WHAT'S UP DOC

Software Design Document - Fall

9/30/2022

Project:
C & I Doctoral Tracking Tool

Project Sponsor:
Gretchen McAllister

Faculty Member:
Michael Leverington

Team name:
What’s Up Doc

Team Members:
Adam Larson (Lead), Brandon Shaffer, and Eddie Lipan

Team Mentors:
Daniel Kramer

Version 1.1

Table of Contents:

1.0 Introduction

Attending and graduating college constitutes some of the most formative and adversarial years of a person's life. According to a census taken by the Education Data Initiative in 2021, 3.1 million post baccalaureate students are enrolled in graduate programs at universities in the United States[1]. Generally these students are striving to learn and better themselves each day of attendance but many will likely face similar adversities, such as timeliness, deadlines and examinations, all while balancing social lives or work schedules. With all these stressors, graduate students should be able to reliably track and manage their graduate program without fear of adding additional stress.

Most undergraduate courses are implemented through Learning Management Systems (LMS) such as Blackboard Learn or Canvas, which act as course delivery and grade feedback tools. This is possible as most undergraduate courses classify as "pass/fail" or fall under some version of the standard A-F grading system. Assignments and examinations from class to class are normally quite rigid in that they pertain to lectures or chapters of an assigned reading, thus they may be outlined and planned before a semester even begins. This rigidity is absent in graduate school as graduate students perform activities that are less quantifiable, such as shadowing mentors and researching. Therefore, it is considerably harder to deliver the graduate program using a LMS.

The Coordinator of the Curriculum and Instruction (C&I) doctoral program at Northern Arizona University (NAU), Gretchen McAllister, Ph.D., and Administrative Services Assistant, Michele Benedict, have both experienced the shortcomings of Blackboard Learn firsthand. As a result, neither is able to track the milestones of their graduate students. In order to attain candidacy in the C&I Ph.D program, students must complete a minimum of 60 graduate-level course units, where the average course is 3 units. C&I graduate students must also complete professional development requirements, comprehensive written and oral exams, a qualifying research paper, get the approval and assignment of a dissertation committee, and get the submission of an approved dissertation prospectus.

[1] Hanson, Melanie. "College Enrollment Statistics [2022]: Total + by Demographic." *Education Data Initiative*. 22 Jan. 2022. <https://educationdata.org/college-enrollment-statistics#:~:text=Report%20Highlights.,students%20are%20in%20graduate%20programs.>.

Feedback is a vital aspect of learning and for undergraduates, this can be achieved through a simple three step process; take an exam, turn in said exam, and receive a grade. However, as discussed above, providing graduate student feedback through an LMS is not feasible. At present, Gretchen and Michele are having graduate students funnel all deliverables to a single computer via email which are then held locally in folders labeled with the student's name. No further organization is performed other than storing the deliverable into the appropriate students' folder. Graduate student grading and submission inquiries are achieved by emailing Michele directly, who then browses the files locally, and provides feedback via email. The current data management process is inefficient and taxing for both parties involved.

Team What's Up Doc's solution is an intuitive website application, allowing graduate students to track their progress and receive daily motivational messages up until candidacy. The website application will also allow faculty to view student progress and create detailed reports based on their remaining milestones. Given that some students may not have a technical background, the app would require an intuitive, visually appealing dashboard making it easier to upload deliverables and track progress, all while avoiding the time-consuming correspondence with Michele. Another goal of the website application is to avoid manual data entry for either Gretchen or Michele.

The purpose of this Software Design document is to provide a blueprint for our final website application. This will include outlining our overall architectural design and outlining our implementation plan of said design. This document will expose any design/implementation flaws ahead of time so that we may address them accordingly before they become real issues. Team What's Up Doc will be sure to keep this document updated in order to accurately reflect our current software design.

2.0 Implementation Overview

The two main goals of Team What’s Up Doc’s solution is to offload the data management roles required by faculty, and to provide graduate students with a proper way to track their status in the doctoral program. Our solution to the data management problem within the NAU C&I doctoral program is an approachable website application that helps track the graduate students' milestone progress. Considering this project is for a CoE and not a business, there will not be a producer-consumer pattern. Instead, the website application will require a certain level of accountability out of the graduate students as they will be tracking their own milestones, but if used properly it could be invaluable.

In order to implement our website application, Team What’s Up Doc is managing a MySQL database to better tackle the sensitive data at hand. On the front end, we will use HTML forms for the website application. We then chose to style our HTML with CSS and incorporate the back end and front end using Spring Framework.

Team What’s Up Doc chose to move forward using the relational-database MySQL for a number of reasons. The first reason was that our team members had familiarity and experience with MySQL which would greatly help us in the implementation process. Second, MySQL is open source, and trusted by many programmers throughout the community. This entails that MySQL is rich with great documentation and ensures that if any problems are met, we can receive guidance from the MySQL developer community.

HTML and CSS were chosen on the front-end for similar reasons. Both of Team What’s Up Doc’s front end programmers have experience with web programming and both have used HTML and CSS to successfully style and create intuitive web pages. Not only do we have experience and familiarity with the two web programming languages, but we also appreciate its simplicity and the fact that HTML and CSS web pages can be opened on all platforms.

Lastly, Team What’s Up Doc chose to move forward with Spring Framework to meld together our front end and back end. We chose Spring Framework due to its abundance of packages, such as Spring Boot, Spring Authentication, and Spring Security, as well as its great documentation. Each package can be implemented through dependency-injection which allows us a significant level of control throughout our web application.

3.0 Architectural Overview

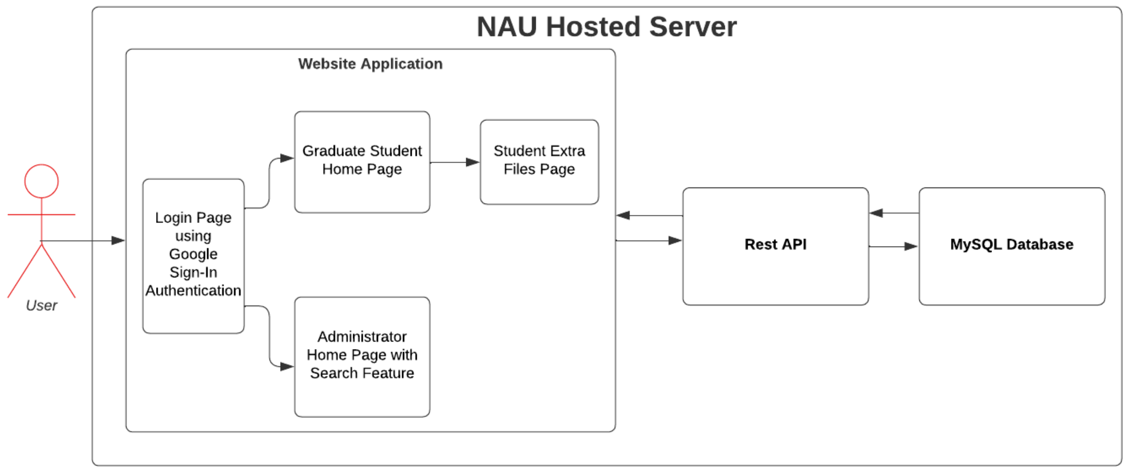

NAU Hosted Server
Website Application
User
Login Page using Google Sign-In Authentication
Graduate Student Home Page
Student Extra Files Page
Administrator Home Page with Search Feature
Rest API
MySQL Database

Figure 3.1 - Architecture

The architecture of our system involves a web application, server, and database, all hosted through NAU servers. Communication between components requires an internet connection to access and retrieve data.

Figure 3.1 begins by having a user attempt to access the web application and be redirected with Google's Sign-In option to the NAU servers. If a graduate student email is recognized, they will be taken to the graduate student home page. From this home page, students will be able to view milestone widgets horizontally across their screen, and upon clicking each milestone, they will be presented with a taskbar on the left-hand side of the interface that tracks smaller tasks within a larger milestone. Once signed in with their school email account, the web app will request said students' files and information based on their alphanumeric student ID. The server will receive the request and query the MySQL database, returning either that student's information or an error message if the student doesn't exist. This information is returned to the web application from the server and displayed by code on the home page. Modifying requests are handled similarly but change, add, or remove data within the database. Administrative roles, for developers and faculty, will have additional post and delete requests to add and remove students. If a student wants to upload or download a document, they can do so straight through this student home page. The only other page the student will be able to access is an extra files page for extraneous files that may not pertain to a particular milestone.

In the event an administrator email is recognized during Google Sign-In, the administrator will be taken to an admin home page. From this home page, the administrator will be able to search and filter sensitive program information in order to view the overall progress of a graduate student or of the program as a whole. The administrator will also be able to add and remove students from the program through this admin home page. Due to the inherent capabilities available to an administrator from the admin home page, it is imperative that after Google Sign-In is bypassed, the web application is able to properly discern whether the email used is a student email or a faculty email. This vital security check will be done by our REST API middleware, which allows us to maintain great levels of security, authentication and control. The implementation of our architectural design will result in a responsive and secure website application that assures the right people are seeing the right information.

4.0 Module and Interface Descriptions

Web Application

User-Interface

The User-Interface will consist of a webpage where a student or faculty member would be able to access the database. HTML forms will take user input and simplify the database request process from a user's perspective. This benefits the users who will not all be computer literate, and would otherwise struggle with accessing their files. The faculty will have the ability to access various students' information, while a student will only be able to access their own. Whether a user is faculty or a student will be acquired by the implementation of Google Sign-In. Google Sign-In will be implemented in order to determine whether a user is faculty or a graduate student. This authentication process will affect the way the User Interface looks, depending on what type of user they are identified as.

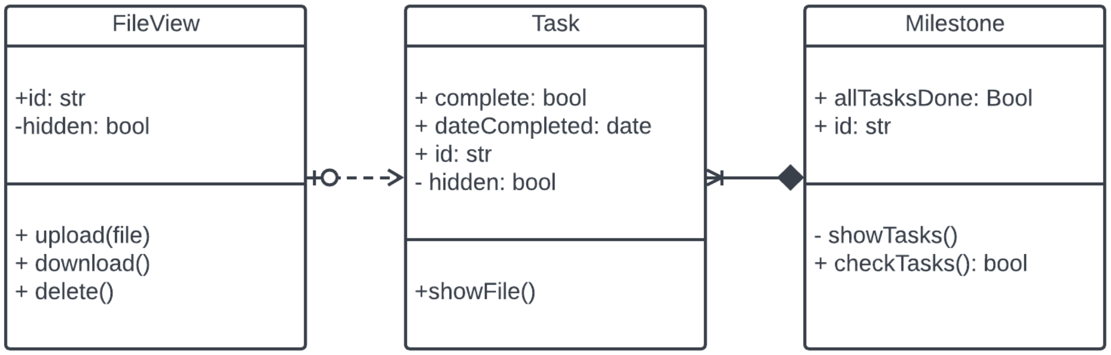


Figure 4.1 - Interface UML Diagram

Google Sign-In

While no sensitive information will be stored within our program, Google Sign-In is implemented to securely authenticate and deliver the appropriate information. The benefit of this is that all NAU students and faculty school accounts are managed by Google, and is a work around to not being able to connect directly to student accounts. Students and faculty will provide Google with their NAU email address, routed to the CAS login page, then back to our page to have their information displayed. The REST API will process this information based on login approval.

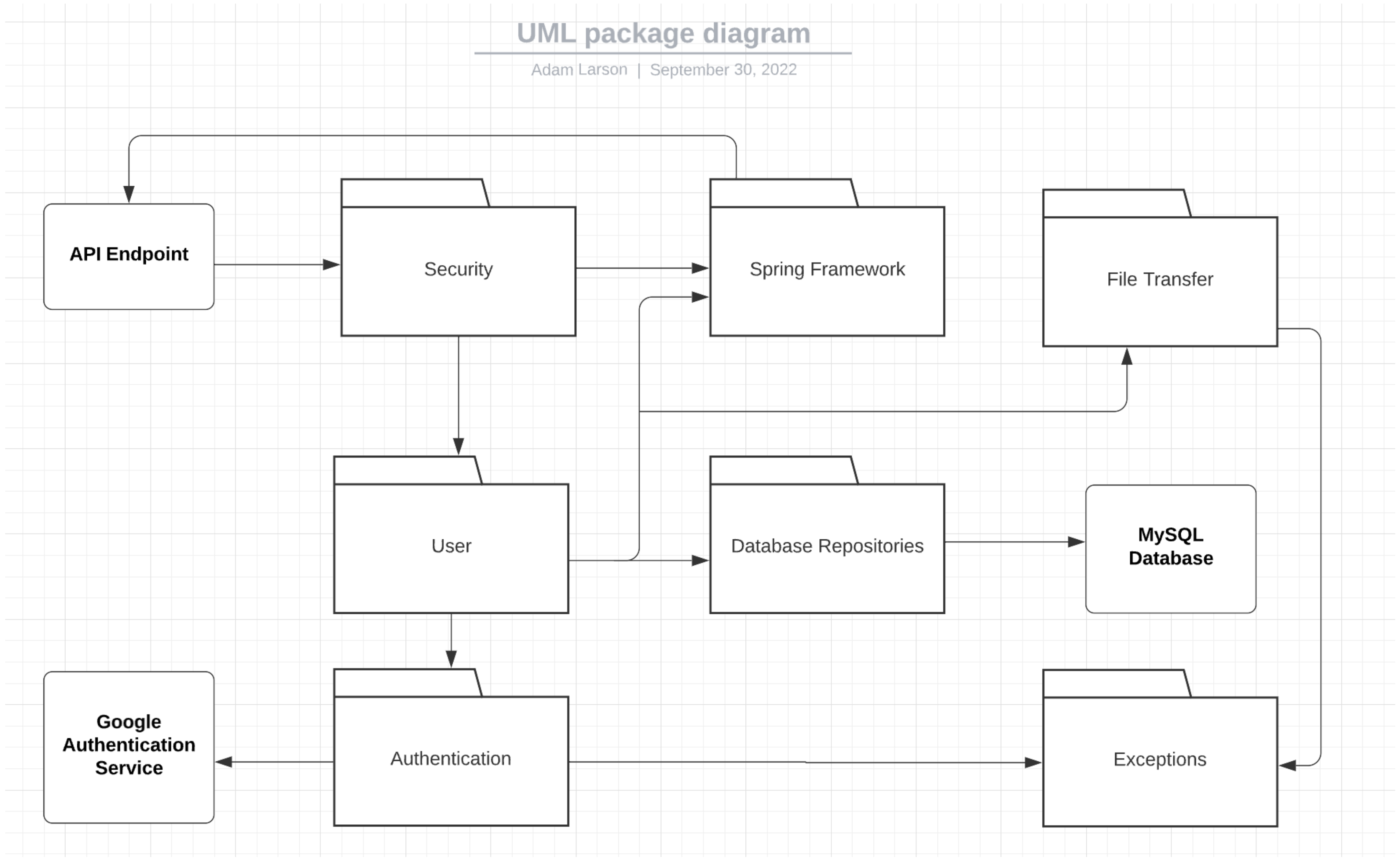


Figure 4.2 - UML Package diagram for all rest API components

Rest API

Spring Boot

The API being used for our server application is with Spring Boot. This is a Java based application that allows for significant extension through the use of available libraries and manages the overhead set up to allow more focus on functionality. It is an entirely standalone, Maven (Gradle also available) based application using a pre-configured Apache Tomcat web server. Initialization through https://start.spring.io provides all required dependencies by means of a .zip file and allows additional modules to be included automatically. Once initialized, web controllers responsible for receiving and sending information are necessary and operate on desired HTTP requests (get, post, put, and delete for this product) and data. Annotations are used on all classes and many fields and methods allowing the inner working of the framework to find and utilize the desired functionality.

Spring Security

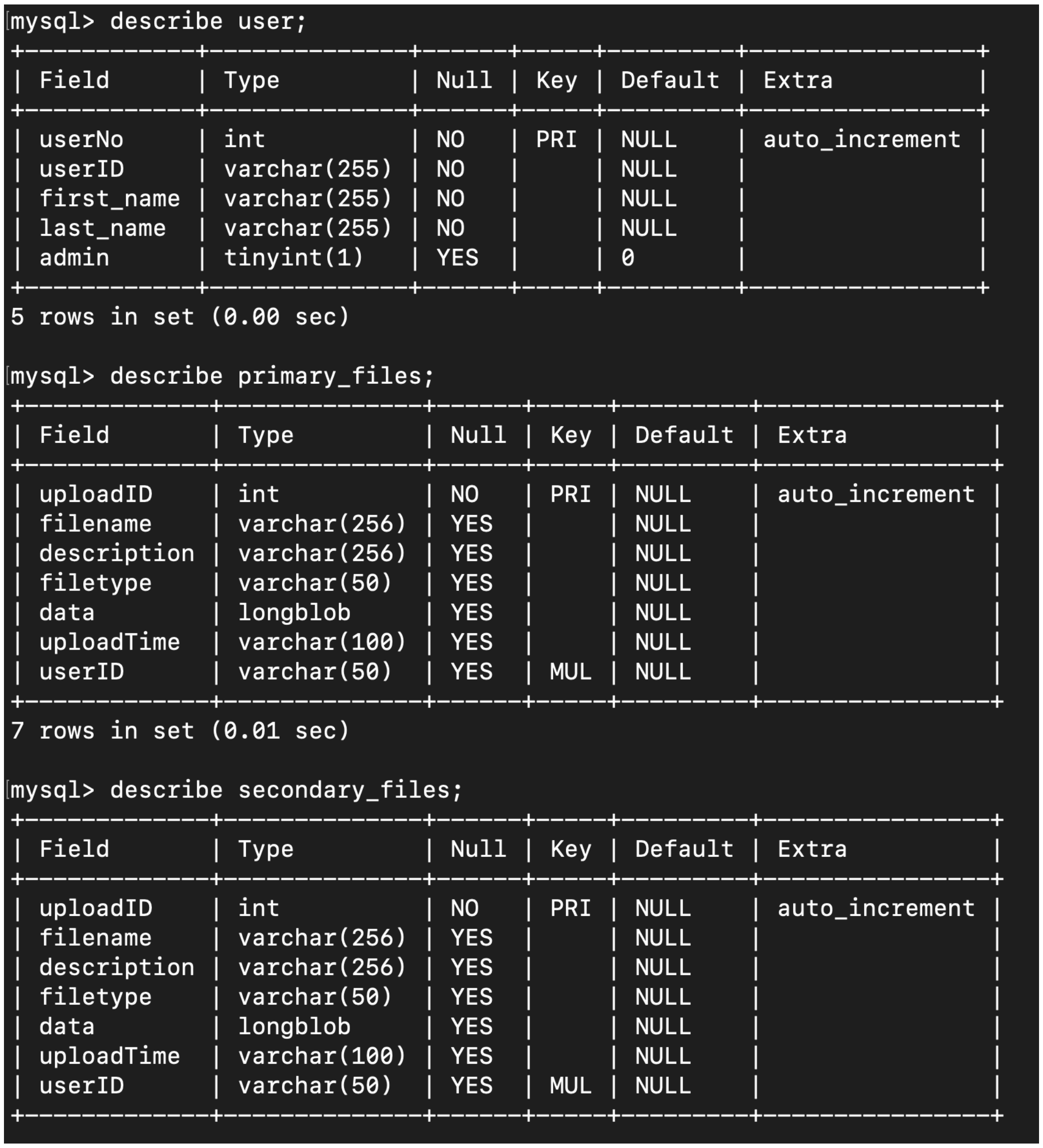
Spring Security is a module that is not included in the base Spring Boot program. This additional feature allows security features and communications to be easily integrated into the application. Adding this module to the Maven project is done simply by adding a dependency to the pom.xml and rebuilding the project if not done automatically. Policies are required to be set to utilize this module and are provided via either the application.properties file or an additional class with connections to the security module.

The additional features provided by this module include connectivity to large tech company login API's (Google, Facebook, GitHub, etc.) for login functionality without setting up a local authentication system. This application is also taking advantage of credentials that must be supplied when the web application is making requests. Currently this is set using a randomly generated alphanumeric name:key credential, both of which are 25 characters long, varying case, and numbers 0-9. This module includes a significant amount of features, however these two are in use.

Spring JPA

This is another module that is supplementary to the Spring Boot program. For our program to effectively track student data and files a database must be used. Spring JPA allows for incredibly fast connectivity and implementation. Unlike the Java Database

Connectivity (JDBC) module, JPA prioritizes minimal code and utilizes keywords in methods and fields via a CRUD (create, read, update, delete) repository interface. If necessary, custom SQL commands can be created to retrieve desired information within this interface. A JPA repository interface is also available if pagination or sorting is required, though the CRUD interface will suffice.



```
mysql> describe user;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| userNo     | int          | NO   | PRI | NULL    | auto_increment |
| userID     | varchar(255) | NO   |     | NULL    |                |
| first_name | varchar(255) | NO   |     | NULL    |                |
| last_name  | varchar(255) | NO   |     | NULL    |                |
| admin      | tinyint(1)   | YES  |     | 0       |                |
+------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql> describe primary_files;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| uploadID    | int          | NO   | PRI | NULL    | auto_increment |
| filename    | varchar(256) | YES  |     | NULL    |                |
| description | varchar(256) | YES  |     | NULL    |                |
| filetype    | varchar(50)  | YES  |     | NULL    |                |
| data        | longblob     | YES  |     | NULL    |                |
| uploadTime  | varchar(100) | YES  |     | NULL    |                |
| userID      | varchar(50)  | YES  | MUL | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)

mysql> describe secondary_files;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| uploadID    | int          | NO   | PRI | NULL    | auto_increment |
| filename    | varchar(256) | YES  |     | NULL    |                |
| description | varchar(256) | YES  |     | NULL    |                |
| filetype    | varchar(50)  | YES  |     | NULL    |                |
| data        | longblob     | YES  |     | NULL    |                |
| uploadTime  | varchar(100) | YES  |     | NULL    |                |
| userID      | varchar(50)  | YES  | MUL | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
```

Figure 4.3 - MySQL Table Structure

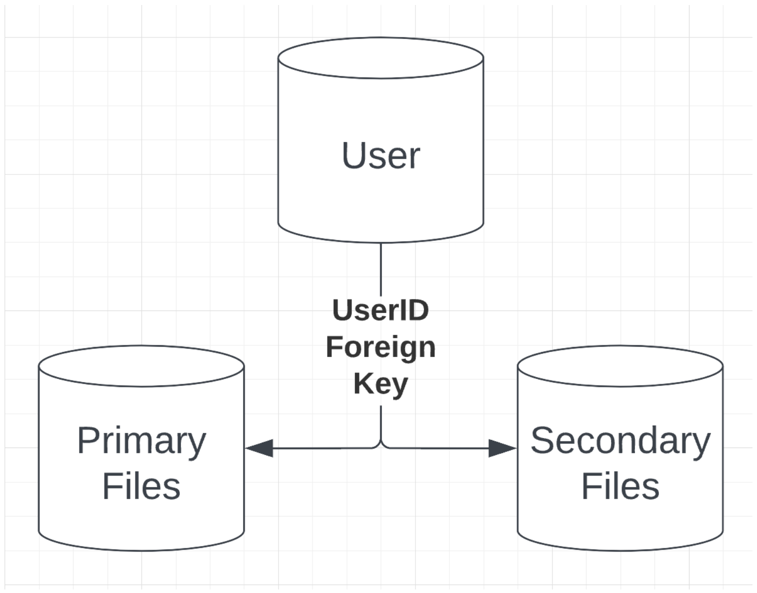
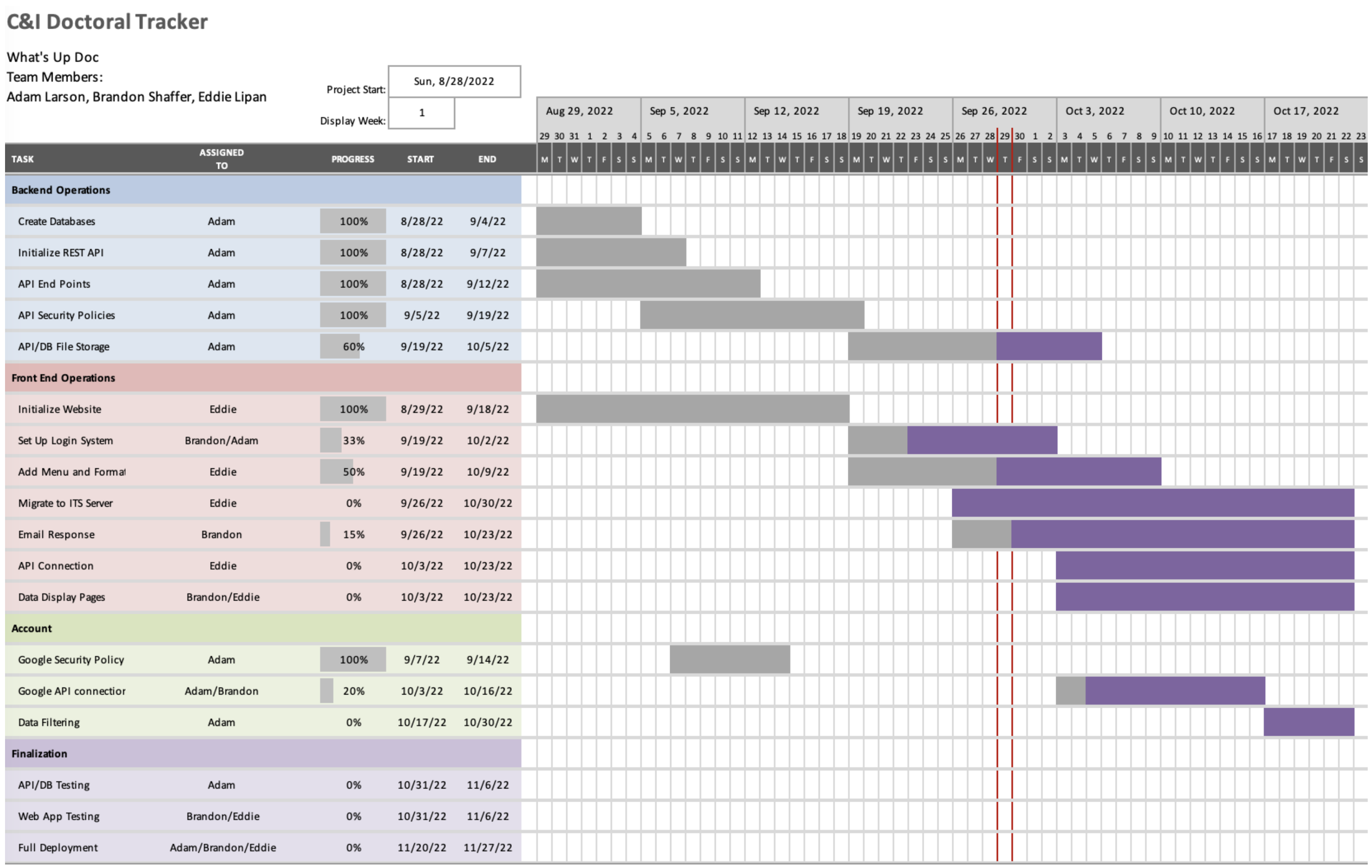


Fig 4.4 - MySQL Database connections

Database

MySQL

The database that will be used is MySQL. This was chosen due to familiarity, popularity, and documentation. The database structure consists of three tables: a primary user table; a primary file table; and a secondary file table. The user table holds the student first/last names, student ID (alphanumeric), and admin status (boolean). Admin status in its current state refers to either group members or faculty. File tables are identical structurally, however files will be filtered into these based on file name by the REST API. Structure of these tables include the file name, description, type, data, upload time, and student ID (alphanumeric).



C&I Doctoral Tracker

What's Up Doc
Team Members:
Adam Larson, Brandon Shaffer, Eddie Lipan

Project Start: Sun, 8/28/2022
Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END
Backend Operations				
Create Databases	Adam	100%	8/28/22	9/4/22
Initialize REST API	Adam	100%	8/28/22	9/7/22
API End Points	Adam	100%	8/28/22	9/12/22
API Security Policies	Adam	100%	9/5/22	9/19/22
API/DB File Storage	Adam	60%	9/19/22	10/5/22
Front End Operations				
Initialize Website	Eddie	100%	8/29/22	9/18/22
Set Up Login System	Brandon/Adam	33%	9/19/22	10/2/22
Add Menu and Forma	Eddie	50%	9/19/22	10/9/22
Migrate to ITS Server	Eddie	0%	9/26/22	10/30/22
Email Response	Brandon	15%	9/26/22	10/23/22
API Connection	Eddie	0%	10/3/22	10/23/22
Data Display Pages	Brandon/Eddie	0%	10/3/22	10/23/22
Account				
Google Security Policy	Adam	100%	9/7/22	9/14/22
Google API connectior	Adam/Brandon	20%	10/3/22	10/16/22
Data Filtering	Adam	0%	10/17/22	10/30/22
Finalization				
API/DB Testing	Adam	0%	10/31/22	11/6/22
Web App Testing	Brandon/Eddie	0%	10/31/22	11/6/22
Full Deployment	Adam/Brandon/Eddie	0%	11/20/22	11/27/22

Figure 5.1 - Fall Semester Gantt Chart

5.0 Implementation Plan

(See Figure 5.1 above for Timeline)

Back-end implementation has largely been completed, with the exception of the API/Database storage. The task is awaiting allocation of server space from NAU ITS, which is also delaying the migration task. Front-end implementation is underway, though has encountered several delays. Initializing the website was extended to readjust the design to better meet client expectations. During our first meeting of the semester with Gretchen, she expressed more specific expectations for the appearance of the front-end, as well as brought attention to a miscommunication on what user data would be connected to. The confusion was whether the database would be sorted by a student's student ID, the seven digit ID, or the NAU login. A work around was agreed upon due to the fact that we do not currently have access to student ID's. This delay was further exacerbated by illness and temporary complications with communication within the team. Connecting the website to google's login system and creating the website's primary interface is on track for being completed on schedule. Our current schedule is set to ideally have our product ready for its alpha demonstration a week early to allow for any further delays or complications.

Testing is being conducted during the associated tasks as well as plans for future tests after the alpha demo to refine our application. To ensure data security we are testing to ensure data cannot be accessed by sources lacking the proper credentials. The database's access via GET, POST, PUT, and DELETE methods as well as admin access to methods will be tested, as well as HTTP return codes, during the website's API connection. HTTP error codes will be tested for and should lead to redirection to an error page. The access methods were tested for on the database side using the Spring Unit Test Suite. After the alpha demonstration testing will shift towards ensuring a positive user experience, using a class that Gretchen has selected. These tests will largely focus on elements like the color scheme of the interface, readability of the font, and other more subjective aspects of the user experience.

6.0 Conclusion

Successfully completing a graduate program is no small task, and it becomes even more challenging if a student has trouble tracking their progress within the program. Team What's Up Doc aims to reimplement the data management process for graduate students enrolled in the NAU C&I doctoral program through an easily accessible website application. A working College of Education website application would introduce an efficient data management mechanism for C&I graduate students and offload a tremendous amount of stress for our clients, Gretchen McAllister, Ph.D., Coordinator of the NAU C&I doctoral program and Michele Benedict, Administrative Services Assistant. Not only would this application reduce the stress of our clients, but if utilized properly, it would greatly reduce the stress associated with tracking a graduate level program and give the students valuable feedback on their progress within the program.

This Software Design Document is meant to provide the architectural details or "blueprint" of our final product. By explicitly outlining our architectural design and our implementation plan, we are able to better identify any potential problems that might arise from our particular design choices. This ability to plan and foreshadow possible outcomes based on our design choices could save Team What's Up Doc from headaches in the near future during implementation of the website application. If any changes need to be made, the continual updating of this document will serve to reflect Team What's Up Doc's current software design elements and implementation plans.

As of the completion of this Software Design Document, Team What's Up Doc has implemented the majority of their back end components as well as the middleware that will be connecting our website application to our MySQL database. We are currently in the process of designing and implementing the front end of our website application, specifically our login page, which will utilize Google Sign-In, as well as our two different user home pages. Our client, Dr. Gretchen McAllister, is requiring a certain look and feel for her website application, therefore the front end User Interface will constantly be changing and adapting as we meet her needs and the students needs. Finally, Team What's Up Doc has been in the process of acquiring a permanent server location for our website application and database through NAU. As a result, we will have to migrate to NAU servers when available.