Digital Backpack Final Report

May 4, 2022

**Team Logout**

****

Sponsored by Dr. Morgan Vigil-Hayes

Community Aware Networks & Information Systems

**Team Members:**

Volodymyr Saruta – Team Mentor

Nicholas Caporusso – Team Leader

Kevin Kilbourne

Jonathan Nation

Jacob Novin

**Table of contents**

[**1.0 - Introduction**](#_jeam0wdd2y8r) **3**

[**2.0 - Process Overview**](#_uqd4z3n11sq) **4**

[**3.0 - Requirements**](#_awo17mmy0ocw) **5**

[**3.1 Functional Requirements**](#_6lpfm1q82cu4) **5**

[**3.2 Performance Requirements**](#_m87uwwtpydrd) **10**

[**4.0 - Architecture and Implementation**](#_3udm9olemdu4) **13**

[**4.1 - Digital Backpack Overview**](#_mgbgu6q7foe8) **13**

[**4.2 - Student/Teacher UI**](#_w8i914rs96rk) **15**

[**4.3 - Resource Searching and Downloading Algorithm**](#_so8hn1yl7zy5) **17**

[**4.4 - Resource Rating System**](#_ij8wzj89weae) **17**

[**4.5 - Student Flagging System**](#_xjry6ti1mvaw) **18**

[**4.6 - Student Connectivity Heatmap**](#_bh60nv64n7kb) **19**

[**4.7 - Changes Over Time**](#_vua17oihckrf) **19**

[**5.0 - Testing**](#_f0vmdvak4hha) **21**

[**5.1 - Resource Searching and Downloading Algorithm**](#_eypwefwfjlq) **21**

[**5.2 - Resource Rating System**](#_copzmxqfr48x) **23**

[**5.3 - Student Connectivity Heatmap**](#_937pmz37ffv8) **24**

[**6.0 - Project Timeline**](#_mtrv8gnt01tm) **27**

[**6.1 - Gantt Charts**](#_tkez3rn4g6ir) **27**

[**7.0 - Future Work**](#_tlrqfi6jg6bl) **29**

[**7.1 - Fixation of Legacy Code**](#_dkwrouttoqis) **29**

[**7.2 - Profile Customization**](#_yra3rdgewy0w) **29**

[**7.3 - Gradebook Management System**](#_p7zsx7aln1jk) **30**

[**7.4 - Parent User Model**](#_dmyc9yktuhyr) **30**

[**8.0 - Conclusion**](#_xb82rl95i7eg) **32**

[**9.0 - Glossary**](#_aji1xousm1lo) **33**

[**10.0 - Appendix A: Development Environment and Toolchain**](#_6z5ew55n3j5e) **34**

# 1.0 - Introduction

From recent history in the field of education, the world has seen a continual shift towards the integration of technology into the process of education, from K-12 and beyond. This shift has been incredibly prominent during 2020, when countless schools throughout the world closed their doors and went to being 100% online resulting from COVID-19. This transition, however, has also left many people with incredibly limited access to their education due to the newfound coupling of the internet and schooling. In the United States today, there are over twelve million K-12-aged children who do not have adequate access to the internet in their homes. This means that these students are reliant upon alternative sources of internet, such as public libraries, cafes, or other options that offer a consistent connection. But those sources are not always reasonably available to each student. This disparity in students’ abilities to access the web and, therefore, educational content, is known as the homework gap.

Our team has been working with Dr. Vigil Hayes, assistant professor at NAU and director of CANIS lab, to expand upon and improve The Digital Backpack, the first teacher/student communication platform for academia that is targeted for online use. The Digital Backpack is a web application that will allow a fluid transition between online and offline learning. This is enabled with the use of the system’s main components, allowing for more utility to be provided for both students and teachers.

# 2.0 - Process Overview

When initially starting this project, the team had laid out specific roles and responsibilities designated to each of the team’s members. These can be found within Team Logout’s “Team Standards” document. This was done to make the development process much more effortless as the divided work by the team was able to be completed more controlled and efficiently. The team roles are divided as follows: Nicholas Caporusso as the Team Lead, Kevin Kilbourne as the Architect, Jake Novin as the Release Manager, and Jonathan Nation as the Team Recorder. Despite these individual roles, each team member is also given the role of Coder.

Communication between team members, mentor, and client was of the utmost importance throughout this process. This led to the team creating a Discord server, acting as a project workspace where everyone can remain on the same page in terms of development. This Discord server was divided up into different chat rooms, designated for specific tasks, messages, or documents. This server also allowed direct access to Team Mentor, Volodymyr Saruta. This allowed for either communication through messages, voice channels, or sharing screens possible for the whole team. The majority of the Team Mentor meetings were conducted through this Discord server once a week on Tuesdays. In terms of client communication, this was done mostly through Zoom, letting vital project information be presented and discussed between team members and Dr. Morgan Vigil-Hayes.

In order to make the programming workflow the most streamlined, the team produced a GitHub repository. This repository established a way for the team members to share code with one another and contain the entire project’s codebase, in one place. Through GitHub’s extensive tools and features, numerous ways to promote project modularity and documentation were offered. This made the team sure that all project components were in the necessary places and running smoothly.

# 3.0 - Requirements

This section describes the key requirements necessary to significantly improve upon the Digital Backpack. These requirements are broken down into functional, performance, and environmental requirements. The domain-level requirements of this project establish the overall goals of the Digital Backpack. These are listed as follows:

* The Digital Backpack will aid remote education.
* The Digital Backpack will provide tools that help support seamless transitions between online and offline functionality.
* The Digital Backpack will support two types of users: students and teachers.
* The Digital Backpack will be highly accessible in both design and device requirements in order to accommodate a wide range of users.

## 3.1 Functional Requirements

The functional requirements describe the features of the Digital Backpack system and its requirements for the user.

General Functional Requirements

F.R. 1: The Digital Backpack will have a family-friendly, eye-appealing User Interface.

F.R.2: The Digital Backpack will be able to grab student/teacher information from the central database for Django web pages.

F.R.3: For the searching algorithm, The Digital Backpack will query and download websites for use as a supplement by students using keyword search terms provided by teachers.

Student User Functional Requirements

F.R.4: For students, The Digital Backpack will view their own internet connectivity.

F.R.5: For students, The Digital Backpack will be able to rate websites found by the web resource searching algorithm.

Teacher User Functional Requirements

F.R.6: For teachers, The Digital Backpack will allow them to provide keyword tags when the algorithm is searching Google.

F.R.7: For teachers, The Digital Backpack will have the ability to flag students.

F.R.8: For teachers, The Digital Backpack will be able to view individual students' internet connectivity.

F.R.1: The Digital Backpack will have a family-friendly, eye-appealing User Interface.

When a user wishes to utilize the Digital Backpack, we want them to be greeted by a family-friendly UI which is also very easy to maneuver through and be able to find where you want to go. In addition, we wanted to have each of our features very clear in where they can be easily identified because of our very large age range, which also has a range of knowledge and experience of using technology.

F.R.2: The Digital Backpack will be able to grab student/teacher information from the central database for Django web pages.

Throughout this project, the Digital Backpack is connected to the central database in multiple ways. From the teacher’s perspective, we want to grab the student’s name if they signed in, or their email address and display it on the class roster area and an area for what are the active classes that teacher has been teaching. We also wanted to have the database to store assignments that will be created by teachers for students to work on. Once the teachers are done creating assignments, they will be able to see the assignment from the database on the active assignment box. To the student’s perspective, the central database will be mainly used for grabbing their student information when viewing their own connection.

F.R.3: For the searching algorithm, The Digital Backpack will query and download websites for use as a supplement by students using keyword search terms provided by teachers

When the searching algorithm is searching google for other websites to use for supplement. The way for this to grab the websites is by using keyword searches that the teachers provide and go to Google.com to use those keywords. The searching algorithm will be able to grab the website names and store them into a structure. Teachers will be able to remove websites from the list before continuing to download the resources. Then the searching algorithm will take the queue into another system that will download the websites in the queue as pdfs that will be used for the student.

F.R.4: For students, The Digital Backpack will allow for viewing their own Internet connectivity.

As the students connect to and utilize the Digital Backpack, the web application detects the user’s connectivity timeframe of when they accessed the web server. This is carried out with help from Python’s very own datetime library. The date and time collected is then input into a .csv file for that student and it is from this .csv file that the heatmap is generated. Students are able to access their own online connectivity heatmap through the “View My Connections” button found at the bottom of their homepage.

The Online Connectivity Heatmap’s components includes the following:

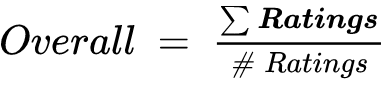
* X-Axis: This consists of the days of the week. Each column represents a day of the week, with Monday being the leftmost day and Sunday being rightmost.
* Y-Axis: This consists of the hours in a day. Each row represents the hours in a day, ranging from 12:00 A.M, located at the bottom, and 11:59 P.M., located at the top.
* Color bar: The purpose of the color bar is to display the range in values from the heatmap in a gradient. The minimum value is immutable as it depicts that the student has logged in zero times. The maximum value is the highest value found in the heatmap and can grow as more successful logins are collected. For example, if the most amount of times a student has logged into the Digital Backpack within one hour is seven times, the colorbar will show a gradient ranging from zero to seven.

F.R.5: For students, The Digital Backpack will be able to rate websites found by the algorithm.

When students take advantage of the extra, supplementary resources a teacher can provide for use on assignments, it is crucial that students are able to provide feedback on if the resource is either advantageous or unusable. The data collected consists of ratings gathered from students in the form of “stars” ranging from 1-5. The usefulness of the website resources can then be measured and provided to instructors for any revisions needed to be made for the curriculum.

The values for the ratings will be stored as such:

* The equation we will be calculating for the overall rating will be as follows:



* + Overall: Standing overall rating of the website
  + 𝝨 Ratings: The total sum of all ratings ranging from 1 - 5
  + #Ratings: The total number of reviews given for the website

F.R.6: For teachers, The Digital Backpack will allow them to provide keyword tags for when the algorithm is searching Google.

Used for when teachers want to give extra resources to students in which they might want to utilize to work on the assignments. They will be able to use a Google Searching Algorithm that will grab the websites that they students may want to use for their assistant on the assignment. Teachers will use keywords that will be used in the search bar of Google to find other websites that can help students on their assignments. This is how we will be taking the keywords that the teachers put into the keyword text box and search on google to grab the websites.

F.R.7: For teachers, The Digital Backpack will have the ability to flag students.

With the help of the time series view of student’s connectivity periods, it becomes easy for teachers to notice when certain students begin to show lack of motivation towards school work. As this can potentially evolve into an issue, it becomes imperative a teacher takes swift action. This is where our “flagging” system comes into effect as it allows teachers to have a delimited area within their workspace for displaying students in need of extra assistance.

F.R.8: For teachers, The Digital Backpack will allow individual students' internet connectivity to be viewed.

For the time collection and heatmap generation process, see F.R.4. Considering teachers are working alongside students with poor internet connection, it is convenient for the teachers and students to be on the same page in regards to when a student is capable of carrying out assignments. In order to aid this relationship, the application’s teacher counterpart of recording connection activity comes into effect. The data collected will be displayed inside of the individual student’s account page, accessible from the teacher’s workspace in the form of a heatmap. Here, specific students are able to be individually looked at and addressed. This allows patterns of when a student *can* use the Internet to easily be seen by their instructor.

The Online Connectivity Heatmap’s components includes the following:

* X-Axis: This consists of the days of the week. Each column represents a day of the week, with Monday being the leftmost day and Sunday being rightmost.
* Y-Axis: This consists of the hours in a day. Each row represents the hours in a day, ranging from 12:00 A.M, located at the bottom, and 11:59 P.M., located at the top.
* Color bar: The purpose of the color bar is to display the range in values from the heatmap in a gradient. The minimum value is immutable as it depicts that the student has logged in zero times. The maximum value is the highest value found in the heatmap and can grow as more successful logins are collected. For example, if the most amount of times a student has logged into the Digital Backpack within one hour is seven times, the colorbar will show a gradient ranging from zero to seven.

## 3.2 Performance Requirements

The performance requirements describe the metrics in which the application is expected to perform. The following subsections will describe the measurable standards for the functional requirements outlined in section 2.1.

P.R.1: The Digital Backpack will have a family-friendly, appealing User Interface.

This was made with the intent of having the users, teachers or students, be able to easily access and navigate through the Digital Backpack system.

**Target Metric:** We have made the UI look very simple yet modern to where the users can get a welcoming feel to our system and have everything laid out for them to where they don’t need to go searching for one thing.

P.R.2: The Digital Backpack will be able to grab student/teacher information from the central database for Django web pages.

In order to see the student’s information or upload an assignment into the database, the Digital Backpack must be able to have a database that will be able to hold multiple tables and information that can be accessed by the Django Web Pages that call for specific information/data.

**Target Metric:** The database must be able to hold multiple types of tables for different kinds of information like classes, assignments, student information, etc.

P.R.3: For the searching algorithm, The Digital Backpack will query and download websites for use as a supplement by students using keyword search terms provided by teachers

When a homework assignment has been assigned, the algorithm will then start a search to Google for relevant websites pertaining to that assignment. It will take in the class given keyword tags given by the teacher. Once 10 links of websites have been found, the teacher will be provided a list of the resources for them to review, and they will be given the option of deleting resources not related to their teachings, and be able to create assignments and the system will then download the resources into an object with the assignment title, description, due date and other files given.

**Target Metric:** We will use a searching algorithm to find websites and store the website URLs into a queue that can download the websites into PDFs and provide them as a PDF viewer. When students open the assignment on the student side of Digital Backpack, the assignment will download all of the resources that were found by the searching algorithm.

P.R.4: For students, The Digital Backpack will view their own internet connectivity.

This gives the student their own personalized view of when they are generally online and offline. It can generally be used to supplement a student with scheduling meetings, planning when to do school assignments or be in class, etc.

**Target Metric:** For this, students will be able to see accurate times of when they can have access to the internet again. This will allow students to know when the next time they will have internet access.

P.R.5: For students, The Digital Backpack will be able to rate websites found by the algorithm.

This allows for a way of constant and clear communication between the student and teacher when they are reviewing and looking at the websites found by the algorithm and lets them know which are good and which are bad.

**Target Metric:** This will allow students to rate the websites that the teachers provide using the google search algorithms. If the students find the websites provided helpful then they will rate the website out five stars, and when the student has access to the internet. The DigiLearn Web Application will send the ratings to the teacher and self so that the next time students want to use the websites given by the google search algorithm.

P.R.6: For teachers, The Digital Backpack will allow them to provide keyword tags when the algorithm is searching Google.

When a teacher is creating an assignment for their students, they will be required to give at least 3 and no more than 32 keywords that describe the assignment and will help the algorithm guide its way to proper websites to download.

**Target Metric:** We will provide a text box for the teachers to use and input keywords inside. The keywords that the teacher provides will then be used in the search box on google to find the top results of the search.

P.R.7: For teachers, The Digital Backpack will provide the ability to flag students.

This is how the teacher will be able to keep track of specific students that aren’t doing well in class, missing class, or any other scenario. Teachers will be able to head into the specific student’s information page that displays their email, name, connectivity and flagging status. Teachers will be able to click one button to change the student’s flag to remind them to change what they are doing.

**Target Metric:** For this, if the students are not turning in the assignments, teachers would want to flag the students so the teachers can remind the students to work on their homework next time they are in class.

P.R.8: For teachers, The Digital Backpack will be able to view individual students' internet connectivity.

In order for teachers to keep track of their students, in regards to assigned tasks, when outside of school, they are provided with graphs containing time information on when a student connects to the application.

**Target Metric:** The time it takes Python’s datetime library to recognize a user is online, write the collected time to the student’s .csv file, and update the connectivity heatmap on the webpage.

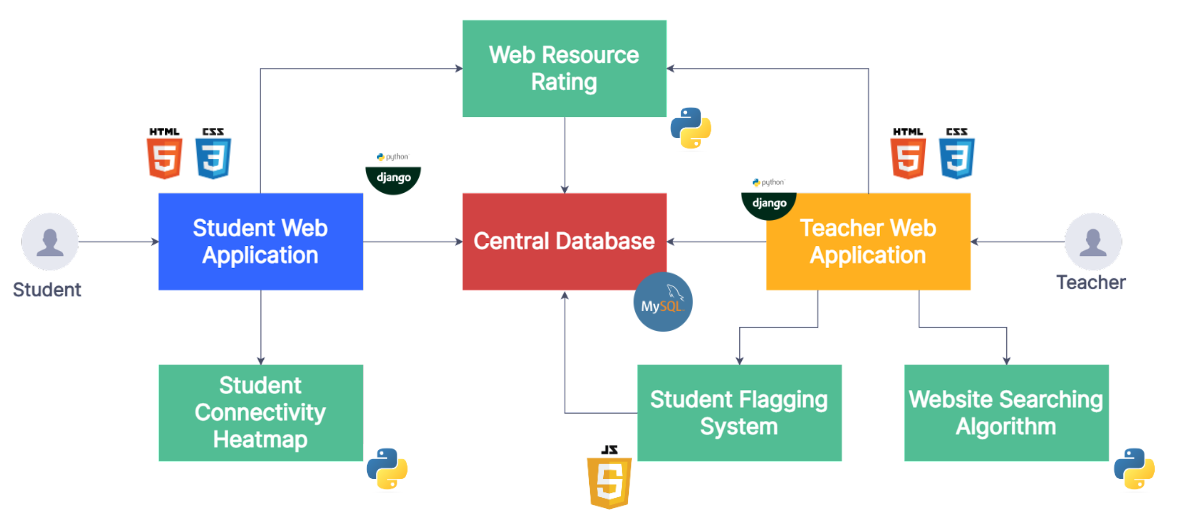
# 4.0 - Architecture and Implementation

This section will cover the entirety of the Digital Backpacks overall architecture, as well describing each of the features we have implemented and how they connected with each other.

## 4.1 - Digital Backpack Overview

The figure below, (Figure 4.1) is the master diagram of the Digital Backpack Team Logout has produced over the last two semesters. In this diagram, there are symbols on the corners of some of the boxes which represent the programming tools used to make our version of the Digital Backpack. For the Digital Backpack, Team Logout used Python as the main programming language. With this, we used the Django web framework to hold our progressive web app and to interact with the features. For the database, we used MySQL to hold the information that students and teachers provide. Lastly we used HTML, CSS, and JavaScript for the web pages that the teachers and students will use in the progressive web application.

In the diagram below (Figure 4.1), we have our boxes in different colors along with the symbols of their programming tool. Green boxes represent the different main features the team has created for the Digital Backpack. The red box represents the main database that is used for our features or pages that require the database’s information. Blue and Yellow boxes represent the student’s and teacher’s web pages respectively. This is the main way that each feature and the database will interact with the users. On the far left and right of the diagram we also have the students and teachers users nodes, respectively, to show what the prospective users will have access to in the master diagram.

*Figure 4.1: Digital Backpack Version 1.5 Master Diagram*

To begin the deep dive into the Digital Backpack, we have broken down each of the features we have implemented and how they would interact with each other. Both teacher and student web pages were created using HTML and CSS with the use of Django to connect everything that was needed for the page. The main feature of these web pages is to provide a student or teacher-friendly user interface that will help users understand what/where things are. The teacher web application will be able to access the Student Flagging System, Resource Rating System, Website Searching Algorithm features and the central database. The student web application will be able to access the Student Connectivity Heatmap System, Resource Rating, and the central database as well.

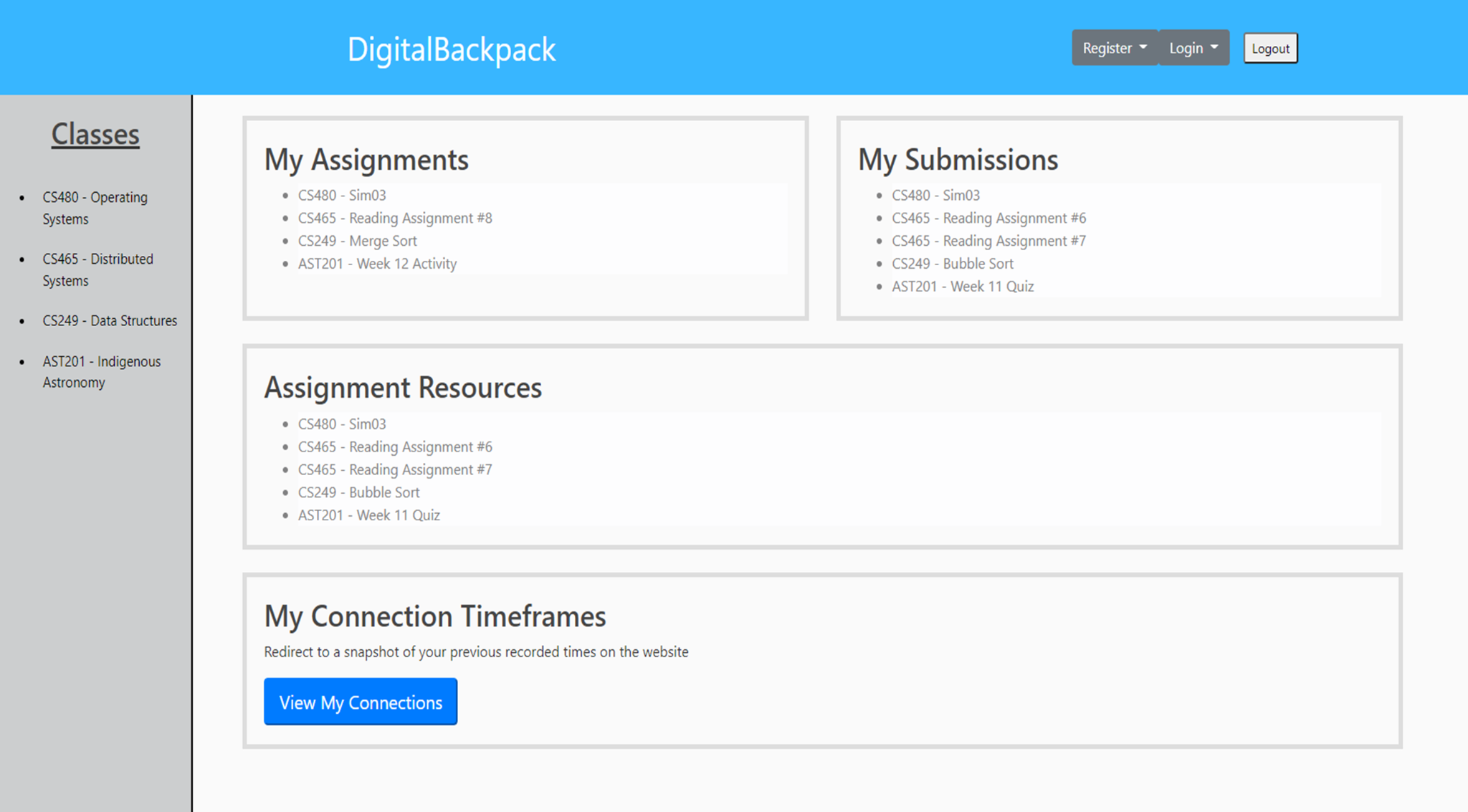
First for the teacher’s web application is the Student Flagging System, where this feature will allow teachers to flag students. This component was made using JavaScript and the main responsibility of this is to access the central database and change a specific student’s flagging status to let both the student and teachers know that they were flagged. Next is the Website Searching Algorithm where teachers will be able to fetch resources and create assignments. This component was programmed in Python with help of the pdfkit package that changes the links into pdfs for anyone to use. The main feature of this is to go on google and search using the keywords the teachers provide. Once the teacher is ready to create an assignment, this communicates with the central database, downloading the resources and saving assignment details (designated class, title, description, due date, and extraneous files). A feature that is also used by the teacher is the Resource Rating, this helps the teacher on whether the resource they will be using helps the student’s learning. This component was created using Python and Django as communication with the central database is easily performed. The main idea on the teacher's side is to let the teacher know if this resource actually helped the student’s completion of the assignment or not. Lastly is the central database for the teacher’s perspective. The main key feature of the database being used here is to show all the information needed. This could be seeing the list of students that are currently enrolled into their class, seeing the student’s information when the teacher would view them and creating an assignment as they will be able to choose which class from the list of classes this teacher teaches.

In the case of the student’s web application, the first tool here is the Online Connectivity Heatmap system. This component was created using Python and also uses Python’s very own datetime package to help grab the time and date. The main feature of this is to calculate when students are using the Digital Backpack, indicated by a successful connection to the student homepage. This communicates with the Django server and stores the date and time data into the student’s specified folder where it can be accessed by both users. The next component that the student can use is the Resource Rating System. This was mainly programmed in Python that can also connect to the database. This main feature is to accept ratings from 1 to 5 stars from students and store the ratings that connect to the website the resource used was from. This communicates with the database by storing the data into it for the teachers to review it when it comes time to creating assignments.

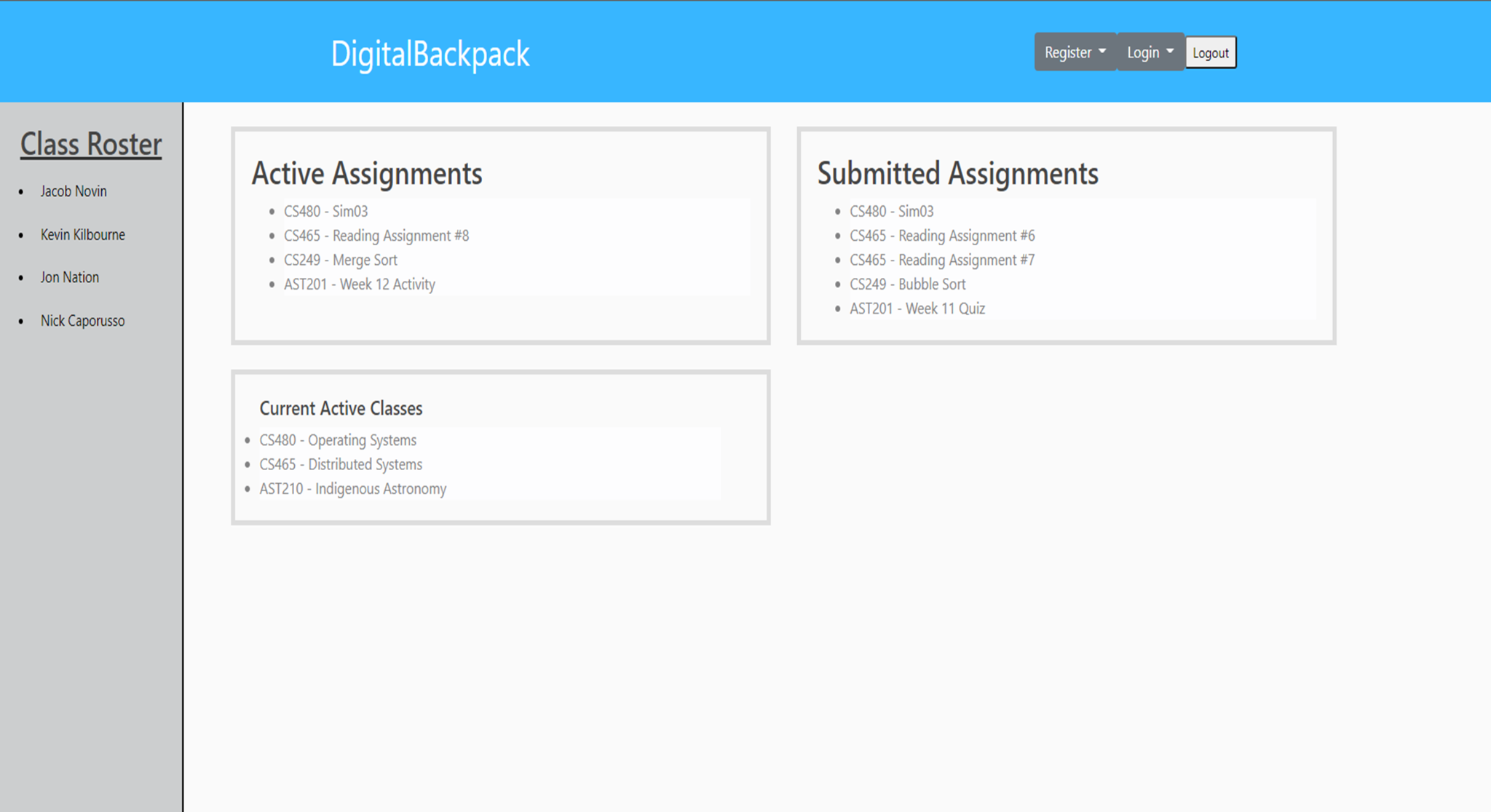
## 4.2 - Student/Teacher UI

This is the main source of interaction with our system for each of the Student and Teacher users. Each user will have to login to our system, which will automatically redirect them to their respective homepages.

For Students, they will mainly see their current assignments, assignment submissions, their own personal connectivity heatmap, as well as the classes they are enrolled in.

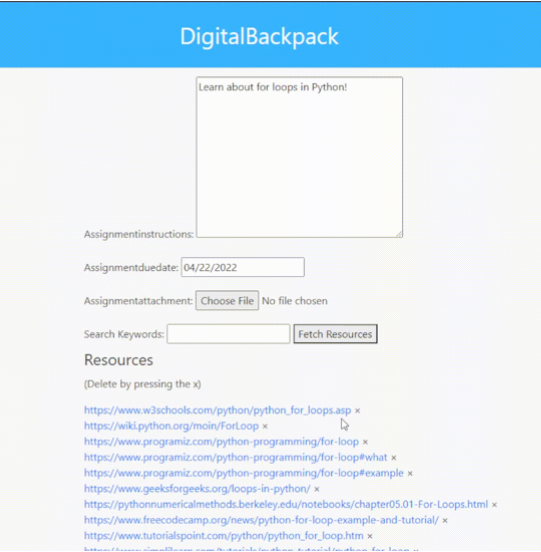


For Teachers, they will be able to create assignments for students in their respective classes, which will also include the use of our Resource Searching and Downloading Algorithm, they can also see each of their students account information, such as name, email, flag status, and connectivity heatmap.



## 4.3 - Resource Searching and Downloading Algorithm

This feature is how our system will be gathering additional outside resources for students to utilize while they are not connected to the internet. When the teacher creates an assignment, the last thing they will be asked to do is provide a list of at least 3 keywords to provide accurate results that the teacher can look at, as well as being able to remove certain links or do a completely new search for new links, before the system will automatically start the download process of each of the resources with the click of a button.



## 4.4 - Resource Rating System

This is the main source of communication between the student and teacher based on how well the resources were to the student and their further understanding of the assignment. When a student submits an assignment, our system will automatically redirect them to the ratings pages and prompt them to give each of the resources a rating from one to five stars. Once they submit the ratings, the system will update each of the ones that received a rating in our database for the teacher to see which are good and bad moving forward in the year.



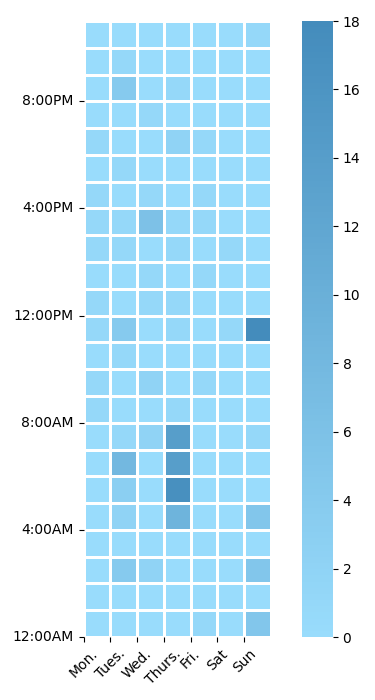
## 4.5 - Student Flagging System

The primary purpose of the Flagging System is to allow the teacher to clearly identify students in their class roster of ones that they might need to keep a close eye on or track underperforming students. If the teacher sees fit for a certain scenario, they can also send an email to the students' parents/guardians of their concerns or to get a deeper understanding of what might be going on in why they are underperforming.



## 4.6 - Student Connectivity Heatmap

The students will be able to view their own connectivity on their trends of the times they have reliable access to the internet. The teacher is also able to view this in case they want to schedule a meeting with them for any reason they see fit.



## 4.7 - Changes Over Time

From what we planned in our Software Design Document and now in the Final “As-Built” Report, there are a few changes that we did from what we first envisioned. One example would be the Resource Searching and Downloading Algorithm. The original would use three different functions, one to grab websites, check the websites, and download the websites. When we created this feature in the end, we removed the checking website function as we decided that the teachers should be checking the web pages themselves and removing which website that they do not want to be used by students.

Another thing that was changed from our first design was the User Interface. In this, we changed how the UI would look and make it more clean and family-friendly. Our old design looked very rugged and not appealing to both us and the client. When the client explained how they would like the UI to look better, we took the time to create a user interface that looks appealing for anyone who comes to the Digital Backpack. We received opinions from our mentor as well as other people's work from Google to gain inspiration for how to make our UI look much more modern and less rugged.

After these changes, the Digital Backpack became more utilized and efficient. But what do outside people of the Digital Backpack project think about the product we created? Is what we created able to get the message across on what they need to do in certain situations? Does the Digital Backpack satisfy them for them to use in the future?

# 5.0 - Testing

For over the past few months, Team Logout has been working on the Digital Backpack and its features to help with the current problem with the learning gap. The features that we created for the Digital Backpack involve the new Student/Teacher User Interface, Resource Searching and Downloading Algorithm, Resource Rating System, Student Flagging system and the Student Connectivity Heatmap. Once we have created these features into the Digital Backpack, we began the testing phase where our participants will be testing our features and giving us feedback on what they liked or disliked about the features they used.

## 5.1 - Resource Searching and Downloading Algorithm

**5.1.1 - Unit Testing**

Unit testing the Resource Searching and Downloading Algorithm, we would be checking if the code that we were creating would go on Google with the provided keywords. One of the things that we tested with this feature would be testing that resources can be fetched. We used different sets of keywords on the searching algorithm so that we can get different kinds of resources. Ranging from 1 to 32, we checked if the searching algorithm will handle the different lengths of keywords to use for resources. Everytime the searching algorithm would return the 10 possible links, we checked them to see if the links were accurate to what the keywords were describing.

**5.1.2 - Integration Testing**

Testing the Resource Searching and Downloading Algorithm, we incorporated the rest of creating assignments to the feature. Creating assignments involves putting in the title of the assignment, the description, the due date, and their own files that would also help students with the assignment. Here is where the Resource Searching and Downloading Algorithm is utilized and grabs resources that are related to the keywords provided. After checking the websites and deciding which resource will remain for the assignment, pressing the create button would start the process to create the assignment and download the resources into our database. Once completed, the page will head back to the teacher’s page and show the assignment that was just added.

**5.1.3 - Usability Testing**

Our last test for the Resource Searching and Downloading Algorithm is the usability test. We asked our testers to create assignments using the feature. We asked if there was anything that the testers disliked about the process in either creating assignments or finding resources using the searching algorithm. The feedback that we got from the testers using this feature were all positive. The feedback that mostly stands out explains how easy the system was to create assignments. The testers also liked how the searching algorithm would display all of the links that it grabbed and let them look into the links so they can double check that the link is accurate to what they want the assignment to be. One thing that the testers did talk about in the feedback was the process of creating assignments once they are fully satisfied with the assignment they created. They were hoping that the downloading process would be faster, but were okay with the current speed. We did explain that we would look into ways to make it faster and more efficient.

**5.1.4 - Testing Results**

The results of our testing concluded with not a lot of changes from what we have created originally. The testers were very satisfied with the process of creating assignments and the process of the resourcing fetching. After the testing phase, we came up with the idea to make the downloading process of the resources for the assignment easier using multi-threading. This idea would make the process of creating assignments faster than what we currently have, but after a few days to a week of working on implementing this, we were not able to get the multi-threading working. We are happy with the current process of creating assignments and could possibly have the process go faster for the next Capstone Team that will pick this project up.

## 5.2 - Resource Rating System

**5.2.1 - Unit Testing**

Unit testing the Resource Rating System consisted of primarily ensuring that data provided to the system was being communicated accurately and that updates were being correctly made to the database values. This process was fairly simple, as there are a limited number of logical paths to take; new entries would simply assume the rating given and a rating count of one, and pre-existing entries were updated using the formula mentioned in Functional Requirement 5.

**5.2.2 - Integration Testing**

For integration testing the Resource Rating System, there were only two major plugin points that were of reasonable concern. These were the correct linking of assignment items from the student’s submission to prompt them for ratings, and the graceful handling of incorrect data being sent in. For linking purposes, we tested utilizing pre-created links that utilize a GET request to pass data along. This will allow for an easy plugin to a system with an integrated submission system, which we did not have at our disposal, as that is not in the purview of our additions to the system. For the graceful handling of incorrect data, form cleansing was added to ensure that website-rating pairs were being sent exclusively, or they would simply be ignored.

**5.2.3 - Usability Testing**

Our last test for the Resource Rating System is the usability test. We asked our testers to rate any number of hypothetical resources using the ratings page via links to simulate how students would be redirected after submission. We asked if there was anything that the testers found difficult to use with the ratings, or if there were any major areas of improvement. Feedback on this portion was almost exclusively positive. Providing ratings was simple and easy to do, and users did not have issues moving past certain sites that they were ignoring (to simulate not using a certain resource). The only critique brought up was that the actual appearance of the page could use some work, and we have noted that as a possibility for the future.

**5.2.4 - Testing Results**

The results of this testing process closed out with virtually no changes to the system as it is now. The functionality appears relatively bulletproof at the moment, and test users expressed that the interface was very simple, making it quite easy to use. The one critique brought, the changes to appearance, mostly centered around implementing star icons to utilize, as well as spreading items out a bit more. These would help round out the page a bit more, but ended up not fitting into our development schedule by the end of the semester.

## 5.3 - Student Connectivity Heatmap

**5.3.1 - Unit Testing**

In order for the heatmap to be generated correctly, it is reliant on the correctness of the incoming time data. For this module, the date and time information coming in is derived from Python’s own ‘datetime’ module.

The process of running this module through unit testing started off by making sure the tests are conducted in the same environment as it would when the product reaches the hands of actual users. This means being within views.py of the Digital Backpack Django server. At the core of this unit testing is the utilization of the ‘freezegun’ package. With this package, all calls to datetime.datetime.now(), datetime.datetime.utcnow(), datetime.date.today(), time.time(), time.localtime(), time.gmtime(), and time.strftime() will return the time chosen to be frozen.

For the testing in the heatmap module, we only need to interact with datetime.datetime.now(). Confirming the base case here is to not alter datetime.datetime.now()’s value and seeing if it is in fact equivalent to the present day’s date. For further analysis, “freeze\_time”, from the freezegun package, gives us the ability to change the time value that datetime.datetime.now() records, allowing us to compare this newly set value against a concrete date. For example, after calling “@freeze\_time(“2000-04-18)” the value that datetime.datetime.now() holds is now equivalent to this frozen date specified. Now, using Python’s assert statement, we can tell whether this new datetime.datetime.now() is identical to the set datetime of April 18, 2000 by identifying an AssertionError being raised or not.

Another significant point to cover with unit testing is the process of determining if a .csv file has been written to correctly. Achieving this is, also, done through the utilization of the freezegun package. This testing has been done in tandem with the tests for accurate time collections as the write operations are performed within the same function. With that being said, we check to see if the correct datetime is being collected, output this newly picked time to a test .csv file of the same format found in the Digital Backpack system, and look to see if the change was in the correct position or not.

**5.3.2 - Integration Testing**

A crucial component for the view of the individual student’s connection timeframe heatmap is making sure correct write operations are performed on the .csv file Matplotlib reads from in order to generate the corresponding visualization. Upon a student visiting the student homepage, the method student\_page() within views.py is called. In the student\_page() view, the utils.py function makeHeatmap() is called with a student’s username being passed in. A part of the responsibility of the makeHeatmap() function is to take in the current date and time the page was visited by that individual, convert the .csv file into a 2D array, and write the collected time information in the correct position within the array. This array is then converted back into a file of .csv format and saved onto the server. It is this updating process’ execution that is imperative to the system.

Also in this method is where the read portion of the connection heatmap takes place and as this directly coincides with the student\_page() function, both being seen as equally important. The .csv file’s location needs to be immutable, as the specified file path will stay static.

**5.3.3 - Usability Testing**

From within the student’s homepage, students have the ability to access their own heatmap visualization consisting of previous times that they have been connected to the webpage. The access point for the timeframe can be found in two places. One is found from the Teacher’s class roster in the teacher homepage. Clicking on an individual student will redirect the teacher to the corresponding student’s account page. The other location is from the student homepage. There, students can find a button near the bottom of the page and can easily be maneuvered to by student users. Upon clicking the button, the student is then redirected to the their view\_myself page that displays this heatmap, alongside some personal information. Considering this action appears to be somewhat inconsequential, the only significant testing point is determining if the requests, deriving from the button, make their way through the server correctly, and receive the correct HTTP status code (200). This test can be carried out with the use of a 3rd-party package called ‘requests’. After importing the package, we are able to set a specific URL to be visited. For this case, we will be sending requests to the webpage found from the following path: “../student/view\_myself/” and “../teacher/view\_student/”. Using this set URL variable, we can loop for a set amount of requests to be sent to the URL, determine which HTTP status code is found, and record the frequency of said returned codes.

**5.3.4 - Testing Results**

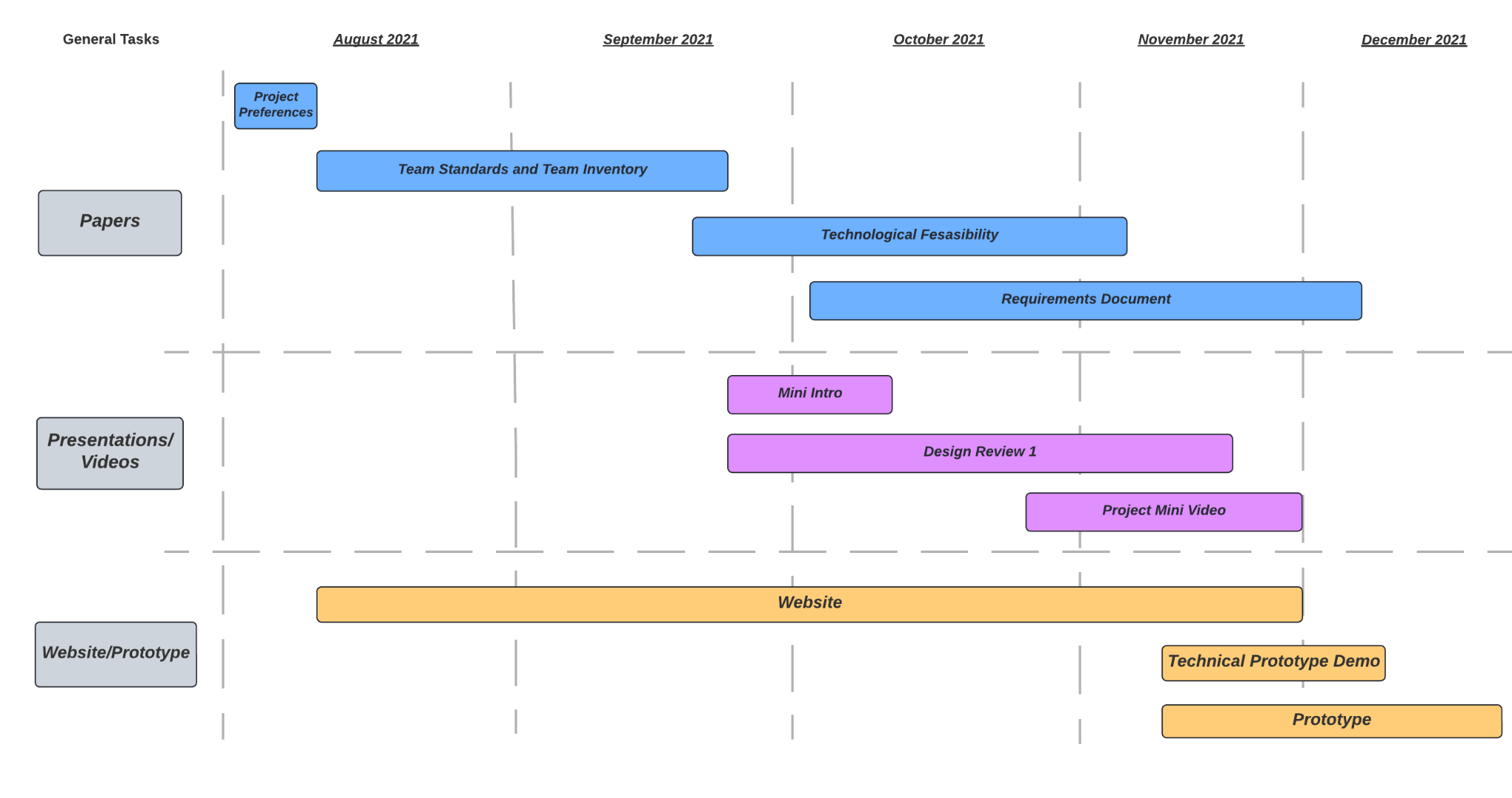
Testing of the student connectivity heatmap yielded a number of changes throughout the development process, primarily based on the scope of the tool and the visual representation of connectivity trends. In our final round of testing, testers found no issues with the functionality of the heatmap system, and after the alterations made along the way, our test users were able to understand the meaning of it with ease. Of the feedback given, most was based around formatting the heatmap representation closer to a ‘weekly calendar’ style of representing time periods. This is still a possibility later down the line, but was unable to be fit into our development schedule.

# 6.0 - Project Timeline

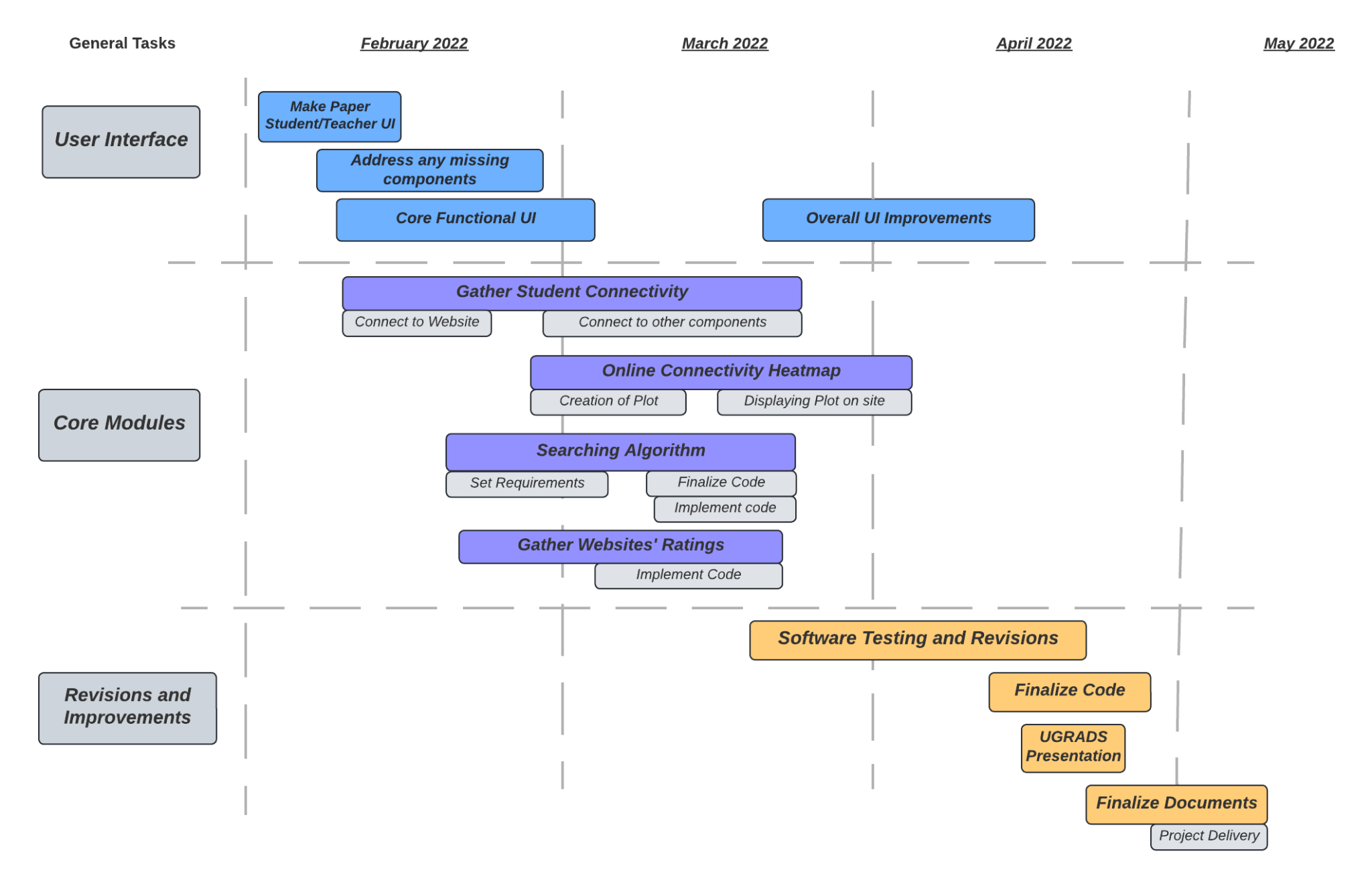
Throughout this year of Capstone, we have provided our timeline of what we have accomplished, how much time each took, and what exactly we had to do. We have broken it up into two separate Gantt charts for each Fall and Spring respectively.

## 6.1 - Gantt Charts

*Fall 2021 Semester Schedule*



*Spring 2022 Semester Schedule*



## 

# 7.0 - Future Work

As we conclude with the Digital Backpack for the 2021-2022 year, we are excited to complete our minimal viable product for what our client, Dr. Morgan Vigil-Hayes, has asked us to make. But as we finish our version of the Digital Backpack, we believe that there could be improvements to this project that will benefit the overall goal of assisting students with little to no internet access. The ideas listed below are what Team Logout offers to assist with an outline to what the next minimal viable product will be for the next that picks up the Digital Backpack.

## 7.1 - Fixation of Legacy Code

One of the biggest challenges Team Logout had when creating our product was the deterioration of the legacy code from the last capstone team that worked on the Digital Backpack. When we tried to work on fixing the problems in the old server, the Digital Backpack was not able to run when we fixed the problems we found. Our solution to continuing this project was to make an entire server that will hold the features of our MVP. Resulting from the deterioration of the legacy code, Team Logout had to drop two of our initial features that we planned on working on in our Requirements doc. Those two frameworks are, having a homework submission upload queue and having the Digital Backpack being able to functionally work on Chrome Operating Systems. Team Logout has agreed that fixing the legacy code and plugging in our features into the old server would help make the Digital Backpack whole again. This will also help other future teams taking the Digital Backpack as they will not have to work with multiple servers that have different sets of features to work with.

## 7.2 - Profile Customization

Another big thing that Team Logout has thought of was to give the Digital Backpack more characteristics. To do this, we thought that the Digital Backpack should have a way for teachers and students to personalize their profiles. This would give students and teachers creative control over their accounts and show what they like. Change their profile picture to a recent trip they went to, a television character they appreciate, or and other images they would like to use. Another part of the profile customization would be changing their emails and passwords. This would allow them to change them in case their old one is not up to date to their other accounts they used outside of Digital Backpack. In the same area about updating emails or passwords would be having basic information to the account. Giving students and teachers the ability to put more information on their account would help students get a better understanding of what their teachers are like or vice versa. There are more that would benefit from profile customization, but the next team that picks up the Digital Backpack will have access to what can be used as a profile customization feature.

## 7.3 - Gradebook Management System

The overall utility of the Digital Backpack at its current state is to create assignments, rate the resources used for the assignments, flag students, show online connectivity, and upload student submissions from the Digital Backpack 1.0. There is not a way for teachers to put student’s grades into the Digital Backpack. The teachers would have to write down the grades and calculate them or use a separate system like Microsoft Excel that will calculate the grades. To help teachers to not use multiple systems to grade students and look at submitted assignments, the nest iteration of Digital Backpack should include a grading system that will accept assignments and what their grade is. This will also be able to calculate the total grade the student has using given weight percentages and categories that the assignments are (E.x. Quizzes, Homework, Test, etc.). Team Logout believes that this will improve the overall utility of the Digital Backpack with this feature.

## 7.4 - Parent User Model

The last feature that Team Logout thought of as an improvement to the Digital Backpack would be to include parents into the database and project. Parents sometimes would want to know about their child’s grades or overall performance in their classes. Parents play a pivotal role in their child’s education as well as teachers. Giving parents the ability to also log into the Digital Backpack will help them understand how their children are doing and give them an idea where their child could get more help in subjects outside of school. This will also help teachers let parents know if their child is not doing well in classes and is flagged. Creating a communication platform that allows teachers to contact parents on the Digital Backpack will be of utmost importance to the next iteration of this project.

# 8.0 - Conclusion

As the internet has become more and more of a necessity for students of all levels of education the “homework gap” has grown considerably. Students without regular or reliable internet access are at a significant disadvantage compared to others that are able to access the internet consistently. The Digital Backpack project aims to aid students struggling with this issue by offering an oCDN for educational content. The Digital Backpack application will help give users the ability to help better connect them with their education, despite it being offline. This gap can start to be bridged with the introduction of our four main components: the web searching algorithm, website resource rating, student account options such as the flagging underperforming students, and a modular online connectivity heatmap. By providing these tools for users, it can help bring the students closer to others with consistent internet connections.

As the internet has become more and more of a necessity for students of all levels of education, the homework gap has grown considerably. Students without regular or reliable internet access are at a significant disadvantage compared to others that are able to access the internet consistently. The Digital Backpack project aims to aid students struggling with this issue by offering a new educational workspace which aims to provide the content for offline use. This will come in the form of a fresh, refined interface providing a clean workspace alongside a number of improvements to make schooling easier for students and teachers alike. The overall goal this project sets out to accomplish is helping the people who are already struggling. So, let’s close this digital learning gap for good!

# 9.0 - Glossary

This Glossary defines common terms found throughout this document.

**CANIS** - Community Aware Networks & Information Systems

**Homework Gap** - barriers students face when working on homework assignments without a reliable Internet source at home

**UI** - User Interface

**IDE** - Integrated Development Environment

**HTTP** - Hypertext Transfer Protocol

**URL** - Uniform Resource Locator

**oCDN** - opportunistic Content Delivery Network

**Digipack** - Digital Backpack

**MVP** - Minimum Viable Product

**HTML** - Hypertext Markup Language

**CSS** - Cascading Style Sheets

**JS** - JavaScript

# 10.0 - Appendix A: Development Environment and Toolchain

1. **Hardware**

For the hardware that we used for this project, we mainly were developing our product to work on Windows operating systems and Linux operating systems. Below is what the team’s hardware (In no particular order) we used to create, test, and produce our product in.

| **Operating Systems** | **Processor** | **Memory** |
| --- | --- | --- |
| Windows | i5 | 24GB |
| Windows | i7 | 8GB |
| Windows | Ryzen 5 | 16GB |
| Windows | i7 | 16GB |

There should not be any minimum hardware requirements for effective development besides having a decent machine, capable of maintaining a Django server.

1. **Toolchain**

We used a few software tools to assist us in completing this project. The main environment we used to program was Pycharm. The list of IDEs are as follows:

* Pycharm:
  + Purpose: Allowed us to connect directly to our GitHub repository
    - Pull all of the code to use on our local machine and to make any pushes to the repository that we make while editing.
* Database:
  + SQLite: Held all of our user models, classes, assignments, etc.

1. **Setup**

We do not have a specific working environment that was detrimental to the creation of this project. However, one IDE in specific was very helpful and that was Pycharm, which can be downloaded here <https://www.jetbrains.com/pycharm/>. This was helpful because it allows us to connect directly to the GitHub repository and pull code and push changes directly from there, making the process that much easier.

1. **Production Cycle**

This section will run you through how to install and run the server. The source code is located here: <https://github.com/jakenovin3/DigitalBackpack>

**Windows Installation**

1. SSH into your desired web server - Once this is done you can create your desired directories and drag the Django project files into this directory.
2. Create a Python virtual environment - Ensure you are in the directory with the Django project files. Create a python virtual environment by entering the command:

**> python -m venv [virtualenv name]**

1. Activate your virtual environment - Enter the virtual environment you just created by entering the following command:

**> [virtualenv name]\Scripts\activate.bat**

1. Install requirements - A requirements.txt file is included within the Django server files. Install the requirements in the virtual environment using the following command:

**> pip install -r requirements.txt**

1. Make migrations - The student database is handled using Django models. Make migrations to initialize the database with the following commands:

**> python manage.py makemigrations**

**> python manage.py migrate**

1. Run the server - At this point, the installation process for Django is complete. The final step is to run the server. This can be done by running the following command:

**> python manage.py runserver 127.0.0.1:8000**

**Linux Installation**

1. SSH into your desired web server - Once this is done you can create your desired directories and drag the Django project files into this directory.
2. Create a Python virtual environment - Ensure you are in the directory with the Django project files. Create a python virtual environment by entering the command:

**> python3 -m venv [virtualenv name]**

1. Enter your virtual environment - Enter the virtual environment you just created by entering the following command:

**> source [virtualenv name]/bin/activate**

1. Install requirements - A requirements.txt file is included within the Django server files. Install the requirements in the virtual environment using the following command:

**> pip3 install -r requirements.txt**

1. Make migrations - The student database is handled using Django models. Make migrations to initialize the database with the following commands:

**> python3 manage.py makemigrations**

**> python3 manage.py migrate**

1. Run the server - At this point, the installation process for Django is complete. The final step is to run the server. This can be done by running the following command:

**> python3 manage.py runserver 127.0.0.1:8000**

You can find more about how to run and maintenance the Digital Backpack through our User Manual.