NMTA

Interim Report

Christopher Esquibel, Jaxen Bujold, Tyler Del Prete, Zixi Li

# 1. Project Overview (10 points)

We plan on developing a website that allows users to create a visual representation of their college semester schedule in an aim to clarify the process of course registration. The main website features will consist of a calendar graphic, a search bar, and an LLM chat bot trained on New Mexico Tech course data. A user should be able to add a course using the search bar functionality, which will then be reflected on the calendar. The calendar will support overlapping for the class graphics, which will provide the user with an indication of any course conflicts. The chat bot will serve as an advisor. It will respond to any questions that the user may have about specific classes or instructors. It will be trained on scraped course data from the NMT catalog along with instructor data from a review site such as Rate My Professors. An example interaction would be the user asking about the software engineering course. They should be able to ask in plan text such as “tell me about software engineering”, and the chatbot should provide every instance of software engineering that is available during that semester along with the instructors attached. If more than one instructor exists, the chatbot should be able to suggest the most favorable one.

**Statement of Proposal:** *We propose to create a website that allows students at New Mexico Tech to create their class schedule with optional advice from an artificial intelligence advisor.*

## 1.1 Scope and Objectives

The scope of the data set used for the website database and chatbot training will be limited to courses and instructors within the domain of the computer science department. We want to limit the capabilities of the chatbot to the aforementioned functionality for the time being as the difficulty scaling to make it more functional may cause a disproportionate amount of time, which could add additional risk elements to the project. For the purposes of demonstration, we want to host the website on a local machine. There it can be accessed via localhost and reduce the chances of something going wrong before or during the demo.

## 1.2 Supplementary Requirements

As we decided to utilize an Agile methodology for the development of NMTA, we have little in the way of documentation and rely on comments within the code files in order to document our knowledge of the subsystems. We believe our system is reusable as the subsystem that reads in courses can be used in classes for various majors other than just Computer Science. We kept in mind that the system should be extensible as possible new majors are added, so we design the system in such a way that each major course selection page will be its own html page. During the creation of the course reader, it was kept in mind that the regex used should be able to handle formatting differences between the majors in the course catalog, making the course reading reliable for all majors provided. Also, the end goal of the application is for it to be used for many semesters to come, able to continue reading the most recent course catalogue information.

# 2. Customer Requirements (15 points)

Since we used an Agile-based approach, we used a User Story instead of a Use-Case Diagram:

| Epic | User Story | Acceptance Criteria |
| --- | --- | --- |
| As a **Student User, I need** to access the schedule builder calendar **so that** I can create a schedule. | As a Student User, I need to select my major on the homepage so that I can begin creating a schedule for that major. | Ensure the Student User is able to:   * Access the homepage * Navigate to the page of their desired major |
| As a Student User, I need to be able to add classes and remove classes from the calendar | Ensure the Student User is able to:   * Click on a course entry from the course log to add it to the calendar * Click on a course on the calendar to remove it |
| As a Student User, I need to be able to access the chat bot so that I can ask it questions about what classes to take | Ensure the Student User is able to:   * Click on the chat assistant box and have the chat bot appear * Input text and have the chat bot respond to the input |

# 3. Architectural Design (20 points)

We decided to build our website in the Python, HTML, and JavaScript languages. We utilized a flask server as a development environment. When requirement elicitation was undergone, only one alternative was considered. If Python web development was too difficult or time consuming to learn, we would instead use Java as we all have experience with Java web development. This was not needed as we all learned Python, specifically Flask, in a relatively short amount of time and continue to learn new features as we need them.

.

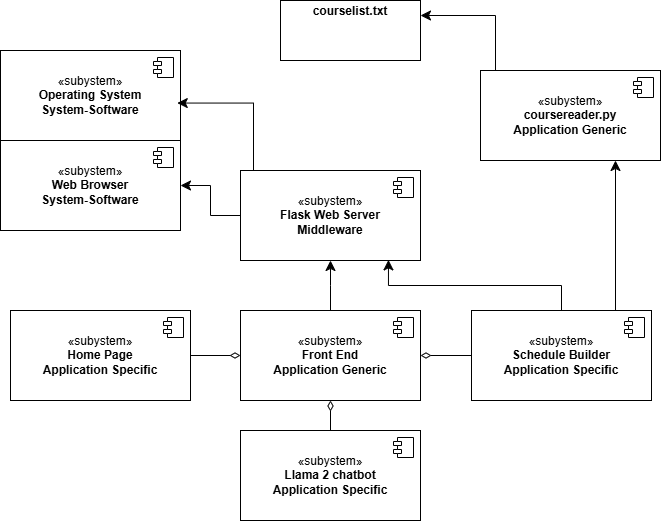
## 

## 

## 

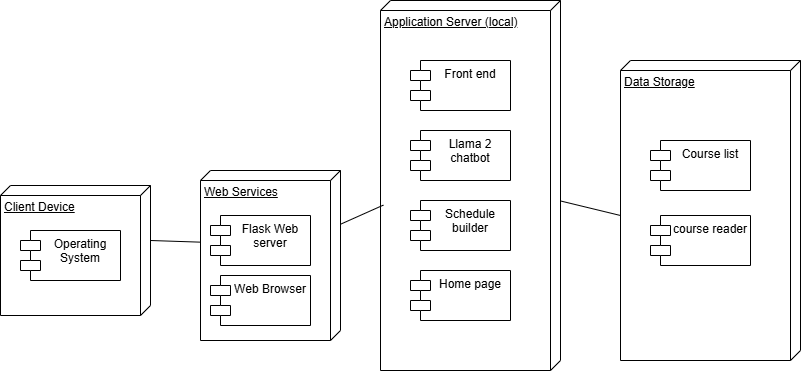
## 

## 3.1 Subsystem Architecture



At the application specific level, we have the home page, schedule builder page, and Llama 2 chat bot systems. These are application specific because they provide direct user support when utilizing the application. In the application generic layer, we have the front end and the course reader python program. These are at this level because they are the backbone of what the application specific layer relies on. Next, the middleware layer contains the flask web server. The flask server bridges the gap between the system-software level and the application levels.

## 3.2 Deployment Model



The program will first run locally on the client’s device. Then, since our application is web-based, the flask server will run to create a local server that displays the application using the client’s web browser. Once, in the application, the full functionality of the system deploys to the user. They can view the front end, including the chatbot, schedule builder page, and the home page. The application receives its information from the course list, which is filtered using the course reader.

# 4. Use Case Realization Design (20 points)

As we used an agile methodology, we opted for a Sprint Table:

| Task Name | Story Points | Priority | Sprint | Assignee(s) | Status |
| --- | --- | --- | --- | --- | --- |
| Scrape NMT course catalog | 8 | Medium | Sprint 1 (Feb 16 - Mar 1) | Jaxen, Chris | Blocked (Returned to Backlog) |
| Load scraped info into database | 2 | Medium | Sprint 1 (Feb 16 - Mar 1) | Jaxen, Chris | Blocked (Returned to Backlog) |
| Create front page | 3 | High | Sprint 1 (Feb 16 - Mar 1) | Tyler, Zixi | Done |
| Create schedule builder front end | 5 | High | Sprint 1 (Feb 16 - Mar 1) | Chris, Jaxen | Done |
| Create schedule builder back end | 8 | High | Sprint 2 (Mar 2 - Mar 15) | Zixi, Tyler | In Progress |
| Project Demo | 3 | High | Sprint 2 (Mar 2 - Mar 15) | Jaxen, Chris, Tyler, Zixi | In Progress |
| Scrape RMP | 8 | Medium | Sprint 3 (Mar 16 - Mar 29) | Jaxen, Chris | To Do |
| Upload scraped data into file | 2 | Medium | Sprint 3 (Mar 16 - Mar 29) | Jaxen, Chris | To Do |
| Feed LLM data | 8 | High | Sprint 4 (Mar 30 - Apr 12) | Zixi, Tyler | To Do |
| Integrate LLM | 13 | High | Sprint 5 (Apr 13 - Apr 26) | Jaxen, Chris, Tyler, Zixi | To Do |
| Product Finalization and demo | 3 | High | Sprint 5 (Apr 13 - End of semester) | Jaxen, Chris, Tyler, Zixi | To Do |

# 5. Subsystem Design (20 points)

The software is hosted on a Python Flask server, which serves as the foundation for multiple user-facing pages. Upon launching the application, users are directed to the homepage, where they can navigate to various parts of the system. This homepage provides links to schedule builders for different majors, including Computer Science, Mechanical Engineering, Civil Engineering, Electrical Engineering, and Biology. Currently, only the Computer Science schedule builder is operational, but plans to expand and implement schedule builders for additional majors are in progress.

The Computer Science schedule builder itself is composed of several subsystems that are integral to the overall functionality and maintenance of the application. These subsystems include: coursereader.py, (cs\_style.css, script.js, selected.js) Each of these components plays a key role in ensuring the smooth operation of the Computer Science schedule builder.

## 5.1 Flask Web Server Subsystem

Each subsystem design should include a description of subsystem interfaces, a class diagram of subsystem design classes and interfaces, sequence diagrams that depict the internal interactions that support subsystem interfaces, and any nested subsystems.The flask web server subsystems are responsible for routing, handling requests, and serving the frontend pages.

## 5.2 Homepage Subsystem

Displays the main page where users can navigate to different schedule builders. These homepage subsystems include ( computer science, mechanical engineering, electrical engineering, civil engineering and biology). Currently the only one of these subsystems that is functional is computer science however there are plans to implement the rest in the future.

## 5.3 Schedule Builder Subsystems

Individual schedule builders which a user can select and use to view and select courses based on available courses. These subsystems will be reliant on both the Fontened UI subsystem to display information and the schedule builder as well as the course reader subsystem to get the required information to display.

**5.4 Course Reader Subsystem**

Reads in from a text file containing all the courses and their related information and organizes them into a way where it can be utilized by the schedule builder subsystems. This subsystem will eventually be updated to scrape the online NMT course catalog and get the most recent course information for the schedule builder.

**5.5 Frontend UI Subsystem**

Manages the visual design and user interaction of the schedule builder. This system mainly relies on HTML for structure, CSS for styling and Javascript for dynamic functionality such as adding classes and updating visuals without reloading the page.

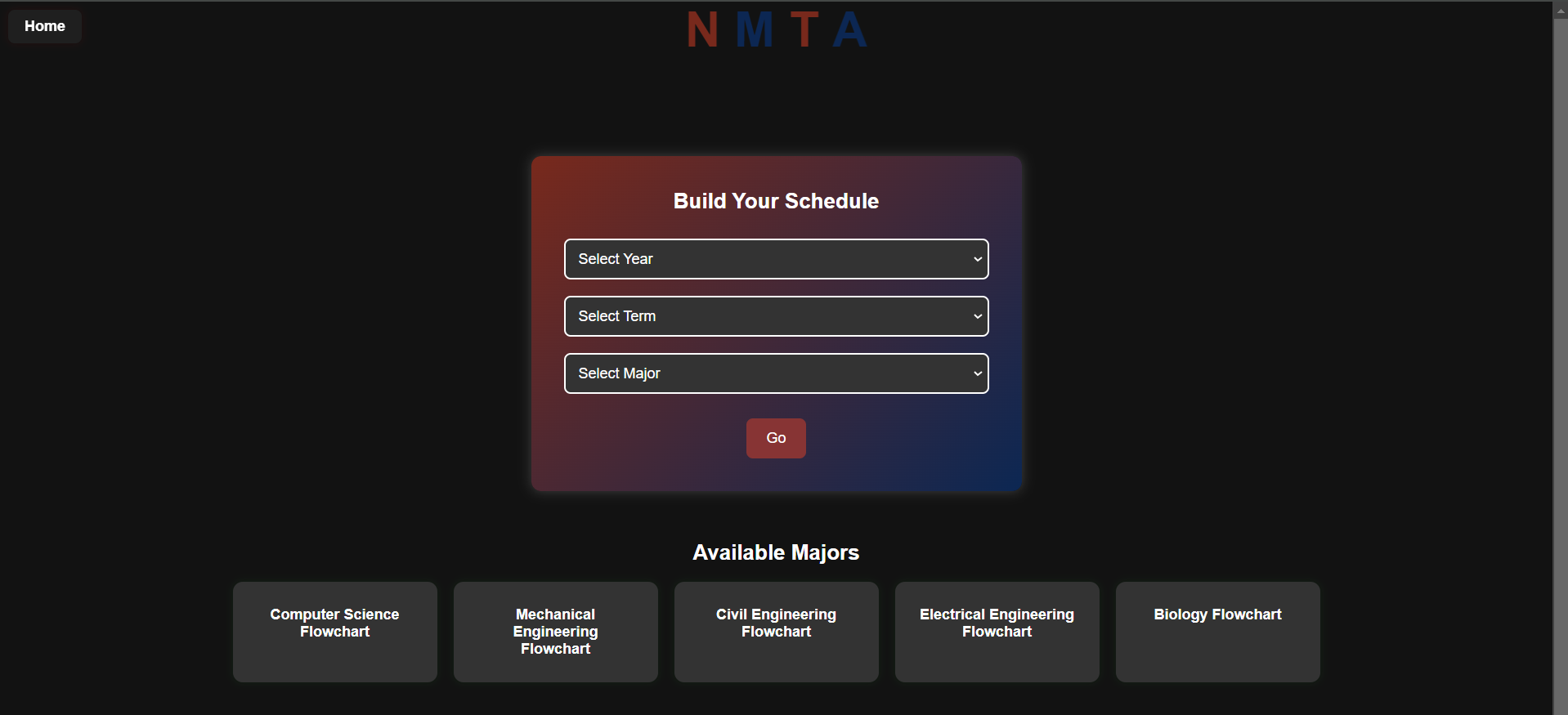
**5.6 Llama 2 Subsystem**

This subsystem is a chatbot responsible for assisting the user with navigating the application and building their schedule. This chatbot will be accessible through a chatbox that will be displayed on each page so the user has constant access to it. **(Note: This subsystem has not been implemented yet.)**

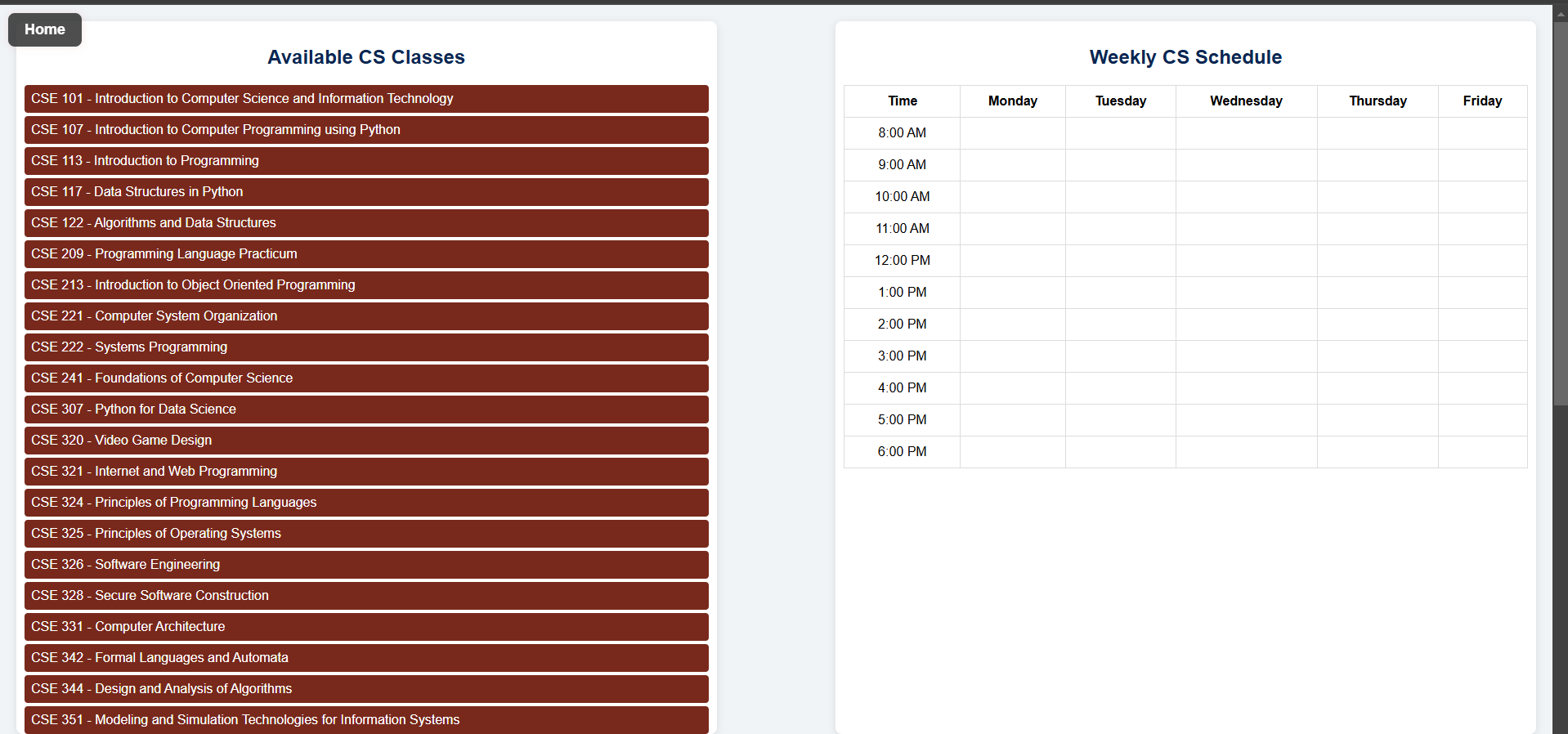
# 6. Human Interfaces (10 points)

Mockup designs for each user interface.

Home Page:



Schedule Builder Page:



# 7. Testing Plan (5 points)

Once our implementation has reached penultimate design, we will ask colleagues to create their own schedule that they have for this semester (as the schedule builder is using this semester’s course times) and receive feedback from them in order to create the final product.

# 8. Appendices (5 points)

## 8.1 Project Status

We have been making steady progress on the project and are currently on track to have the majority of our expected functionality and design completed by early May. Currently we have a majority of the user interface complete which includes the homepage, flowcharts and schedule builder pages which the user can interact with. As for the backend functionality of our application there are still a few key elements we need to implement before we can move to the testing phase of the project. These include scraping the class information directly from the online NMT course catalog so we can have the most recent data on the offered NMT courses, this will involve updating the course reader subsystem to access the online database and scrape information from it. Additionally the Llama 2 subsystem plays an integral part in this project and we have plans to implement this system soon.