Milestone 7
CSCI 3308-103

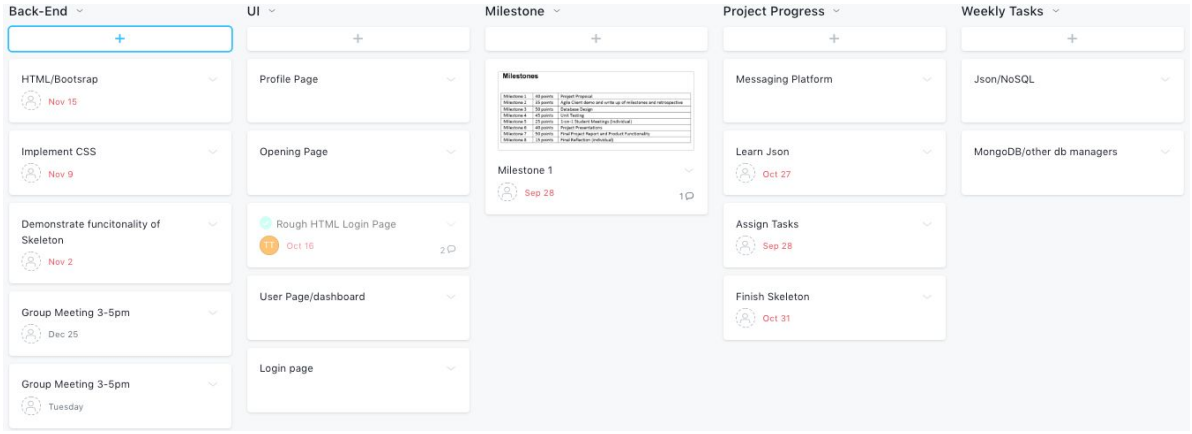
BuffsGitBuff

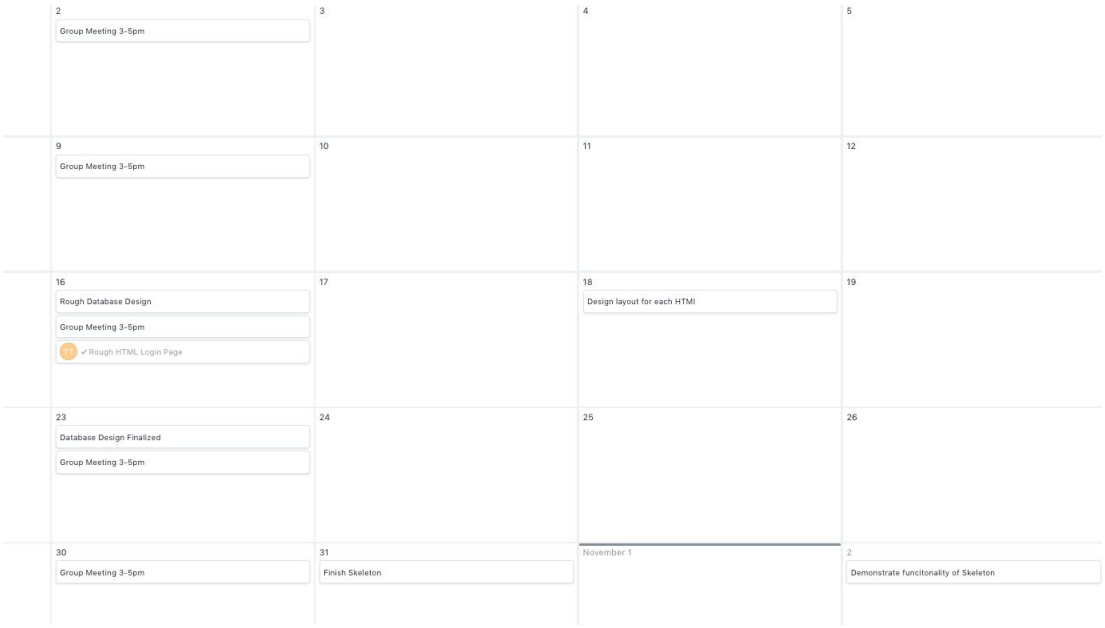
Names

Casey Durham (Github username: DockJumper86)
Aaron Lu (Github username: aaronlu56)
Willem Mirkovich (Github username: willemmirkovich)
Daniel Taylor (Github username: 1s0m0rph)
Tyler Tokumoto (Github usernames: tyto8880 and drayneman)
Carlos Salazar (Github username: Carlos2020Salazar)

Project Tracker

Asana Project Tracker
link

- 

- 

VCS Tool

[Github Repository]

Deployment

Currently running on our VM provided by GCP

[Project Website]

Contributions

Willem

Note to Instructors: I added all of my contributions to the master branch through merging with my own branch, “willem”, throughout the semester. This led to Github not keeping track of my contributions. Therefore, I will be putting screenshots and a description of my work throughout the semester below.

These were the main files/features I worked on

- routes.py : added the functionality for the user page to get send http requests (screenshots of work provided)
- User.html: built the html page and script section for js calls
- createDatabase.js: made the script to populate database with our info, this was later modified by Daniel, and with the info collected by Aaron
- Initial creation of the flask environment: built the bare bones of the flask environment at the beginning, which was fully implemented later by teammates

```
        except Exception as e:
            print(e)
            return 'error: create workout did not go through'
    # call for get the users workouts
    elif (request.method == 'GET'):
        try:
            userInfo = db.getUserInfo(session['username'])
            workouts = userInfo['favoriteWorkouts']
            userWorkouts = []
            for workoutID in workouts:
                workout = db.getWorkoutFromIDForUser(int(workoutID), session['username'])
                userWorkouts.append(workout)
            # returns all user favorite workouts if they exist
            return jsonify(userWorkouts)
        except Exception as e:
            print(e)
            return 'get workout did not work correctly'
    return 'error: not get or post request'

# handles add workout to user favorites calls from userpage
@app.route('/addWorkout', methods=['POST'])
def addWorkout():
    #adds workout to current session username
    try:
        workoutID = request.form['id']
        db.addWorkoutToUserFavorites(workoutID, session['username'])
        return 'added workout'
    except Exception as e:
        print(e)
        return 'workout not added'
```

routes.py (Willem)

Aaron

I switched around jobs a couple time throughout the course of the project. At the beginning I was tasked to populating the database with all of the equipment, exercises, and muscle groups. I moved to the front-end halfway through the project and worked mostly on taking the html web pages we already created, changing them to utilize jinja templates, and making them look nice.

The files I contributed to were:

- Everything in the databaseScripts directory
- base.html, user.html, and edit_profile.html

I also played a role in editing all of the other html scripts in this project.

Daniel

I stuck generally to the backend and integration layer area. My most major contribution to the project was the algorithms we used to actually generate workouts. However, I also worked on both cleaning and standardizing the database. In addition to this, I also wrote all of the automated tests we used in our product. I used my general Github user for this project, whose username is 1s0m0rph. The files I contributed to were:

- database.py, workoutgenTests.py, & databaseTests.py
    - I wrote these files, including the all of their functions and functionality.
- createDatabase.js
    - I standardized the formats for the exercises as well as cleaned this file up.

Tyler

I laid the groundwork for the backend of the project and managed the deployment instance. This includes setting up and implementing much of the routes.py file, creating a skeleton for most of the templates currently being used, and maintaining an updated build on our Google Cloud Platform server. As one can see from the Github contributions, my code is centered on the Flask app directory.