FAU College of Electrical Engineering
& Computer Science

Principles of Software Engineering
CEN4010 – Summer 2023
Group #11 – Team Snorlax

Milestone #1 – Project Proposal & High-Level Description

Project Proposal, High Level Description of Features & Functionality, Use Cases, Team Roles, Project Architecture

Authors

Name	Email
Dylan Cohen	dylancohen2018@fau.edu
Maximo Mejia	mejiam2017@fau.edu
Owen Levine	Olevine2020@fau.edu
Sienna Gaita-Monjaraz	Sgaitamonjar2019@fau.edu
Cassidy Wilson	cassidywilso2017@fau.edu

Table of Contents

Executive Summary

Every student has experienced the stress of planning out a semester of classes. Many sources have to be referenced in order to make an intelligent decision regarding relevance to career goals. The purpose of this project proposal is to address the issues associated with existing course registration interfaces and improve the overall experience for students. The proposed solution aims to streamline the course selection process by leveraging database systems to provide a clearer pathway towards career goals.

The proposed solution is a web-based application that shows students exactly what courses they are able to take next based off the courses they have already taken. A list of courses publicly available from FAU will be processed and saved to create a relational database that will be used by the web application. The database will hold corequisites, prerequisites, (held as nodes in a tree) and skills that relate to each class. With this data, we can display to the user the classes they can take next via depth first search. This provides a seamless way for students to understand their academic progression based on their course history.

In addition to helping with course selection, the application will also allow the user to see a visual map of all the skills they have acquired from each of their completed course.This feature provides a visual representation of a student's strong and weak points through the connectivity of the map–facilitating further skill development.

Another unique feature of the platform is the job fitting simulator. By leveraging the data from the skills mapped out by the system, an algorithm matches students with potential job titles tailored to their current skill set. This feature bridges the gap between education and professionalism, easing the transition from college to the engineering industry.

While the current scope of the project focuses on computer science students at Florida Atlantic University, the flexible design of the relational database allows for potential expansion to other academic majors or universities. This proposal represents not just a tool for better course selection, but a comprehensive platform for fostering educated career decisions from the beginning of a college career.

Competitive Analysis

Company	Static vs Real Time	Personalization	Cost	Target Demographic
myAdvisor (our product)	On Demand	Skills student specifically acquires in their classes	Free	FAU Students
FAU Auditing System	On Demand	shows courses you have taken	Free	FAU Students
Right Management	Scheduled live consulting	leadership development track	pay per session	Professionals
Mercer	Scheduled live consulting	Consulting to find relevant jobs	pay per session	Professionals
Better Up	Scheduled live consulting	Skills youve said you gained	$89-$275 monthly	Students & Professionals

Competitive Advantage of Our Platform Over Traditional Audits
Traditional degree audits do not show course prerequisites they just show what classes you have not completed. To find prerequisites you need to go into registration and look for them. Our platform displays clearly what courses you can and cannot take given the current classes you have completed. When comparing this advantage it becomes clear that our solution offers numerous unique competitive advantages.

Comprehensive Prerequisite Mapping
Our platform takes the uncertainty out of course selection by providing a clear, user-friendly table display of courses leading up to the current semester. Traditional audits often leave students in the dark about course progression, leading to inefficient planning and even sometimes delayed graduation. By contrast, our platform lets students plan their academic journey- reducing uncertainty and enabling them to take absolute control of their education. This is especially useful when it comes to specialized upper level electives which all have different prerequisites inside that specialization.

Skill Acquisition Tracking
Moreover, while traditional audits simply list completed courses, our platform goes further by detailing the specific skills students have acquired from each class they have marked as taken. This insight gives students a tangible sense of their progress and equips them with the means to articulate their capabilities. This is a huge advantage when it comes to writing resumes, preparing for job interviews, or applying for

internships. Instead of simply mapping out what you recall learning, our product takes an advanced approach by listing with certainty what you have learned from each course as detailed in the syllabus.

Visual Representation
The visual skill map offered by our platform further enhances this advantage, offering an intuitive representation of a student's competencies. Students can clearly see their strong areas (where many skills are linked to each other) as well areas that could benefit from further coursework or experience (where skills are not linked together). Traditional audits offer no such insight, leaving students to guess which skills they've acquired and how they all fit together in the scope of becoming a well-rounded computer scientist.

Facilitating Career Progression Through Targeted Job Suggestions
Another major competitive edge of our platform is its ability to link academic achievements to real-world career opportunities. While student pick their certification, often coursework is more nuances and complex. Student might not take classes just from one certification, thus leaving them in the dark about the jobs that have opened up with these new skills acquired. Our product fills this critical gap by giving students insight into what jobs they could reasonably attain based on the skills they've acquired throughout their academic journey. With knowing the skills the student has gained from classes, our product can suggest the careers best suited to ones individual academic skillset.

Tailored Job Suggestions
Our platform uses an algorithm to analyze each student's unique skill set and match it with the qualifications required for various careers in their degree. The algorithm considers the specific skills students have developed in their courses, rather than focusing solely on the course title or a specialization earned.

This personalized approach ensures that the job suggestions are relevant and realistic for each student. Instead of providing a broad overview of potential jobs related to a field of study, our product provides specific job roles that a student is directly qualified for in their current state. This also helps students to progress in their professional journey by showing them what they can actively do with the skills they've learned in class. This targeted method reduces the overwhelm often associated with job hunting and provides a clear direction for students' career paths.

Preparing for the Job Market
By making clear connections between academic skills and specific job roles, our platform does more than just suggest potential careers—it helps students understand how to present themselves effectively in job applications, resumes, and interviews. By

seeing which skills are valued in their potential job roles, students can focus on highlighting these skills, increasing their chances of success in todays competitive job market.

Bridging the Gap between Academia and Career

Overall, our platform represents a significant leap forward from traditional audits and career planning services. Through providing comprehensive course flow mapping, detailed skill acquisition tracking, and data-driven career guidance, we empower students to make informed decisions about their not only their career paths as well. This holistic approach sets our platform apart, offering students an invaluable tool to navigate their academic journey with confidence and clarity.

Data Definition

Name	Meaning	Usage	Comment
Member	Actor	Use Case Scenarios	A user who is registered with the system
Non-Member	Actor	Use Case Scenarios	A user who is not registered with the system
System	Platform Hardware and Services	Use Case Scenarios	The mySQL database, all code, back end services and front end design
Website	User Interface	User Interface	Front end display for user interaction
Home Page	User Interface	User Interface	The first webpage that the user sees
Navigation Bar	Service/User Interface	Site User Service	A toolbar that helps the user navigate to different areas of the site
Dashboard	Service/User Interface	Site User Service	The primary place to display tools and student information
Account	Data	Use Case Scenarios	Location on a network sever that stores user information
User ID	Data	Use Case Scenarios	A unique name for each user to distinguish between different users.

Password	Data	Use Case Scenarios	A confidential string of characters defined by the user that is used to authorize a user log in to an account with an associated User ID
Register	Service	Site User Service	Process by which an unregistered user becomes a registered user
Log In	Service	Site User Service	Allows user to access their stored account data and system tools
Session	Service	Site User Service	Activity between the log in and log out
Log Out	Service	Site User Service	Ends the login session and redirects user to homepage
Course ID	Data	Data Field	A unique identifier for each course
Course Name	Data	Data Field	The name of the course
Department	Data	Data Field	The department to which the course belongs
Semester	Data	Data Field	The specific semester that the course is offered
Difficulty Level	Data	Data Field	The level of difficulty of the course
Prerequisites	Data	Data Field	The prerequisite courses required for the intended course

Roadmap ID	Data	Data Field	A unique identifier for each career roadmap entry
Career Goal	Data	Data Field	The long-term academic/career goal that is associated with the roadmap
Required Courses	Data	Data Field	The courses required to achieve the academic/career goal
Elective Courses	Data	Data Field	The courses that are recommended to achieve the academic/career goal
Status	Data	Data Field	The registration status (pending, approved, or rejected)
Support	Service	Site User Service	The way a user may report issues and bugs experienced to the developers
Account Settings	Service	Site User Service	The settings and configurations associated with the user account
GitHub	Repository Hosting	Use Case Scenarios	A website where developers can store their software projects, making it easy to collaborate, track changes, and manage different versions of their code

Firebase	Database/Back End Framework	Use Case Scenarios	A platform for backend development that provides developers with many tools and services to create/manage web/mobile applications without worrying about server management.

Overview

For reference – current concept for website

- User logs in/signs up
- User prompted to select their major from list (only doing computer science for now)
- Interface with a table for semester planning on one side, and a list of available classes on another
    - Only classes with met prerequisites appear and classes that are corequisites are grouped together
- Button to “commit (save)” semester into the table
- Display is refreshed with new classes available based on entered semester/s

For reference – basic framework for website

- Back-end
    - Python sqlite database
    - Data compiled from public fau course descriptions and program track using beautiful soup library
- Front-end
    - React framework for user interface
    - Firebase for user authentication

Scenarios/ Example Use Cases:

Use Case - User Profile Creation/First Time Use

Preconditions:

- User visits the website
- System is up and accessible
- User does not already have an account

Flow of Events:

1. User opens the app and selects the option to create a new account.
2. The app prompts the user to enter their personal information, such as name, and age.
3. User enters the required information.
4. The app validates the information and saves it to the user's profile.
5. The app displays a confirmation message, indicating that the profile has been successfully created.

6. App then has the user go through any back logged classes they may have taken prior to the account creation and adds those skills to the user
7. User then selects from a list their desired career paths along with being shown careers that line up well with their current skills
8. Users can now proceed to explore career options and select intended careers.

Actors:

- User
- System

Use Case - Viewing Degree Completion Progress

Preconditions:

- User has created an account.
- User has inputted the classes they have already taken.

Flow of Events:

1. User opens the website and logs into the system
2. User selects degree completion page
3. App retrieves any previously taken classes the user has, user's intended final degree, and any other necessary user data
4. App determines how many classes the user still needs to take and compares it to the classes the user has already taken
5. App displays the users previously taken classes and details about them along with the intended next classes if available
6. User can track their progress over time and make adjustments to their plan if necessary

Actors:

- User
- System

Use Case - Selecting Future Classes

Preconditions:

- User has created an account.
- User has added their intended career
- User has inputted the classes they have already taken

Flow of Events:

1. User visits the website and logs into the system
2. User Selects Class Advising/Future Classes Section
3. System retrieves previously taken classes, users intended career, related skills, and users intended degree.
4. App calculates what classes the user needs to take that can be taken this semester and that can be taken based on the prerequisites for each class. App also calculates what electives the student can take and what skills they will gain from each class.

5. App then displays all of the class options to the user with different recommendations based on what their intended career is, and which classes will have the most relevant skills.
6. User can then select which classes they wish to take and they will be added to their class list/degree progress

Actors:

- User
- System

Use Case – Getting Career Title Advice

Preconditions:

- User has created an account.
- User has added their intended career
- User has inputted the classes they have already taken

Flow of Events:

1. User visits the website and logs into the system
2. User selects career page
3. App retrieves users current intended career, and current skills based on relationships established with each class they have taken
4. App displays user's current skills and the skills their intended career requires along with a few other careers that have closely related skill requirements
5. User can choose to either just view their skills and career, view more details, or change their intended career to anything they want.

Actors:

- User
- System

High-Level Functional Requirements

Initial list of functional specifications:

Non-Member Experience:

1. **Account Creation**
   - **1.1 Summary**

     During the account creation process, the system will guide users through a user-friendly interface to enter and validate their personal information. Users will set a unique username and a secure password, agree to the terms and conditions, and verify their email address. Once the account is created and activated, users can log in and explore the features of the course registration interface.
   - **1.2 Stimulus/Response Sequence**
     1. User will access the course registration interface and locate the "Register" or "Create Account" option (available on the homepage or in the navigation menu).
     2. User will then provide the required personal information, such as their full name, email address, and student ID.
     3. User will choose a unique username and a strong password to secure their account.
     4. User will read and review the terms and conditions presented by the system.
     5. User will acknowledge their acceptance of the terms by selecting a checkbox or clicking an "Agree" button.
     6. System will check if required personal information is valid.
     7. System will enforce specific password requirements and validate that the user has met those requirements.
     8. System will store the user's personal information and password.
     9. System will confirm to the user that the account was created.
     10. System will have a button for the user to return to the home page to be able to log into their newly created account.
   - **1.3 Function requirement label**
     1. REQ 1.1 Account Creation

Member Experience:

2. Account Login

2.1 Summary

The login process requires the user to input their username and password into the fields provided in the user interface. If a user account is not found that matches with the username input, or the password is incorrect a message is displayed to the user notifying them of this error.

2.2 Stimulus/Response Sequence

1. User accesses the login page which displays an input text box labeled “Email/Username” and another input text box labeled "Password”
2. User inputs their username or email address in the first input text box
3. User inputs their password into the second input text box. The characters of their password are displayed as black dots for security purposes
4. User clicks the button labeled “Login”
5. If login is successful, main page is displayed, and user can continue to use the application as described in (2) below
6. If login is unsuccessful, the login page will load again, except that a message will be displayed above the input boxes that reads “Username and/or password combination was incorrect. Please try again.”

2.3 Function requirement label

1. REQ 2.1 Account Login

3. Creating a record of classes taken in a semester

3.1 Summary –

The user is able to click a button that notifies the app that a semester of classes that have already been taken are going to be entered into the system. The user can scroll through a list of classes or search the list by course code. Only classes in the college of engineering and computer science are displayed, but more majors can be added in future iterations of the product.

1. User signs onto their account
2. User clicks “Add Completed Courses” button on the main screen
3. User chooses term that classes were taken from a dropdown menu.
4. User scrolls through classes that were offered that term, and selects a course that they took.
5. User clicks “submit” button, which adds the course to a database record for their account
6. System recalls the semesters in the database and displays each semester as an item in the dropdown menu
7. System is able to query the database for a list of classes offered during the chosen semester and display them in a scrolling box
8. System enters chosen class into the database under that user’s account

4. **Searching for class by name**
   4.1 Summary –
   Users have the option to enter text in a search box which filters the classes in the list and only displays classes that contain matching strings in the course identifier or the course title
   1. User sees a list of classes available in their major, to choose and add to their record of taken classes
   2. User enters a string into the textbox labeled “Search for courses”
   3. User clicks the button labeled “search”
   4. System queries for a list of available classes offered in major of study
   5. System performs search for classes in this list that have strings in course id or course title that match the string input by the user
   6. System displays “courses matching *user input string*” above the new scroll box with updated list of matching courses
   7. System displays only the courses out of the list that match the string, and functionality resumes as described in **4.1**

5. **Viewing available courses to take based off already taken classes**
   5.1 Summary –
   The web application prompts the user to pick from a list of future semesters, and shows a list of courses available to take based off of classes entered by the user as previously taken in past semesters. The future
   1. User logs in and views the main page of the application
   2. User selects the “Explore Classes” button on the main screen
   3. User selects the semester from a dropdown menu of future semesters
   4. User is able to scroll through a list of classes that they can take based on prerequisites met
   5. User selects a class that they want to take by clicking the check box next to the class in the scroll menu.
   6. User can view supplemental information as outlined in **6.1**
   7. User clicks the “Submit” button to add the course to their record of classes
   8. System queries the database for a list of courses available to take in the semester the user specified
   9. System searches the list of courses for classes that meet the prerequisite requirements
   10. System displays the list of courses to the screen in a scrolling text box, with check boxes that the user can click
   11. When user clicks submit for a class, system performs a check to see if there are any co-requisites, in which case the system notifies the user which class needs to be added
   12. System adds the class to the user’s schedule for the specified semester

6. Viewing supplemental information about classes

6.1 Summary –

When a class is selected in the user interface, a button can be pressed that fetches relevant information from the database, as well as from RateMyProfessor.com.

6.2 Stimulus/Response Sequence

1. User selects a class by clicking on the corresponding check box next to the course ID
2. User clicks the button labeled “Display Info”
3. User sees a text box which displays the course description, and the professors that are teaching the course. Next to each professor name is a numerical score fetched from RateMyProfessor.com, with a color code from green (good) to red (bad) that matches the professor recommendation
4. User can click the “x” on the top of the displayed info box to exit out of the supplementary information view.
5. On click, System fetches data from the database concerning the selected class
6. System also fetches data from URLs like RateMyProfessor
7. System displays information to the screen

7. Editing semester stored in database

7.1 Summary –

User can choose a semester and have full edit controls of its contents

Non-Functional Requirements

Interoperability Requirements:

1. Browser Compatibility: The system will be web-based and operate on major browsers, including Google Chrome, Safari, Mozilla Firefox, Internet Explorer, etc.
2. Operating System Compatibility: The system will operate on multiple operating systems, including OS X, Windows and Linux.

Storage Requirements:

Storage requirements depend entirely on how many users we have using the system

1. Class Data: Determine the amount of data required to store information about each class, such as class name, description, prerequisites, and any additional details. Estimate the average size of this data and multiply it by the number of classes.

2. Skill Map: Consider the size of the skill map data, including the skills and their relationships. This could be stored as a graph or a structured database. Estimate the size of the data based on the number of skills and the average size of the skill entries.

3. Job Matching Data: If you have a dataset for job matching that includes information about different jobs and their required skills, estimate the size of this data based on the number of jobs and the associated information.

4. User Data: Determine if you need to store any user-specific data, such as the classes the user has taken and their progress. Estimate the size of this data based on the number of users and the average amount of data associated with each user.

5. Indexes and Metadata: Consider the additional storage requirements for indexes and metadata associated with the data, which may be needed for efficient searching and retrieval.

Security Requirements:

1. Login/Password System: The system will have a login/password system for the user to access their account. This is to keep all of their information safe and

secure. We are using Firebase, which offers a range of authentication options, including email and password authentication, OAuth-based sign-in with leading platforms such as Google, Facebook, Twitter, and GitHub, and phone number authentication with SMS verification codes.
2. Access Control: The user will have limited access to using the system based on the user interface. Everyone on the development team will be able to access the front end and back-end code.
3. Spam Protection: Firebase Authentication supports the integration of reCAPTCHA, a technology that helps prevent automated bots from abusing registration or sign-in forms. This way, developers can add an additional layer of security to verify that the user is a human and not a bot.

Supportability Requirements:

1. Coding Standards: The system will be coded using HTML, CSS, JavaScript and Python. The code will be produced, routinely reviewed, validated and tested by a third-party software.
2. Naming Conventions: HTML classes and id tags will be written in lowercase, except where there is more than one word in a name, then camelCase will be used. All other programming languages will follow this same format. All Python programming will follow PEP standards.

Availability Requirements:

1. Accessible times: The system should be available for use 24 hours a day, 7 days a week, as it is hosted on GitHub (unless GitHub were to go down for some reason).
2. Downtime Impact: Downtime of the system will be minimal, mostly due to maintenance. A splash page will be displayed letting the user know of this maintenance and how long the system will be down for.
3. Support: The system will have a support option on the navigation bar that will pull up a template that allows the user to report bugs or express any concerns. Their response will be routed to the development team's email account.

High-Level System Architecture

- VSCode (IDE)

Hyper Text Mark-up Language (HTML) – will be the language that will allow the browser to display the website.

Cascading Style Sheets (CSS) – will be the language used to decor the web pages

JavaScript – language used for client-side functionality that will be handled for User Interface (UI) needs to make the user experience enjoyable

Python – is a general-purpose language often used to build websites and software, automate tasks, and conduct data analysis.

chart.js - provides a set of frequently used chart types, plugins, and customization options. In addition to a reasonable set of built-in chart types, you can use additional community-maintained chart types. On top of that, it's possible to combine several chart types into a mixed chart (essentially, blending multiple chart types into one on the same canvas).

React. js – an open-source JavaScript framework and library developed by Facebook. It's used for building interactive user interfaces and web applications quickly and efficiently with significantly less code than you would with vanilla JavaScript.

BeautifulSoup - is a Python package for parsing HTML and XML documents. library that makes it easy to scrape information from web pages. Providing Pythonic idioms for iterating, searching, and modifying the parse tree.

- GitHub/Git

A web-based platform for version control and collaboration that is widely used by software developers to manage and share their code. It provides a centralized location where developers can store their code, track changes, and collaborate with others on projects. At its core, GitHub is based on the Git version control system, which allows developers to create and manage multiple versions of their code. Developers can use Git to create "branches" of their code, which allow them to work on different features or aspects of a project independently. They can also use Git to "merge" changes from different branches back into the main codebase.

- SQLite

Open-source software. The software does not require any license after installation. SQLite is serverless as it does not need a different server process or system to operate. SQLite facilitates you to work on multiple databases on the same session simultaneously, thus making it flexible. SQLite is a cross-platform DBMS that can run on all platforms, including macOS, Windows, etc. SQLite does not require any configuration. It needs no setup or administration.

- Firebase

Backend framework developed by Google which allows for automated authentication, authorization, and database functionality using NoSQL.

- Trello

Software application that allows teams to track issues, manage projects, and delegate tasks.

- Discord

Primary communication application between team members.

- Canvas

Secondary communication application between team and instructor for upcoming Milestone assignments and deliverable requirements.

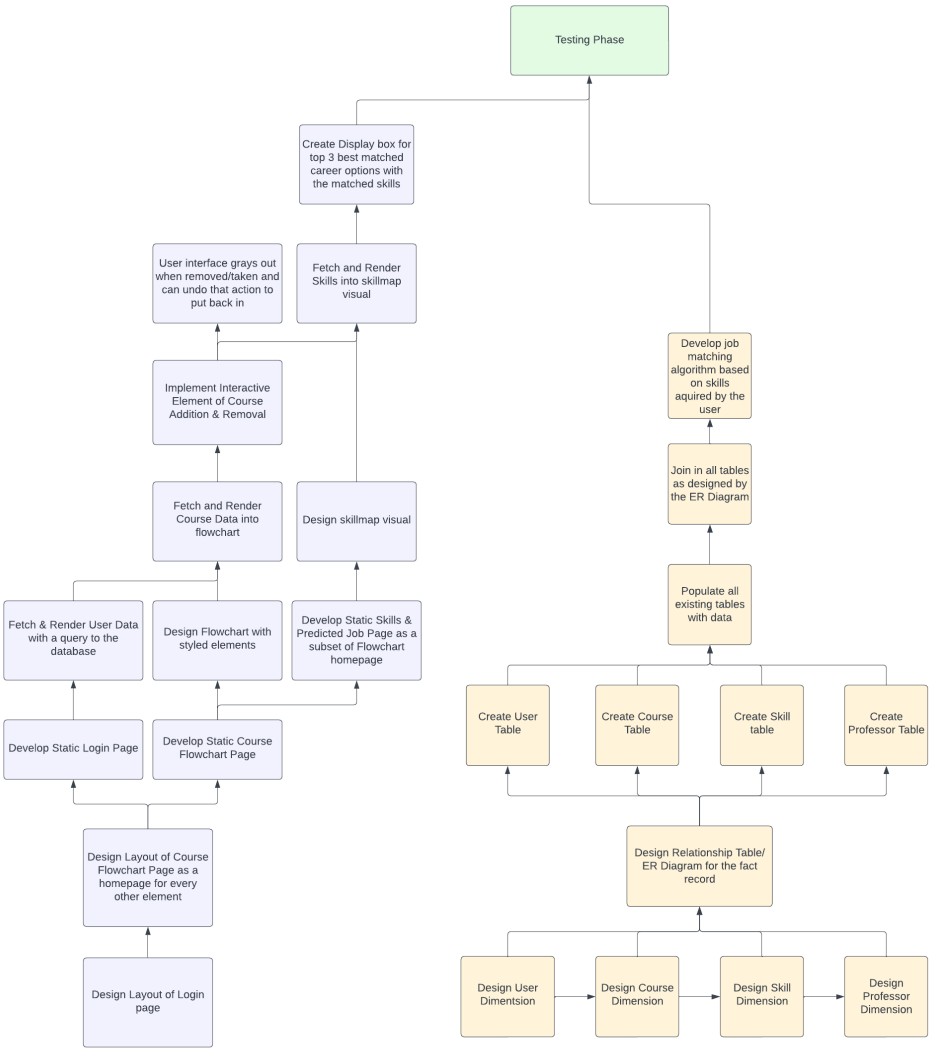
Team

Team Leader:
- Dylan Cohen

Front End Developers:
-Sienna Gaita-Monjaraz
-Maximo Mejia (Front End Lead)
- Dylan Cohen

Back End Developers:
-Sienna Gaita-Monjaraz (Back End Lead)
-Owen Levine
-Cassidy Wilson

GitHub Master:
- Dylan Cohen



Checklist

For each item below you must answer with only one of the following: DONE, ON TRACK (meaning it will be done on time, and no issues perceived) or ISSUE (you have some problems, and then define what is the problem with 1-3 lines). Reflect these items in your Trello project space:

a) Team decided on basic means of communications -DONE
b) Team found a time slot to meet outside of the class -DONE
c) Front-end and back-end team leads chosen -DONE
d) GitHub master chosen -DONE
e) Team ready and able to use the chosen back and front-end frameworks -ISSUE
    There is a disagreement about the structure of the database.
f) Skills of each team member defined and known to all -DONE
g) Team lead ensured that all team members read the final M1 and agree/understand it before submission -ISSUE
    Poor communication on the due date.

Project Links

GitHub Repository:

- https://github.com/dylancohen2018/FAU_FlowTrack

Trello Dashboard:

- https://trello.com/w/thesquadflowtrackappcen40101