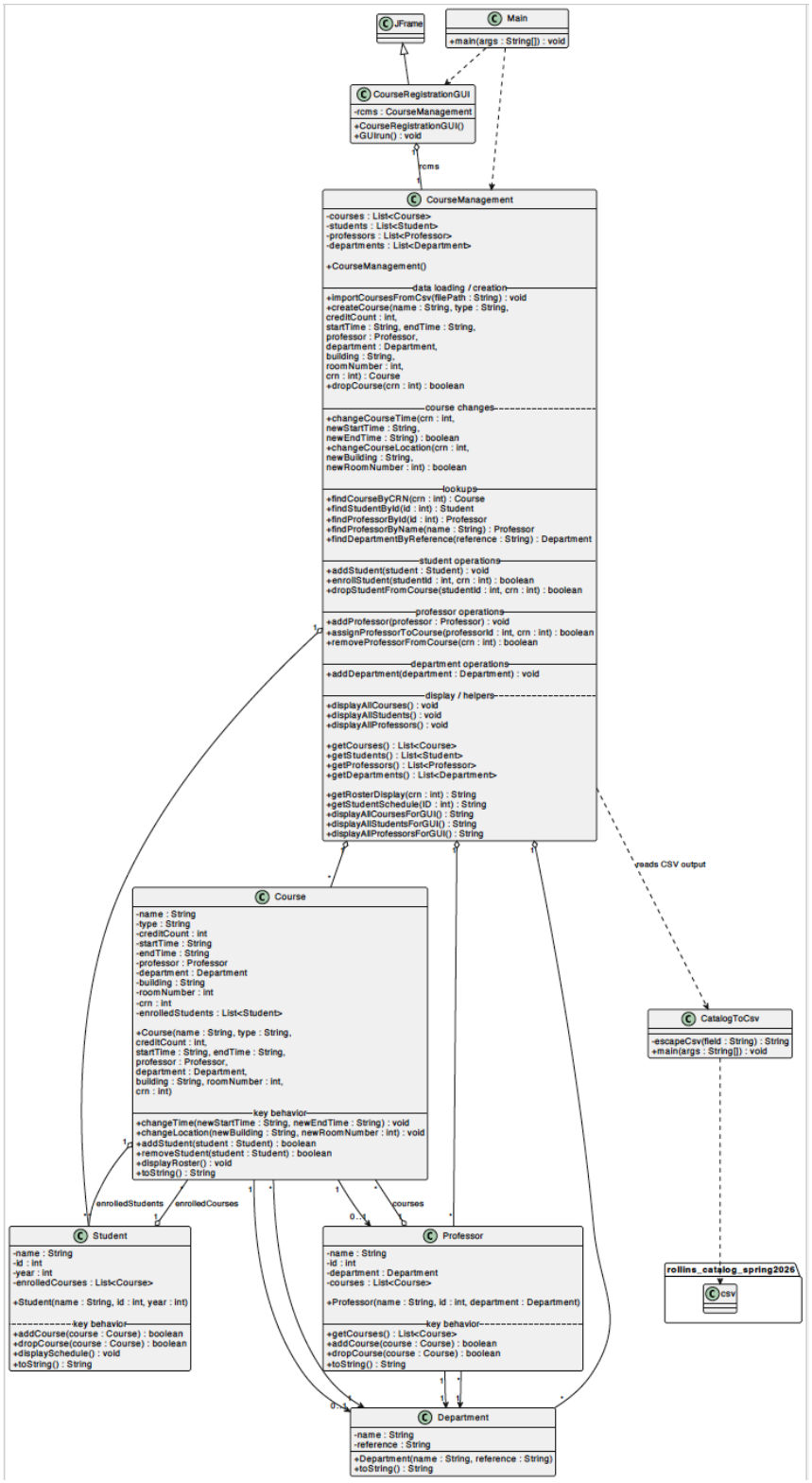
Rollins College
Course Management System Report

Phuc Dao: System Integration and Design
William Slowey: Back End Programming
Alex Octuk: Front End Programming

Introduction:

This project is the creation of a course registration system using Rollins College's course organization logic. All objects have references or numbers associated with the instance of that object, and thus, all objects can be easily searched for using their references or identification numbers. We chose to create a course registration system because the one Rollins currently uses is prone to crashing and glitches when there is too much activity on the system. Our system aims to simplify the system so usage is easier and improve the processing to prevent crashes or glitches.

UML Diagram:



Implementation:

First, the overarching skeleton of the classes was constructed, defining which variables are held within which classes, including the visibility and structure in which they will be held. The student, department, professor, and course classes hold the information regarding their respective objects, then the CourseManagment class acts as a bridge between all classes, holding the methods to find, change, and display all information requested by the user in the Eclipse window. Methods were made to:

- Create students, departments, courses, and professors
- Change the information of instances of students, departments, courses, and professors
- Get any requested information from any class, regardless of visibility
- Manage arrays within classes to create or remove instances of courses
- Manage arrays within classes to add to or drop from professor or student schedules
- Display full course catalog
- Display course rosters
- Display student and professor schedules

Next, the GUI system was created and implemented. The final form of the GUI creates multiple panels, one for each login type (Student, Professor, or Admin). Each panel shows the functions available to that account's permission level. For example, students can only see the course catalog and their schedules, in addition to being able to add and drop courses from their schedules. However, the admin account can change anything about a course, professor, student, or department. Each panel has a series of buttons or text fields and labels, showing the function of that panel. Some buttons lead to further panels where more specific functions are offered until only one action can be taken. The layered panels created a simple design that allowed one direct action for the user to choose at a time, providing ergonomics and understandability. This GUI design came with complications in bookkeeping, whether panels were visible or not, leading to complications. Parallel to the GUI implementation, the Rollins College Course Catalog was imported and converted into a usable CSV file, and methods were added to the CourseManagment class to take the courses and other information and import it into the CourseManagment system. Input validation was added to make sure the proper values were being added. The implementation made sure that the proper data types were inputted when prompted and inputs matched the current course catalog. The implementation of exception handling helped improve user experience and limit errors with data management.

Challenges and Solutions:

The GUI implementation brought about issues with the data access permission and the static variable references. Most of the issues were resolved by using public static variables, so each class is able to access all data, and the data is difficult to change, avoiding confusion as the system is used. Some of the data was not able to be static due to the dynamic changes that happen through user input, particularly with the student schedules and course rosters. These changed so often that making them static variables caused more issues; instead, a function was added to collect all the data requested when a course roster or student/professor schedule needed printing. This function then stored the requested data as a large string and passed it to the GUI for display.

Another major challenge we faced was importing the real Rollins course catalog for spring 2026 and transforming its raw, web-formatted information into a structured dataset that our system could use effectively. The catalog’s inconsistent HTML formatting and

varying layouts made it difficult to directly map the content to our course objects. To solve this, we learned how to use the JSoup library to locate and extract the specific elements we needed, such as course titles, meeting times, and CRNs, while cleaning the data to remove unnecessary tags. Another challenge involved storing this data locally, which required us to work with Java's BufferedReader and handle potential errors through IOException. We spent time understanding the correct way to read from and write to files while ensuring the program remained stable. Through collaboration, debugging, and experimenting with different parsing strategies, we successfully converted the catalog into a format compatible with the system's data structures.

Contribution Summary:

Alex Octuk designed and implemented the GUI and user interface, facing challenges with formatting and computational organization. Within the report, Alex synthesized the implementation of the GUI, the associated challenges and solutions, as well as creating the framework for the report to follow.

William Slowey worked on the backend with designing the methods and writing the classes. William worked on creating and managing. Additionally, William worked on creating the login page for the GUI. He also created the exception handling and input validation.

Phuc Dao was responsible for the system integration and design. Phuc imported the official Rollins Spring 2026 course catalog by scraping the data and converting it into a usable CSV format, then integrated that data into our database to support real course search and enrollment functionality. Also, Phuc designed the overall system architecture and core features that enable students, professors, and courses to interact smoothly within the platform. He also created the complete UML class diagram to clearly represent our object-oriented structure and guide development. Additionally, Phuc designed the presentation materials to communicate our project's goals, functionality, and technical design.