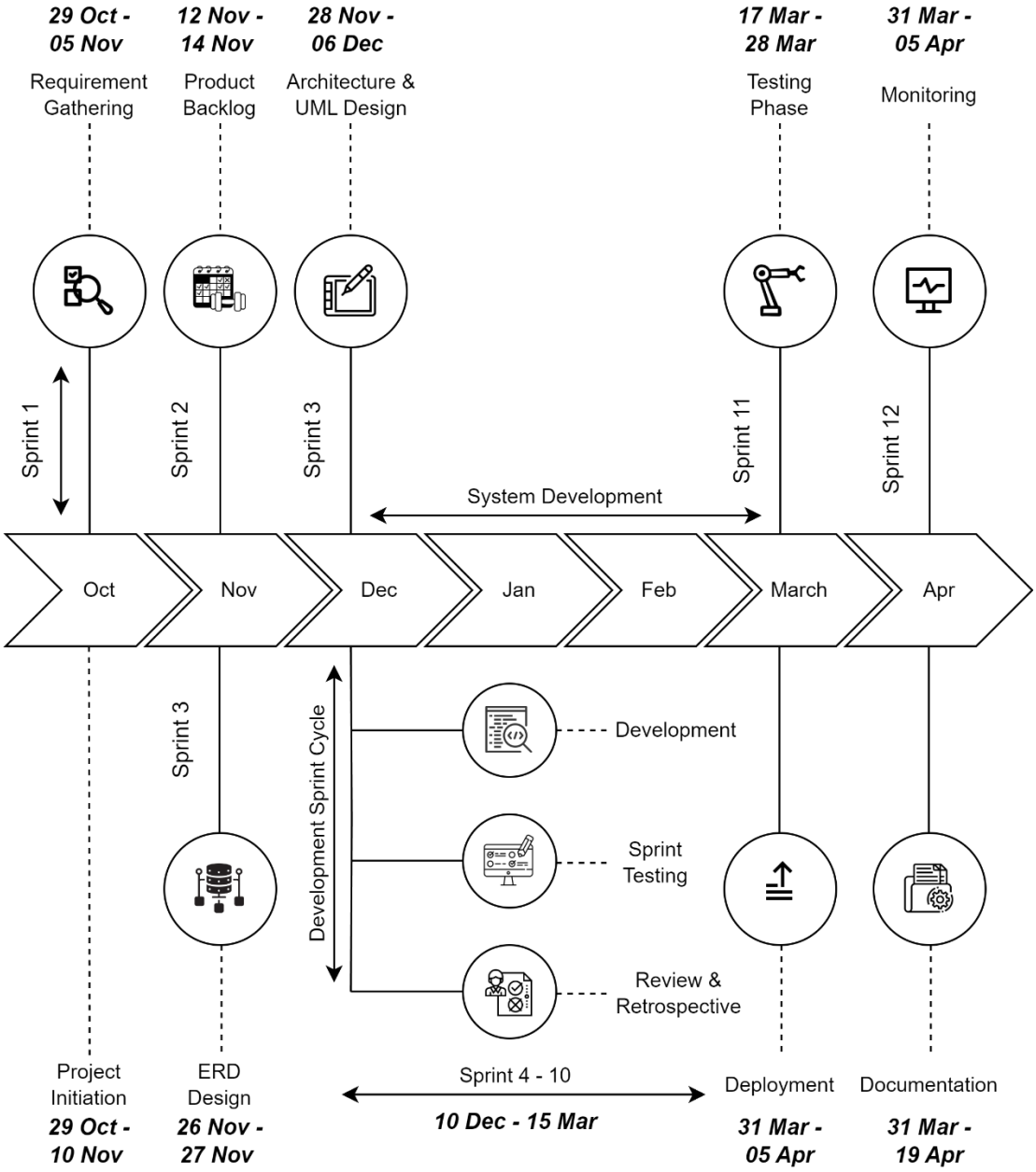


Figure 106: Milestone Chart: Revised Version

- **Milestone 1:** Requirement Gathering (17th Oct 2023 – 26th Oct 2023)
  The first step to initiate the project is to meet up with the client and then gather requirements accordingly. Thus, business cases will be defined, and the project scope will be defined.

- **Milestone 2:** Product Backlog (27th Oct 2023 – 9th November 2023)
  In order to release and iterate on all of the planning, a well-prioritized agile backlog is developed. As a result, expectations could be set with the many parties involved, and engineering time became a more stable resource. Based on the roadmap's specifications, it outlined a sequence of tasks for the development team to complete.

- **Milestone 3:** ERD Design and Architecture Definition (10th November 2023 – 29th November 2023)
  An initial entity relationship diagram was so defined in this milestone which decapitated the overall system on a surficial level based on all the designing stages. Further on the system architecture was defined, the pattern of development was stated, and the programming application architecture was thus finalized.

- **Milestone 4:** Development Sprint Cycle (10th December 2023 – 15th March 2024)
  The sprint of development has begun and is currently 30% complete. The business logic has been designed based on a developed basic coding pattern that includes the definition of all repositories and their corresponding services. Standard CRUD operations and descriptive backend logical activities have been launched, and database migrations have been submitted to the database engine. From here on out, each sprint's worth of work will be completed in tandem with the others.

- **Milestone 5:** Testing Sprint Cycle (17th March 2024 – 28th March 2024)
  After the coding and development sprint is complete and all functional requirements have been met, the system will be put through a full sprint of testing, including unit tests and system tests, to detect any instances of bugs and issues.

- **Milestone 6:** Deployment (31st March 2024 – 5th April 2024)
  Deploying the application into the server so that it can be accessible by the client is the next crucial component for this system. This is a pivotal stage of the application to remember, since it will yield invaluable data about the system's behavior and performance in a live network setting.

- **Milestone 7:** Monitoring (31st March 2024 – 5th April 2024)
  It's not guaranteed that the system will function exactly as it does on a local environment once it's been deployed to the server. Given that server load and network congestion might affect system performance, it's important to trace all API calls made by the deployed application across the network and investigate those that have encountered problems. In the future, appropriate optimization and bug fixes will be applied to define the system in its optimal state.

- **Milestone 8:** Documentation and Finalization (31st March 2024 – 19th April 2023)
  A well-documented application program is about more than just coding. Complete documentation of the development process will be defined, including test cases that characterize the system's behavior at each stage of its creation.

7.11. Revision of Gantt Chart

The below diagram shows the proposed Gantt Chart representation for the project. At the outset, there was little consensus on how long each sprint should last and what information should be included in each sprint's presentation. The time required to complete the navigating activities for each sprint was only roughly estimated.

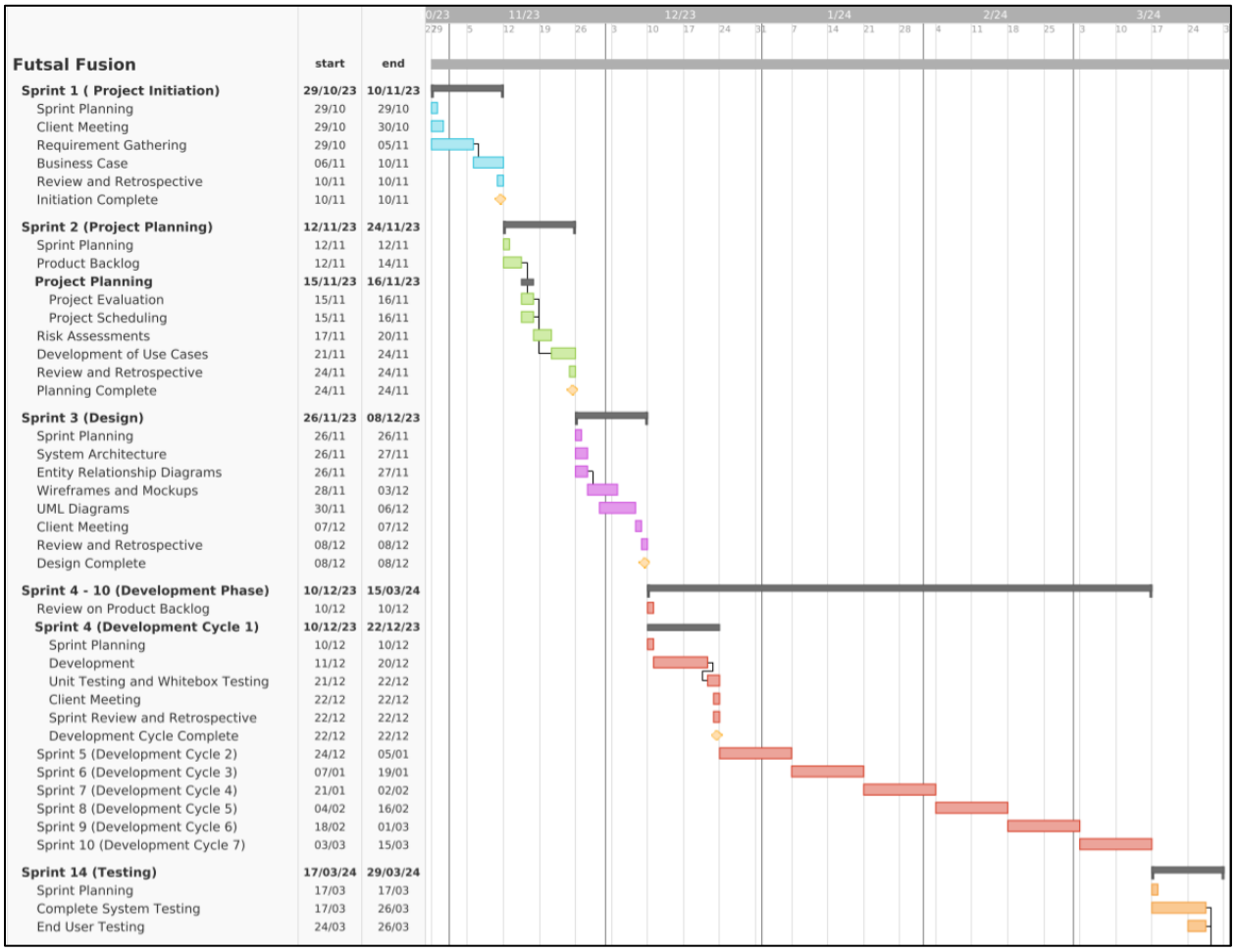


Figure 107: Gantt Chart: Proposal (1)

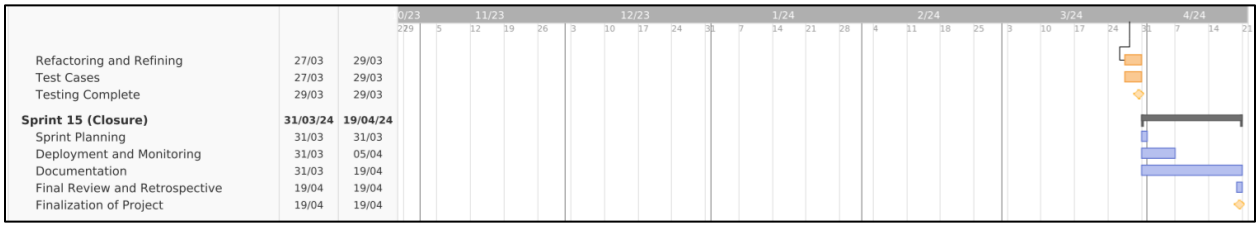
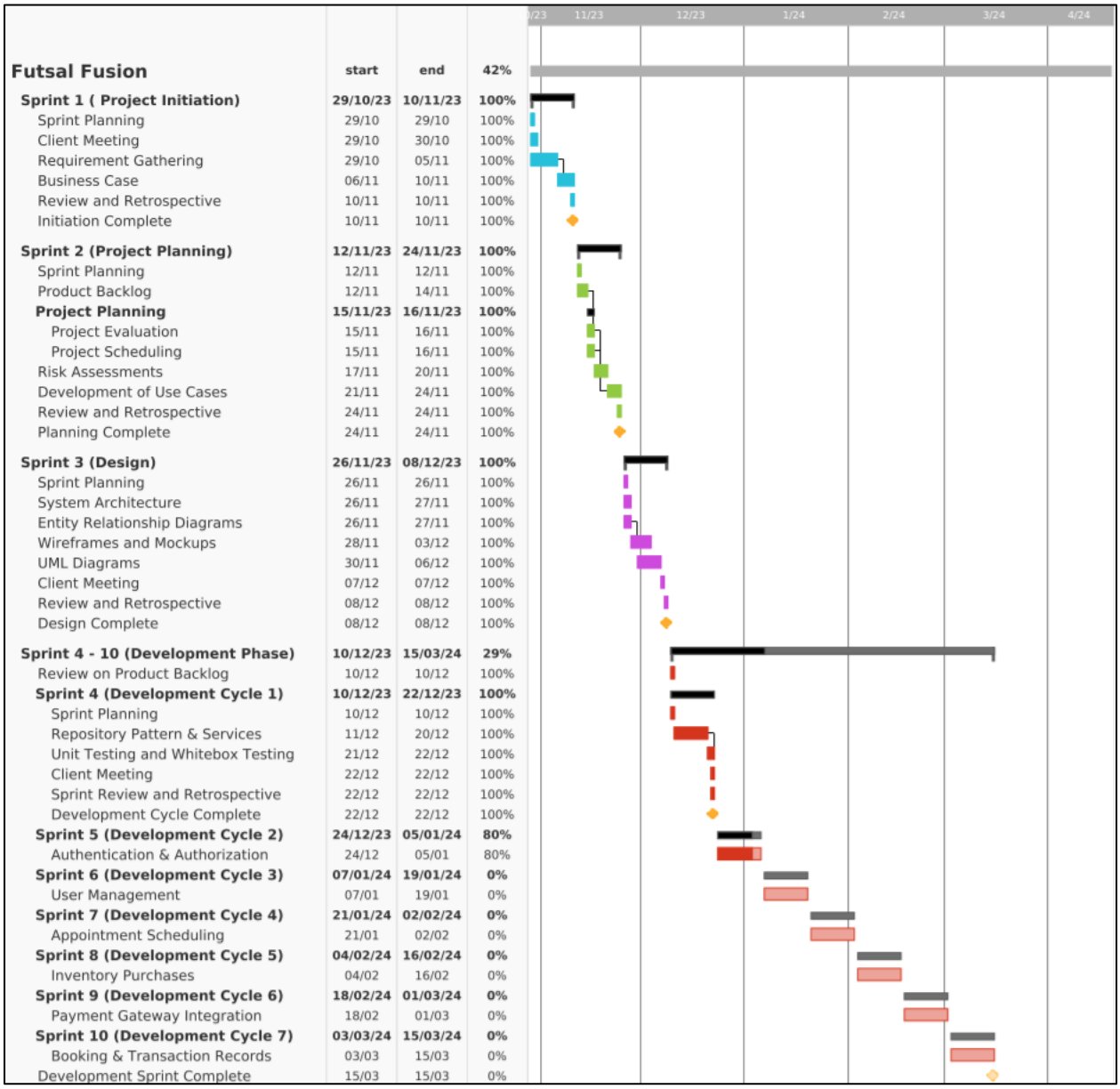


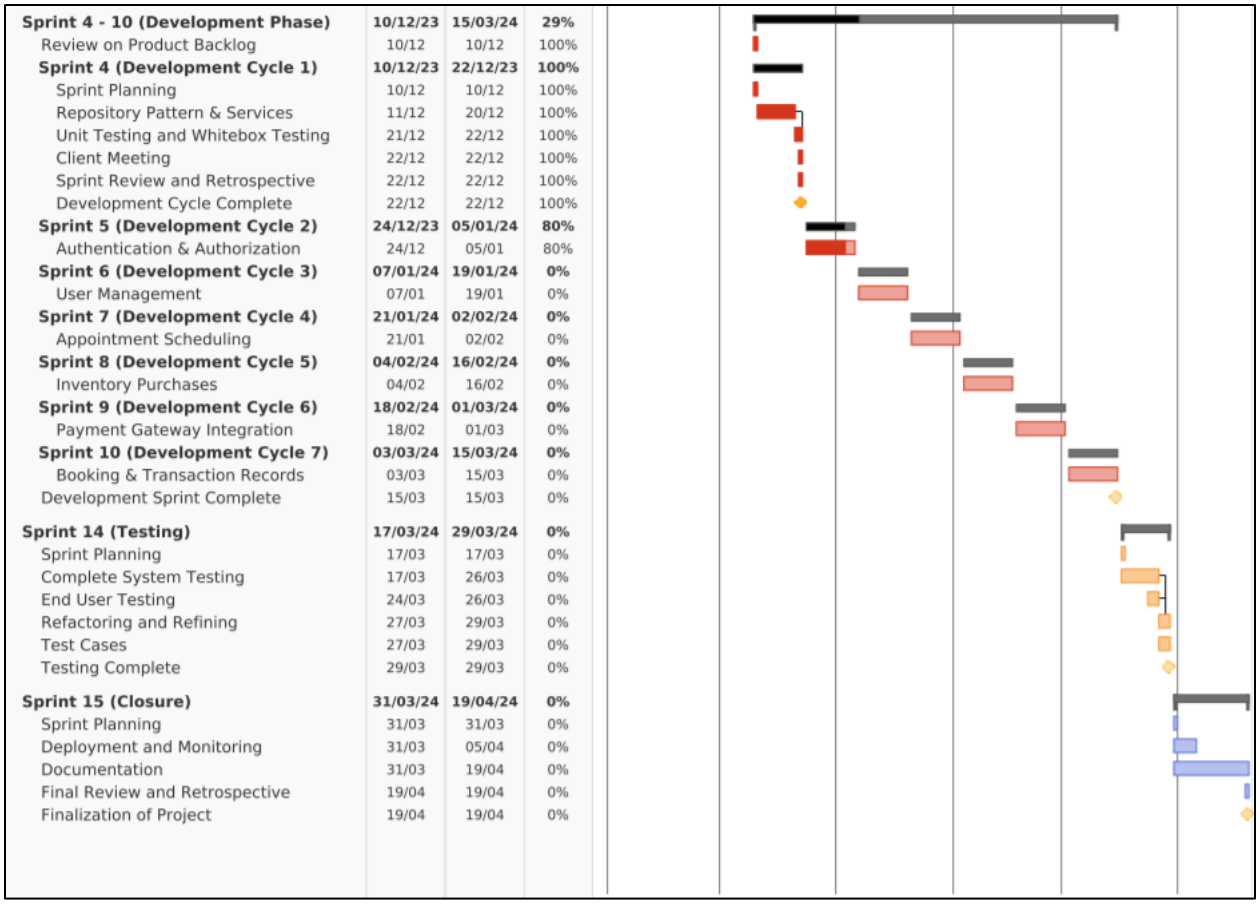
Figure 108: Gantt Chart: Proposal (2)

However, when the development process nears its halfway point, it becomes evident how much of the task can be done and which tasks need to be prioritized. The list of essential application functionalities that need to be implemented has been formally stated in a backlog. Another estimation of tasks and its division can be described in accordance with the given Gantt Chart now that there is an overall gain in information regarding the work to be accomplished in each of the sprints. The sprint, which incorporated a coding cycle, is thoroughly explained, and it will be adhered to the letter until the end of the project.



Futsal Fusion	start	end	42%
Sprint 1 (Project Initiation)	**29/10/23**	**10/11/23**	**100%**
Sprint Planning	29/10	29/10	100%
Client Meeting	29/10	30/10	100%
Requirement Gathering	29/10	05/11	100%
Business Case	06/11	10/11	100%
Review and Retrospective	10/11	10/11	100%
Initiation Complete	10/11	10/11	100%
Sprint 2 (Project Planning)	**12/11/23**	**24/11/23**	**100%**
Sprint Planning	12/11	12/11	100%
Product Backlog	12/11	14/11	100%
Project Planning	**15/11/23**	**16/11/23**	**100%**
Project Evaluation	15/11	16/11	100%
Project Scheduling	15/11	16/11	100%
Risk Assessments	17/11	20/11	100%
Development of Use Cases	21/11	24/11	100%
Review and Retrospective	24/11	24/11	100%
Planning Complete	24/11	24/11	100%
Sprint 3 (Design)	**26/11/23**	**08/12/23**	**100%**
Sprint Planning	26/11	26/11	100%
System Architecture	26/11	27/11	100%
Entity Relationship Diagrams	26/11	27/11	100%
Wireframes and Mockups	28/11	03/12	100%
UML Diagrams	30/11	06/12	100%
Client Meeting	07/12	07/12	100%
Review and Retrospective	08/12	08/12	100%
Design Complete	08/12	08/12	100%
Sprint 4 - 10 (Development Phase)	**10/12/23**	**15/03/24**	**29%**
Review on Product Backlog	10/12	10/12	100%
Sprint 4 (Development Cycle 1)	**10/12/23**	**22/12/23**	**100%**
Sprint Planning	10/12	10/12	100%
Repository Pattern & Services	11/12	20/12	100%
Unit Testing and Whitebox Testing	21/12	22/12	100%
Client Meeting	22/12	22/12	100%
Sprint Review and Retrospective	22/12	22/12	100%
Development Cycle Complete	22/12	22/12	100%
Sprint 5 (Development Cycle 2)	**24/12/23**	**05/01/24**	**80%**
Authentication & Authorization	24/12	05/01	80%
Sprint 6 (Development Cycle 3)	**07/01/24**	**19/01/24**	**0%**
User Management	07/01	19/01	0%
Sprint 7 (Development Cycle 4)	**21/01/24**	**02/02/24**	**0%**
Appointment Scheduling	21/01	02/02	0%
Sprint 8 (Development Cycle 5)	**04/02/24**	**16/02/24**	**0%**
Inventory Purchases	04/02	16/02	0%
Sprint 9 (Development Cycle 6)	**18/02/24**	**01/03/24**	**0%**
Payment Gateway Integration	18/02	01/03	0%
Sprint 10 (Development Cycle 7)	**03/03/24**	**15/03/24**	**0%**
Booking & Transaction Records	03/03	15/03	0%
Development Sprint Complete	15/03	15/03	0%

Figure 109: Gantt Chart: Revised Model (1)



Sprint 4 - 10 (Development Phase)	**10/12/23**	**15/03/24**	**29%**
Review on Product Backlog	10/12	10/12	100%
Sprint 4 (Development Cycle 1)	**10/12/23**	**22/12/23**	**100%**
Sprint Planning	10/12	10/12	100%
Repository Pattern & Services	11/12	20/12	100%
Unit Testing and Whitebox Testing	21/12	22/12	100%
Client Meeting	22/12	22/12	100%
Sprint Review and Retrospective	22/12	22/12	100%
Development Cycle Complete	22/12	22/12	100%
Sprint 5 (Development Cycle 2)	**24/12/23**	**05/01/24**	**80%**
Authentication & Authorization	24/12	05/01	80%
Sprint 6 (Development Cycle 3)	**07/01/24**	**19/01/24**	**0%**
User Management	07/01	19/01	0%
Sprint 7 (Development Cycle 4)	**21/01/24**	**02/02/24**	**0%**
Appointment Scheduling	21/01	02/02	0%
Sprint 8 (Development Cycle 5)	**04/02/24**	**16/02/24**	**0%**
Inventory Purchases	04/02	16/02	0%
Sprint 9 (Development Cycle 6)	**18/02/24**	**01/03/24**	**0%**
Payment Gateway Integration	18/02	01/03	0%
Sprint 10 (Development Cycle 7)	**03/03/24**	**15/03/24**	**0%**
Booking & Transaction Records	03/03	15/03	0%
Development Sprint Complete	15/03	15/03	0%
Sprint 14 (Testing)	**17/03/24**	**29/03/24**	**0%**
Sprint Planning	17/03	17/03	0%
Complete System Testing	17/03	26/03	0%
End User Testing	24/03	26/03	0%
Refactoring and Refining	27/03	29/03	0%
Test Cases	27/03	29/03	0%
Testing Complete	29/03	29/03	0%
Sprint 15 (Closure)	**31/03/24**	**19/04/24**	**0%**
Sprint Planning	31/03	31/03	0%
Deployment and Monitoring	31/03	05/04	0%
Documentation	31/03	19/04	0%
Final Review and Retrospective	19/04	19/04	0%
Finalization of Project	19/04	19/04	0%

Figure 110: Gantt Chart: Revised Model (2)

7.12. Study on Considered Methodologies

For the purposes of development, design, and planning, more software development strategies were explored beyond those already mentioned. For that reason, what follows is a synopsis of the techniques that were considered.

7.12.1. RUP Methodology

Rational Unified Process, or RUP for short, is a well-defined and well-structured software development approach that is recursive, architecture-centric, use-case driven, and also accounts for the need for modification early on and delivers a gradual integration (by iteration) of the various components to allow the exploration of consequences and their resolution in a timely manner, as mentioned on its five phases of inception, elaboration, construction, transition, and finally production (Study, 2021).

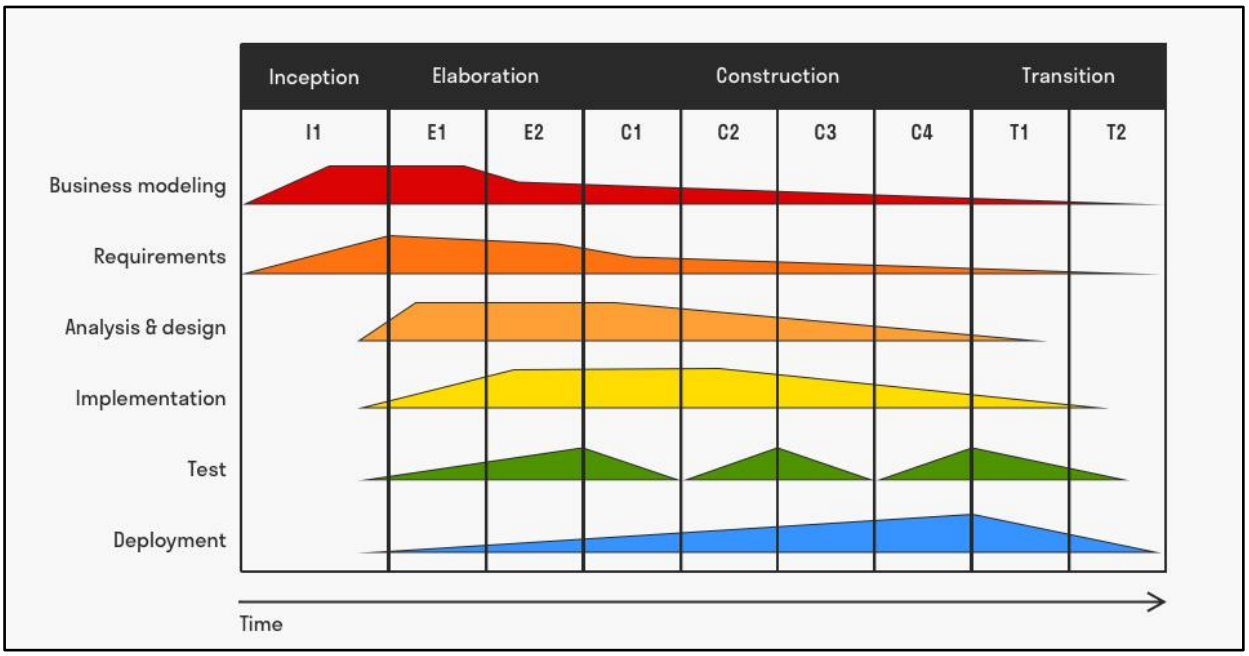


Figure 111: Considered Methodology: RUP

RUP is a comprehensive methodology that emphasizes accurate documentation and is capable of proactively resolving the project potential risks with the client's changing requirement that require careful change request management. Similarly, the reuse of modules within the product cycle drastically reduces development time. Similarly, this strategy is highly recommended for producing high-quality software within time-cost

restrictions. Out of the outlined advantages, a few reasons that were considered for not implementing this model are thus listed below.

Case Study Scenario	Developing a simple yet organized system
Features	The development process is too complex and unorganized
Justification	As the proposed application would be a simple application, following the RUP Model, the development process would be overly complex, unorganized, and less adaptable than an agile method, making it a slower development cycle model.

Table 43: Justification Model: RUP (1)

Case Study Scenario	Developing the system as a beginner
Features	The development process requires experts in their fields
Justification	As the main focus of this project is to solve a problem and deliver a problem in the most basic way as a learner, integration of this model sounds appealing in theory but on projects with several development streams with professions and expertise at their field, the model's approach produces confusion and testing problems in contrast to the theoretical integration.

Table 44: Justification Model: RUP (2)

Case Study Scenario	Developing a system to solve an issue with just available resources
Features	The development process seems to be a costly measure
Justification	Since, the project is about learning and implementing a system based off available resources, implementation of RUP would be reliant on the Rational toolset, which is prohibitively expensive, rendering it as inadaptable; and this strategy is no longer generally employed in huge organizations as well.

Table 45: Justification Model: RUP (3)

7.12.2. RAD Methodology

Rapid Application Development (RAD) is a model of software development that emphasizes rapid prototyping and rapid feedback over lengthy development and testing cycles. Rapid application development enables software developers to make multiple iterations and updates without starting from scratch each time. This helps ensure that the final product is more quality-oriented and in line with the needs of end-users. Business modeling, data and process pattern design with prototyping, and application development and testing are the primary phases that justify a RAD model (Geeks for Geeks, 2022).

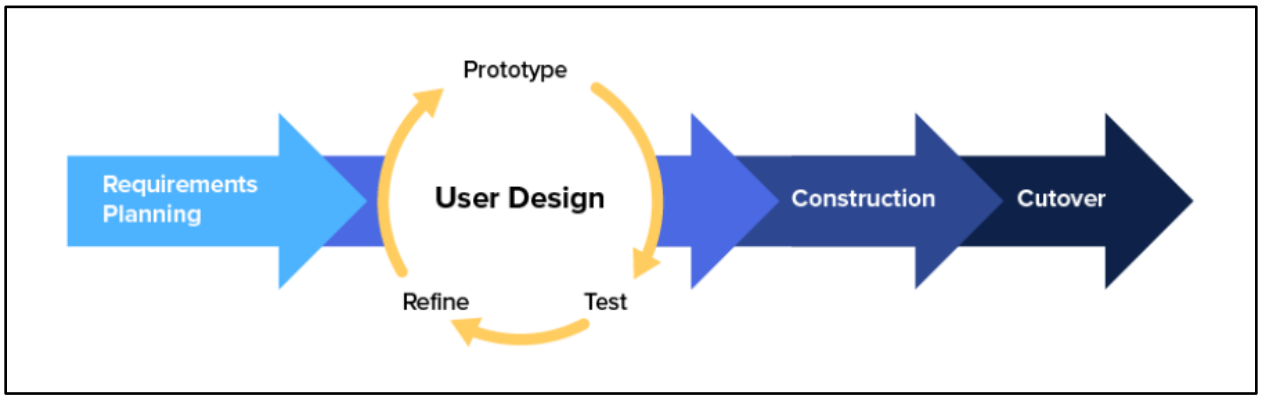


Figure 112: Considered Methodology: RAD

It is anticipated that a system based on the RAD methodology will be developed in a short period of time, between four and five months, with extensive user participation to clarify requirements and identify areas for prototyping improvement based on user feedback. Therefore, feedback from reputable sources can be extremely beneficial before an actual development takes place. However, the following positive constraints were insufficient for this model to be suitable for my project; consequently, it was avoided for the following reasons.

Case Study Scenario	Development doesn't require external tools and is cost-efficient
Features	The development process requires multiple factors of considerations when it comes to actual progress
Justification	The cost of utilizing automated tools and techniques may exceed the entire project resources and budget, even if the project is of a moderate size.

Table 46: Justification Model: RAD (1)

Case Study Scenario	Development could result to be scalable as per the requirement of the client
Features	The development process is heavily dependent on prototyping.
Justification	As a RAD application begins as a prototype and evolves into a final product, its scalability is diminished.

Table 47: Justification Model: RAD (2)

Case Study Scenario	Development requires strong monitoring and documentation to track every progress routine
Features	The development process doesn't follow project management technique but just focuses on development.
Justification	Since the project is not only about prototyping and development but with proper documentation regarding retrospective and outcome to track down progress and learning, there would be difficulty in monitoring progress and problems, as there is no documentation to illustrate what has been accomplished.

Table 48: Justification Model: RAD (3)

7.12.3. Kanban Methodology

Popular in the implementation of agile and DevOps software development, Kanban is a visualization method for organizing work as it moves through a process and focuses on creating a continuous workflow. The intended outcome is a visual representation of the workflow as well as the tasks involved in that workflow defining both the doing, done and to dos. By locating and eliminating any potential bottlenecks, Kanban ensures that work is completed efficiently and at the lowest possible cost. The goal of lean and just-in-time (JIT) production is to maximize the flow of the raw materials and components used to make a final product by using a scheduling system called Kanban board to inform what to produce, when to produce it, and how much to produce (Digite, 2022).

Even when a Kanban model's propensity to foster client interaction, flexibility in the face of change, and continuous integration of all the visualization done in a board to aid in the achievement of productivity over the course of application development, it has a few considerable reasons on why it is to be avoided, a few are as follows.

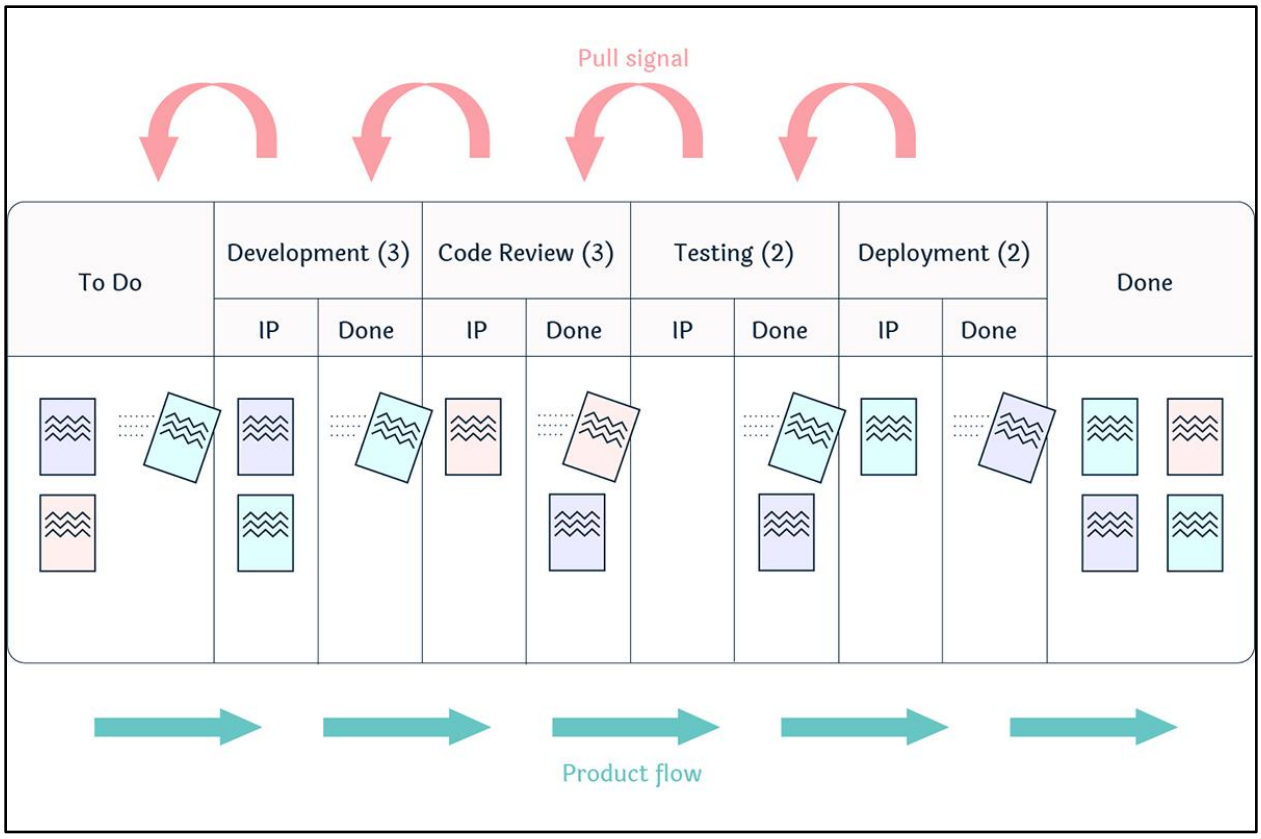


Figure 113: Considered Methodology: Kanban

Case Study Scenario	Development requires strict following patterns of a methodology
Features	The development process isn't considered an official methodology without having is proper documented guidelines.
Justification	Since the project emphasizes on properly structuring a software development methodology

Table 49: Justification Model: Kanban (1)

Case Study Scenario	Development needs to follow precise time scales
Features	The development process can't describe the timescales and timeframes involved in getting the task done.
Justification	As a project needs to be progressed and be delivered within time frames, applying a Kanban methodology would make it difficult to describe timescales and time spans involved in getting a task done.

Table 50: Justification Model: Kanban (2)

Case Study Scenario	Development requires progress in an iteration pattern following dynamics pattern
Features	The development process doesn't offer an iteration process making it unfit for dynamic environment.
Justification	Since the project is to be delivered in iterations with ongoing user feedback and adjustments based on the client's approach, the Kanban Model appears ambiguous because it lacks a dynamic setup.

Table 51: Justification Model: Kanban (3)

7.13. Selected Methodology (Personal Scrum Model)

Scrum is an agile framework that focuses on producing value progressively and collaboratively through continuous experimentation and feedback loops. As an empirical approach based on observation, experience, and experimentation, scrums display feedback that occur during sprints usually short cycle of two weeks, allowing for inspection and customization of the process and what will be delivered incrementally in a collaborative way (Shore, 2021).

The sequence of events, ceremonies, or meetings that scrum teams undertake on a regular basis is one of the most well-known parts of the scrum framework. The most important routines that a scrum team might perform are outlined below.

- Creation of a product and scrum backlog to arrange and prioritize items for sprint's activity, defining the scope and the deliverables at the end of the sprint.
- In a typical two-week sprint, the set of increments should be ordered so that continuity is maintained and past experiences learnt may be used to subsequent sprints so as to make small yet incremental progresses.
- Organization of daily scrum meeting to ensure that everyone is working toward the same sprint objective and to visualize the accomplishment and what remains undone.
- At the end of each sprint, a retrospective should be held to present the completed items from the prior sprint's backlog and solicit suggestions for modifications (Drumond, 2022).

Similarly, a few of the benefits that a Scrum model provides are as follows:

- The implementation of rules, artifacts, and events is straightforward.
- Software delivery is accelerated when ambiguities are eradicated by a semi-prescriptive method.
- Complex tasks are broken down into simple user stories, making it excellent for challenging projects.
- Enhances project visibility and facilitates modification of requirements based on feedback loops (Sutherland & Schwaber , 2020).

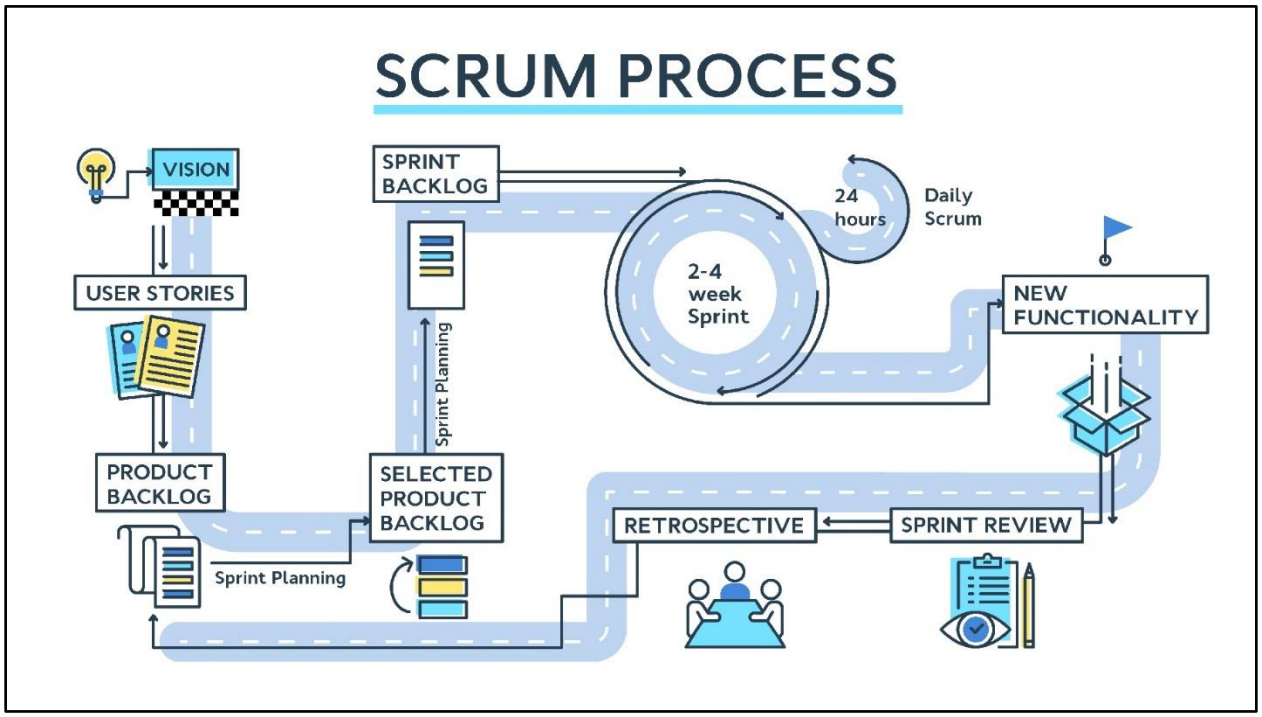


Figure 114: Selected Methodology: Scrum Model

Because of its emphasis on client participation, flexibility in the face of change, and iterative development and visualization to boost productivity, the Scrum model is a solid approach to take while creating an application. The second most important factor was the fact that this approach views the app service as an ongoing endeavor, rather than a one-off endeavor.

It is possible to accomplish a project using the Scrum process even though it requires the Scrum Master, the Product Owner, and the engineers to all wear multiple hats at once. Being an individual contributor, I anticipate alternating between scrum master and product owner at the beginning of each sprint, and then between front- and back-end engineers as the sprint develops. In the future, the client's role will be played by the individual client, and I will be able to facilitate these changes by consulting with my project managers in my capacity as Scrum Master. This will enable me to maintain a steady stream of output in both the development and strategic planning domains, and to visualize all the backlogs, tasks completed, and tasks left undone. In this model, each module must be completely ready to be used before it is passed on to the next sprint.

7.14. Online Forms Survey

The bulk of the project's needs and information was gathered through client meetings and integration, but it was still important to know how the public felt about the system and its uses, thus an online survey was also necessary. In order to facilitate public discourse, the suggested application offered queries about the system's capabilities and appropriateness in the present setting. The following is a list of questions that were specified in the survey, also a brief description on each of them will further on be provided.

Google Forms was utilized to prepare and arrange a set of questions to facilitate public with their opinions on the respective futsal app and highlighting all the major features on consideration with its key implementation. Further on these forms were presented to a set of people and a total of 76 responses were collected.

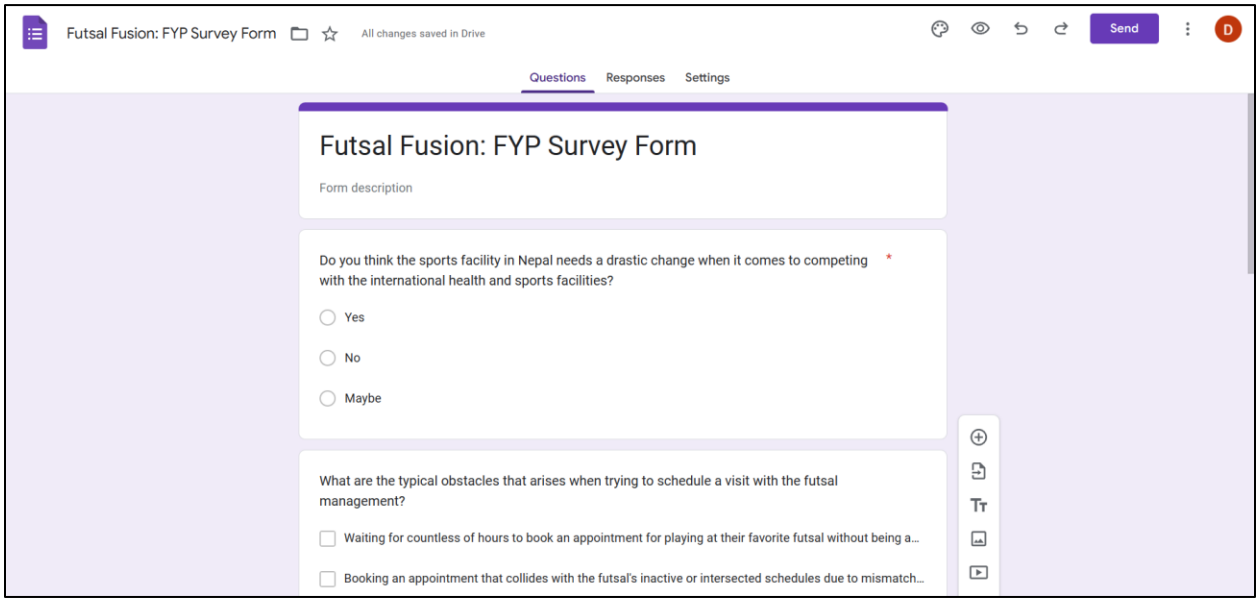


Figure 115: Online Survey: Preparation

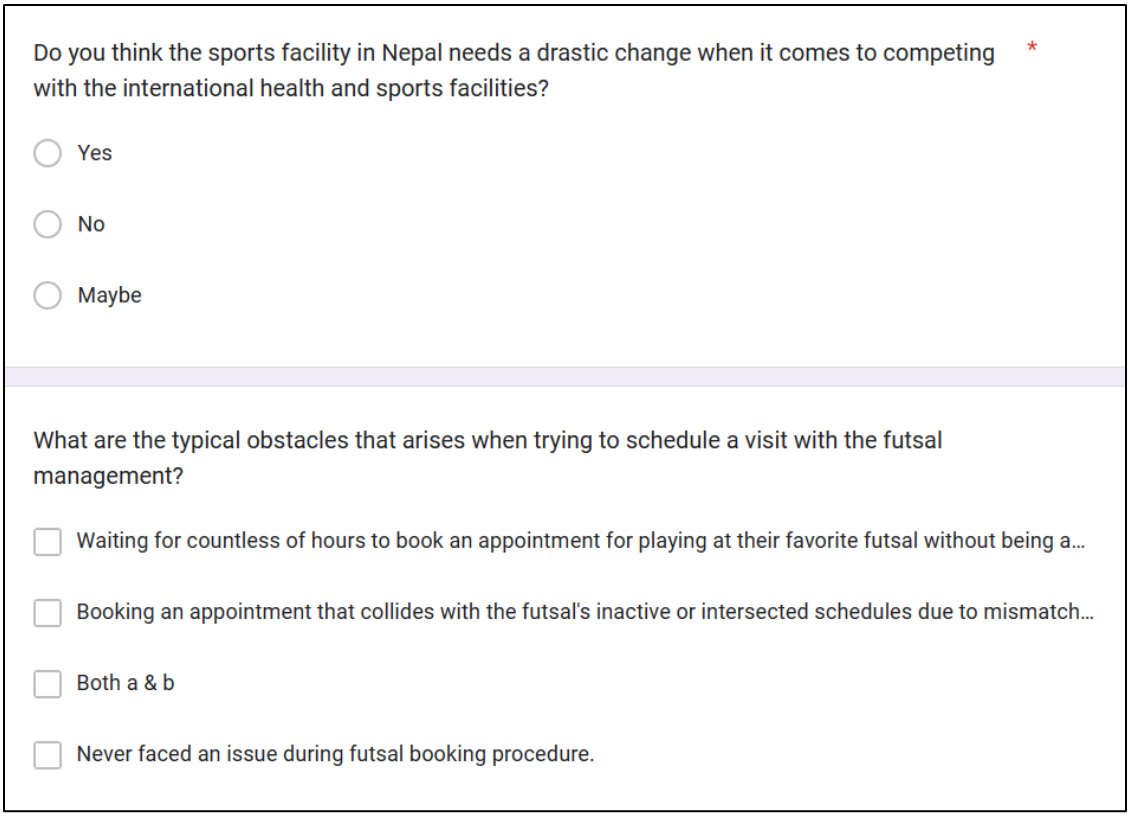


Figure 116: Online Survey: Questionaries 1

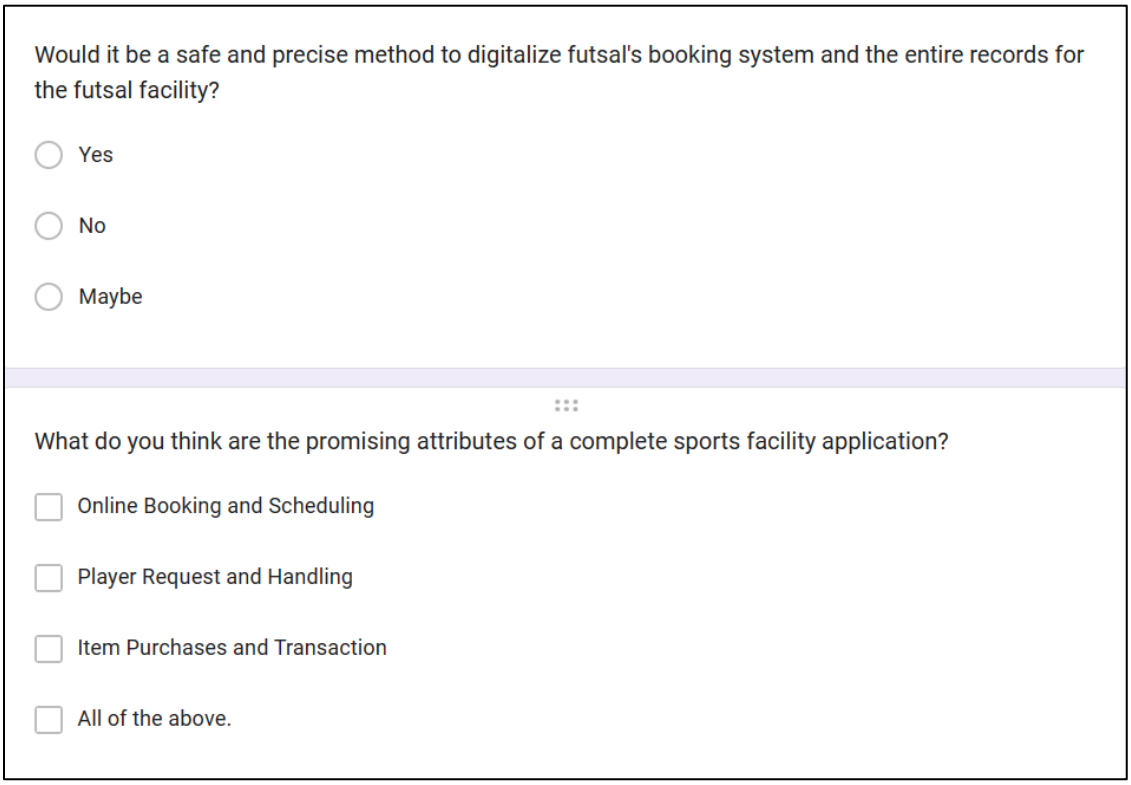


Figure 117Figure 96: Online Survey: Questionaries 2

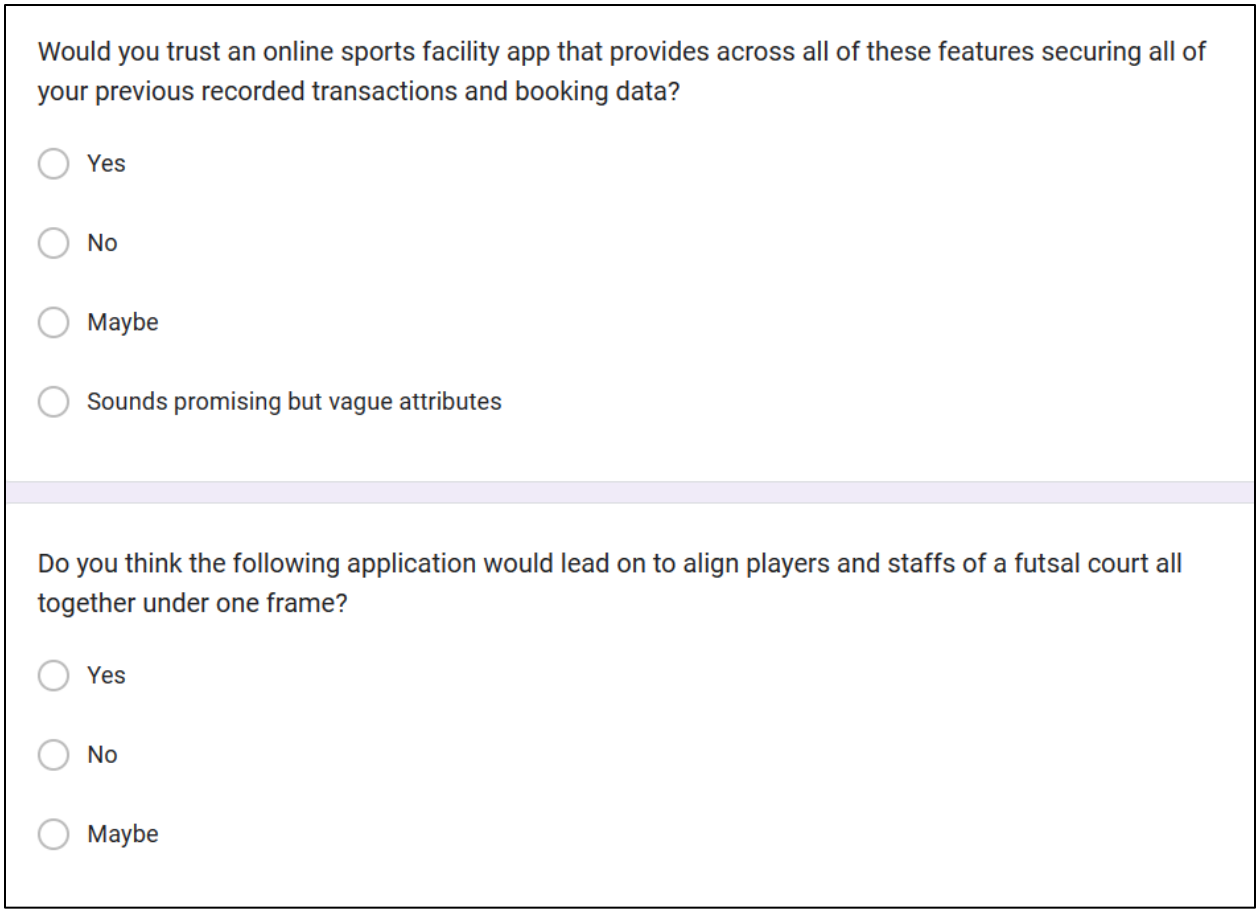


Figure 118: Online Survey: Questionaries 3

Survey Responses

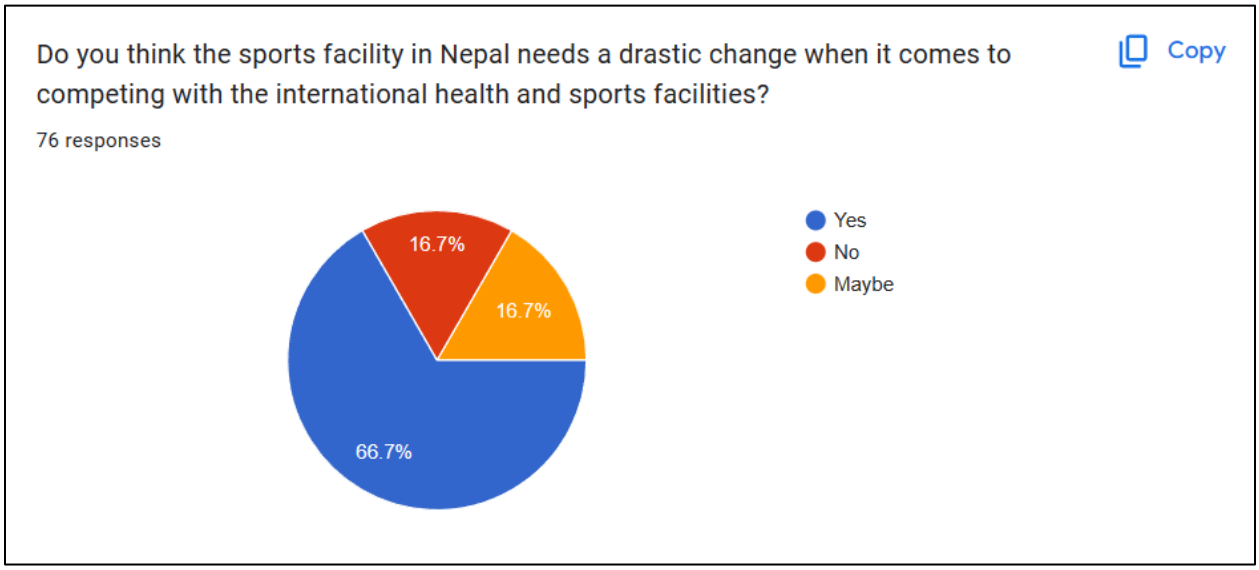


Figure 119: Survey Response: Question 1

Result: Based on the conducted survey, most of the respondent acknowledged a requirement for a change in sports and facility industry, so it does provide a wonderful opportunity to work on a system addressing the situation and the general's needs.

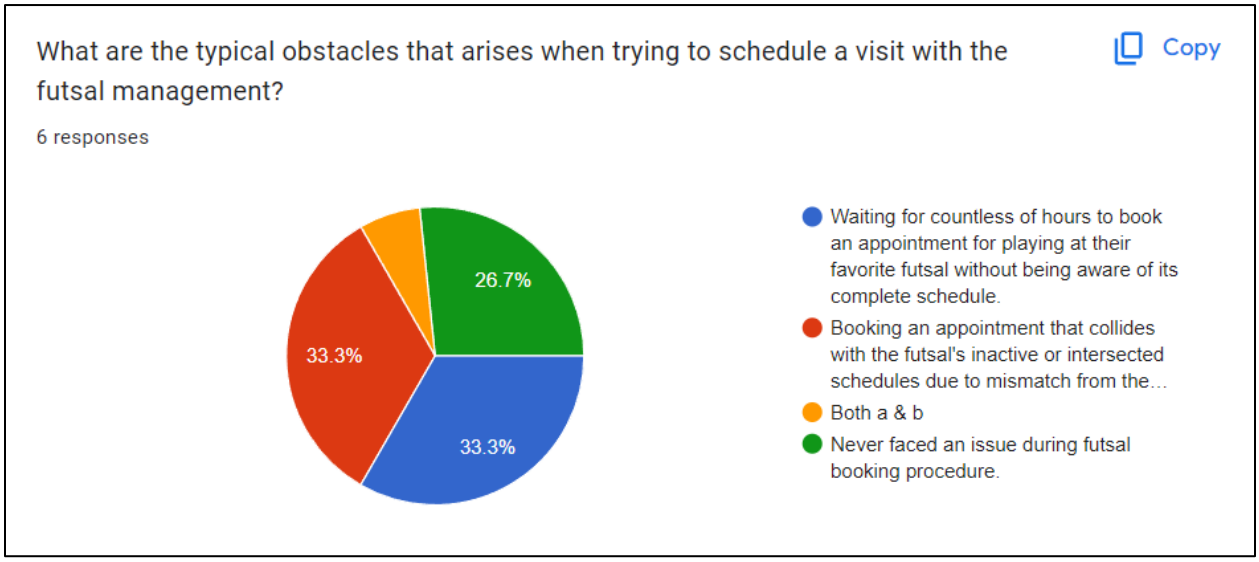


Figure 120: Survey Response: Question 2

Result: More than a quarter of survey respondents found challenges in arranging and maintaining appointments to be an inconvenience, making this program a needed aid.

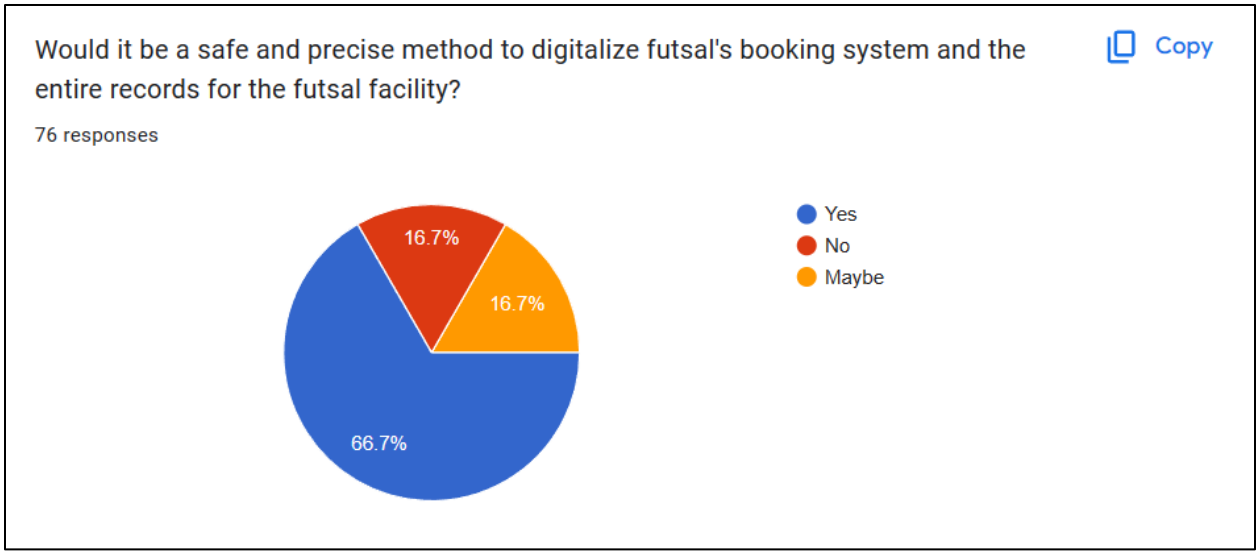


Figure 121: Survey Response: Question 3

Result: Even if the application focuses on digitizing a player's and their respective futsal's day to day records, it can seem vague to some point as everything would be digitized, making it sensible to allocate on what the majority wants.

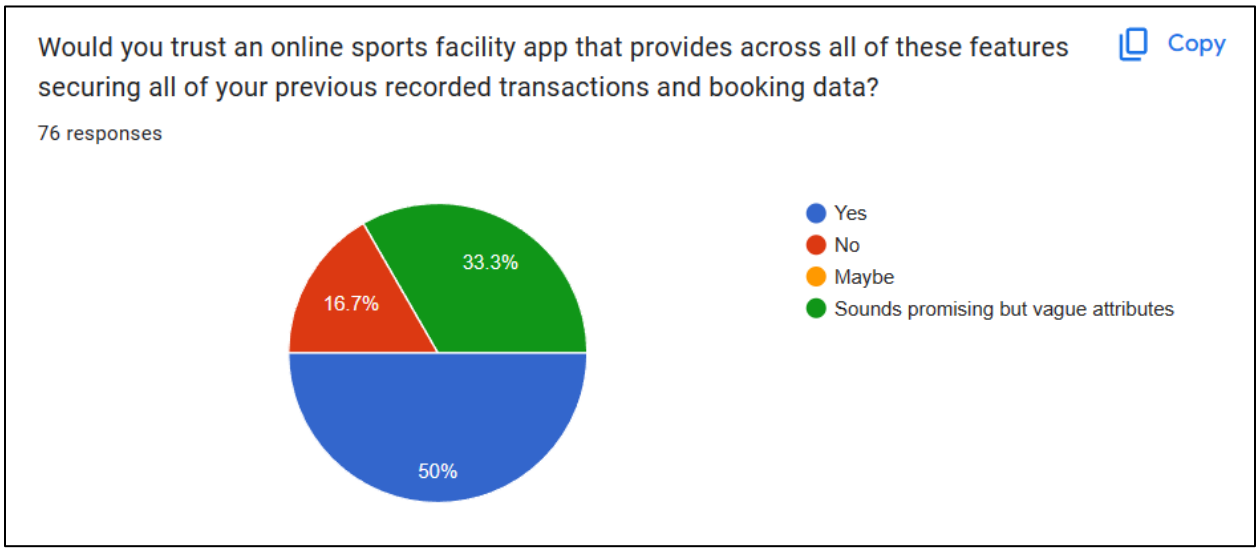


Figure 122: Survey Response: Question 4

Result: As a huge amount of people were surveyed regarding the application's features and feasibility, it was obvious that not all would agree and approach to the presented ideas. However, the realistic approach did trace out that half of them did require a change in the sports and facilities of Nepal where they could trust with all of their sensible data as well.

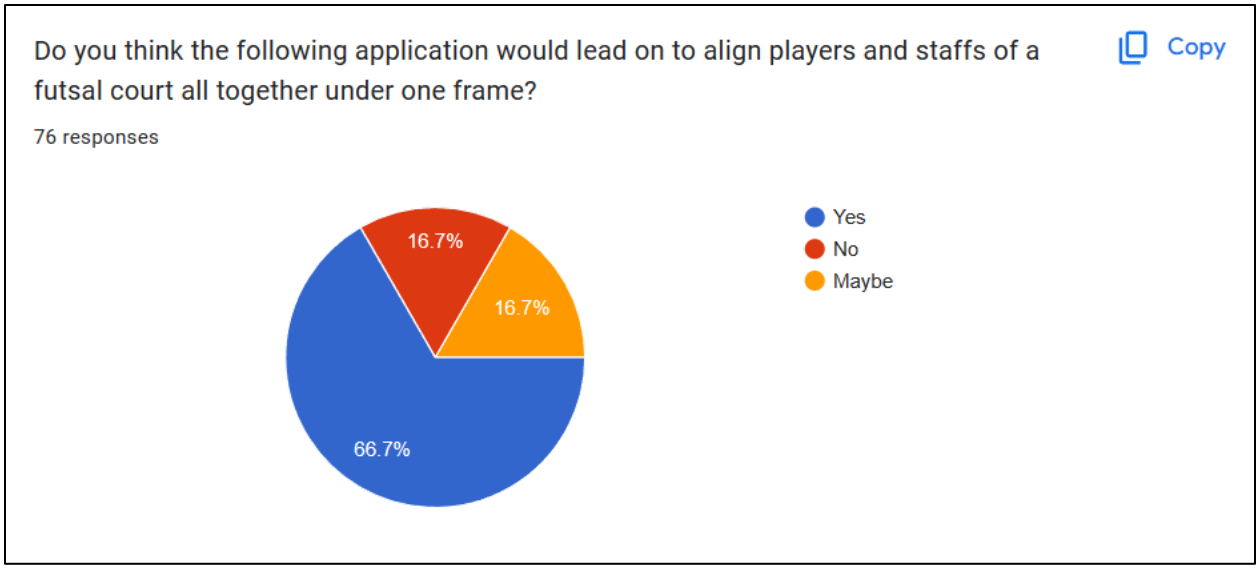


Figure 123: Survey Response: Question 5

Result: One of the major focus of the app would be co-aligning futsal enthusiasts and individuals together under a single frame, which is suggested by the app's approach and majority of them did approve their wants and needs to centralize players and staffs.

7.15. Development and Logs

According to the schedule, Git will be used as the standard version control system to optimize the application's development journey, track its progress, and monitor the repository platform. In a similar vein, GitHub will be the engineered platform that incorporates version control to keep tabs on progress, identify errors during program compilation, and roll back to a previous commit. Rest assured that your development work will not be compromised in the event that the local device becomes corrupted or lost, as the source code will be stored in a secure, remote repository accessible through a reliable network.

As the first sprint has already been completed, which included allocating development and implementation of business logic over its repositories and services, and also the process of setting up the project's architecture has been finalized, the changes can be viewed over the Git history logs and source codes. However, in order to protect sensitive information, the repository has been set to **private**. This means that the source code will not be publicly available. The relevant supervisors will have access to it so that they can monitor the project's development as needed.

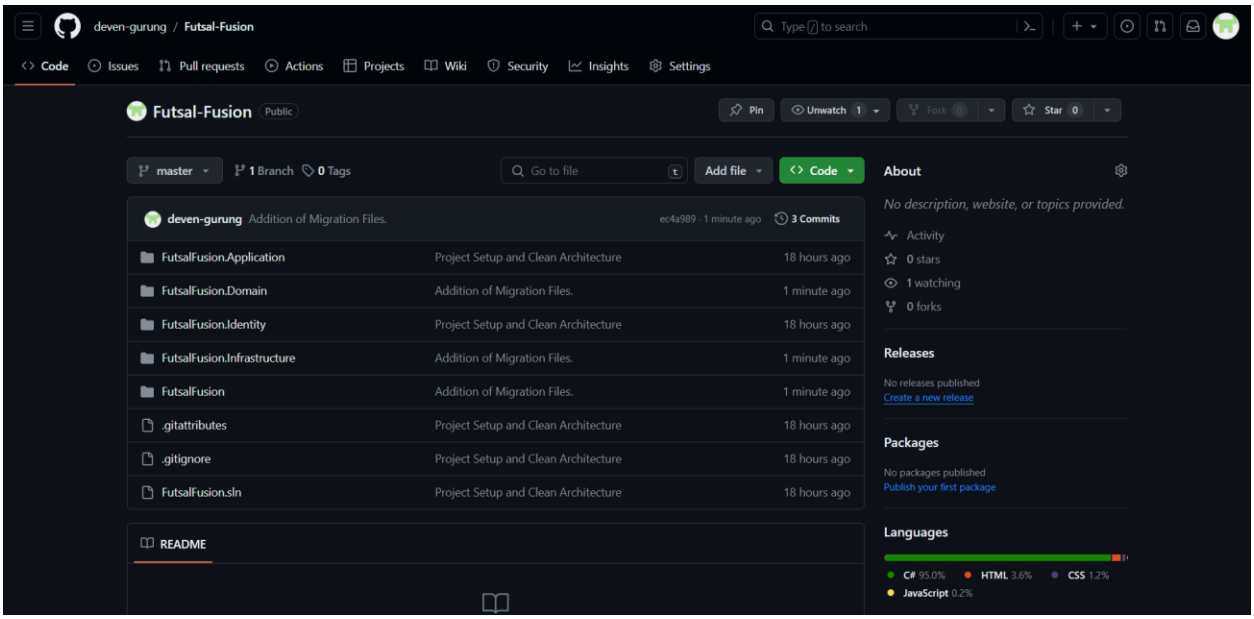


Figure 124: Git Log: Application Repository

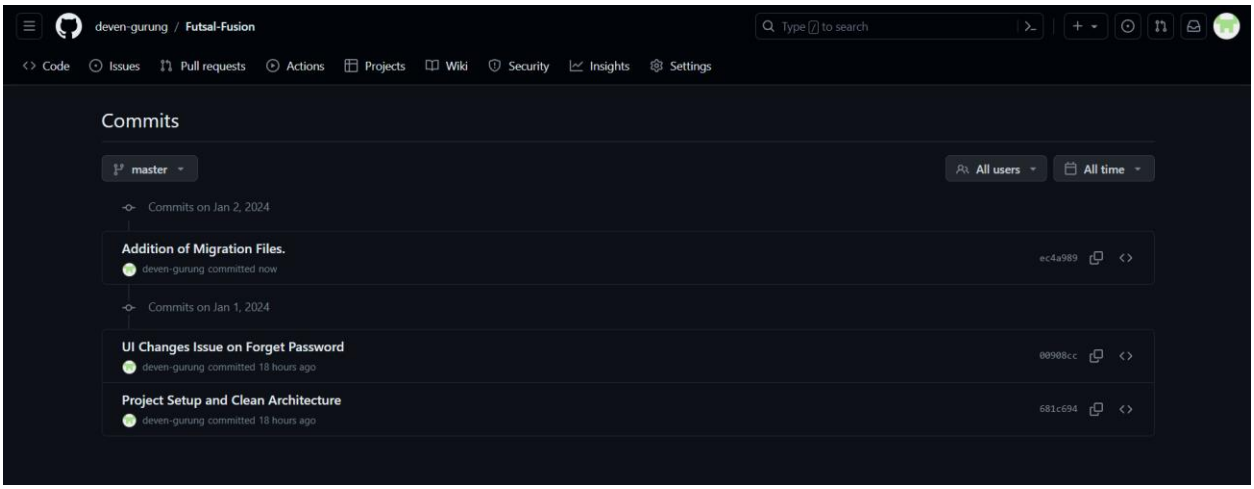


Figure 125: Git Log: Commits

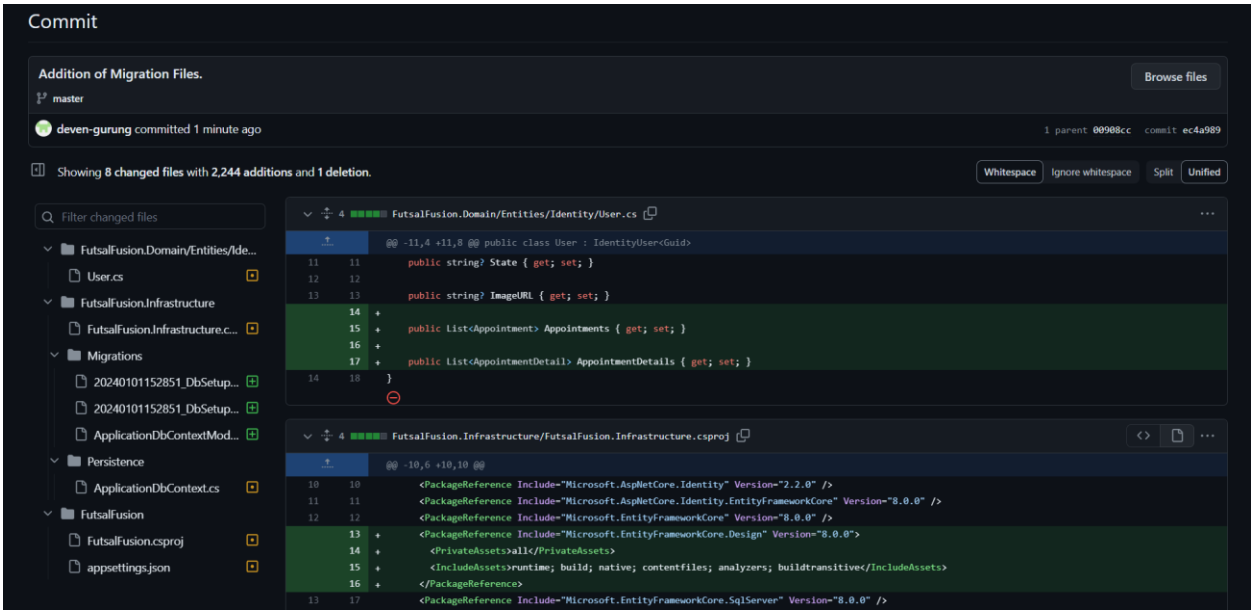


Figure 126: Git Log: Committed File

7.16 Client Approval Letter

Kapil Gurung,
Dhuku Sports Hub,
Baluwatar, Kathmandu

Date: 25th November 2024
Deven Gurung
Futsal Fusion: Futsal Scheduling & Booking App
Islington College
Kamalpokhari, Kathmandu

Subject: Permission Granted for the Role of Client

Dear Deven,

I have reviewed your request for the role of client, and I am pleased to inform you that I have agreed to be your client for the project of the application. We are willing to provide the requirements and the necessary information that we need in the project application. Considering our recent phone conversation, I am prepared to supply all of the details and requirements listed in the (FYP Project) application. Call me or set up a meeting so we can talk about the problems and show you where the app development is so far if you have any questions or need any more help.

Sincerely,
Kapil Gurung,
Deven Gurung,
Futsal Fusion: Futsal Scheduling & Booking App,

Kapil

Signature

Figure 127: Futsal Fusion: Client Approval Letter