FAU College of Electrical Engineering
& Computer Science

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

Milestone #3

More Detailed Requirements, Architecture and a Vertical Software Prototype

Authors

Name	Email
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 on the user inputting courses they have already taken. A list of courses publicly available from FAU will be processed and saved to create a relational database using Firebase Realtime Database that will be consumed by the web application in javascript & HTML. For each class, the database will hold professor, semester available, corequisites, prerequisites, and skills that relate to each class. With this data, we can display to the user the classes they can directly take next semester. 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 seeall the skills they have acquired from each of their completed courses along with their proficiencies in each of these skills.This feature provides a visual representation of a student's strong and weak points–facilitating further skill development and helping students make educated decisions in semester courses.

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. Along with that we take the users intended career and show them how their skills match up with employers desired skills for that career. This similarly helps students to decide which classes to take next semester based on the skills they need to further develop to match up with their intended career.

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 table 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). Our platform offers class suggestions in weak parts of skills. 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
Navigation Bar	Service/User Interface	Site User Service	A toolbar that helps the user navigate to different areas of the site
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
- There is a page that can be accessed from the Navigation Bar called “Career Planner”, where user can select a career from a drop down menu and generate the skills for that career.
    - A double bar graph will be generated that compares the user’s current skills with the skills required for the selected career to determine their qualification status.
    - The screen will either display that the user qualifies for the career or doesn’t, and if they don’t, it will generate and display a list of classes that the user can take to qualify.
-

For reference – basic framework for website

- Back-end
    - Firebase Real-Time Database
    - Using test data, based on FAU Computer Science Flowchart.
- 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 email, first and last name, and age.
3. User enters the required information.
4. The app will then have the user create a password.
5. The app validates the information and saves it to the user's profile.
6. The app displays a confirmation message, indicating that the profile has been successfully created.
7. 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
8. User then selects from a list their desired career paths along with being shown careers that line up well with their current skills
9. 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 inputted the classes they have already taken

Flow of Events:

1. User visits the website and logs into the system
2. In the registration tab, the user is given menus to select the semester and reason(core/elective) for classes they are eligible to take.
3. System retrieves previously taken classes, separates the classes by semester and reason.
4. User can click on classes in the list to view class information.
5. 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 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. The page will display a box with a drop down arrow containing all of the possible careers for the computer science major and a “generate skills button”.
4. App displays user’s current skills and the skills the selected career requires. It will also generate the user’s qualification status.
5. If the user does not qualify for the selected career, it will populate a list of classes that the user can take that teach the required skills for the selected career.

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,,. Once the account is created and activated, users can log in and explore the features of the course registration, skills, and career interface.
   - **1.2 Stimulus/Response Sequence**
     1. Users will provide by student email and create a password.
     2. System will check if required personal information is valid.
     3. System will enforce specific email and password requirements and validate that the user has met those requirements.
     4. System will store the user's personal information and password.
     5. .
     6. System will have buttons for the user to navigate to the. other pages.
   - **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 emailand 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. They are then directed to register.
   - **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,the user is prompted to select classes taken, 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 –

After login in, the page displays a table of classes.The user selects the classes taken by checking the boxes. 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 into their account
2. User marks the box of the class taken.
3.
4. The user can enter in first name, last name, and intended career.
5. User clicks “submit” button, which adds the course to a database record for their account
6.
7.
8. System enters chosen class into the database under that user’s account

4. Viewing available courses to take based off already taken classes

4.1 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 registration page.of the application
2.
3. User selects the semester from a dropdown menu of future semesters
4. User selects between core and elective classes from a dropdown menu.
5. User can click on a class to view class information, such as class name, instructor, etc.
6. User can view supplemental information as outlined in **6.1**
7.

8. System queries the database for a list of courses available to take in the semester the user specified
9.
10. System displays the list of courses to the screen in a scrolling text box,
11. System adds the class to the user's schedule for the specified semester

5.

7. Using the Career Page

7.1 Summary

User can select a career from a drop down menu and generate a bar graph comparing the user's current skills and the required skills for the selected career. The page will also display the qualification status and if the user does not qualify, it will generate a list of classes that the user can take to qualify for the selected career.

7.2 Stimulus/Response Sequence

1. User selects the Career Page on the navigation bar.
2. User is prompted to select a career from the drop down menu and presses the button "Generate Skills."
3. System grabs the user skills and required skills data from the database and generates a double bar graph comparing the two.
4. System grabs the classes that teach the required skills for the career and compares those to the classes that the user has already taken, generating a list of classes that the user can take to obtain the skills required for the selected career. If the user has already taken the class, it is not added to the list.
5. System displays the qualification status of the user.
6. System displays a list of classes that the user can take to qualify for the selected career if they do not qualify already.

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 is handled by Firebase for the back-end. GitHub for the front-end.

1. Class Data: Every class holds the name of the class, instructor, prerequisites, corequisites, semester available, and skills gained from that class along with the proficiencies of those skills. In total we have about fifteen classes holding each of those facets. Every class is sorted by wether its a core class or an elective class. In total we have 36 classes that a user can select from or mark as taken.

2. Skill Table:. . For every class there are at least three skills associated with that class. Locally we populate the proficiency level instead of storing it inside the database so it doesn't take up to any extra storage in the backend.

3. Job Matching Data: We store in the database careers and each of these careers have skills associated with the career and desired proficiencies for that career. In the user we store what the users intended career is. In total we have around eleven classes that have around five skills with desired proficiencies stored in firebase realtime.

4. User Data: We store users name, last name, their intended careers and the classes they have taken along with the skills they have acquired from those classes.

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 –which our team chose to use.

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.

Supportability Requirements:

1. Coding Standards: The system will be coded using HTML, CSS, JavaScript and
2. Json in firebase . The code will be produced, routinely reviewed, validated and tested by a third-party software.
3. 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.

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).

High-Level System Architecture

Database Organization

The system utilizes a database to store information about careers, classes, and users. The database can be organized into separate collections or tables for each entity. Here is a suggested structure for the collections/tables:

1.Careers:
    1. Careers:
  - Fields:
    - Career ID: Unique identifier for each career.
    - Career Name: The name or title of the career.
    - Required Skills: A nested object or array that stores the required skills for the career. Each skill is associated with a skill level, represented as a numerical value.

2. Classes:

- Fields:
  - Class ID: Unique identifier for each class.
  - Class Name: The name or title of the class.
  - Class Reason: Indicates whether the class is a core or elective requirement.
  - Prerequisites: A nested object or array that stores the prerequisites for the class. Each prerequisite is referenced by its ID.
  - Corequisites: A nested object or array that stores the corequisites for the class. Each corequisite is referenced by its ID.
  - Professor: The name of the professor teaching the class.
  - Semester Available: The semester (e.g., Fall, Spring) in which the class is offered.
  - Skills Taught: A nested object or array that stores the skills taught in the class. Each skill is associated with a level or importance value.

3. Users:
- Fields:
  - User ID: Unique identifier for each user.
  - First Name: The first name of the user.
  - Last Name: The last name of the user.
  - Z Number: A unique identifier for the user within the system.
  - Intended Career: The user's intended career choice.
  - Skills: A nested object or array that stores the skills possessed by the user. Each skill is associated with a level or proficiency value.
  - Classes Taken: A nested object or array that stores the classes taken by the user. Each class is referenced by its ID.
  - Authentication: Stores the password hash or authentication information for the user.

Media Storage

As of right now we don't have any media but when we start stylizing our website and adding graphics they will be stored in an image file in the code repository.

- 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.

.

- 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.

- 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.

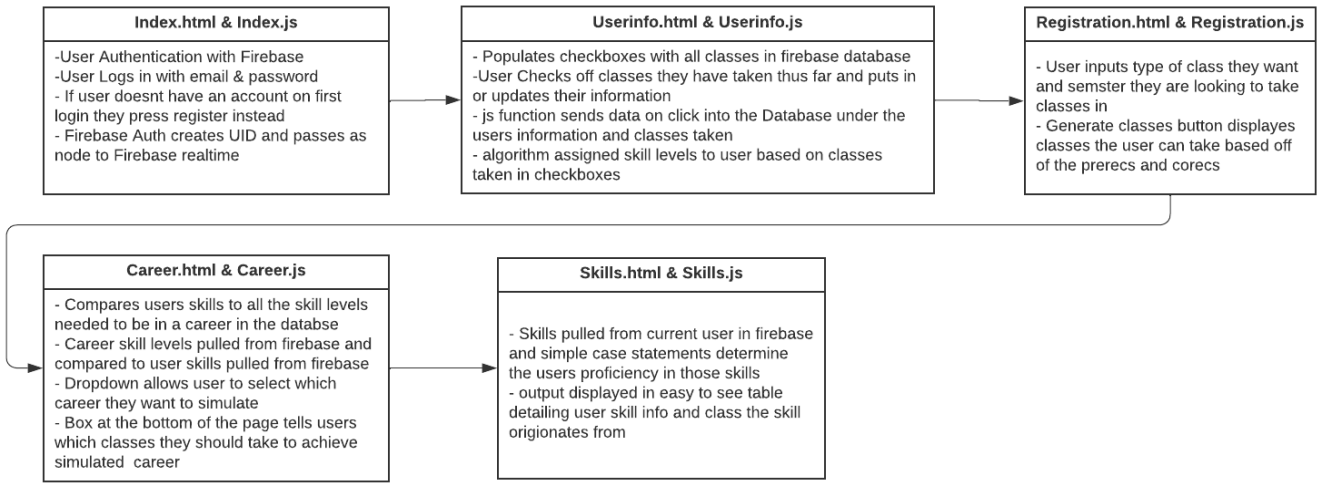
- Discord

Primary communication application between team members.

- Canvas

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

UML Overview



Risks

1) All team members contributed to front-end development. Team members have an amateur knowledge of javascript and HTML/CSS. This leads to a slightly longer troubleshooting time.
2) Despite the extent of the planned features, we are on track to provide a complete product with all the listed features.
3) Interfacing with Firebase is the principal technical issue that arises in development. This has been handled by consulting online resources and platform documentation.
4) Most team members submitted completed assigned tasks. A previous member dropped the course, so the remaining members had to partake in additional work to compensate for the missing effort. Another team member had a family emergency and was unable to complete work so all of the work fell on three of the project members very close to the due date.

Team

Team Leader:
- Sienna Gaita-Monjaraz

Front End Developers:
-Sienna Gaita-Monjaraz
-Maximo Mejia (Front End Lead)
- Cassidy Wilson
-Owen Levine

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

GitHub Master:
-Sienna Gaita-Monjaraz

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 -DONE
f) Skills of each team member defined and known to all -DONE
g) Team lead ensured that all team members read the final M3 and agree/understand it before submission - DONE

Project Links

GitHub Repository:

- https://github.com/siennaphia/FAUmyAdvisor

Trello Dashboard:

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