Team 5 Design Document
Bowling Statistics Tracker

Team Members: Brandon Loi, Hunter Sullivan, James Smagacz, Aaron Althoff, Aaron Nordhoff
Professor Turkstra
CS307

Purpose

The Purdue Bowling Team uses different software to track players and their performance, however they are unsatisfied with the solutions currently available. Our project aims to create an Android app that uses key features of pre-existing bowling apps, and consolidates these features along with additional components into a centralized mobile application.

The application, named "Bowling Statistics Tracker", will be developed for Android phones using Java for both the front and back end. Additionally, user profile information, such as statistics, will be stored in a MySQL database. This database will be hosted on a Raspberry Pi 3 Model B, allowing users of the Android app to retrieve information from the server.

Bowling Statistics Tracker will include a number of features, including:

- Performance tracking
- Live scoring
- Player comparisons
- Player grouping
- Scheduling
- Announcements
- Calendar system

Design Outline

The main design of our project will be of the client-server model, with the database storing players and their data as the server and the android application as the client through which users can access and view this data.

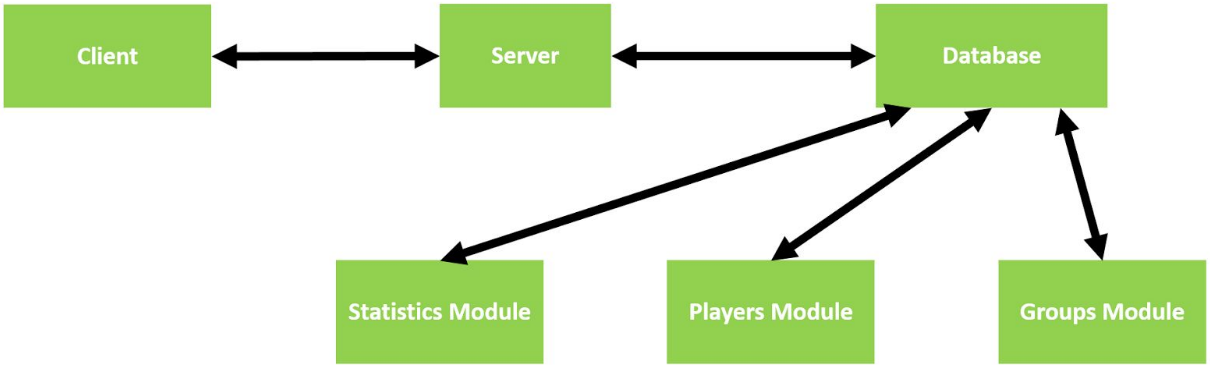
1. Tracking Statistics
    a. Any player can edit their own statistics. These statistics are: strike, single pin spare, multi pin spare, split leave, and filled frames (all measured in percentage).
    b. Coach accounts are also able to edit the statistics of players registered under them
2. Announcements
    a. Coaches can send announcements to students registered under them. These announcements will be present on the announcements page, which will be the initial screen loaded upon launching the application.
3. Calendar

   a. Coaches can add events to the calendar, which will be visible to all players in the given group. These dates will include the date and time of the event, as well as a description of the event.

4. Groups
   a. Any user can make a group consisting of multiple players.
   b. The combined statistics of the group are available to be viewed by all players within the group.
   c. Coaches can make groups of players registered under them.
5. Live scoring
   a. Players can live update the number of pins knocked down after each bowl, or at the end of the frame (in the correct sequence). The application will automatically calculate the current score after each insertion.
   b. Coaches can use the application to track scores during events, and have statistics dynamically calculated from these live scores

Application Overview

Following the structure of the client-server model, our application client will request data through a REST request to the server. The server will query the database and return the relevant information to the client.



Design Issues

Functional

- When should a notification of a upcoming event be pushed to a player as a reminder?
  - **A week before or immediately if the event is sooner**
  - Allow the coach to choose a date for the notification
  - A day before the event

  We chose to have it default to the first option because it automatically chooses for the coach and provides an immediate notification in the case the event is very soon. This leaves no room for user error as a coach may accidentally choose a day after the event to push the notification.

- Who has the privilege to create user groups?
  - Just the coach
  - Coaches and players, with players allowed to create groups that include coaches

    - **Coaches and players, but players cannot create groups that include coaches**
    - Just the players

  We chose to allow coaches and players to create groups. Coaches have the ability to include anybody in a group, including other coaches. Coaches may also create groups consisting of only players -- that is, there are no coaches in the group. Players can only create groups with other players, and must be a member of any group they create. This is to prevent players from creating unnecessary groups, flooding the coaches with groups that are not important, restricting coach groups only to those which are necessary.

- How should a player or coach login to the application?
    - **Through a username and password**
    - Through a Facebook authentication
    - Using a Google account

  We chose to use a simple username and password system of our own due to the complication of using the authentication from outside sources being difficult to integrate into our server.

Non-Functional

- What software should our database/server run on?
    - Option 1: SQLite
    - **Option 2: MySQL Community Server**
    - Option 3: Cubrid

  We decided to work with MySQL as our database software due to its wealth of documentation and support. There are many resources based on creating REST request APIs within MySQL which we can use as a starting point for setting up the connection between our client and server.

- How long should data be kept in the database before being deleted?
    - Option 1: Delete data after a set amount of time
    - Option 2: Delete data after a set amount of events
    - **Option 3: Delete all statistical data at the end of the semester**

  We decided to keep data until the end of the semester to ensure that data is not kept after it is no longer relevant, as well as getting rid of data from players that are no longer on the team. Because the data we are storing is not very memory-intensive, the database should have no problem storing the data for the time period of a semester.

- How do we add coach accounts?
    - **Option 1: Have one hard-coded in coach account and make others invite-only**
    - Option 2: Give coaches a string key to use on account creation
    - Option 3: Let anyone make a coach account

  We have decided the best and most secure way to limit who can make coach accounts is to initially create one default root coach account. Coach privileges will then be given by invite, starting with the default account.

Design Details

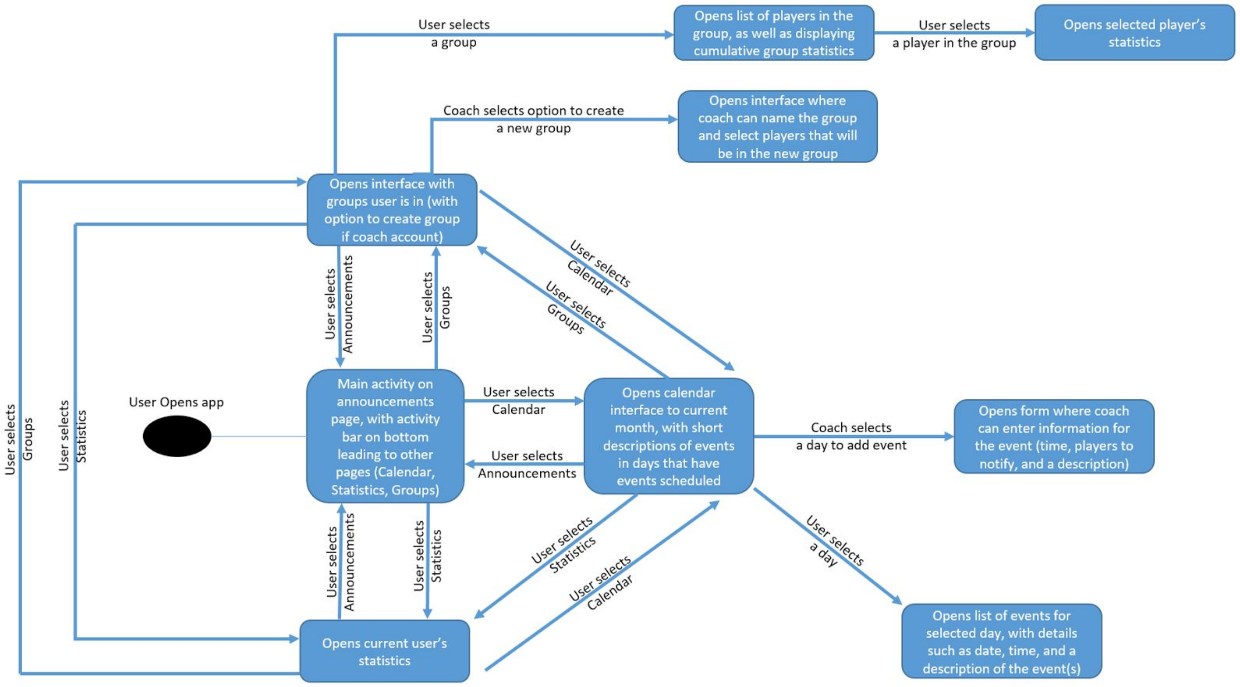
Coaches:

- Can create/end live event and edit past statistics from previous event.
- Can create user groups
- Can add events to the calendar
- Can notify players of upcoming events (push notifications)
- Can edit player statistics

Players

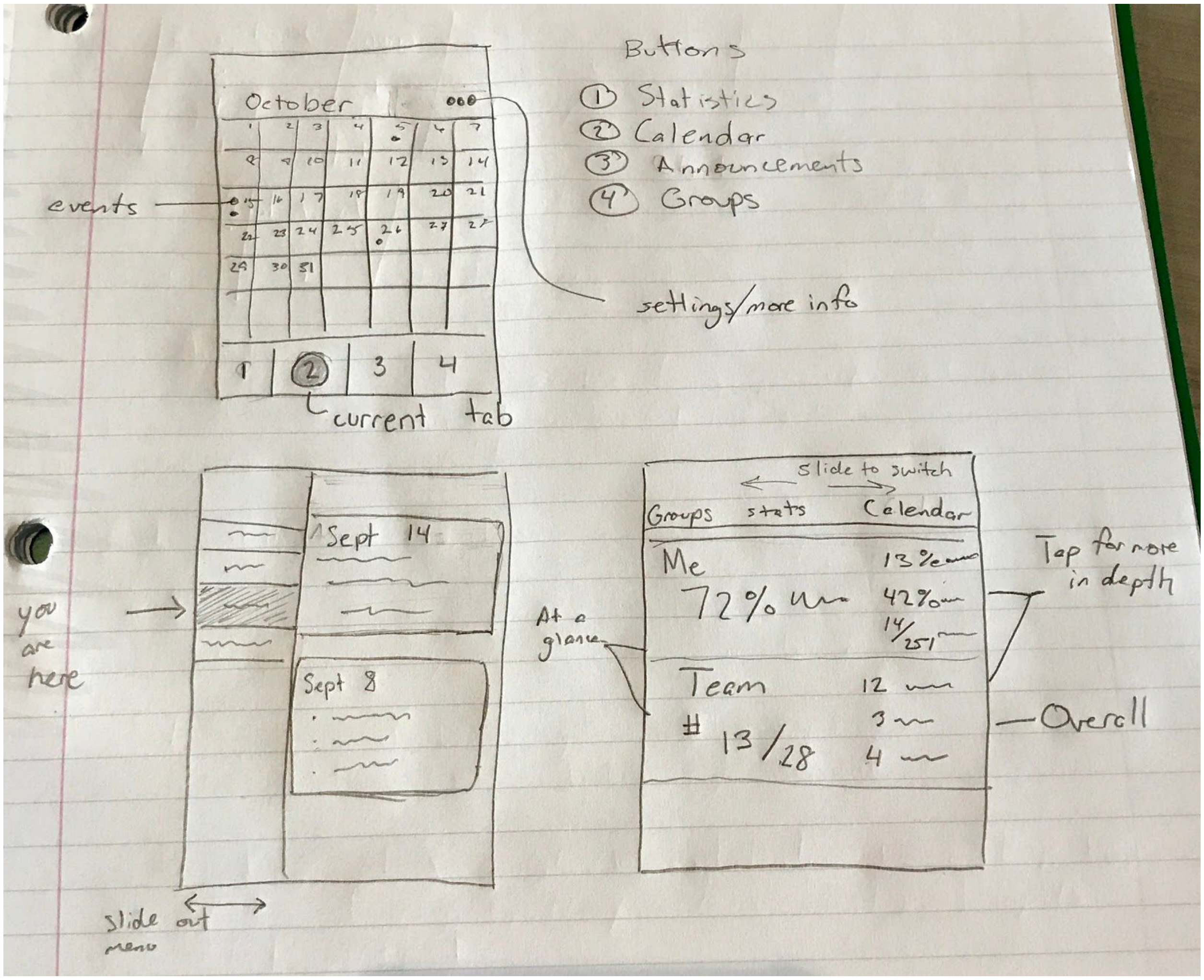
- Can create user groups but not with coaches.
- Can add statistics of their performance in live events as it happens.
- Can look up statistics of themselves, of the team, and of a group they are a part of
- Can view announcements
- Can view the calendar

Activity Diagram


User selects a group
Opens list of players in the group, as well as displaying cumulative group statistics
User selects a player in the group
Opens selected player's statistics
Coach selects option to create a new group
Opens interface where coach can name the group and select players that will be in the new group
Opens interface with groups user is in (with option to create group if coach account)
User selects Announcements
User selects Groups
User selects Calendar
User selects Groups
User selects Groups
User selects Statistics
User Opens app
Main activity on announcements page, with activity bar on bottom leading to other pages (Calendar, Statistics, Groups)
User selects Calendar
User selects Announcements
Opens calendar interface to current month, with short descriptions of events in days that have events scheduled
Coach selects a day to add event
Opens form where coach can enter information for the event (time, players to notify, and a description)
User selects Announcements
User selects Statistics
User selects Statistics
User selects Calendar
User selects a day
Opens current user's statistics
Opens list of events for selected day, with details such as date, time, and a description of the event(s)

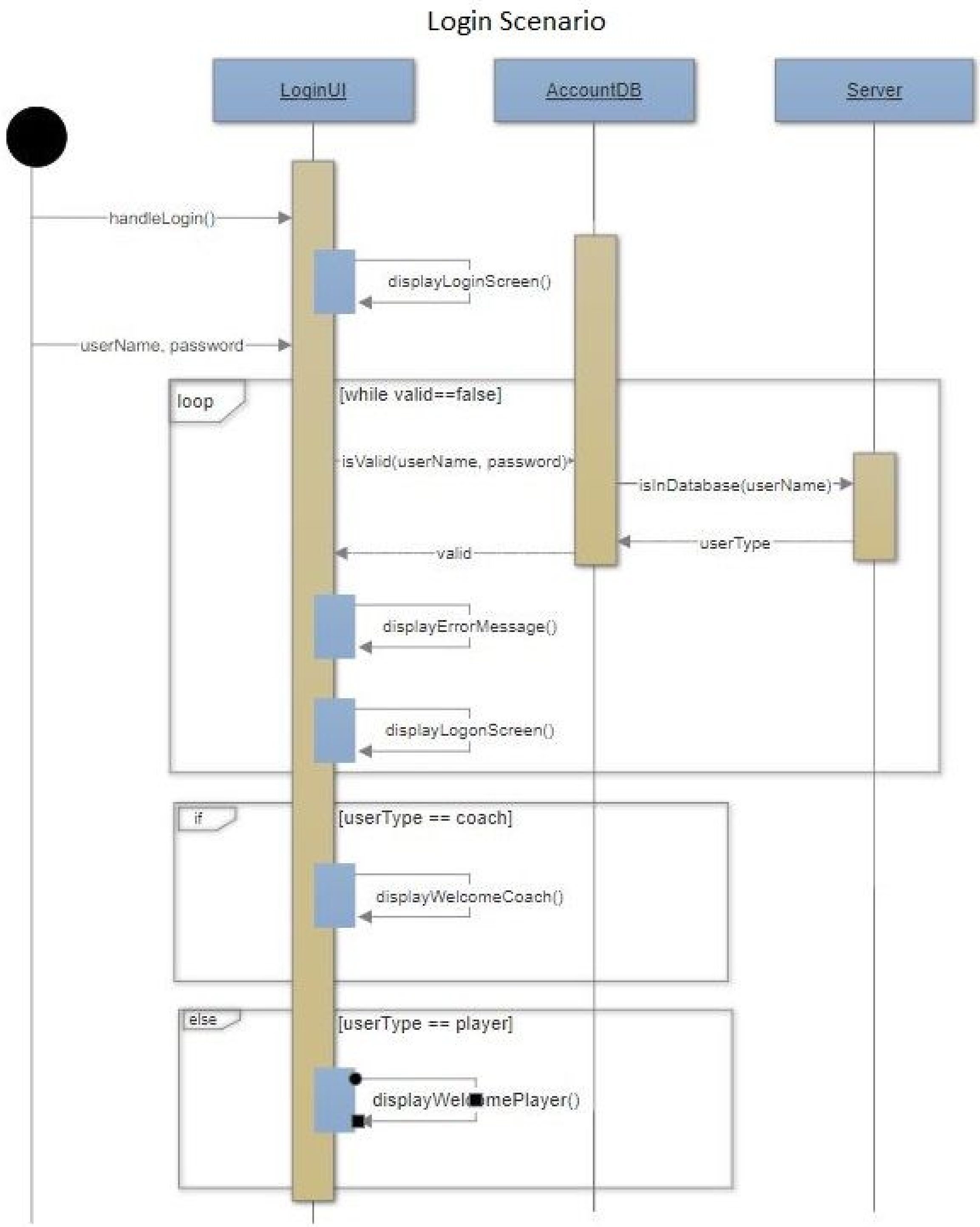
This is a rough overview of the interface mapping of the application. This map does not include the live scoring module for the sake of readability, however it will act similarly to the announcements, statistics, calendar, and groups interfaces in that it will be a separate module accessed by an activity bar/drop-down menu.

UI mock-up



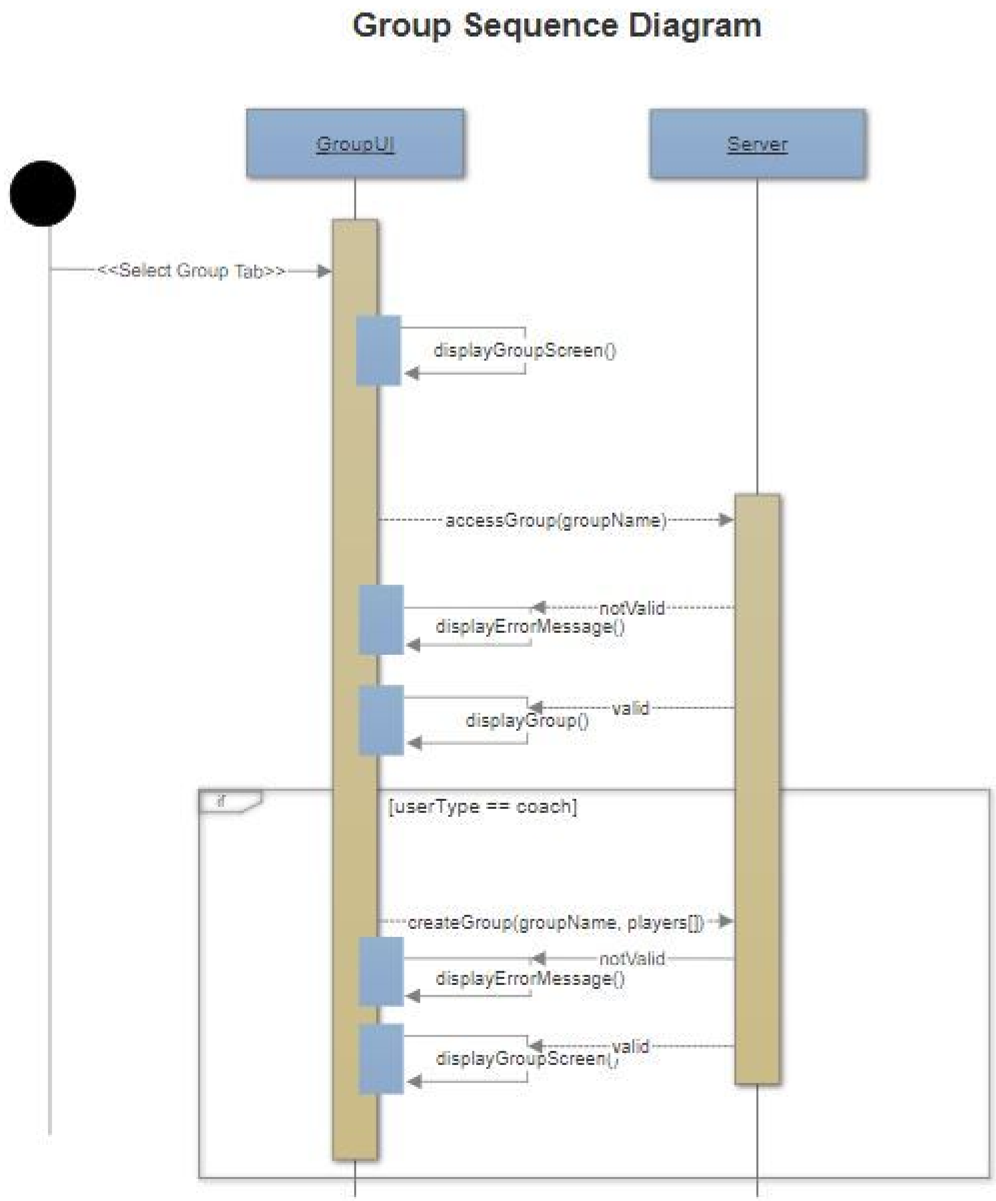
These are a few rough sketches for the potential User-Interface models we can use. The two main models from which we will choose are an activity bar at the bottom of the screen for users to navigate to the different modules, or a slide-out menu accessed from a more compact button to save screen space while the user is navigating a particular module.

Login Scenario Diagram



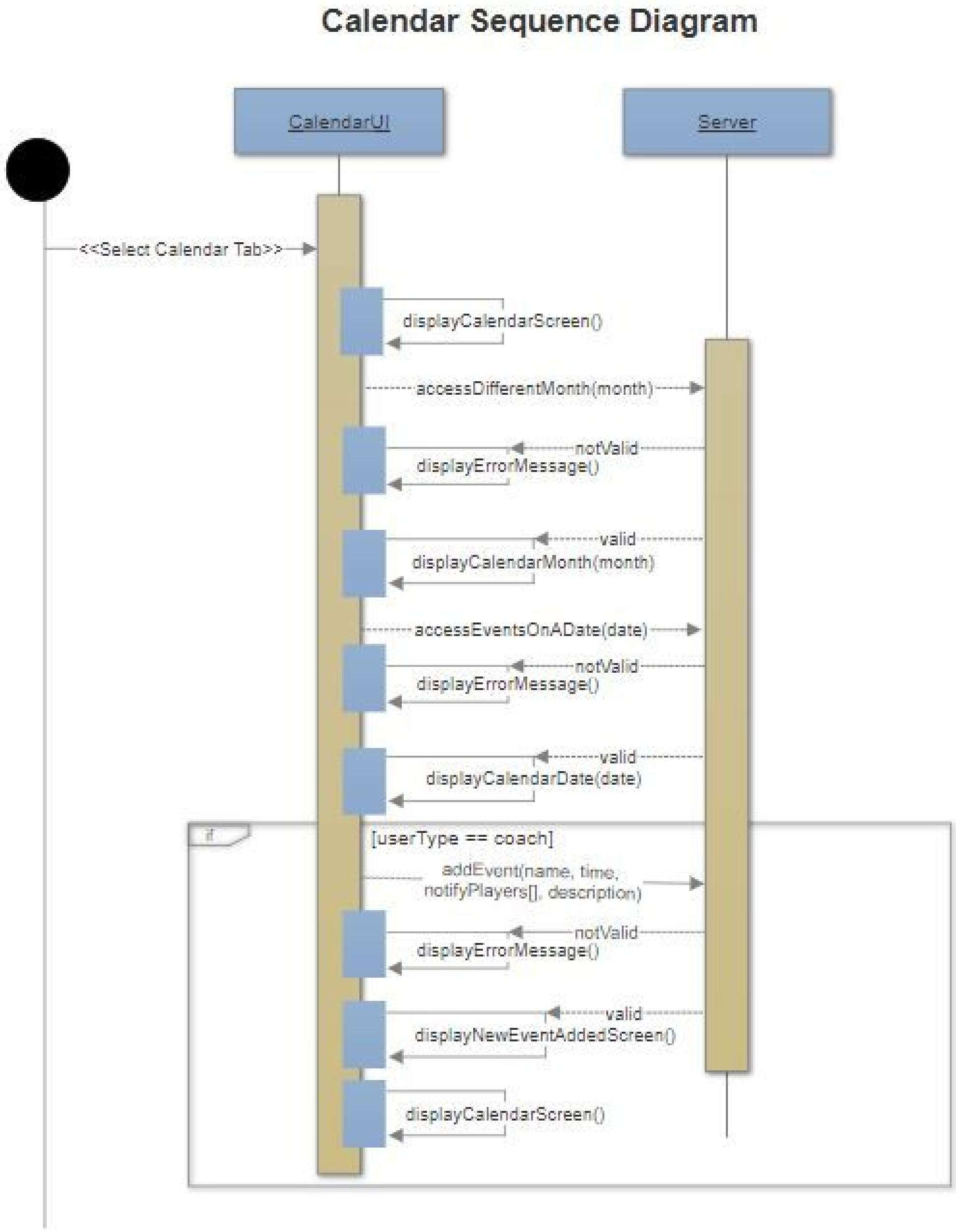
This is a rough overview of the user login process. The user will input their username and password, and the login UI will check with the account database and the server to ensure that the entered information is valid, repeating the login process if not. If the username and password are valid, then the user will be logged in and a specific welcome page will be displayed corresponding to their user type.

Group Sequence Diagram



The groups tab displays the groups that the user is a member of. If the user selects a group, the app displays details about that group. If the user is a coach, they can create groups, which will then update the groups page of the players added to the group.

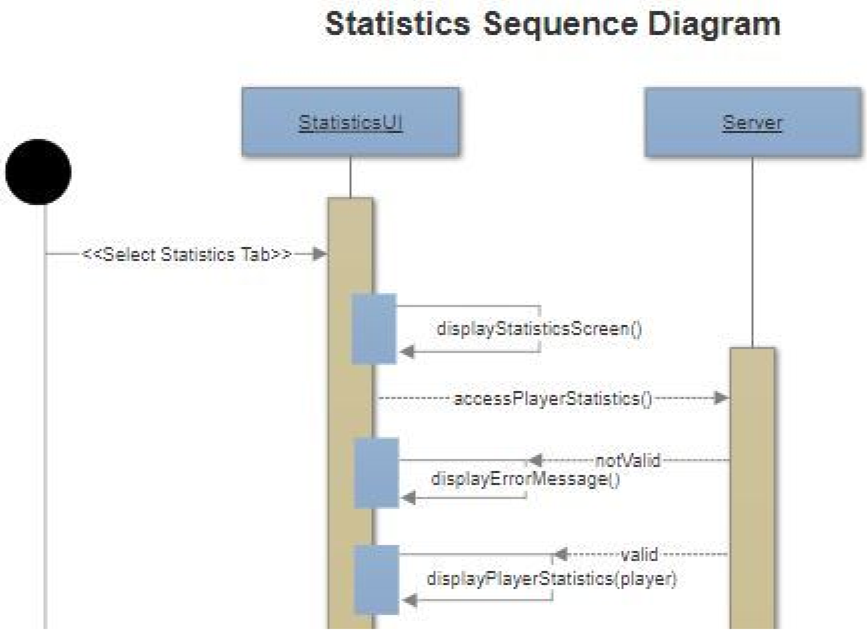
Calendar Sequence Diagram



Calendar Sequence Diagram

The calendar tab displays a calendar of events that the user is scheduled to participate in. When the user opens the tab, it displays a calendar of the current month. If the user tries to access a future or previous month, the app will display the calendar from that month. If the user selects a specific date, the calendar will display a list of events that the user has scheduled for that day. If the user is a coach, they can add an event to the calendar, which will then be added to the calendars of every user involved with the event.

Statistics Sequence Diagram


Statistics Sequence Diagram
StatisticsUI
Server
<<Select Statistics Tab>>
displayStatisticsScreen()
accessPlayerStatistics()
notValid
displayErrorMessage()
valid
displayPlayerStatistics(player)

The statistics page is fairly straight forward. When a user accesses the statistics tab, it displays their statistics, if they have any.