Overview:

We developed an NFL app to allow users to check up-to-date information about the NFL, such as scores, standings, players, and teams, as well as to be able to mark teams and players as favorites to find information about them more easily. The way it works is that we obtain the up-to-date information from API-Sports (https://dashboard.api-football.com/), which requires the use of a key that we can use in our HTTP get request. We have defined a class to store player, team, standings, and game information, so that we can more easily work with the data. Using the data, we are able to display the information and images in the different tabs and pages of our app. We also use a favorites model to keep track of the teams and players currently marked as favorites so we can display information about them, as well as a games model to keep track of the currently selected game that the user has clicked on for more information about.

The motivation for our sports app was to display up-to-date information about the NFL in a visually appealing way and provide features to facilitate a user's ability to find the sports information that they are interested in. The motivation of developing the sports app was also to demonstrate and acquire skills related to programming handheld systems, especially with processing and storing data retrieved from API calls, keeping tracking of app state, and displaying the information in a nice way.

List of Goals:

1.) Implement Core Features using the Third Party Football API:
    - Integrate a reliable third party NFL api in order to retrieve live information for scores, standings, and players. This is a crucial goal as this allows for the app to dynamically update its content and provide users with accurate real-time information.
2.) Develop a User-Centric Interface with Five Main Tabs:
    - Scores: Display Scores for the current/previous weeks with favorite team matchups prioritized
    - Players: Showcase favorited players' game and season stats with expandable views
    - Standings: Present the league's current standings
    - Favorites/Home: Provide quick access to favorite teams and their latest/upcoming games along with information about their favorite teams
    - Teams: List all teams and allow access to detailed team info.
3.) Enable User Customization via favorites tab:
    - Allow users to personalize their app experience by selecting favorite teams/players. This will allow users to more quickly access the information that

they want for their favorite teams/players, rather than having to search throughout different pages of the app for the data they want.

4.) Implement Push Notifications for Enhanced Engagement:
    - Introduce notifications to keep users informed on scores for the latest week, prioritizing their favorite teams.
5.) Improve Data Integration Across Pages:
    - Ensure seamless data flow between all tabs, with consistent design for storing and retrieving data. This helps make the app scalable and easier to implement new features into.
6.) Uniform Style:
    - Ensure the styling and formatting of the tabs are consistent throughout the entirety of the application to help maintain internal app consistency and style.

User Interactions:

When you open the app for the first time, you will be prompted for notification permissions:

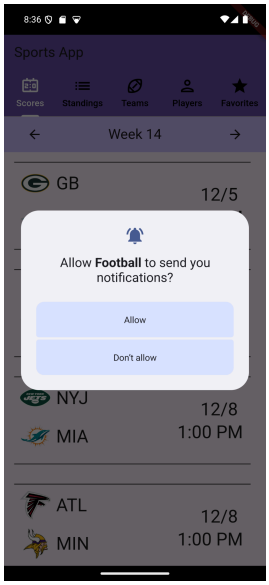


Figure 1: Initial

Once you click on "Allow" you will see the following view:

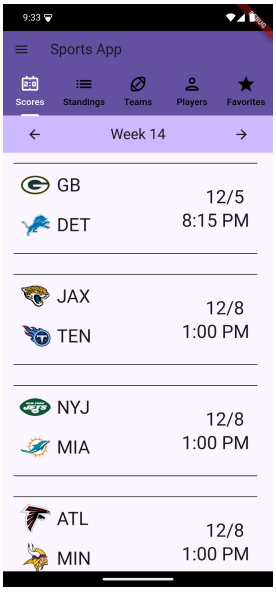


Figure 2: Scores

The scores tab will show the scores/game times for the current week of the NFL season. The week being viewed can be changed by clicking one of the arrows at the top, the left arrow to the previous week, and the right arrow to the following week. By clicking on the left arrow, you will see the following view:

You can also see the current standings for each division and every team's basic stats by clicking on the standings tab after the back button:

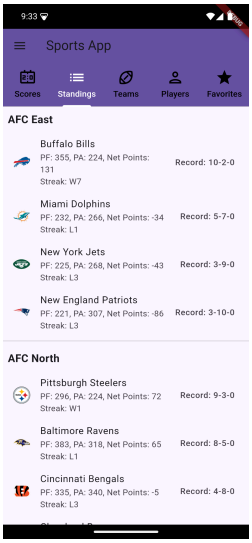
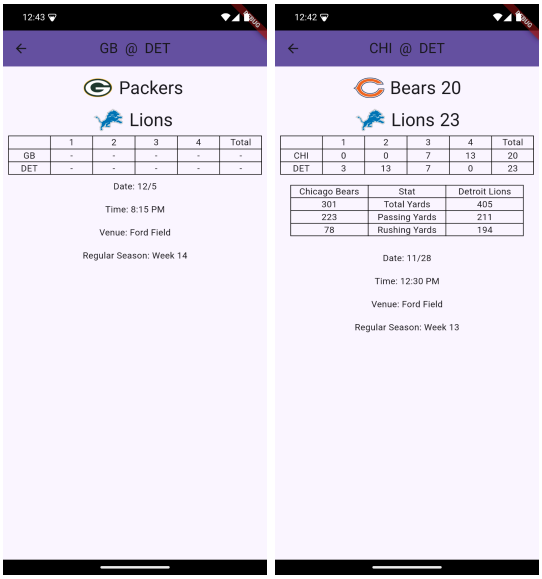


Figure 3: Standings

Clicking on a game will take you to a game details page. A game that hasn't started will look like the left image and a game completed or in progress will be similar to the right image:



By clicking on a team, you will get the following view with the team's basic information, previous game, and subsequent game:

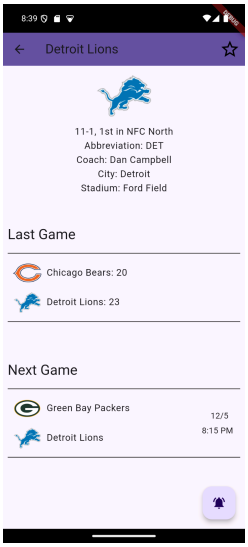
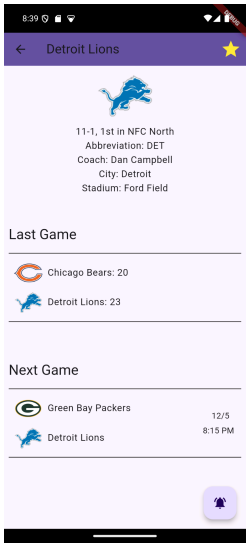
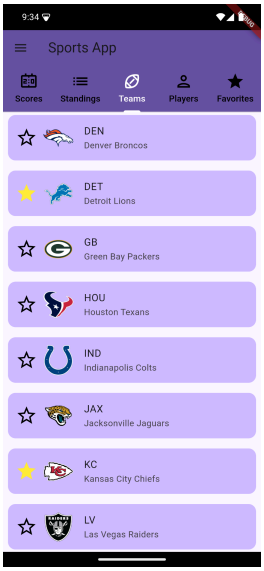


Figure 4: Team Page

We have a feature to favorite teams and players, so to favorite a team in this view, simply click on the star in the top right corner:



Pressing on a team will redirect you to the same view as Figure 2, displaying the team's basic information and the previous and following games. You can still favorite a team by clicking on the star in the top right corner of that team's page, but another way to do this is by clicking on the star next to the team in the Teams Tab itself:



Now, if you press the back button at the top left corner and navigate to the teams tab, you will see the following view with all the teams in the NFL:

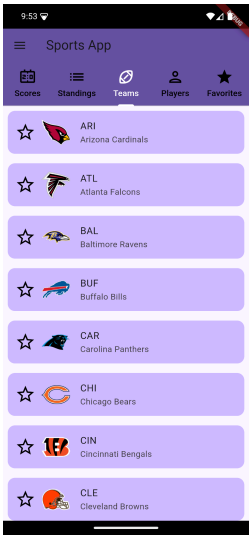
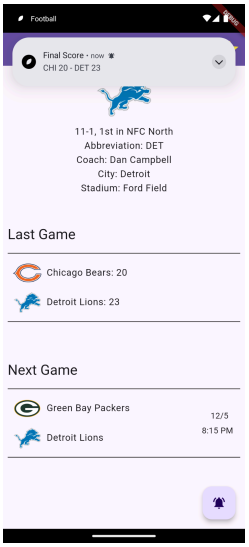


Figure 5: Teams Tab

To simulate notifications for a favorite team, we added a button at the bottom right corner of the team's details page, which, when clicked, will cause a notification to appear with the score of the team's previous game:



Next, by selecting the Players Tab, you will see the following view:

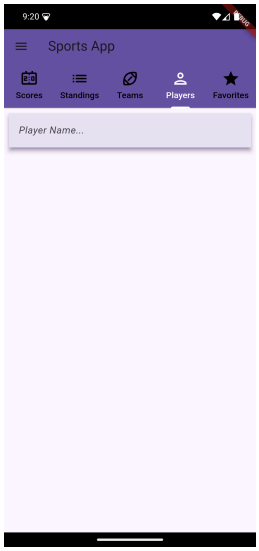


Figure 6: Players Tab

After searching for a player by their name, say “Jackson” the page will be populated with a list of players with Jackson as their first or last name. Just like the teams tab you can favorite a player by selecting the star next to that player:

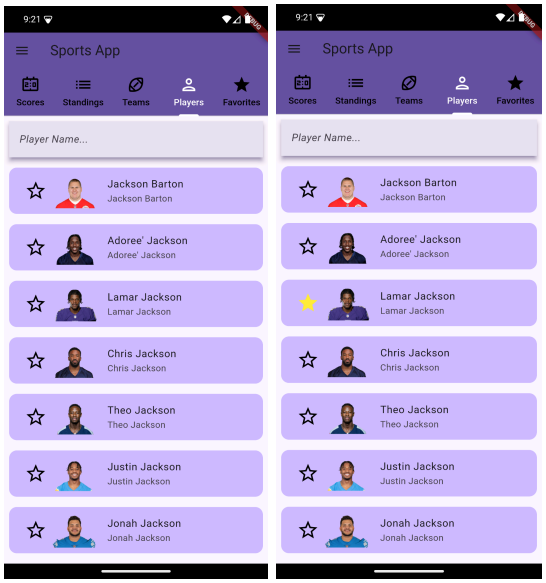
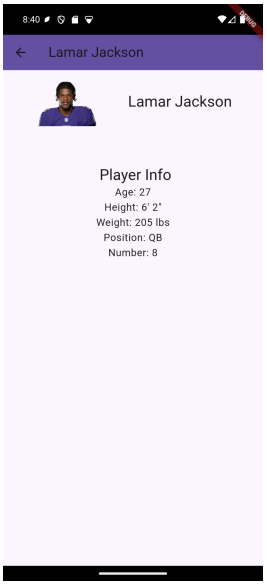
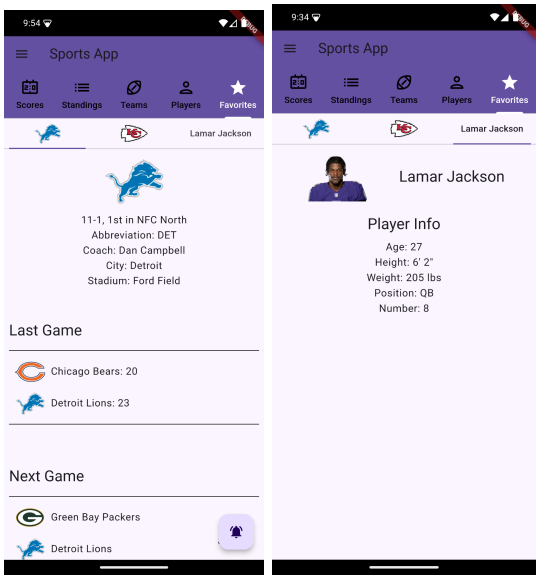


Figure 7: Players tab after Search

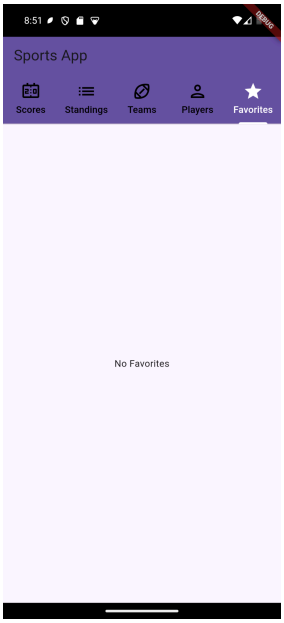
By clicking on one of the players, you will be brought to the following view with that player’s basic information:



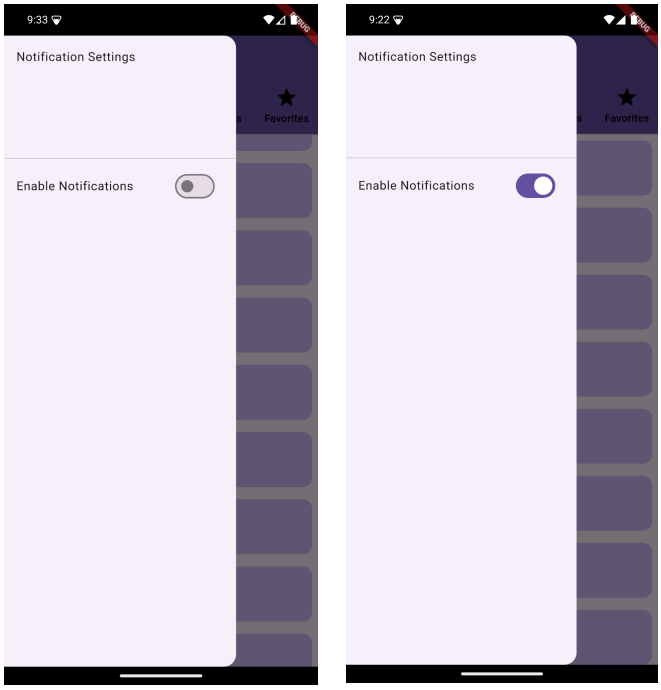
When you press the back button and navigate to the favorites tab, you will be brought to a view that shows the basic information of all the teams and players that you have favorited, such as the following:



If no such teams or players have been favorited, you will see the following view:



Lastly, if you select the drawer at the top left, you will see a switch to turn notifications on and off. Toggling this off will remove the notification button that you see in Figure 4.



Development Process:

Our development process was simple, yet effective. After we decided on what we wanted to do for our app, we then brainstormed what data we wanted our app to display. We ultimately chose to follow the normal abilities of apps that track and display data from various sports. We decided to focus on just the NFL and to display the scores of games throughout the regular season, the standings, a list of all the teams and players with the ability to favorite them, and a tab that displays your favorite teams and players. We then decided how to split up the work. We ended up each taking a page and implementing a bare-boned hard coded version of our given page in time for milestone 1. By the end of milestone 1 we had hardcoded games on the scores tab. A standings tab for each division in each conference using data from the API. A list of the teams from the API on the teams tab. This page also had the ability where clicking on a team would direct you to a page with basic information about that team. A hard coded list of players from the API on the players tab. A hardcoded favorite team and information about that team from the API on the favorites tab. At this point we were just about right on schedule. However, we did plan on accomplishing having true scores/games from the API on the scores page but we did not. Now we shifted our focus onto our goals for milestone 2. The work was still divided into each of us continuing to build on the page we started in milestone 1. During this milestone was when we realized that splitting the work solely into the different tabs was not the most ideal. Having the function of favoriting a team and/or player and having it display on the favorites tab would have a similar functionality, but when we each individually made a way to favorite a team or player the way that occurs was slightly different and would have to be worked out in the following milestone. At the end of milestone 2 we had completed the ability to favorite a team and have that appear on the favorites tab. The ability to click on a player and get basic information about that player was also completed. Finally, the scores tab's data was fully implemented using game data from the API. However, it was hard coded to only display games for week 7 of the NFL regular season. We set out to also be able to favorite a player and have it dynamically appear on the favorites tab. The ability to favorite a player was about 75% of the way done, but not shown on the app at the end of this milestone. In the time between milestone 2 and milestone 3 we would make a lot of progress on catching back up to our initial timeline. By milestone 3's completion we were able to favorite a player and have that player appear on the favorites tab, push notifications for a teams last game, the ability to see very basic game information when clicking on a game from the scores tab, and the ability to click on a team and get that team's details on the standings tab. We also added a custom app icon to our app in this milestone. Now that milestone 3 was complete we regrouped to determine what we still needed to accomplish. We realized that we needed to create a more consistent style across all pages to make the app look uniform. We also needed to make the scores tab be able to show games for all 18 weeks in the NFL regular season. For the players tab we still needed to add a search bar in order to have it fully integrated with the API. We first implemented background notifications that would tell you the final score of your favorite teams' last game. We set these to be sent at 6am on

Tuesdays in order to make sure that the notification was about the team's most recent game. Next, we implemented a way to change between the weeks of the NFL regular season in order to see past weeks and future weeks. We stylized how the games would be displayed here by making each one a list tile and having a solid black horizontal line above and below each game. If the game had started or was over it would show the score on the right side of the screen as well as bolden the winning team and score. If the game hadn't started it would show the date and time of that game on the right of the screen. This style would then be implemented on the team details page for their last and next game. We made sure that the box decoration behind each team and player on the teams tab and players tab respectively were the same as well. We then added more in depth details for each game that are shown when you click on a game from the scores tab. This includes the box score, date, time, venue, week, total yards, passing yards, and rushing yards. Next we added a drawer with a switch that would turn notifications on and off. You could trigger a notification by clicking a button on a team's details page. This button will only appear if the switch is set to on. Finally, we implemented a way to search for players by name. The API had a built-in feature for searching for player(s) with a given name. All we had to do was make a search box for the user to type in a name and make the API call with that given name and populate the tab with player(s) returned from the API call.

Setbacks that were Encountered:

Development of our app was generally smooth sailing for the most part with the largest setbacks being encountered at the beginning of the semester when everyone was meeting and trying to figure out how to set things up.

The first setback we encountered then was figuring out our project's direction and decisions for what kind of app we would design. We didn't have a lot of common interests or overlapping ideas and weren't able to really find anything in zones we were all skilled in so we instead decided to base our App off of the tools available to us instead of the tools we already knew how to use. Because of this we all ended up deciding to work on a Football stat keeping app since using API requests to get the information about football player was a process with some relatively publicly available documentation and actually accessing the information through the API was free (although limited to 100 requests per day). Using a free and publicly documented tool like that would be useful for us to develop and good for us to understand since there were a lot of other similar stat keeping apps we could use as examples.

The second and last major setback we encountered was something I already mentioned in the paragraph above. That being the fact that our API limited us to 100 requests every day when we wanted to get information for our app. Our app makes about 5 API requests upon startup. Depending on the data accessed and how much said data was accessed while using the app, it

could potentially use all 100 requests in a single use. We solved this by each making an account with the API and only using our API code when testing and making changes to the app. This would allow each of us plenty of requests to make changes throughout the development of our app.

Decisions and Changes made to the Initial Plan:

We mainly decided on the initial plan after two meetings in the beginning of the semester where we first decided to use API as the focus of our project and then decided to have Seasonal Football Statistics be the focus of our App. Beyond those basic plans and ideas not a whole lot was changed but the main way our project really morphed and evolved over the course of the semester was in how we shifted the focus towards prioritizing user experience and visual comprehensiveness towards the end. The main capabilities of our API was to give basic stats and details about Football players so after we implemented features that used every stat the API could request we just naturally changed our focus towards improving the presentation of that information instead of just giving more of it.

What was learned about the Development Process:

One important lesson that was learned was how communication and research is vital to executing a mostly online project. Every team member understanding their own task and understanding how to do it and how it would interact with other team members' task was a vitally important part of our coding process. Clarity in role organizations was something we really focused on when communicating with each other online and it ended up being a vital way our development was kept efficient and timely. The second part of the lesson we learned was about how using 3rd party features impacted our development. We used a publicly available and free API for our development but one result of this decision was that documentation on the use of that API was not clearly structured or easily readable and we had very little resources for help on how to use such specific and unique resources. Because of this we had to spend a lot of time internalizing and researching the ways the 3rd party tools we used actually worked before working with them since there were no simple and clean explanations available to us. Eventually we got the hang of our new API and how its commands worked but we initially underestimated the documentation needed to use 3rd party libraries and had to learn the reality of the situation because of it.

New Features not taught in Class:

The main new feature we learned was how to request information from the Football API we used. While seemingly simple in concept there were a lot of specific formats in how to request different types of information and we also had to make sure once a piece of information

was only requested once when a user opened the app so as to conserve API calls (recall the 100 limit). Beyond this syntax was also different between different versions of the API and there were multiple sources of documentation for it which were not all consistent so just condensing a large and intricate library of Football Statistics requests into a simple series of API calls that gave simple pieces of information was the feature we taught ourselves.

Potential Future Directions:

For the potential future directions, we could look to implement more advanced push notifications, so that when the user's favorite team starts a game or scores, for example, the user can be notified of that event and be able to click on the notification for more information. This could be done by using the flutter_local_notifications package. Adding more data to teams, games and players would be a must add in the future. We could also look to implement more up-to-date information so that as the scores of the games and player statistics change, the app could display those in real time (rather than only updating when the app is relaunched) on the scores tab and on the scores details page for each of the different games. This would involve making more API calls to check if any scores or statistics have changed since the last update, and using the data in those API calls to display on the app. We could also look to make it possible so that when a user is on the game details page, they would be able to click on any of the players or coaches involved in that game, sending them to a page with more statistics or info on that player. Lastly, we could look to implement a way that would allow users to keep track of player's fantasy stats based on an inputted point format.