**Group 11**

**Derryck Ramos – Student Enrollment System**

**Project Description**

Enrollment in college has always been a hassle, between all the paperwork one had to fill out and all the offices one had to visit. A lot of processes are still done by hand, and what few online solutions exist leave something to be desired in terms of performance or user experience.

This project aims to answer the question of whether it is possible to create a Student Enrollment System where the process of adding and dropping classes is quick and easy for the student, and configuring what classes are available is equally simple for administrative staff.

**Project Scope and Expected Results**

**Overview**

This project is primarily an MVC web application, but it also has a few API services available for consumption. The following section will describe the primary functions the project will have. Said functions will be classified by the user role able to perform them (namely: the administrator user and the student user). Under each function, the expected result will be described.

**In Scope**

**General (available to both user types):**

* Logging in and out of the system
* Changing their password

**Adminstrator Functions**

* **Managing Departments** 
  + The administrator will be able to create, update, view, and delete Department information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific Department with the click of a corresponding button.
  + e.g. Applied Sciences department
* **Managing Courses** 
  + The administrator will be able to create, update, view, and delete Course information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific Course with the click of a corresponding button.
  + (e.g. Computer Science, under the Applied Sciences department)
* **Managing Faculty** 
  + The administrator will be able to create, update, view, and delete Faculty information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific Faculty with the click of a corresponding button.
  + e.g. Charles Xavier, faculty member under the Applied Sciences department
* **Manage Subjects** 
  + The administrator will be able to create, update, view, and delete Subject information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific Subject with the click of a corresponding button.
  + e.g. Web Programming, under the Computer Science course
* **Manage Sections** 
  + The administrator will be able to create, update, view, and delete Section information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific Section with the click of a corresponding button.
  + e.g. MAB, a three-letter code that corresponds to a subject’s schedule during a week; this example means that the subject is a Monday class with a time of 7 AM to 8:30 AM
* **Manage Student Accounts** 
  + The administrator will be able to create, update, view, and delete Student Accounts information. Creation and updating is done via filling out a form with pertinent information, while deletion is accomplished via a corresponding button. Users will have the option to view the full list of departments in a tabulated form (with the table having additional features such as a live search bar, sorting, and pagination) as well as more details on a specific account with the click of a corresponding button.
  + The resulting student accounts created through this process can be used to log into the system as a Student User and perform all the functions available to them.

**Student Functions**

* **View Subject list**
  + The student will be able to view a table list of all available classes similar to that of the administrator. The aforementioned filtering, sorting, and pagination features are also available here, but what the student can do with the presented information is different.
* **Enroll Subject/s**
  + The student will be able to enroll a class by clicking a corresponding button next to the desired subject
* **View Semester Schedule**
  + The student will be able to view a table list of all the subjects they have chosen to enroll. The aforementioned filtering, sorting, and pagination features are also available here.
* **Drop Subjects/s**
  + The student will be able to drop a class from their schedule by clicking a corresponding button next to the desired subject, thereby removing it from their schedule view.

**Subject APIs (available to both user types)**

* **View all Subjects**
  + Returns a full list of all subjects available in the database
* **View Subjects by Course**
  + Returns a list of subjects under a specified course
* **View Subjects by Schedule**
  + Returns a list of subjects with a specified section code (which corresponds to a schedule)
* **View Subjects by ID**
  + Returns a single subject based on the given subject ID

**Enrollment APIs**

* **Log In**
  + Using the login credentials used in the MVC application, this API generates a JSON Web Token that grants users authorization to use the below APIs
* **View Own Enrollment Schedule (Student User only)**
  + Returns the list of subjects a logged-in student has chosen to enroll
* **View Enrollment Schedule by Email (Administrator User only)**
  + Returns the list of subjects of a student has chosen to enroll. The student in question is specified via the input of an email address corresponding to the login credentials they use to log into the application.

**Out of Scope**

Student Enrollment is a multi-step process, and realistically speaking, it would have been an arduous undertaking to develop an application that covers every single aspect of that process. Thus, this project focuses specifically on the step that involves the actual adding and dropping of classes to one’s semester schedule. Below are a few features that were deliberately passed over, and can be considered as recommendations for a potential next iteration of this application.

Processes that would normally take place prior to (e.g. pre-enlistment advising to decide which subjects a student would be permitted to take) or after this step (e.g. tuition calculation for all the enrolled subjects) were not implemented. Either integrating these processes into the project or creating APIs that allow this project to give and receive data to standalone implementations of these processes would be an ideal addition.

Faculty Users could potentially be allowed to use the system as well, permitting them to select which subjects they would teach during a given semester, as well as the ability to view other pertinent information. An attempt was made to implement this function very early on into development, but the complexity its addition presented in terms of the database structure was too significant a hurdle for a single developer to manage with the given time frame.

The Enrollment APIs were created with the intention of integrating them into third party calendar applications (so one could import a weekly schedule). This idea was beyond the developer’s knowledge at the time.

Currently, the application operates in a vacuum, with no outside context defining when the enrollment features should be available. Normally, there is a specified enrollment period for college students, and as of now, the application has no concept of only allowing access to enrollment features within a given time period.

As it stands, students can only log in via the accounts created for them by the administrator user with provided credentials. It would be more recommended that a single sign-on feature that utilizes a student’s university/college email address be implemented in place of this.