CU Fueling Station Web Portal (CUFS)

Project Milestone 4
Team 0x1: No. 014-1

Jake Derouin, John Aldrete, Michael Metz, Ella Hernaik, Ryan Chaknova, Tyler McCormick

Table of Contents

1 - Updated Features List

1.1 - Identikey Login

The CUFS Web Portal will allow the user to login using CU Boulder's Federated Identity Service. Once we get the correct credentials, we can have the user sign in with their existing CU Boulder Identikey, and we can display certain information about the user (name, profile picture, etc.)

1.2 - CSV/Spreadsheet Creator

This is a feature being developed for the coaching staff to handle some administrative duties regarding athletes check ins.

1.3 - Notes Editor

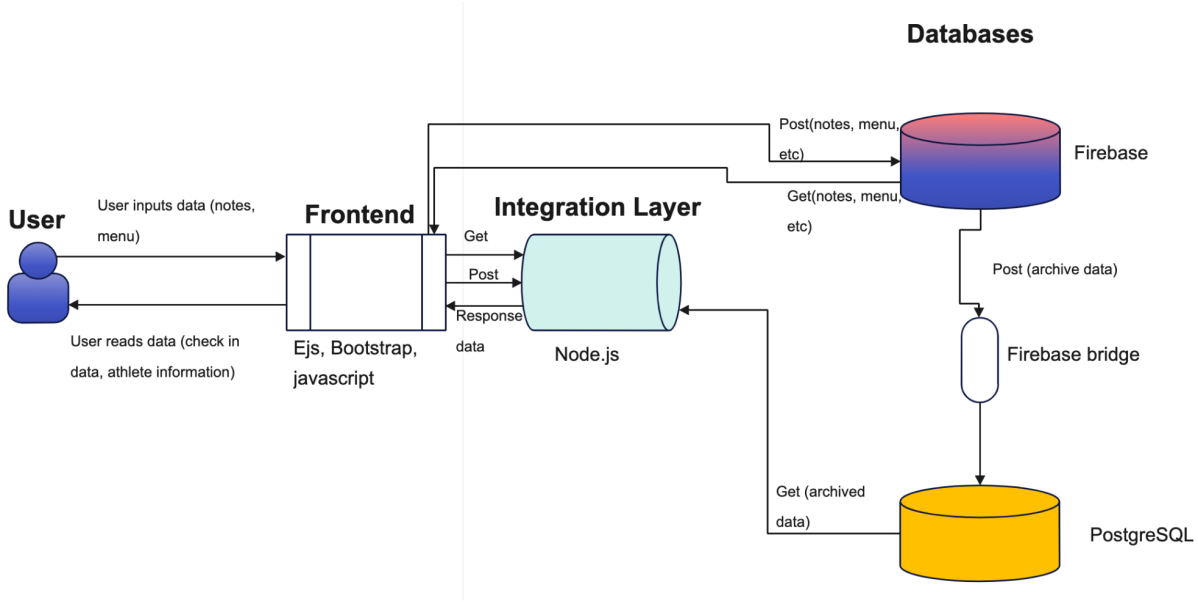
The notes editor will be available to athletic staff to give notes to specific teams about training, nutrition, and more.

1.4 - Menu Creator

The menu creator is a feature for the nutrition staff to have an organized and simple way to create menus each week.

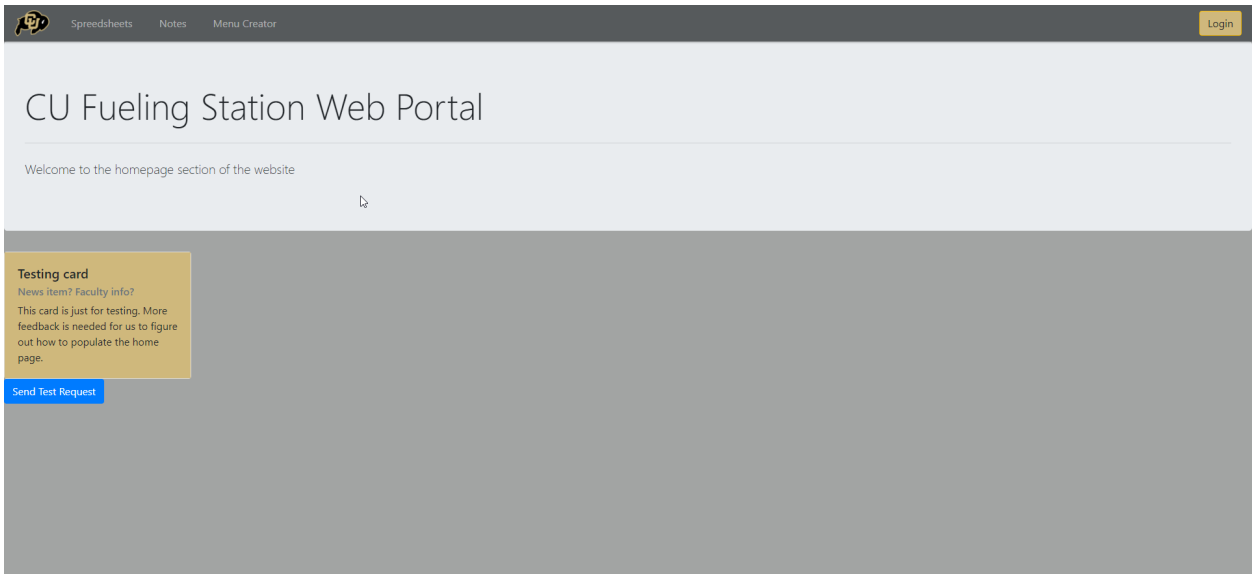
2 -Architecture Diagram

Also important to note. The website (and mobile apps) will communicate with the CU Boulder Federated Identity System through Firebase to obtain user information (name and identikey) and sign in. This will use the SAML protocol.
PostgreSQL is used for storing and retrieving long term check in data while Firebase is used for everything else.
Additionally, a reCAPTCHA system will be in place to protect the firebase system from attacks by using App Check API.



3 - Front End Design

Landing page (tentative)- Because some web pages will be accessible to non cu affiliates and others require a user to be logged in to view, the landing page of the web portal for now is the homepage. Users will then manually log in, or get automatically directed to a log in page when attempting access to certain web pages



4- Web Service Design

Our website will use the Firebase Javascript API to communicate directly with the Firebase database.
Types of data: checkins, authentication, ralphie run high scores, nutrition notes, minimum build version.
reCAPTCHA API will be used to use the Firebase App Check API for security.
Type of data: Site interaction pattern results.

5 - Database Design

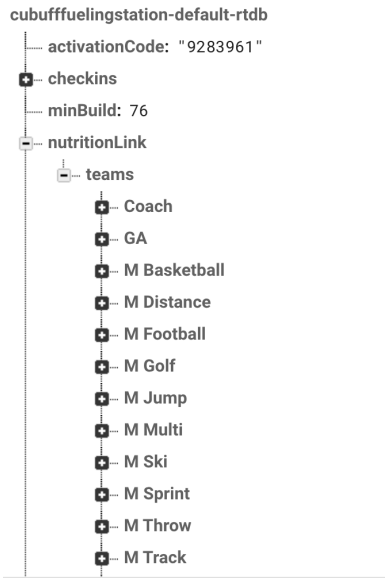
Our website currently utilizes 2 different databases for different types of data, handled by Firebase and PostgreSQL. Starting with PostgreSQL, below is a summary of our DBMS.

5.1 - PostgreSQL Database

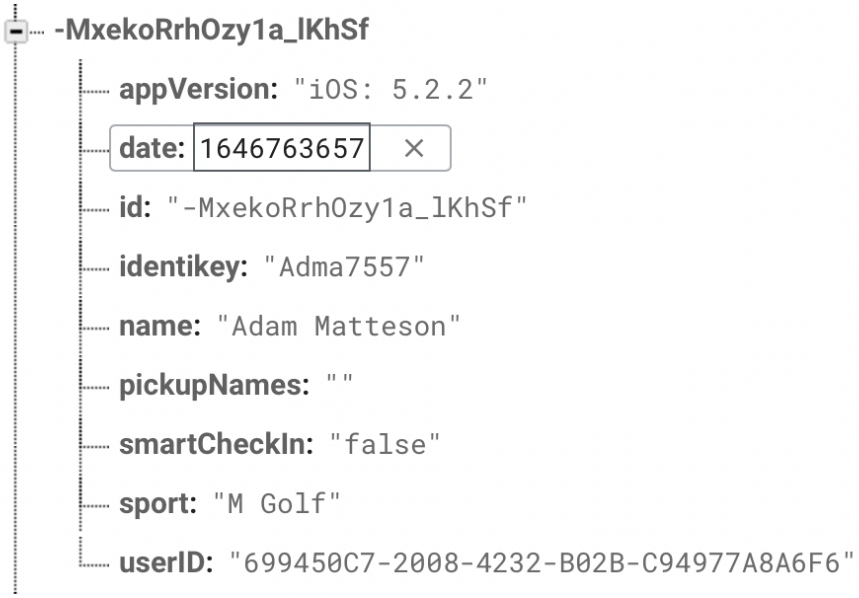
The Postgres database is mainly concerned with past data. We are concerned with this data since the coaching staff uses it for administrative duties (one of our features is based off of this). Since many students check in to fueling stations around CU every day we need to have a cheap solution to archive past data, using Postgres with Heroku fits here. To the right is a list of the columns in the checkindata table, which gets perennially updated from Firebase (explained below).

5.2 - Firebase Database

Firebase is what we are using for more current data. This is where the app writes the most recent check-ins to, and is where the team notes (feature in development) will be stored and displayed to athletes through the app. To the right is the main table for the check-in data in Firebase, nutritionLink is where the team notes will be stored..



Example of checkin:


-MxekoRrhOzy1a_lKhSf
appVersion: "iOS: 5.2.2"
date: 1646763657
id: "-MxekoRrhOzy1a_lKhSf"
identikey: "Adma7557"
name: "Adam Matteson"
pickupNames: ""
smartCheckIn: "false"
sport: "M Golf"
userID: "699450C7-2008-4232-B02B-C94977A8A6F6"

6 - Challenges

- It is very possible that the backend portion of our project eats up most of our time and results in a lacking front end. This is purely based on the nature of the application, and if we cannot resolve this, then we will just stick with basic bootstrap elements for the website.
- Things have felt a little disorganized, sometimes we would have multiple people working on the same feature, so we need to figure out a better way to make sure everyone is working on something unique. Whether that's splitting up into groups for each group to work on one feature, or something else.
- The SAML (federated identity service) setup is a bit difficult (though we are making good progress). If we can't get this up and running we will switch to manually creating user accounts for authorized staff members.

7 - Individual Contributions

7.1 - Tyler's Recent Commits

1. Updated '/' endpoint to display current number of check-ins for current day (WIP)
2. Updated table to display error if no data to be shown
3. General UI Updates

7.1 - John's Recent Commits

1. Enabled PostgreSQL connection with Node.js.
2. Ported old HTML to EJS.
3. Added dynamic EJS table for data from backend.

7.1 - Jake's Recent Commits

1. Connected firebase as a JS Module (along with webpack) to website.
   -Continued working with the OIT department on SAML