The City College of New York

Grove School of Engineering

CSc221 Software Design Laboratory

Fall Semester 2023

Final Project Report

MySportsLive App

Instructor: Professor Albi Arapi

Names: **Mahir Asef** & **Matthew Ching**

Table of Contents:

Introduction:

We are currently in the mix of 21st century modern technology where technological advancements keep pushing the boundary every single day and the most revolutionary piece of hardware that mostly makes it all possible is a smartphone. It helps us to view content and information instantaneously, that is either stored in the phone's memory or accessed from the web and software is what makes it possible to show contents on a phone's display. One of the most integral parts of software design is application or app. Apps make finding information that the user is looking for convenient and faster. However, there are different kinds of software in different operating systems (OS) that can run apps. Two of the most popular smartphone OS in the present day are Apple and Android; Android is the most used OS in the world. So, with that spirit of creating wonders in the competitive world of software development, this project will be about building an Android application that'll kickstart our journey to the real-world experience of becoming a software developer and their tasks. In this project we will create an Android application which we named "My Sports Live". The application will show various player statistics from multiple sports within a single application. We will use Android Studio and utilize the Android emulators to display the functionality of our app and present the optimized version of a complete 'ready to use' Android app to the best of our ability.

Objective:

The purpose of this project is to show the cumulative knowledge that we have gained over the course of studying computer science and in our Software Design class to ultimately build a program that appeals to real world usage. This project will showcase our knowledge in a higher-level programming language, while also teaching us about collaborative programming. Through this process, we will learn how software developers use all the resources available in their arsenal while collaborating with a peer to build programs that are efficient, easy to use, and convenient. This project will test and expand our knowledge of Java, a programming language, and our ability to use classes/subclasses and objects while also testing our capability for handling errors, and exceptions in programming. All this knowledge will compile together to test our understanding of back-end programming's ability to feed information to the front-end which ultimately will show our competence, as well as creativity of designing an interactive user interface of an Android application as well as showing the extent of our coding skills to make the application successfully run.

Information:

The Android application we are building is called "My Sports Live". The application is built to display various player statistics from different sports. Upon opening the application, a user is met with a sign-in screen. On this screen you can enter your username and password if you have a registered account and once a user can log-in, it will take the user to where you can search for sports or simply display all the options available. The user will then be directed to whichever option they choose. Depending on which option the User may be met with Buttons to view a sport or already be viewing a sport. Users will be able to select any sport they wish to find more details about by clicking on any sport's name, which takes them to the next page which will show a list of players for the selected sport. It will show the greatest players in their respective sports. Upon clicking on any sports' name, they will be able to view a detailed version of season statistics of an individual player. The player statistic page will be the last site within this application and the user will be given an option to log out of the page. If they click "sign out", they will be brought back to the homepage where if they want to re-enter the app, they will have to go through user authentication again. Each page within the app will also have a "previous" option which if the user clicks, they will be taken back to the previous page.

Software Structure:

This project will have two main parts. The front-end and the back end. The front-end part will be built using Android Studio. We will utilize everything provided within the Android Studio project. We created a blank Android project with min API 24 and min SDK 34. Once the work studio is ready, we will design the UI in resources using layout. We will have 6 different layout files for 6 designs which include homepage, search page, menu page, NFL page, La Liga page and NBA page. The sports pages are statistics pages which is the main purpose of this app. We will also have separate java files under "java" to program the functionality and the commands we want our app to execute as well as connecting the content layout pages with the back-end code. We will also have "Android manifest" file where we program our code to be able to launch the activities in the project. As well as connecting to the internet. Another important part is "build" Gradle where we include all our dependencies, implementations and all the building features. Finally, we will use an Android emulator which allows us to select and launch a virtual pixel device of our choice. The emulated pixel device will start running upon running the application and launch the application on the phone's display to test the functionality and user interface of the app. The second part of the application is back-end. We will use Spring Boot that will run using the Maven library. We will launch our Spring Boot application using the IDE called "IntelliJ IDEA". The Spring Boot back-end will be used to mainly store the data that will be later fed onto the front-end. The Spring Boot application will contain all the data that needs to be displayed once the Android Studio project is running. All our files in Spring Boot will mainly be divided into 3 different sections named controller, model, and method. Service will have all the logic and methods of storing information that are built from the content of model. Model will provide all the user bodies. Lastly controllers will execute all the logic stored in the service to return or feed the data into an appropriate location. We connect our front-end and back-end using 'retrofit' implementations for the program to be able to run without fault.

Back-end:

As mentioned above, we will be using Spring Boot to store and feed data from back-end and our Spring Boot will mainly have two different type of content classes. User and Search. Each content will have their service, model and controller subclasses. The User classes will provide the necessary data/user credentials for a user to login. In User body, we will declare two private string variables called username and password. We will use the Lombok library to inject getter and setter method. In UserService class, we will kick start the functions with @Service package. This service class will have a public Boolean (String username, String password) function which takes a conditional statement. The conditional statement takes parameters for username and password and sets them to certain values and returns "true" for these exact string values. Otherwise, the Boolean function returns false. We will use "masef" and "mching" as two valid usernames where the passwords are “13579” and “02468” respectively. Then we move onto the UserController class which allows us to direct the logic created under service into another application which in this case will be Android Studio. We will use @RestController and @RequestMapping to start the program. Then we will use POST method (@PostMapping) to direct the public Boolean function into its desired destination. We will connect the service class with controller class using dependency inject (private UserService userService and the constructor). We will use @RequestParam for the String parameters of the function and then we simply return the methods created in UserService.

Next content is Search. The Search classes will provide the necessary data for a user to search for the item they are looking for in the application. In Search body, we will declare one private string variable called sportsName. We will use the Lombok library to inject getter and setter method. In SearchService class, we will kick start the functions with @Service package. This service class will have a public Boolean (String sportsName) function which takes a conditional statement. The conditional statement takes parameters sportsName and set it to certain string values and returns “true” for those exact string values. Otherwise, the Boolean function returns false. We will use “Football”, “Soccer”, “Basketball” and “All Sports” as valid search entries”. Then we move onto the SearchController class which allows us to direct the logic created under service into Android Studio. We will use @RestController and @RequestMapping to start the program. Then we will use POST method (@PostMapping) to direct the public Boolean function into its desired destination. We will connect the service class with controller class using dependency inject (private SearchService searchService and the constructor). We will use @RequestParam for the String parameter sportsName and then we simply return the methods created in UserService.

Front-end:

For our front-end development we used Android Studio. Being that we’ve only experimented with Android Studio in this class we had some of our own learning to do. The simplest page was the Startup Page as we had already seen that be done in lectures prior, we were able to follow that as a template for our design. The file named “user_activity” contained various text views and our logo. Along with that we had input text fields for users to input a username and password. If the username and password were not authenticated by the back-end on Spring Boot the page would show a text view displaying “Username or password not found!”. This text view was invisible until a failed authentication was shown. Upon entering a correct username and password or registering for an account you are met with a new screen where you can quickly search for a sport or view all our app has to offer. This page had a search feature since we wanted it to resemble a google search in a way. This feature with more time could be used to directly search for a player and could ease the process of viewing your favorite players. After selecting your option from our search, you will be met with a page of all our sports if you choose to view all the sports or already be in your selected sport. When selecting all sports, you will be greeted with a page asking you to click a button to select your sport. These buttons are rounded which we had to create an xml file for in the drawable directory.

Lastly you will see your selected sport; since these pages are copies of each other for uniformity we will not go into depth about each page. For each page you will see two rounded buttons on the top left and right. These buttons will serve as a way out of the sport you chose if you are done browsing. The button on the top left is a “Return” button where you can return to the sports page and pick another sport. The button on the top right is a “Sign Out” button which will sign you out and take you back to the Startup Page. Under those buttons you will see the sport’s logo; NFL, LaLiga, NBA followed by the “MySportsLive” logo. A text view under it will instruct you to view all your favorite player’s stats for the sport you chose. The statistics and images displayed are in a ScrollView which may only have one child in Android Development so you must use a different view inside it to add multiple text and image views. The statistics may vary by position and sport. For the NFL the quarterbacks, running backs, and receivers do not share the same stats as all the roles in football have various tasks to perform. For the NBA the players are grouped by position, and all share the same displayed stats since basketball can be positionless at times and everyone can do what the other position does. The LaLiga page has players displayed by their team all sharing the same stats as soccer can also tend to be positionless. The only exceptions to the LaLiga page come from honoring two of the greatest players to ever play for LaLiga in Lionel Messi and Cristiano Ronaldo as they are now playing in America and Asia respectively to track how their careers after leaving Europe are going. And honoring a LaLiga Women’s player “Aitana Bonmati” who won her first Balon d’Or for her amazing campaign through the Women’s LaLiga, Champions League, and World Cup.

Figuring out how to use a ScrollView was exciting as the main thing we wanted to avoid was making multiple pages for what could be displayed in one page. In the future multiple pages with ScrollViews will be necessary as with the addition of more players, scrolling infinitely to find the player you want to see could be tedious and not look clean for the user. Since the ScrollView cannot have multiple children it's children also have to be a larger view. We elected to use constraintView as that was the view that we had most familiarity with from lectures and what was used to design the other pages before this. We did attempt to use ListView as it would make it easier and cleaner to have multiple players and statistics, but while editing one thing it would edit the items before it too. For lack of time, we decided to ditch the idea and brute force the way these image and text views were displayed. Brute forcing does make the code a lot harder to read and very messy, but as we kept a similar order for all the players whenever we had to make an edit, we were able to do so with relative ease. Another thing we were able to do was show which players may be out for the season by constraining a text view in red text explaining their injury. A problem may arise when more than 50 players are to be added to the app as copy and pasting for the brute force will be tedious as it already was tedious for the max of 24 players that we had. Since the player count and stats were so different across the different sports, we also had an issue of uniformity. The project may be a couple of pixels different on different pages and even form text view to text view or image view to image view. This could make the app look less visually pleasing, but to the naked eye some of these pixel offsets may not be as noticeable as we think.

A future development of this project would be linking a database and an API like LiveScore or any public API to instead have statistics be updated in the database live instead of manually having to hardcode the statistics. This would also make the code a lot cleaner while making any edits and bug fixes extremely more efficient. The search box on the second page of our app could also be used to quick search a player instead of having to scroll through various players or we can implement a search for player names in the sports tab itself where it will be limited to searching for a player with the name in that page instead of a global search. We can add a predictions page where people can choose who will win a game and a leaderboard to track who has the most points by predicting most games correctly. This could make the user experience so much better and show that we can still improve on our application.

Program Output:

Once all the back-end and front-end are programmed with no error, we run the android emulator which will launch the app created in the virtual device of our choice. Upon starting, it will show the User-authentication page. If the credentials are wrong, the user won't be able to view the next pages, but if the right credentials are entered, the user will login and view the search page. In the search page, users will be allowed to search for 4 different options as of now guided by the text above. If the user selects a valid search item, they will be displayed on the menu or else the user will be shown an error message. In the menu section, user can select either "NFL" for football, "La Liga" for soccer or "NBA" for basketball. In each of these 3 pages, the app will display current statistics of the best and user's favorite player from the selected sports which user can view scrolling up and down. User can either return to amin menu from there or sign out to quit.

Conclusion:

By this point, we have successfully programmed back-end Spring Boot and front-end Android Studio for our application. We have also successfully run the application on Android emulator. Overall, we believe that this project has taught us a great deal about creating a practical application. Through this project, we were able to learn about UI designing of Android applications as well as programming the designs into the device and implement displaying content and functionalities. We have also learned about back-end and how Spring Boot feeds data into front-end. We also got an idea of how to make front and back work simultaneously. The things from this project we would love to improve on are being able to make the Spring Boot more versatile and being able to direct various type data other than Boolean into the front-end. We would have also liked to incorporate file handling methods to return statistics of players instead of integrating them into the layout design. But ultimately, it has been a great success to be able to progress this far and create a fully functional Android application and we would like to improve and add to our knowledge more in the future.

Reference:

https://m2.material.io/components/text-fields

https://square.github.io/retrofit/

https://stackoverflow.com/questions/20156733/how-to-add-button-click-event-in-android-studio

https://www.geeksforgeeks.org/spring-postmapping-and-getmapping-annotation/