Juhyun Oh, Kevin Kim, Zhisheng Liu

CSE134B HW4

In this assignment, our group implemented Firebase prototyping for authentication and CRUD functionality for both assets and data. We rendered our views using Vue.js.

For our authentication, we have two very simple methods for sign in, Google and Email. The Email sign in is implemented by the instantiating a `signInWithEmailAndPassword` object from Google firebase, which validates the input with the user profile stored. We use the error code return by the object for the sign in and sign up error alert. The Google sign in is implemented by instantiating a `GoogleAuthProvider` object and use it to authenticate user identity with Firebase. Each time, a user login, we will examine if he is a first time user. If yes, we will create a private user folder identified by his unique uid to hold his team data, otherwise we will just search our user/ directory in the real time database and pull his team data the project them onto the team page.

As of now, most of our prototyping is done in the managePlayers.html file. Here, we can create (C-Create) a player by uploading a image of the player and inputting statistics of this player. The image is stored in Firebase Storage while the statistics are stored in Firebase Realtime Database. Each player is rendered through Vue.js and Firebase, thus we have a dynamic list view on this page showing (R-read) all the players in the pool. Using Vue.js, we can also edit (U-Update) each player and update both the player's stats through Realtime Database and the new image into Firebase Storage. Also we can delete (D-Delete) the player which will delete the img file from our Storage and remove the player from the Realtime Database. The major CRUD functionality is set up for future development. For asset management, since we only have one type of asset: player image, we put them in the firebase storage and implement interface for upload and access.

The img file for each player has not been utilized to its full extent yet. However we will use this in our myTeam.html file. Here a user will be able to customize his/her team. We will populate each user's table with the player's table data and also render the player's img from storage. Again for the Create and Update functionality, we have yet to implement any sort of input validation yet. This will be done later as we finalize the project. For the first goal of this homework, we completed most of our specific interface features such as players image upload, dynamic list view of player pool and popup modal for data modification.

Most challenges we faced when prototyping consisted of learning new web technologies and muddling through the respective API's. Because none of us had experience with Javascript, we had to learn how to use jQuery, Vue and Firebase. It is also notable that doing Javascript is much different than doing CSS or Javascript. Any syntax error such as missing semicolon could paralyze the web page's functionality, and it is harder to debug since HTML and CSS are more presentation-oriented, which means anything goes wrong will present itself on the screen, but not Javascript.

Considering performance, because we use Bootstrap, jQuery, Firebase, and Vue from a CDN we will expect a good amount of overhead. However, we put our scripts into our HTML file instead of linking it so that we won't have to load an extra file. Most of our function are implemented by vanilla JS and JQuery, and we only use the framework Vue JS on the managePlayer page since we need to populate the list view on the fly. It is obvious that loading the page implemented with framework JS take more time than the other page. As a conclusion, although framework javascript support powerful functionality, it has some drawback such as loading time expense.

Result:
user data(Auth) and NBA players data(CRUD) is successfully prototyped, the below screen is the response from the real time database