Golf Player Time Manager

Alyssa Beeker, Bailey Carlin, Daniel Teel, Elijah Hunt, Adam Horle

**Table of Contents**

1. Project vision

1.1. Backgrounds

1.2. Socio-economic Impact, Business Objectives, and Gap Analysis

1.3. Security and ethical concerns

1.4. Glossary of Key Terms

2. Project Execution and Planning

2.1. Team Information

2.2. Tools and Technology

2.3. Project Plan

2.4. Best standards and Practices

3. System Requirement Analysis

3.1. Function Requirements

3.2. Non-functional Requirements

3.3. On-Screen Appearance of landing and other pages requirements.

3.4. Wireframe designs

4. Functional Requirements Specification

4.1. Stakeholders

4.2. Actors and Goals

4.3. User stories, scenarios and Use Cases

4.4. System Sequence / Activity Diagrams

5. User Interface Specifications

5.1. Preliminary Design

5.2. User Effort Estimation

6. Static Design

6.1. Class Model

6.2. System Operation Contracts

6.3. Mathematical Model

6.4. Entity Relation

7. Dynamic Design

7.1. Sequence Diagrams.

7.2. Interface Specification

7.3. State Diagrams

8. System Architecture and System Design

8.1. Subsystems / Component / Design Pattern Identification

8.2. Mapping Subsystems to Hardware (Deployment Diagram)

8.3. Persistent Data Storage

8.4. Network Protocol

8.5. Global Control Flow

8.6. Hardware Requirement

9. Algorithms and Data Structures

9.1. Algorithms

9.2. Data Structures

10. User Interface Design and Implementation

10.1. User Interface Design

10.2. User Interface Implementation

11. Testing

11.1. Unit Test Architecture and Strategy/Framework

11.2. Unit test definition, test data selection

11.3. System Test Specification

11.4. Test Reports per Spring

12. Project Management

12.1. 11.1 Project Plan

12.2. 11.2 Risk management

13. References

**1. Project vision**

1.1. Background

The name of our project is Gold Player Time Manager (GPTM). GPTM has two components, a cloud-based player management application and a mobile phone application. The issue we are aiming to resolve with golf is the player frustration due to the lack of easy coordination between each player on a course. With the nature of golf being played on a large field, players cannot reach each other to determine wait times themselves. This is also an issue that golf course staff cannot currently alleviate effectively due to the lack of online communicative technology, as they still rely on manual solutions to inform players. Not only does this lack of technology affect the issue of wait times, but players also cannot easily request assistance or services. Golf course players and staff both lack effective means of communication involving modern online technology.

*The purpose of this project is to help golf players enjoy a smoother and easier game, and to offer quality of life features to golf course employees. The cloud component will track players locations on a virtual golf course to optimize playing time by calculating waiting times and playing speed. On the mobile app, players will be sent messages to advise them of their playing time and to allow other players to pass. Players can also view helpful tips based on location and request assistance or services using the mobile app. These requests can be viewed by employees working at the golf course, further alleviating common issues on golf courses.* (Not sure if we should include the solution in the background, but there is seemingly no other section.)

1.2. Socio-economic Impact, Business Objectives, and Gap Analysis

Our app will improve the down time between rounds for players, this optimization will make playing golf at a participating location more favorable. This means better business for a golf course that utilizes our app as well as the ability to host more players and games. It also gives players easier access to a golf course’s services, boosting sales for refreshments and additional rounds.

1.3. Security and ethical concerns

Since GPTM relies on GPS location tracking, location privacy is the largest security concern. To avoid attacks, we will be using two secure GPS services, Smart Location and the Google Maps API. Another concern is account information privacy, because we require our players and admins to log in to use our services.

The biggest ethical concern of our project is that it introduces more work for golf course employees. This can heighten stress levels for employees and thus it is important for us to make our app highly functional.

1.4. Glossary of Key Terms

* Golf Player Time Manager (GPTM): This is the name of our project, abbreviated as GPTM.
* Golf Player/Player: This refers to the person participating in a game of golf in real life. They do not work for the golf course and are using our app as a standard user. This type of user is referred to as a “Player” in our design.
* Golf Course Employee: This refers to the person who is employed at a golf course and will be involved in our app through golf course management, tending to services and requests.
* Golf Course Manager/Admin: Refers to the person that sets a request to have their golf course be geofenced as well as us (GPTM team)
* Application Programming Interface (API): a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service. More specifically, in this project, things like Firebase, Google Maps, etc.
* Global Position System (GPS): Navigational system that tells the player where they are and if they are in close proximity to or at a golf course
* Geofence: Geographical location in which some arbitrary code will execute / player is given special permissions, i.e. in this case access specific golf course pages

**2. Project Execution and Planning**

2.1. Team Information

The members in our team are Alyssa Beeker, Bailey Carlin, Daniel Teel, Elijah Hunt, and Adam Horle. Together our team has a strong background in mobile and web app development, as well as user interface design. As a team we are adept at Java, Javascript, HTML, CSS which are required for GPTM.

2.2. Tools and Technology

The primary languages we will be using are Java for the mobile app, and Javascript and HTML with CSS for the web app. For our IDEs we will be using Android Studio and the text editor of each team members choice for web development. We chose Android Studio because it is Android's official development software and some team members have experience using it. We will use Microsoft Visio for flowchart design. We chose this because it is accessible to each teammate and is a simple diagramming tool everyone is familiar with. Our AI technology will be from Tensorflow, specifically we will be using Tensorflow lite because it is open source. For GPS services, we will be using Smart Location and the Google Maps API. Smart Location is an android library project which will simplify the usage of location providers and activity recognition. Google Maps API offers a map for us to use in our application for our location features. Our databases are SQLite and Firebase. SQLite is a common database management system that our team is familiar with. Firebase is another backend service that allows us to work on both our mobile and web application, and offers account authentication. Each tool was specifically chosen because it was compatible, easy to understand, and modern.

For collaboration and communication, we will be using Github for project sharing, Google Drive for all of our written and visual documents, and Trello for planning and delegation. For further communication we have set up an accessible chat group for the team to use.

2.3. Project Plan

2.4. Best standards and Practices

Rather than repeating code in multiple places make a function. Write code as efficiently as possible. Functions for common operations in the app will enable scalability and make things a lot more manageable. When working in Angular linting before a push will promote standardization of code.Only one person should be working on one component at a time to decrease merge conflicts. Also, merging master into the current working branch will help with the prevention of merge conflicts. Leaving comments on code where the purpose or flow is apparent is necessary to enable other team members to start working on the code quicker.

**3. System Requirement Analysis**

3.1. Functional Requirements

Administrators need to be able to register a new course and edit the details of the course. Administrators also need to be able to see current players and their locations as well as game requests.

Players need to be able to see the course details along with wait times. Players also need to be able to make game requests. If they have a registered account then players need to be able to register their scores.

3.2. Non-functional Requirements

The application will have two separate components with a simple and intuitive user interface for each. Information needs to be displayed in an easy to read manner.

3.3. On-Screen Appearance of landing and other pages requirements.

The landing page for the Administrator website needs to have all the details of the course divided into the main categories (player details and course details). When an administrator logs in it should be apparent which tab they need to navigate to in order to manage their course efficiently.

3.4. Wireframe designs

**4. Functional Requirements Specification**

4.1. Stakeholders

Golf Players

Golf Course Managers

Golf Course Employees (Caddy)

4.2. Actors and Goals

Player

Admin

Database

Cloud

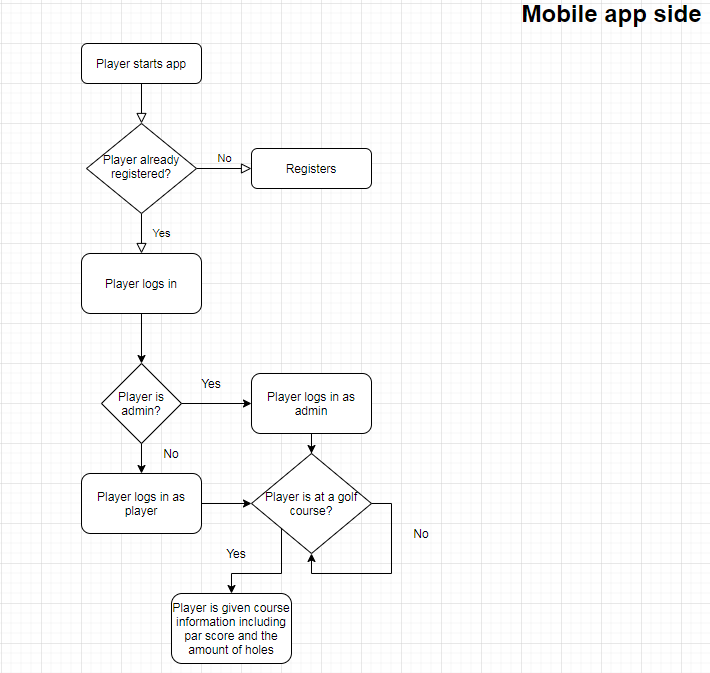
Web App

Mobile App

4.3. User stories, scenarios and Use Cases

4.4. System Sequence / Activity Diagrams

4.4.1 Mobile App Activity Diagram



**5. User Interface Specifications**

5.1. Preliminary Design

5.2. User Effort Estimation

**6. Static Design**

6.1. Class Model

6.2. System Operation Contracts

6.3. Mathematical Model

6.4. Entity Relation

**7. Dynamic Design**

7.1. Sequence Diagrams.

7.2. Interface Specification

7.3. State Diagrams

**8. System Architecture and System Design**

8.1. Subsystems / Component / Design Pattern Identification

8.2. Mapping Subsystems to Hardware (Deployment Diagram)

8.3. Persistent Data Storage

8.4. Network Protocol

8.5. Global Control Flow

8.6. Hardware Requirement

Administrators need a device that has access to the internet. Players need an Android device that is recent enough to be able to download and run apps from the Google Play Store.

**9. Algorithms and Data Structures**

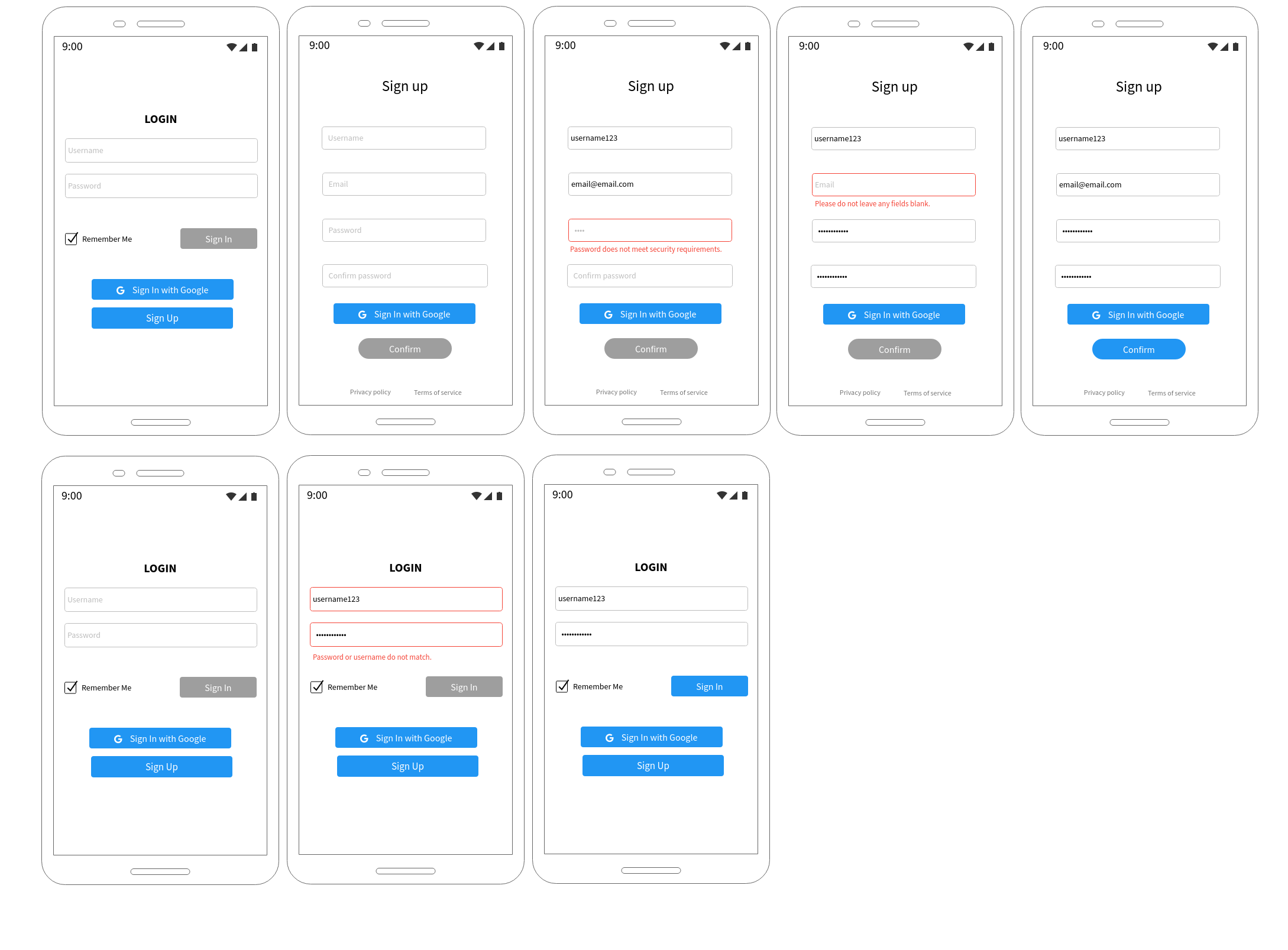
9.1. Algorithms

9.2. Data Structures:

Firebase (database)

**10. User Interface Design and Implementation**

10.1. User Interface Design

Before every development task, we have multiple tasks for design. We first start with a user story to describe what we want a particular feature to do. This is broken up by the user and events involved. Then, we design a mockup based on the user story. This mockup isn’t an exact representation of what the app will look like but it provides the developer a visual aid to work off of.

10.2. User Interface Implementation

**11. Testing**

11.1. Unit Test Architecture and Strategy/Framework

1. Look at the User Stories to determine what the expected behaviour is.
2. Test the scenarios that will yield the expected results and record these scenarios.
3. Test fringe scenarios that could potentially occur with the validation and functionality in mind and record these scenarios.
4. If there are any scenarios that yield unexpected results file a bug ticket with the scenario and the environment in the description.
5. Reguarily test the scenarios that have filed bugs to see if they were fixed. If so, mark the issue (bug) as closed. If not, keep the issue open.

11.2. Unit test definition, test data selection

11.3. System Test Specification

11.4. Test Reports per Sprint

**12. Project Management**

12.1. 11.1 Project Plan

12.2. 11.2 Risk management

**13. References**