Software Design Document

Version 1.1
12 February 2021

Team DigiFolio
Dr. Andy Wang
Fabio Santos

Burbank, Logan(lead)
Braudaway, Jackson
Chen, Kailin
Marschel, Parker
Yang, Zhenyu

Table of Contents

1. Introduction

Millions of students are moving through college on a yearly basis, and most are not well prepared to start their career. Typically, a majority of students will come out of college with their only work experience relating to their major involving the courses themselves. This lack of experience makes it difficult for the recent graduates to find a job in their career field, whether it be a general or specialized position. This is especially true in fields where the employers are expecting employees to have a good understanding of the work environment before coming in. In the end, this can leave hundreds of thousands of college graduates working in fields where their degree is unnecessary, and typically these are not the career the graduate desires.

As it stands, many universities are setting up programs to help get their students on a successful career path. This is often done through online systems that allow for employers to look for students to fill internships. An example of this is with Handshake, which will constantly alert both students and employers when one or the other is in need of an intern/internship. The main issue with these problems is that the programs are voluntary, and the university typically has no system in place for tracking how many students are doing what. The current method of tracking this information relating to career milestones is typically done through exit surveys. These are given to students just before they graduate, and while they are capable of presenting the information to the university, it is often obtained at a point where the information cannot be used to help the students it represents.

Currently, Dr. Wang, the dean of CEIAS, is attempting to have every student achieve two career goals: every student obtains a job offer within 6 months of graduating, and every student has an internship by the time they graduate. As mentioned just previously, this is hard for Dr. Wang to track as there is no easy way for the students to relay this information to the faculty so that Dr. Wang can know what percentage of students still need to work on getting an internship or job offer. Once he is capable of seeing the progress of the students in a clearer way, he can then focus more on improving the university resources given to students in need of career assistance.

Taking all of this into account, our team, along with Dr. Wang, has designed a piece of software that will be capable of handling career information for students, while giving them the power to see their own success. In order to achieve this, our program will build specific web pages that can be filtered and searched through based on the user's desires. Ultimately, the first iteration will allow faculty to see how many students from a specific program have achieved an internship (or similar experience), and they can also see how many have received a job offer. Through these two goals, Dr. Wang hopes to make progress in bringing 100% of students to this point before graduation.

Outside of the statistical side of the program, we expect to implement a profile page, giving students and employers a useful tool. Students will be able to have a profile showcasing their career milestones, as well as giving some contact details. These profiles will be searchable, with the profile owner choosing what people are able to see. Once this is set up, employers will be able to easily find students who have achieved milestones they may be looking for, and at this point they can easily reach out to the student to see about offering a position.

Finally, in order to implement the different pages, we expect to have a backend database that the site will read from. This database will have the ability to not only store user and milestone information, but it will be capable of looking at a user's LinkedIn page and pulling milestones from it, storing them in the database. This will give the program a semi-automated process for filling in the milestones for students, giving them less work to do.

Taking this all together, the main problem we are trying to solve involves the following:

- Dr. Wang has specific career goals for the students of CEIAS to achieve before graduation
- There are no tools that allow for the university to easily track the career progress of the students
- Students are worse off due to not having career experience when leaving the university to build a career

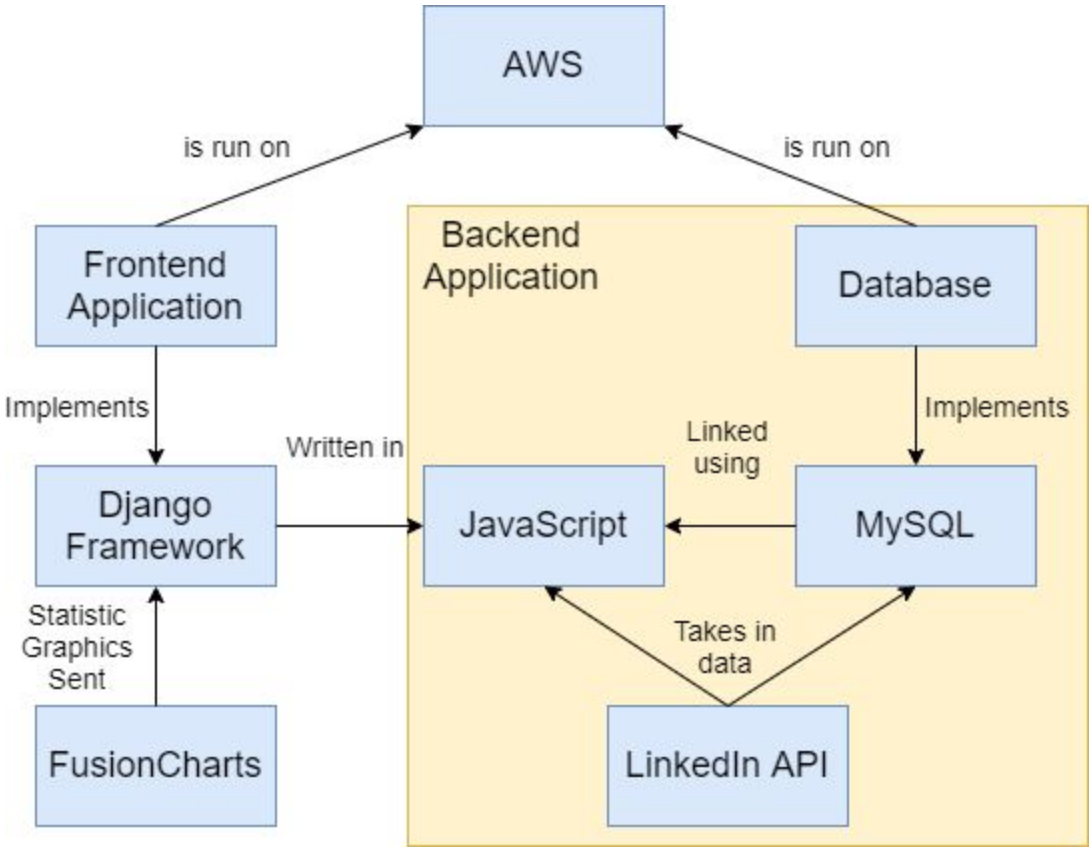
In order to solve these problems, we plan to implement a piece of software capable of the following:

- Student career milestones will be scraped from student LinkedIn accounts
- The information will be taken in and compiled to create statistical graphics of the overall progress of the students using the software
- The milestones will be put together on a profile page for each student, giving the user access to a page displaying their own career progress
- The students and faculty will be able to search through other student profiles, seeing information that the profile owner has set to public visibility

Through all of these pieces, our team expects to build an application that will help Dr. Wang track student success, while also giving the students a useful resource for their career. In order to begin implementing the different pieces, we have come up with our implementation design, and we will be going through the different pieces in the following sections.

2. Implementation Overview

Working with Dr. Wang, we have formulated a blueprint for our software, the Engineering Career Builder (ECB), which will be capable of pulling career information off of student LinkedIn accounts, and compiling the data into a simple-to-use site. One use will allow for students to pull up a profile page highlighting their career milestones. This information can then be used to help students create resumes, or to motivate them to gain more experience when they see an empty page. The application will also allow faculty to pull up statistics on how many students have specific experiences, such as an internship, so the faculty can track student success throughout their college careers, and not just as the students are leaving.



Our general solution is shown above. To begin, the core of the whole system is AWS (Amazon Web Service). Everything included in the software will be implemented onto AWS, including both our frontend web application and the backend database supporting the application. The frontend application will be created by implementing the Django framework. This will provide a framework for our frontend interface so that users can interact directly. The framework will process data and provide users with graphical dashboards using FusionCharts. The Django framework is mainly implemented in Python, but there are some ways to make it interact with JavaScript. The backend database will be implemented through a MySQL database management

system. MySQL can also be implemented through JavaScript using tools in Node.js, which will allow us to query the database using JavaScript. At last, the LinkedIn API does have JavaScript integration tools. Since both the database and the API can use JavaScript, this will allow us to receive data from LinkedIn to populate the database.

In order to implement our solution for the ECB, we will use the tools and technologies listed below. Each tool has been tested and we have decided on the following:

MySQL Database: Data Storage and Synchronization

MySQL is a relational database management system that allows for a database to be set up using tables. We have come across this system through other classes of ours, and the relational model allows the SQL language to be used through multiple languages. Considering our technical feasibility analysis, we know MySQL will be able to accomplish our requirements.

AWS Server: Where the System Runs On

Amazon Web Services are highly customizable with a lot of choices of servers. Amazon also includes free trials, as well as free tiers for their servers. If the free tiers are not enough, they also allow multiple payment options for their services. We will run the whole system, including both the frontend and the backend, on this server.

FusionCharts: Generating Dashboards

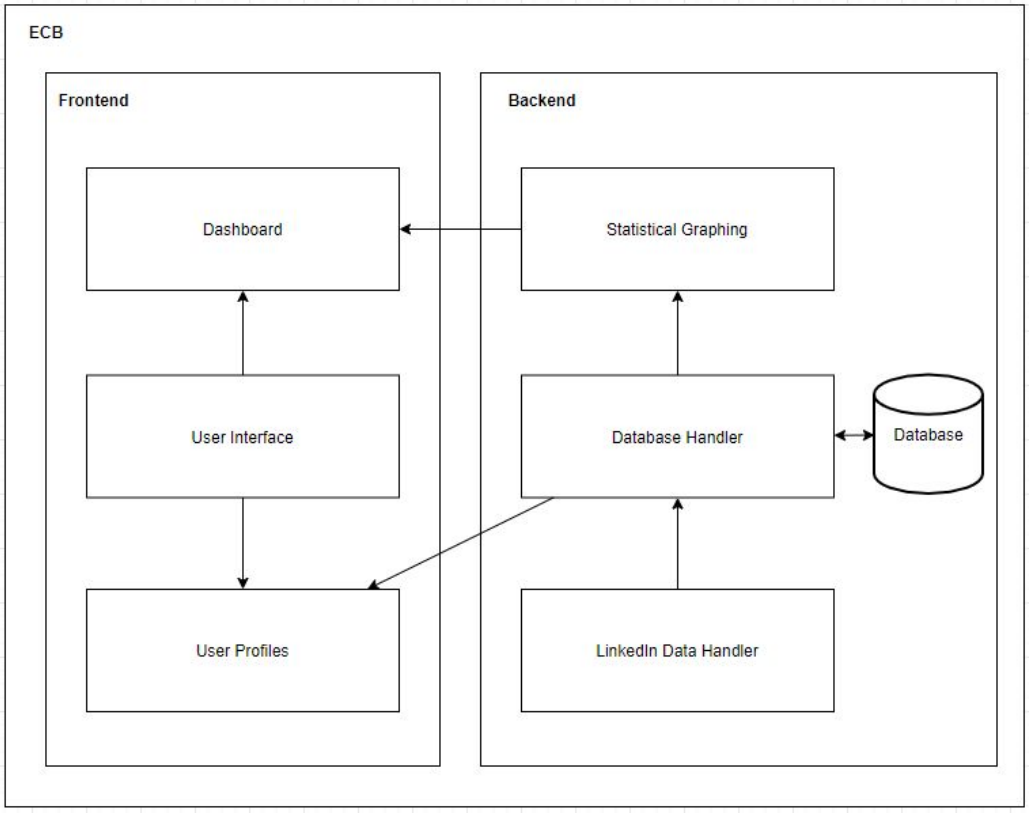
FusionCharts is one of the most widely used charting libraries that uses JavaScript. It can be installed via CDN, npm, or locally. This tool is very easy to learn how to use, it has comprehensive documentation for each library with lots of live examples. It also has great configurability, which includes over 100 charts and 2000 maps. We can generate statistical dashboards using this tool to help users understand which position he or she is in.

Django Framework: Database Integration and Support, API Support

Django follows a model-view-template style and is one of the most popular open-source web frameworks available, while also being compatible with Python, a language we are all familiar with. Another important feature of Django is that one of its main focuses is making database-driven websites easier to develop compared to other available frameworks. It also contains several packages to extend its original behavior, including API provision and consumption. Once again, it is written in Python which adds to the user-friendliness of the framework. This framework will assist in the development and implementation of our web

application by reducing the overhead associated with web development by providing APIs, URL routing operations, user interaction, authentication, etc.

3. Architectural Overview



In the overview of our architecture we have some key major parts. We have the backend, the frontend, and the application as a whole. The frontend will consist of three main parts, the dashboard, the user interface, and the user profiles. The backend will also consist of three main parts with an added database to hold all the data of the application itself. The three main parts of the backend are the Statistical Graphing, the Database Handler, and the LinkedIn Data Handler. Overall, the entire application holds all of these components to create the application as a whole.

The components that the frontend consist of are what the user gets to interact with. The most important component that the user interacts with is the entire user interface itself. This user interface will allow a user to register with the site, create a profile based on their academic success, and navigate throughout the site to look at the overall success of every student. Next, is the user profile. The user profile will allow a user to create their own profile based on all their academic milestones, as well as make their profile to keep as a resume profile through NAU.

Finally in the frontend, there is the dashboard where all the student's milestones and accomplishments will be displayed in graphical form to see the overall accomplishments of the students.

Profiles have many requirements which involves implementing a user system that will allow for user accounts with role-based permissions. There will be three account types, with these types being Student, Faculty, and Alumni. Each of these types will have specific basic functional requirements. A student's use in this application will involve logging in and out, viewing portfolio pages, modifying their own page contents and visibility, and seeing the overall progress. The faculty account is just like the student account, with the exception that it is for the permissions of a faculty member. The biggest difference here is that the faculty member is able to modify any student page, but is not able to select what pieces are public. Also, the dashboard will show faculty individual student progress toward her/his goals, as well as a summative progress for a program, a department, and a college. Finally, the faculty will also be able to modify a student account's status, allowing student accounts to become alumni accounts through a faculty member. Ultimately, an alumni account is similar to a student account, with the ability to request page updates rather than being able to directly modify the page. This is so the information being added can be verified by an administrator. The profiles will talk back and forth between the database and the user's profiles to make sure data is up to date.

The user interface is a combination of all the features including the dashboard, the user profiles, and the website itself. In this user interface a user is allowed to register and create a profile. This profile will then be able to have its own personal profile page which has many features that are included in the profile section. If a user is already registered, then they are able to simply log in. The user interface also includes the dashboard where overall goals will be displayed.

Next on the frontend is a digital dashboard capable of displaying the current progress of the student population in achieving university-specified goals. For now, there are two different goals this dashboard will be capable of tracking: the total percentage of students with an internship (including externship, Co-op, research project with experiential learning outcomes), and the total percentage of students with a job offer. This dashboard will get its data from the database in the backend. It will then calculate anything based on the goals that are implemented on the website.

The components that the backend consist of is where the developers connect all the components together into one big application that the users get to experience. First there is the Linkedin data handler which will scrape data from a LinkedIn account and move all the information into the database. Second, there is the statistical graphing component where it will take all the data and calculate specific goals and display them in a graph on the website. Lastly, there is the database handler and the database itself where all the information will be stored and organized to be accessible by any requests made from the website.

This LinkedIn scraping will be used to populate the student portfolio pages. This module must be able to pull information from LinkedIn, analyze and classify the data, and store the data in its correct place. The main purpose of the scraping is to classify the information pulled from LinkedIn into categories. These categories as we see it will include two important ones to take note of, with the first being any identifying information, and the second being a career milestone. LinkedIn will likely not make it obvious what each career milestone we are pulling entails, so this module will check and place the data in the right spot once it is pulled. This LinkedIn scraping will be used from an API that pulls data from a user's connected linkedin profile which will then be placed inside the database that holds all information in an organized manner.

The statistical graphing will be a function that is done semi-automatically, with the dashboard module needing to be able to calculate how many students have earned an internship, externship, co-op, or research/experiential learning opportunities. As this will be displayed in multiple possible ways, the dashboard will achieve this by keeping a total count of students and consistently updating the number who have had this internship. This way, the two values it needs are always available to display the results as a fraction or as a percentage. Also, the internship will be checked using a flag attached to the student, to allow for quicker calculation when running through all of the students. The module will already be storing the total number of student accounts, and it will now store the number of students who have received a job offer. Then, it works like the previous requirement in how it will check and calculate the results.

The database is where all information will be stored. The database will then communicate with its database handler who then distributes all the information to the designated areas such as the user profiles and the dashboard. The database is the most important part of the website because, without the database, there would be nothing to display about anything or anyone.

The overall architecture design will be a web two-tier. Web two-tier has only two-tiers: the client application and the database. The client application, where the client can send requests to the server side without a middle man having to interfere with any of the requests. The server side then handles the database where it can store information without having to have a third action to keep all the information. Our users will simply be able to create their profile which will go straight to our database and then the request will be sent back with their profile and any information they may have entered into the database from the frontend. Designing our architecture as a web two-tier makes it simple and easy to maintain.

4. Modules and Interfaces

In this section, we will go over the multiple modules and interfaces we will be using in order to implement our software product. These modules are the primary pieces to make the program work, and we will break each one down into the different parts and methods each module will need.

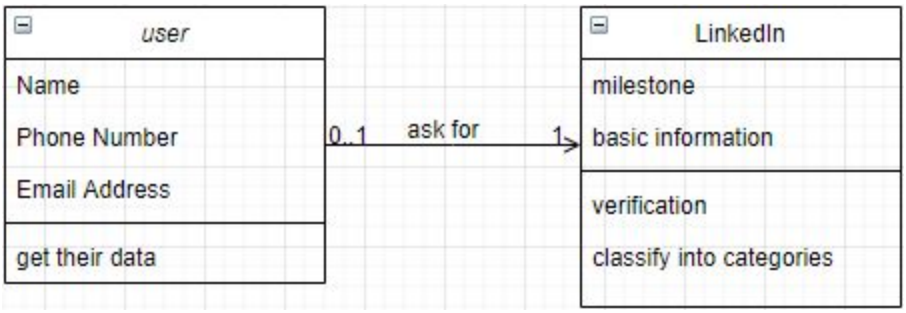
4.1 LinkedIn Data Scraping Module

The LinkedIn module must be able to pull information from LinkedIn, analyze and classify the data, and store the data in its correct place. Therefore, the LinkedIn module has two main functional requirements. One of them is to pull data from LinkedIn servers, and the other is to classify the data into categories.

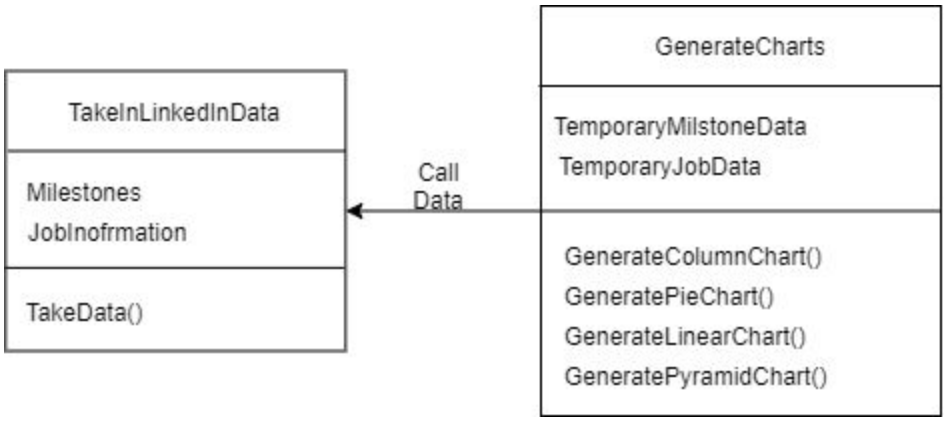
For the first requirement, once the users ask data from their own interface, a request will be sent to the LinkedIn server using its API. Then it will verify the result that is sent, and unpack the result into temporary memory. These stored results will then be used for the other functional requirement, and this step will typically be done automatically, with the choice to scan for new data manually.

After classifying the data, which has been stored at the last step, into categories, these categories as we see it will include two important ones to take note of, with the first being any identifying information (LinkedIn API uses email), and the second being a career milestone. LinkedIn will likely not make it obvious what each career milestone we are pulling entails, so this module will check and place the data in the right spot once it is pulled.

The result that was pulled from LinkedIn successfully is the most important step of the whole module. We are trying to pull data from LinkedIn using different ways to make sure the success rate. Although we have a rough prototype, changes will be made within our modules to ensure that further development will be modular and reusable.



4.2 Statistical Graphing Module



The statistical graphing module will be able to take in the data which is pulled from LinkedIn, analyze it, and then generate statistical graphs based on the milestones of the userbase. In order to achieve this purpose, this module will have two responsibilities:

- Taking in data that pulled from LinkedIn
- Generating statistical charts

The first responsibility will interact with the LinkedIn module. First, the LinkedIn module will pass some data it pulled, and the statistical graphing module will check if the data is valid. Then it will analyze it, and store the data in a temporary data structure for the convenience of calling and reading the information back.

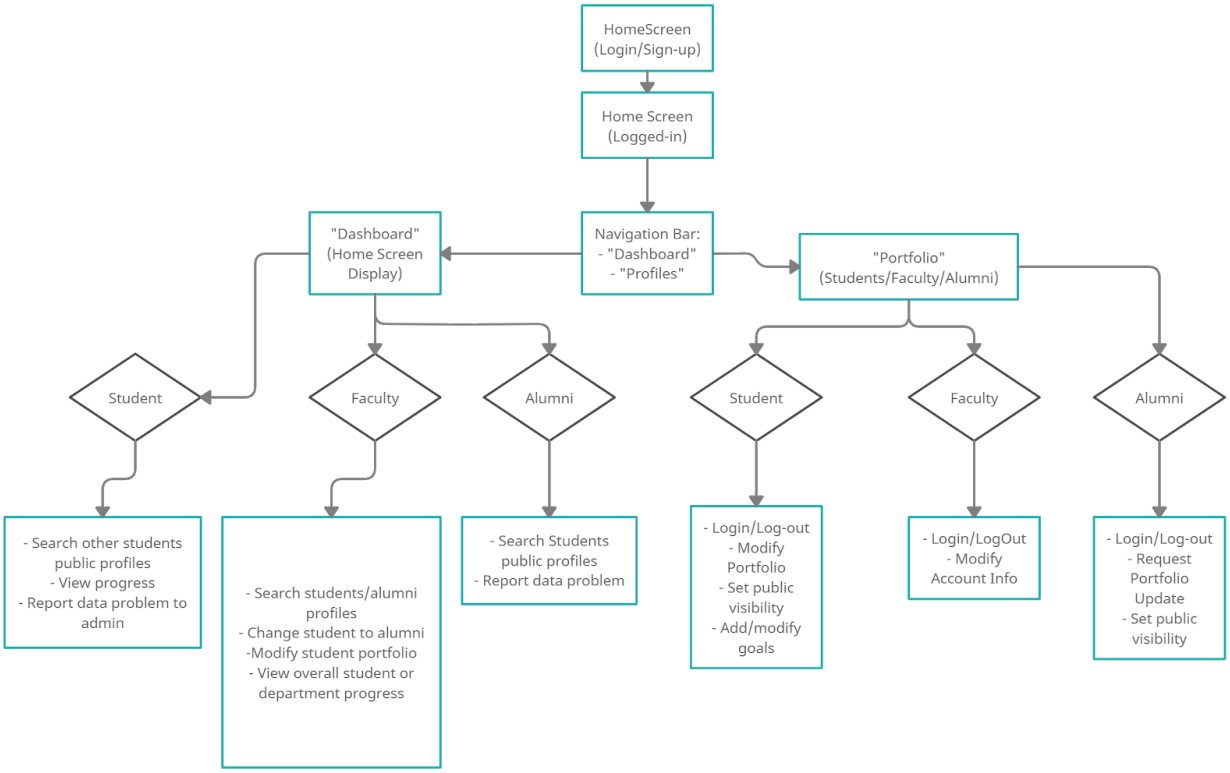
After gathering and storing the data, the second responsibility will check which kind of charts that user wants to have, then call the corresponding charts' function, the charts will show the statistical information clearly, and make it easy to understand. After generating the charts, this module will interact with the user interface, and the charts will be shown in the Dashboard bar.

The TakeInLinkedInData entity has two variables Milestone and JobInformation, and one function TakeData(). The function will get data from the LinkedIn module and pass them into two variables.

The GenerateCharts entity consists two variables TemporaryMilestoneData and TemporaryJobData, and four functions GenerateColumnChart(), GeneratePieChart(), GenerateLinearChart(), and GeneratePyramidChart(). The data stored in two temporary variables is passed by the TakeInLinkedInData entity, and four functions will create corresponding statistical charts using the data in temporary variables.

4.3 User Interface

The user interface will be able to provide the user with all of the necessary operations they will need to do on the application from traversing through different public profiles to editing their own profiles, etc. With three different privileges on the application the user interface will differ slightly for different users. However, it will remain the same for certain pages such as the home screen, which will prompt the user with a login or sign up option that will also be the same no matter who is logging in or signing up (i.e. students, faculty, alumni). As stated prior, once logged into the system this interface will provide different features depending on the user.



Student users, once logged in, will have a home screen (dashboard displayed) which will contain a navigation bar. The bar should have options such as “Dashboard” to return the user back to the home page, “Portfolio” that will take them to their own profile page, and a search bar to look up other students' public profiles. The portfolio page will allow a student to logout, modify their portfolio, set public visibility preferences to different features of their portfolio, and add/modify their goals. The dashboard will allow the student to view their own goals and progression toward them as well as being able to report an issue to administrators.

Alumni privileges will have the same options across the navigation bar as student users. However, upon entering into their own portfolio they will not have any goals to add or modify,

rather they will be able to logout, set public visibility preferences to aspects of their profile, and request to have their portfolio updated. Going back to the dashboard, alumni will be able to search students and other alumni profiles as well as report a data problem to administrators.

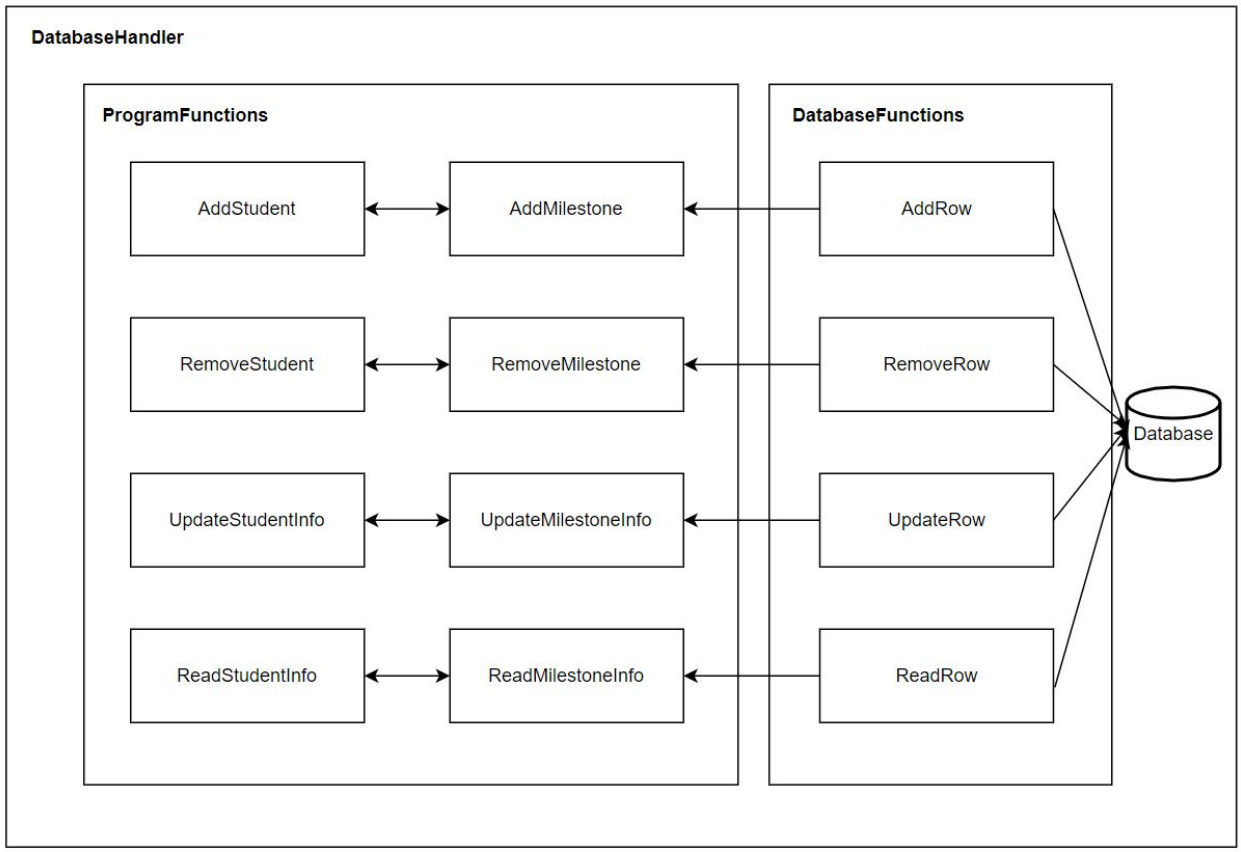
Faculty will be the most different as these users will have the highest level of privileges on the system. Upon login, a faculty user will also have the same navigation bar. However, when searching alumni or students the faculty member will be allowed to make changes to their profiles including changing a graduating students privileges to that of an alumni, or edit any other parts of a profile necessary. The dashboard for faculty members will also provide them with the option to view student profile progress or overall department progress. If the user goes to their profile page they will be able to logout or modify their profile (username, password, etc.).

4.4 Database Handler Module

The database handler module will be implemented in order to easily handle reading and writing to the database our system will run off of. The primary purpose of this handler is to keep all database operations in one place, so if the database is ever worked on or modified, the changes will only be seen in one module. In order to achieve this purpose, the module will have the following responsibilities:

- Adding and removing information to the database
- Modifying information inside of the database
- Reading information from the database

These three responsibilities, while basic, will allow the rest of the modules to be sent information stored in the database, as well as send requests to change the database without having to worry about each module connecting directly to the database.

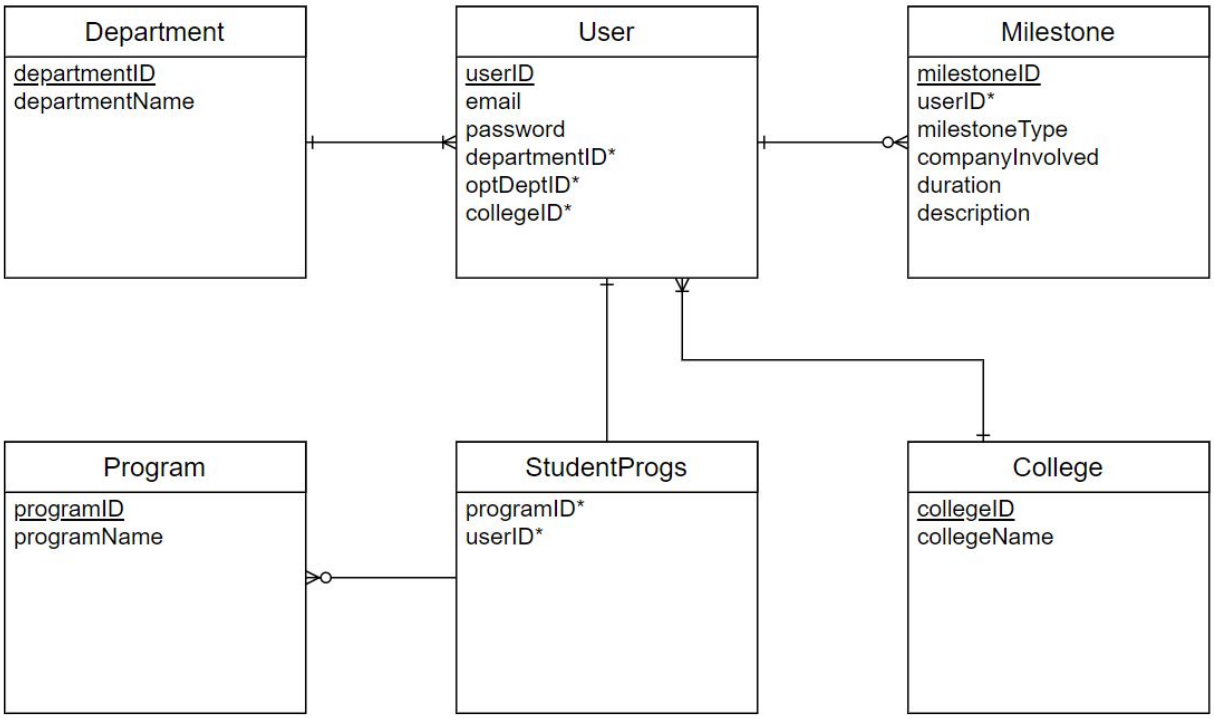


The three responsibilities are then broken down into pieces of two different submodules within the database handler. These submodules are the Program Functions, which will include the public functions the other modules will have access to. The other module is the Database Functions, which will be internally accessible to the database handler, with the only other connection to the module being the database itself.

The Program Functions will primarily focus on the two different entities we will be tracking in the database: students and milestones. For each of these, the functions accessible to the other modules are to add, remove, update, and read a row of information from the table. Each of these will be directly related to a private function in the Database Functions submodule, as both students and milestones will use similar functions to one another. In this regard, the Program Functions will invoke the Database Functions to actually connect with the database.

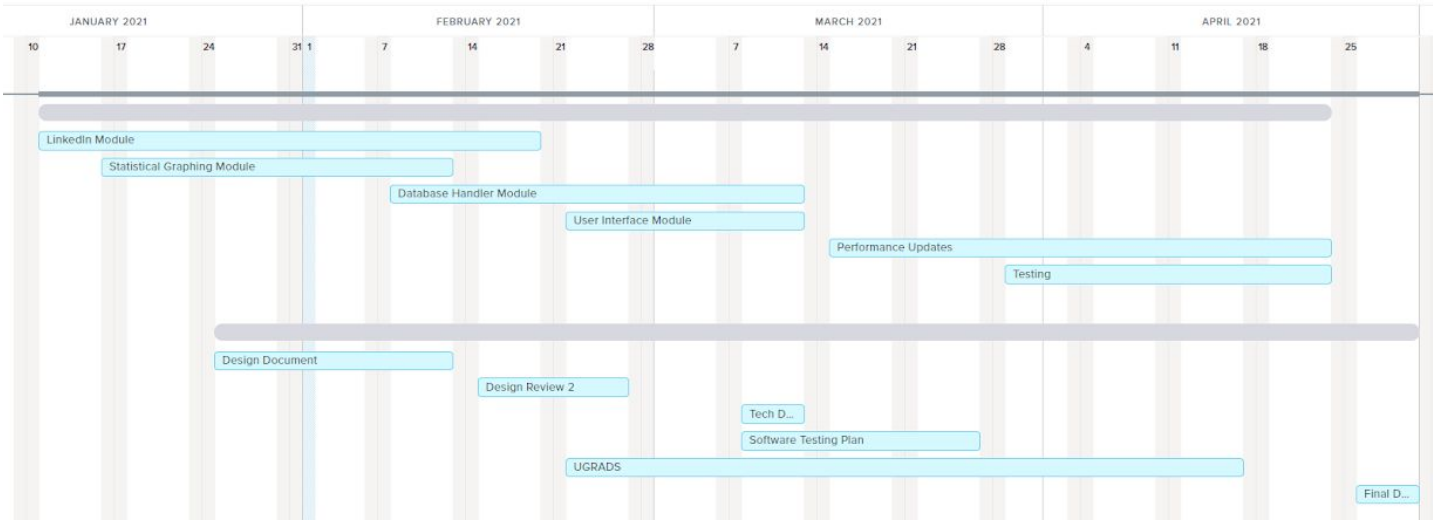
The Database Functions will consist of an AddRow, RemoveRow, UpdateRow, and ReadRow. The AddRow function will return a boolean value, with True being success, and False meaning the row was unable to be added. The RemoveRow function will return the row being removed stored in a custom data structure based on the entity being removed. The UpdateRow function will update a row, and return a boolean once again to show success of the operation. Finally, the

ReadRow function will return the row being requested in a data structure based on the entity being read.



The diagram shown here illustrates the different entities we expect to handle within our database. These are the pieces that will hold information we expect to use for our statistical analysis, as well as the student profile page displays. The user will be the primary entity everything else is centered around, as this is a tool based around user information. In the future, we do expect that companies themselves can be added as entities to allow for better filtering tools, but for now, companies will remain in the milestone information that they apply to.

5. Implementation Plan



In line with our current plan, we are working on the LinkedIn module and the Statistical Graphing module, as those will add the most core functionality to the site. We also expect these two modules to be the more complex ones to work with, so we figured it is best to work on them early to give a better perspective on how quickly we can work through the challenges. We expect to have these modules working within the coming two weeks, where we will start to then work on the Database Handler module.

The Database Handler module is the next option for our work as we plan on connecting the LinkedIn Module, as well as the Statistical Graphing Module, to the database to allow for better testing of the system as a whole. Once enough of the Database Handler is implemented to allow for this testing, we will focus on testing the LinkedIn module's ability to store data, and the Statistical Graphing module's ability to retrieve and use the data for its calculations. We expect this testing phase to take about a week in order to make it sufficient. Following this, we will then begin to implement the User Interface module.

The User Interface module is expected to be a bulky, yet fairly simple module to implement. With this, we expect it to only take us about three weeks to complete, from beginning the implementation to fully testing the module. This is because the module itself has a lot it is handling, like the dashboard and user profiles, but the real work is in the other modules that will already be finished. Using those modules, the User Interface module will primarily involve implementing the actual interface for each page, which makes the implementation expected to be quick.

The current plan is to get all of the functionality of these modules implemented by the middle of March. This way we can prepare a demonstration of the product working by then, and we will then use the remaining weeks to work on updating the performance of the system. A heavy part of this will involve speeding up the lookup on the database, as well as the calculations of the statistics for the graphing module.

In splitting up the modules, each one has been assigned to a different member of the group to allow for a more specialized lead for each piece. The first two modules, the LinkedIn Data Handler module and the Statistical Graphing module, are being led by Zhenyu Yang and Kailin Chen, respectively. Then the Database Handler module will be led by Logan Burbank, and the User Interface module will be led by Parker Marschel. Jackson Braudaway, as our lead design architect, will be the lead on the project as a whole when we come to putting the pieces together.

6. Conclusion

To conclude, our solution will be a web application that automatically pulls user data from LinkedIn and creates displays of the information. The four modules we envision it having include: the LinkedIn Data Handler module, the Statistical Graphing module, the User Interface module, and the Database Handler module. At the moment, the main problem that we face is being able to account for the highlights of students' careers. When a student gets an internship or a job, NAU wants to highlight that fact and encourage every student to pursue anything that will help them get a job in the future.

Dr Wang, our client, plans to set a goal for NAU Engineering students that every single one of the students will have had at least one internship or experience in the degree they're pursuing by 2025. With our website, we hope to create a place where students can constantly be putting in milestones onto their profile page where NAU can see and track each student's progress and even help them get job offers. Our team hopes to create a seamless digital portfolio experience that gives the students tools, while giving the university a better perspective on their student base. We look forward to bringing this design to fruition as we move into these coming months.