ALBUKHARY INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTING AND INFORMATICS

SEMESTER 4 2024/2025

CCC2143

SOFTWARE ENGINEERING

GROUP PROJECT ASSIGNMENT (40%)

PROJECT TITLE	**CENTRALISED COURSE REGISTRATION SYSTEM (CCRS)**	
GROUP LEADER	**MD FARMAN ALI**	**AIU20092185**
GROUP MEMBERS	**AREYAN MUHEMED ALI HUSSEIN**	**AIU21102179**
GROUP MEMBERS	**FUAD HIYBO HUSSIN MOHAMED**	**AIU21102174**
GROUP MEMBERS	**AROUA TALHI**	**AIU22102123**

TABLE OF CONTENTS

1. INTRODUCTION

The ‘Centralized Course Registration System’ is designed to streamline and enhance the course registration process for both students and administrative staff. In today's fast-paced academic environment, managing course offerings, student enrollments, and academic records can be complex and time-consuming. This system aims to address these challenges by providing students with an intuitive platform to search for courses, register, manage schedules, and track academic progress. Additionally, students can efficiently add or drop courses and generate reports on their performance, making academic management more organized and accessible.

On the administrative side, the system equips staff with tools to oversee course schedules, monitor student registrations, and generate comprehensive reports on enrollment data. With a centralized approach, administrators can efficiently update course offerings and schedules while maintaining control over sensitive student data through enhanced security protocols. By incorporating scalability, ease of use, and strong security features, the system is built to adapt to the evolving needs of the institution, ensuring improved operational efficiency and a more seamless user experience.

2. System Project Objectives

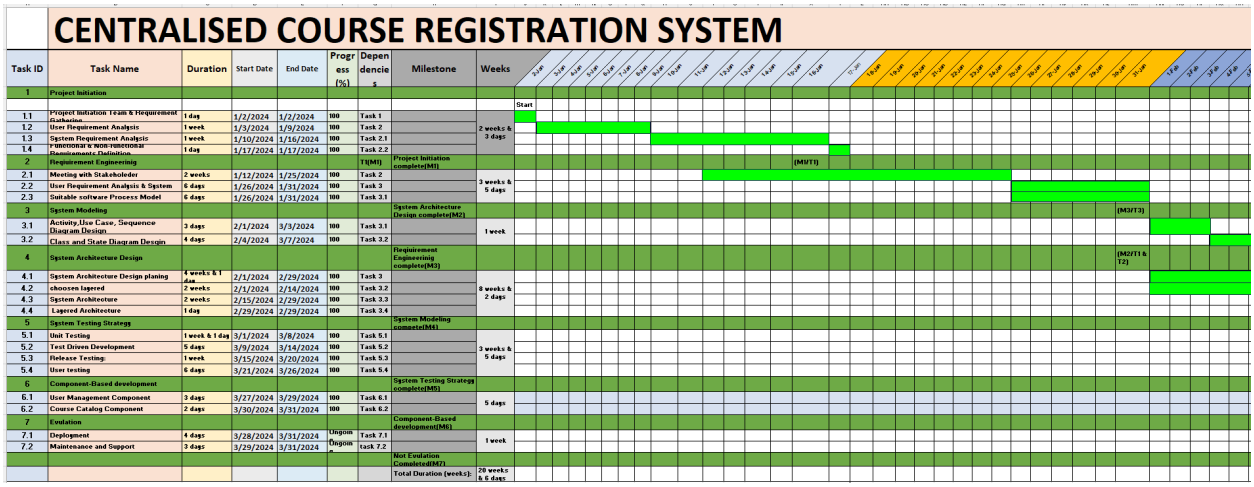
The ‘Centralized Course Registration System’ aims to simplify the course registration process for students and administrators in educational institutions. It is designed to enhance the academic experience by providing an efficient platform for managing course selections and offerings.

a. **Vision:** “To empower students and administrators with a user-friendly system that streamlines course management and enriches the educational experience.”

b. **Objective:** “To provide an effective and accessible platform that allows students to easily search for, register for, and manage their courses while enabling administrators to efficiently oversee course offerings and track enrollments.”

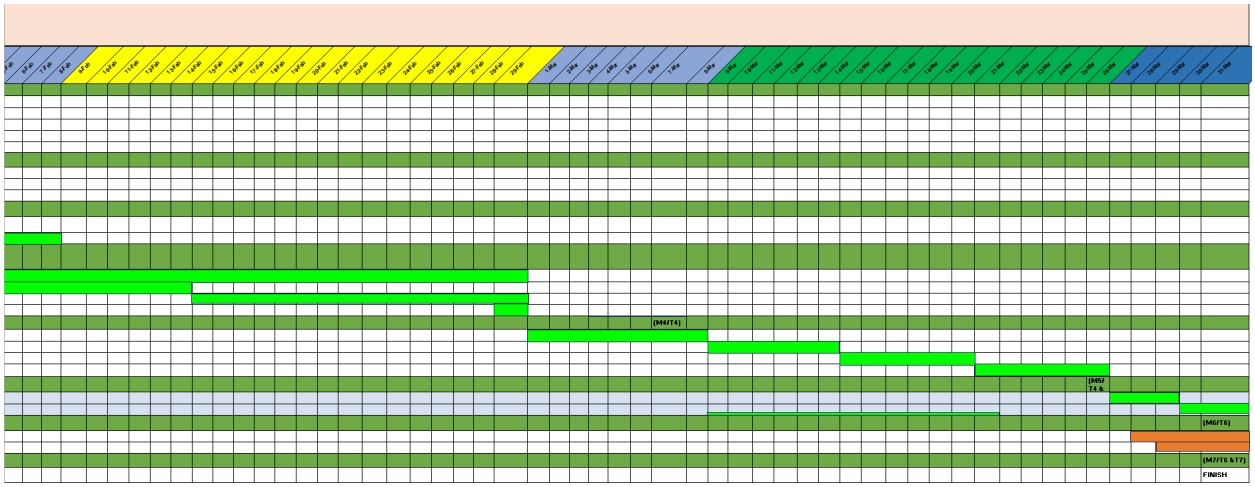
3. PROJECT TIMELINE: GANTT CHART

https://1drv.ms/x/c/6c9f0ed7b5cf5b90/EXMHotyQGP9EmHOuccYewpgBcWVPjQC_1Nd6c5ry4tlfgg



CENTRALISED COURSE REGISTRATION SYSTEM

Task ID	Task Name	Duration	Start Date	End Date	Progress (%)	Dependencies	Milestone	Weeks
1	Project Initiation							
1.1	Project Initiation Team & Requirement Gathering	1 day	1/2/2024	1/2/2024	100	Task 1		2 weeks & 3 days
1.2	User Requirement Analysis	1 week	1/3/2024	1/9/2024	100	Task 2		
1.3	System Requirement Analysis	1 week	1/10/2024	1/16/2024	100	Task 2.1		
1.4	Functional & non-functional Requirements Definition	1 day	1/17/2024	1/17/2024	100	Task 2.2		
2	Requirement Engineering					T1(M1)	Project Initiation complete(M1)	
2.1	Meeting with Stakeholder	2 weeks	1/12/2024	1/25/2024	100	Task 2		3 weeks & 5 days
2.2	User Requirement Analysis & System	6 days	1/26/2024	1/31/2024	100	Task 3		
2.3	Suitable software Process Model	6 days	1/26/2024	1/31/2024	100	Task 3.1		
3	System Modeling						System Architecture Design complete(M2)	
3.1	Activity,Use Case, Sequence Diagram Design	3 days	2/1/2024	3/3/2024	100	Task 3.1		1 week
3.2	Class and State Diagram Design	4 days	2/4/2024	3/7/2024	100	Task 3.2		
4	System Architecture Design						Requirement Engineering complete(M3)	
4.1	System Architecture Design planing	4 weeks & 1 day	2/1/2024	2/29/2024	100	Task 3		8 weeks & 2 days
4.2	choosen layered	2 weeks	2/1/2024	2/14/2024	100	Task 3.2		
4.3	System Architecture	2 weeks	2/15/2024	2/29/2024	100	Task 3.3		
4.4	Layered Architecture	1 day	2/29/2024	2/29/2024	100	Task 3.4		
5	System Testing Strategy						System Modeling comset(M4)	
5.1	Unit Testing	1 week & 1 day	3/1/2024	3/8/2024	100	Task 5.1		3 weeks & 5 days
5.2	Test Driven Development	5 days	3/9/2024	3/14/2024	100	Task 5.2		
5.3	Release Testing	1 week	3/15/2024	3/20/2024	100	Task 5.3		
5.4	User testing	6 days	3/21/2024	3/26/2024	100	Task 5.4		
6	Component-Based development						System Testing Strategy complete(M5)	
6.1	User Management Component	3 days	3/27/2024	3/29/2024	100	Task 6.1		5 days
6.2	Course Catalog Component	2 days	3/30/2024	3/31/2024	100	Task 6.2		
7	Evolution						Component-Based development(M6)	
7.1	Deployment	4 days	3/28/2024	3/31/2024	Ongoing	Task 7.1		1 week
7.2	Maintenance and Support	3 days	3/29/2024	3/31/2024	Ongoing	task 7.2		
							Not Evolution Completed(M7)	
							Total Duration (weeks):	20 weeks & 6 days



4. SUITABLE SOFTWARE PROCESS MODEL

The approach interleaves the activities of specification, development, and validation. The system is developed as a series of versions (increments), with each version adding functionality to the previous version.

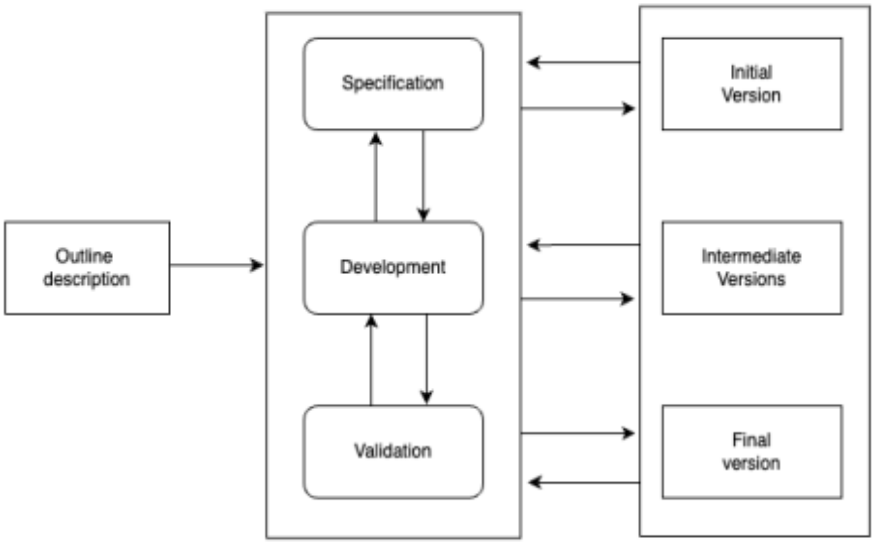


Figure 1.0: Incremental Development Process

Incremental development is a software development approach where the software system is developed and evolved until the required system is developed through feedback from users and others. Specification, development and validation are interleaved instead of separate. The specification is done at initial version, development at intermediate and validation comes as a final version. These three activities are concurrent meaning simultaneous. In an agile approach, the early increments are identified and the later increments are developed based on the progress and user's priorities.

The incremental method has a lot of advantages and suitable to our project with the following reason

1. Early and Ongoing Delivery

- **Early Use:** Some of the functionalities of the system, like the course registration, can be undertaken from the early stage. This contributes to the students and administrators engaging themselves with the system in the right manner.
- **Step-by-Step Improvement:** Modifications are made with the passage of time, and thus, the system is improved with every stage.

2. User Feedback

- **Involving Users:** Users can use the system at an initial stage and also be able to express their views on the system. This assists in guaranteeing that the system meets his/her needs.
- **Easy Changes:** Notes from users help to refine the further stages, and keep the system as close as possible to the expectations of the audience.

3. Reduced Risk

- **Lower Risk:** As the system develops in these small chunks, issue are easily addressed, thus preventing it from escalating into a big problem.
- **Focus on Important Features:** Besides, it could help to prioritize and design the most important parts primarily, while the less important ones could be left for a second, and/or cooperate with other departments.

4. Easier Testing

- **Simple Testing:** One of the beauty of this type of structure is that each part tested individually so it is simple to detect errors in it.
- **Better Quality:** This in a way contributes to the final system where testing is done on subunits because that way, the system will work optimally.

5. Flexibility

- **Adaptable:** What is more, many necessary requirements can be developed further down the line, as the project progresses.
- **Scalable:** It has the flexibility of expanding over a period depending with the additions of other extra features or even modules.

6. Efficient Use of Resources

- **Focused Development:** It allows teams to work on different parts at the same time depending on their skills which would be most effective.
- **Easy Progress Tracking:** Every finished step is defined clearly, it is beneficial regarding orientation in the work, especially for a team.

7. Handling Complexity

- **Manage Complexity:** It is easier to manage the complexity in the process of building the system in parts because all the parts can be well designed and tested.

.

5. SYSTEM REQUIREMENT ANALYSIS:

a. User Requirements:

The **'Centralized Course Registration System'** is designed to meet the diverse needs of both students and administrative users, ensuring a seamless and efficient experience for all. For students, the system provides the ability to **search** for and **register in courses**, **view and manage course schedules**, **track academic progress**, and **generate detailed reports** on their performance. Additionally, students can easily **add, drop, and print course schedules** as needed. On the administrative side, the system empowers administrators to **manage course offerings and schedules, oversee and update student registrations, and generate comprehensive reports on course enrollment** and other critical metrics. This integrated approach ensures that all users can interact with the system effectively, enabling a smooth and organized course registration process.

b. System requirements:

- **Register as a Student:** The system will include a registration feature that allows students to create accounts by providing personal and academic information, which will be securely stored.
- **Log In:** Users (students and administrators) will have a secure authentication process to log into the system.
- **Search Courses:** A search module will enable students to find courses based on various criteria, such as course name, code, or instructor.
- **Register for Courses:** Students can submit course registration requests, specifying the courses they wish to enroll in, along with any prerequisites and conditions.
- **View Course Schedule:** The system will allow students to view their enrolled courses and associated schedules.
- **Manage Registrations**: Students will have the ability to add or drop courses and print their course schedules.
- **Track Academic Progress:** A module will enable students to monitor their grades and academic performance over time.
- **Generate Reports:** The system will provide functionality for both students and administrators to generate reports on course enrollment, student performance, and other relevant metrics. Manage
- **Course offerings:** Administrators will have tools to create, update, and delete course offerings and schedules.
- **Handle Student Registrations:** A management module for administrators to handle student registrations, including approval or denial of requests.
- **Update Course Schedules:** Administrators can modify course schedules and communicate changes to students.
- **Security and Access Control:** Implementation of data encryption and secure access controls to protect sensitive student and course information.
- **Scalability:** The system must be able to accommodate a growing number of users and courses as the institution expands. User-Friendly Interface: An intuitive and responsive design to ensure ease of use for all users, including students and administrators.

- **Notification System:** The system will include notification functionalities to update students about their registration status, course updates, and important deadlines.

c. Functional Requirements:

- **User Registration and Authentication:** Students and administrators must be able to register and securely log into the system.
- **Course Search:** Students can search for courses using various filters (e.g., course name, code, instructor, department).
- **Course Registration:** Students can register for available courses, including the ability to add or drop courses.
- **View and Manage Course Schedules:** Students can view their enrolled courses and associated schedules.
- **Track Academic Progress:** The system allows students to monitor their academic performance, including grades and completed courses.
- **Generate Performance Reports:** Students can generate detailed reports of their academic performance.
- **Manage course offerings:** Administrators can create, update, and delete course offerings and schedules.
- **Student Registration Management:** Administrators can oversee and update student registrations, including approval or denial of course requests.
- **Generate Enrollment Reports:** Administrators can generate reports on course enrollment, student demographics, and other critical metrics.

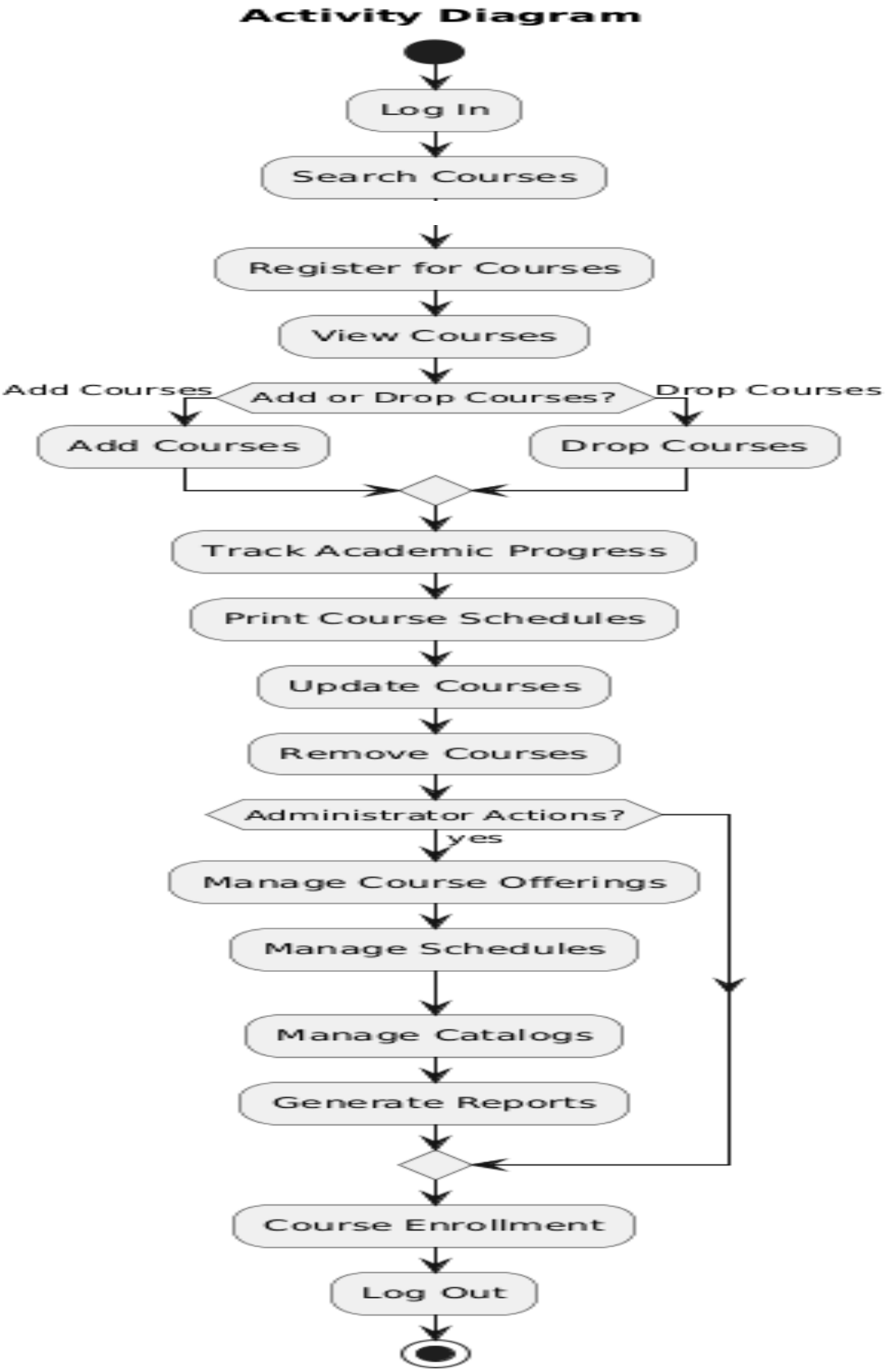
d. Non-Functional Requirements

- **Performance:** The system should handle a high volume of simultaneous users without performance degradation.
- **Security:** Data encryption and secure access controls must be implemented to protect user data and sensitive information.

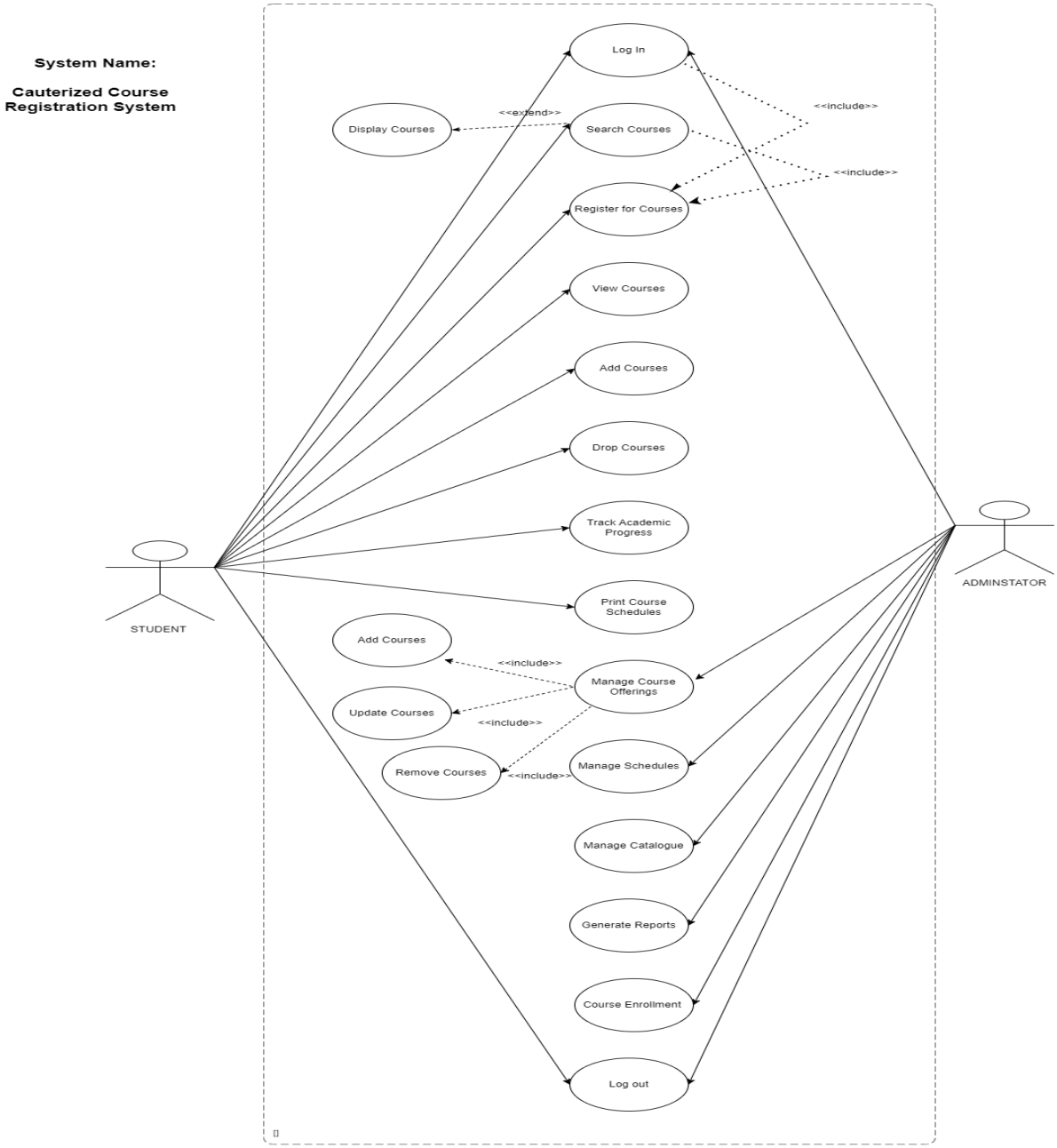
- **Usability:** The system should have an intuitive and user-friendly interface that facilitates easy navigation for all users.
- **Scalability:** The system must be able to scale to accommodate a growing number of users and courses as the institution expands.
- **Reliability:** The system should ensure high availability and reliability, minimizing downtime and ensuring users can access the system when needed.
- **Maintainability:** The system should be designed for easy updates and maintenance to incorporate new features and address issues promptly.
- **Compatibility:** The system should be compatible with various devices and browsers to ensure accessibility for all users.
- **Response Time:** The system should respond to user requests promptly, ensuring a smooth user experience.
- **Flexibility:** Create a system that permits administrators to adjust settings according to evolving requirements.

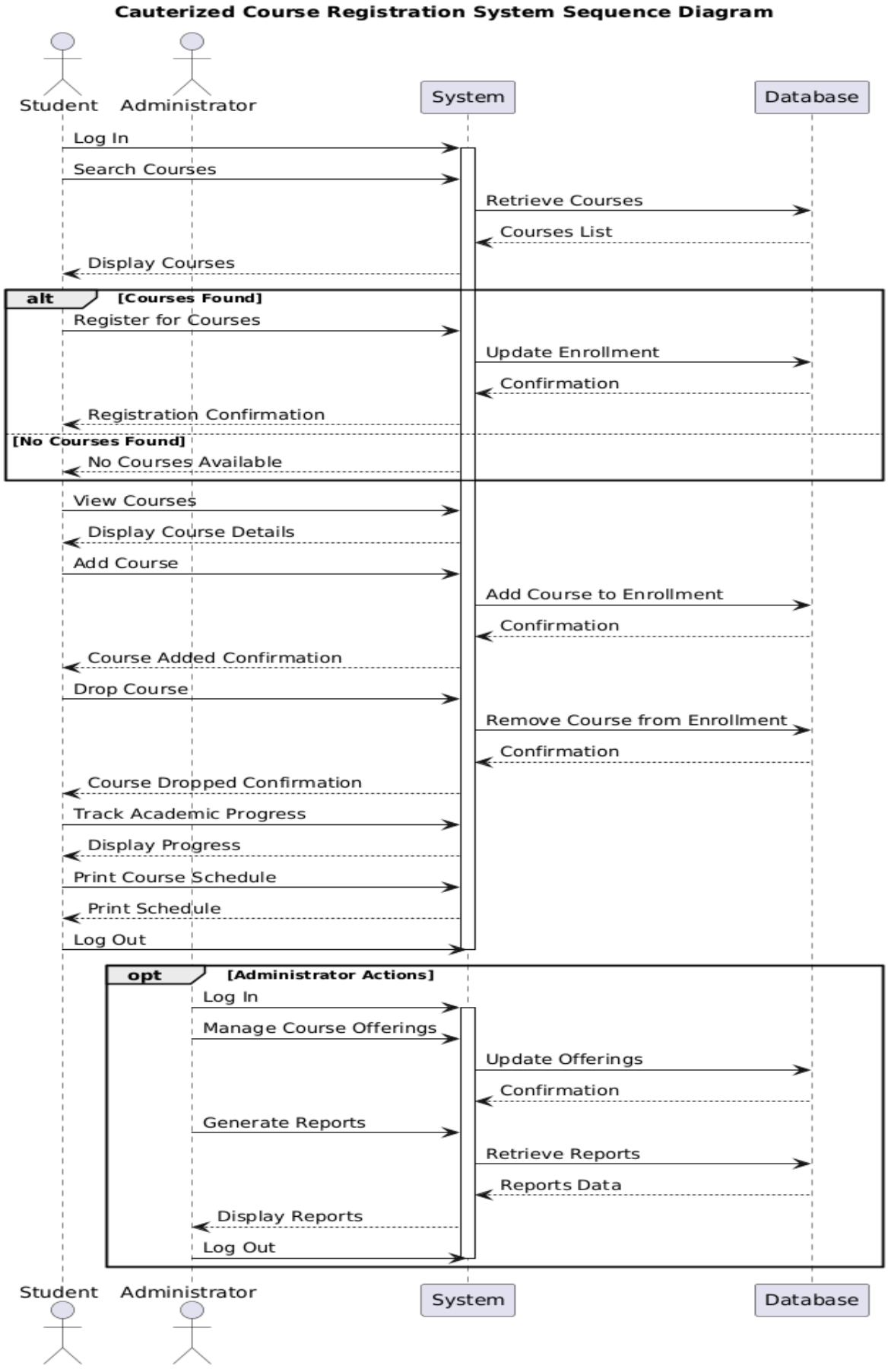
5.1 SYSTEM MODELS

Activity Diagram: This **activity diagram** [Figure 1.1**]** effectively maps out the sequence of actions and decisions involved in the course registration process within the Cauterized Course Registration System. It highlights how students interact with the system, the choices they make regarding course management, and the administrative functions that support overall course offerings. The clear flow of activities and decision points provides a comprehensive view of the user experience, making it a useful tool for understanding system functionality and guiding development.

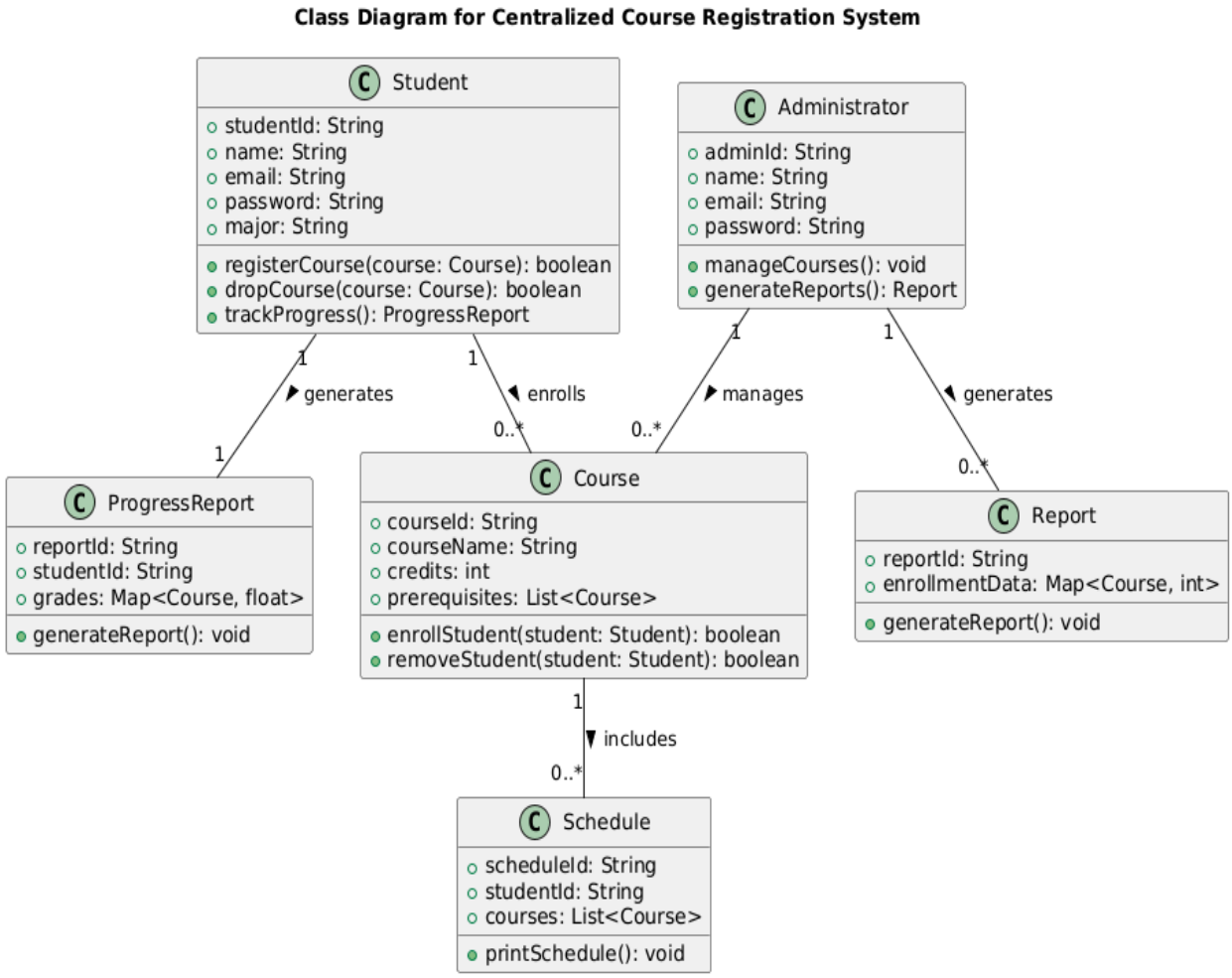

Activity Diagram
Log In
Search Courses
Register for Courses
View Courses
Add or Drop Courses?
Add Courses
Drop Courses
Add Courses
Drop Courses
Track Academic Progress
Print Course Schedules
Update Courses
Remove Courses
Administrator Actions?
yes
Manage Course Offerings
Manage Schedules
Manage Catalogs
Generate Reports
Course Enrollment
Log Out

This **use case diagram** [Figure 1.2**]** effectively encapsulates the interactions between students and administrators within the 'Cauterized Course Registration System'. It highlights the primary functionalities of the system and how users engage with them. By visually representing these relationships, the diagram serves as a useful tool for understanding user requirements and guiding the development process, ensuring that all necessary features are considered

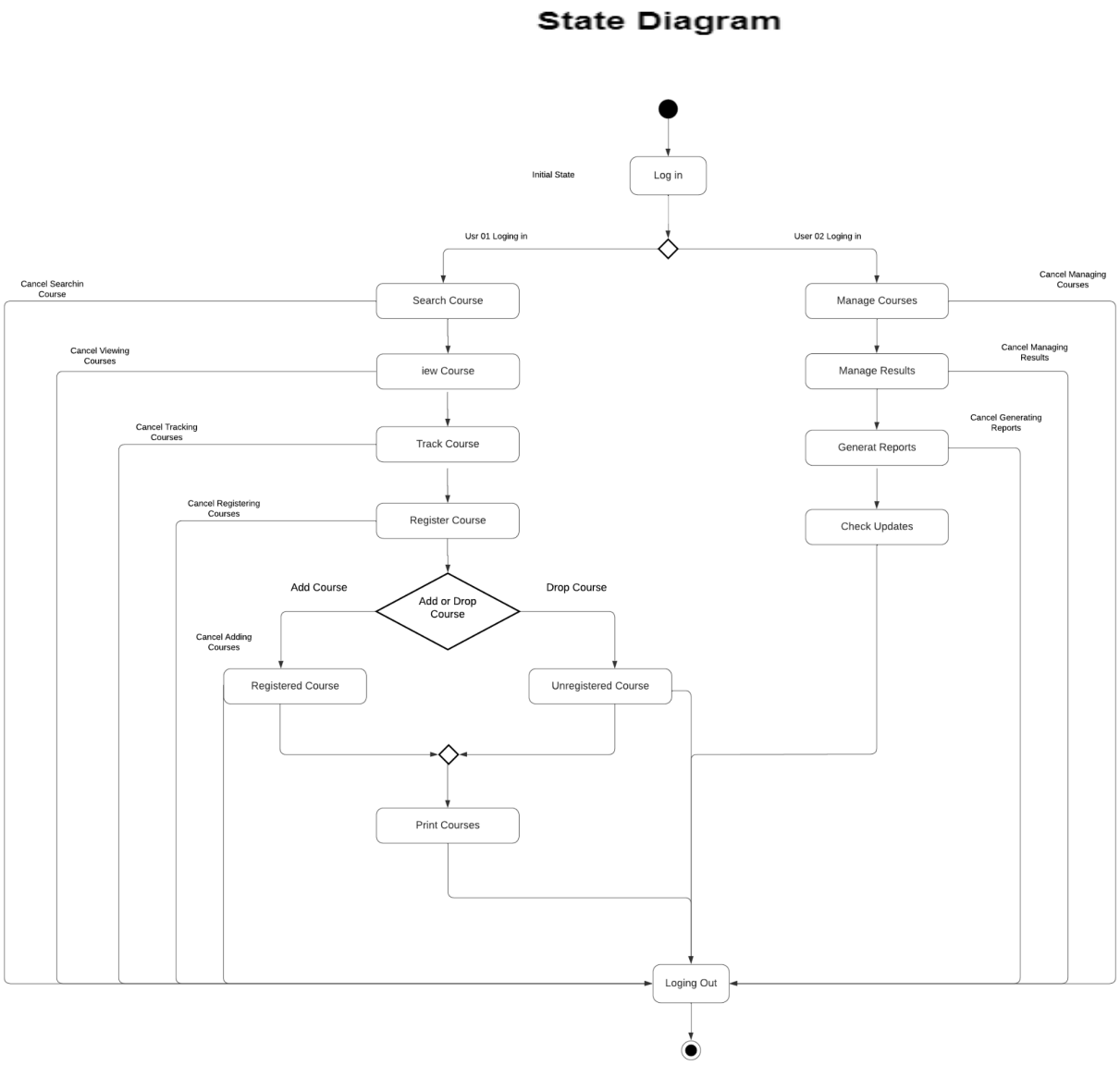



Cauterized Course Registration System Sequence Diagram
Student
Administrator
System
Database
Log In
Search Courses
Retrieve Courses
Courses List
Display Courses
alt
[Courses Found]
Register for Courses
Update Enrollment
Confirmation
Registration Confirmation
[No Courses Found]
No Courses Available
View Courses
Display Course Details
Add Course
Add Course to Enrollment
Confirmation
Course Added Confirmation
Drop Course
Remove Course from Enrollment
Confirmation
Course Dropped Confirmation
Track Academic Progress
Display Progress
Print Course Schedule
Print Schedule
Log Out
opt
[Administrator Actions]
Log In
Manage Course Offerings
Update Offerings
Confirmation
Generate Reports
Retrieve Reports
Reports Data
Display Reports
Log Out
Student
Administrator
System
Database

sequence diagram [Figure 1.3**]** effectively captures the interactions and processes involved in the 'Cauterized Course Registration System'. It highlights the flow of information between users and the system, detailing how students manage course registrations and how administrators maintain and oversee course offerings. The use of conditional paths and options illustrates the dynamic nature of the system, accommodating different scenarios based on user actions. Overall, this diagram serves as a useful tool for understanding the system's functionality and user interactions**.**



This **class diagram** [Figure 1.4**]** provides a clear overview of the entities within the Centralized Course Registration System, their attributes and methods, and how they interact. It lays the groundwork for understanding the system's architecture, which is crucial for both development and maintenance. The relationships and methods illustrated here help clarify how users (students and administrators) will interact with the system and how data flows between different components.



The **state diagram** [Figure 1.5**]** effectively captures the user journey within the 'Centralized Course Registration System'. It illustrates how users interact with the system, the various options available, and the administrative functions that support course management. The clear flow of states and actions helps in understanding the system's functionality and user interactions, serving as a valuable tool for both development and user experience design.

6.SYSTEM ARCHITECTURE

- **Layered Architecture**

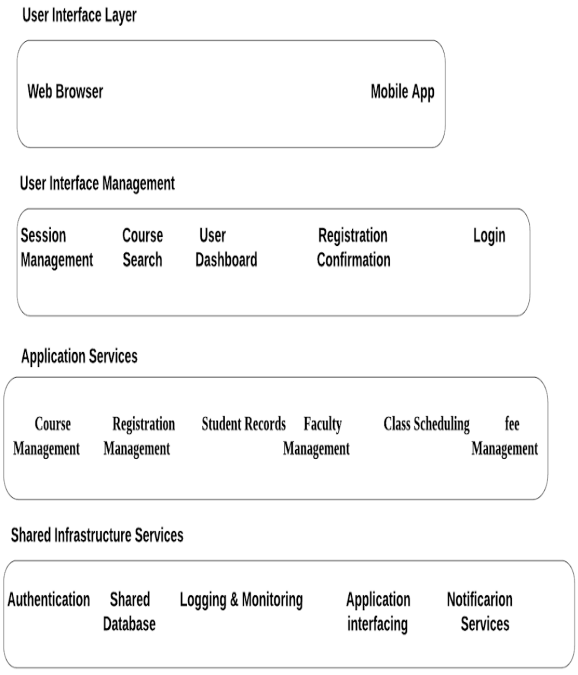


Figure 1.1: Layered Architecture

1. User Interface:

Web Browser & Mobile App: Interfaces through which students, faculty, and administrators interact with the system.

2. User Interface Management:

Session Management: Manages user sessions.

Course Search: Allows users to search for courses.

User Dashboard: Displays personalized information for users.

Registration Confirmation: Confirms successful course registrations.

Login: Handles user authentication to access the system.

3. Application Services

Course Management: Handles course creation and updates.

Registration Management: Manages student course registrations.

Student Records: Maintains student information.

Faculty Management: Manages faculty information.

Class Scheduling: Organizes class times and locations.

Fee Management: Manages tuition and fees.

4. Shared Infrastructure Services

Authentication: Verifies user identities.

Shared Database: Stores all system data.

Logging & Monitoring: Tracks system activity and performance.

Application Interfacing: Connects the system with external applications.

Notification Services: Sends alerts and updates to users.

7. COMPONENT-BASED DEVELOPMENT:

a. User Management Component

The User Management Component is essential for maintaining the integrity and security of the Centralized Course Registration System. It ensures that only authorized users can access the system and perform actions appropriate to their roles. For instance, when a user, such as a student or faculty member, logs in, the system must verify their identity and grant appropriate access based on their role. This process requires secure authentication mechanisms, user data retrieval, and access verification to maintain system security and functionality.

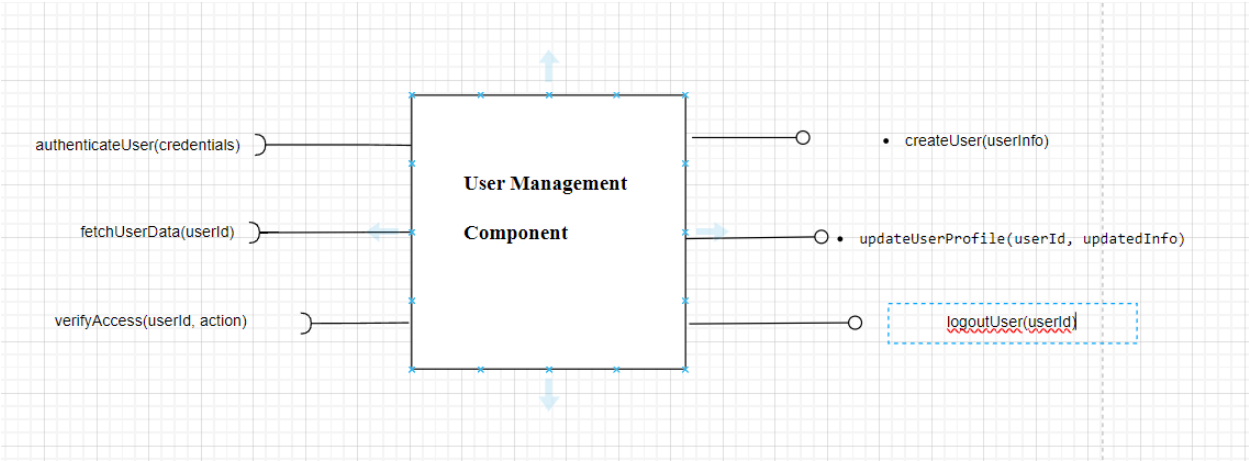


Figure 1.1: Component-Based Diagram for User Management Component.

b. Course Catalog Component

The Course Catalog Component is the backbone of the course selection process. It provides users with the necessary information to make informed decisions about course enrollment. The component must efficiently manage and present course details, prerequisites, and availability. This functionality is critical for students who need to select courses that fit their academic plans.

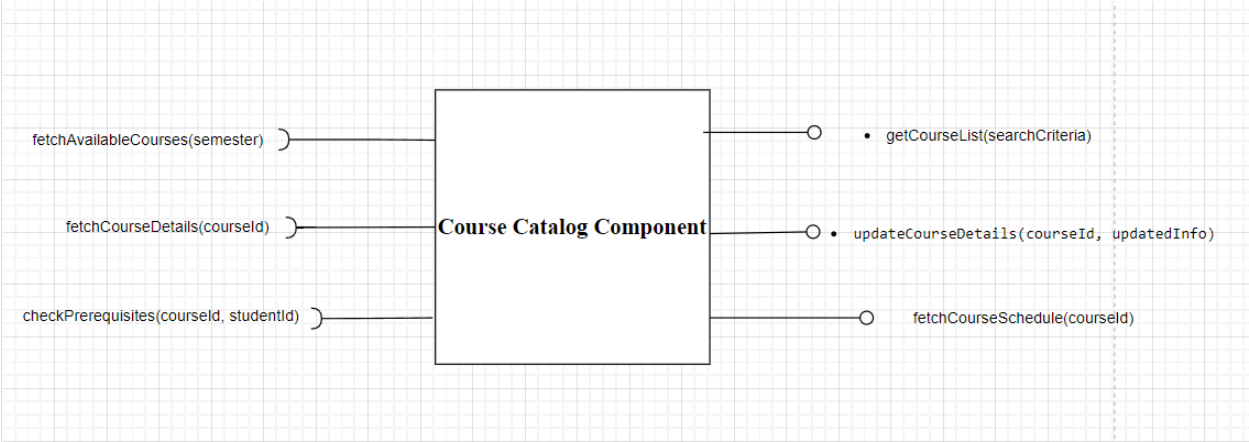


Figure 1.2: Component-Based Diagram for Course Catalog Component.

c. Reporting Component

The reporting component is vital for providing insights into system usage, academic performance, and course popularity. It gathers data from various modules and presents it in a format that supports decision-making by university administrators. The component must be capable of generating detailed reports that cover everything from enrollment trends to individual student performance, ensuring that the institution can respond to emerging needs and maintain high academic standards.

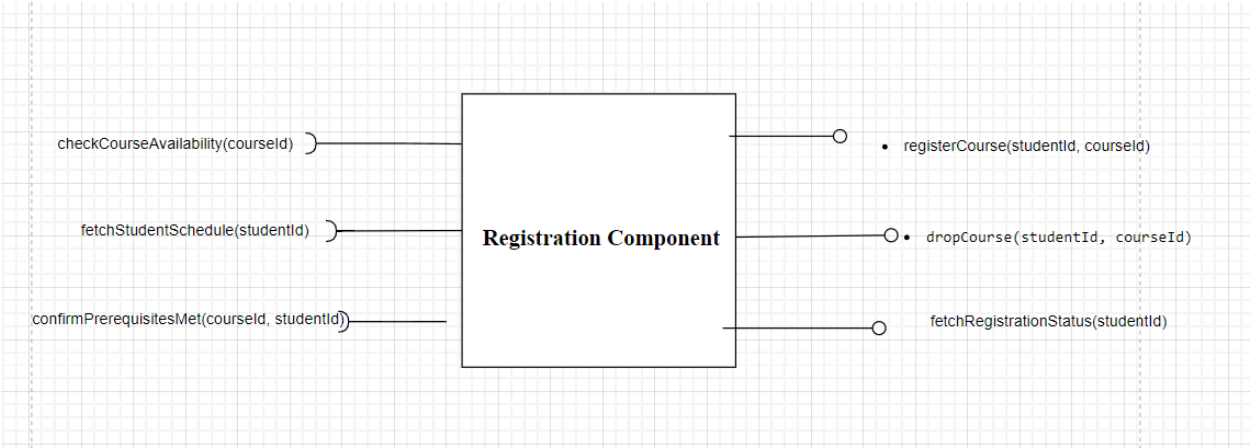


Figure 1.3: Component-Based Diagram for Reporting Component.

8. SYSTEM TESTING STRATEGY

1. **Development Testing:**
    a. **Unit Testing:** Unit testing is the process of testing at the individual methods and the individual object classes level. A unit is an assembly of individual functions or methods within an object. It is also important for constructing the test design when a test is to be made on object classes to cover all the features of the object. It should execute all operations related to the object; set/get the value of all the attributes related to the object; and bring the object to every possible state.

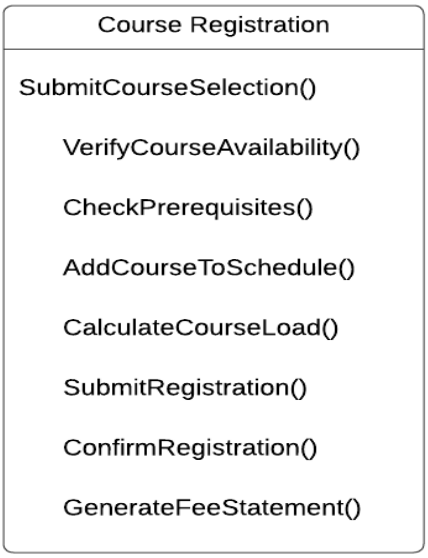


Figure 2. 0: Unit Testing Diagram

Use case testing: due to this characteristic of real-world interaction, use case-based testing is efficiently appropriate for system testing. Typically, the use of components or objects in a system contains multiple minute elements of a use case. Because it requires the use of the application to get to the intended testing of the use case, these interactions are forced. If you have drawn a sequence diagram while modeling the use case implementation then you can determine the objects or components that contribute to the interaction. m. This is illustrated in the sequence diagram (Figure 1. 4: Sequence Diagram) where one can pinpoint the operations that will be tested and also help in developing the test cases that will enable the execution of the tests. Therefore issuing a request for a report will result in the execution of the following thread of methods, therefore issuing a request for a report will result in the execution of the following thread of methods.

b. Test-Driven Development

Test driven development TDD is a development process whereby testing and developmental processes are done concurrently. When establishing the code for the proposed microfinance system, various tests will be carried out on every increment of the implementation. Bear in mind that the next increment will not be developed as soon as possible, if the corresponding code is not tested in all possible variants and outcomes.

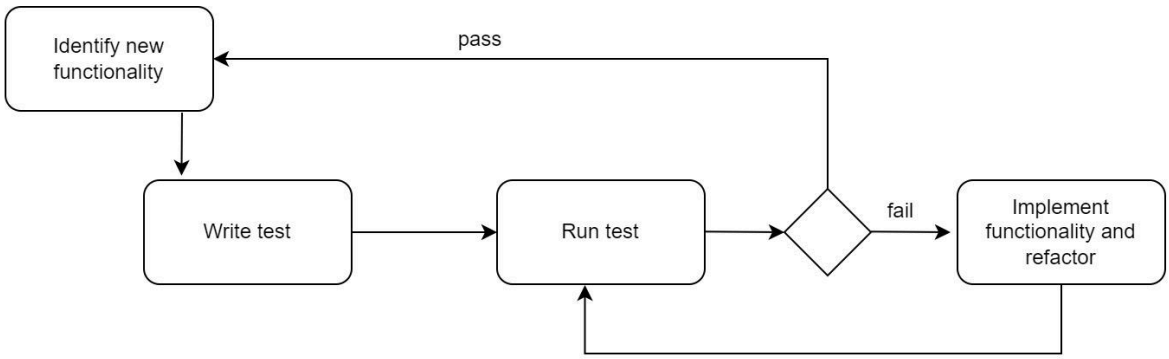


Figure 2. 1:Test driven development process activities

Below is the basic procedure of TDD:

1. Next we identify and ascertain the extent of functionality needed.

2. For this functionality, we build an automated test.

3. We then the give test This is in an effort to evaluate our understanding of what we have learnt consistent with course objectives and outcomes. Otherwise if it fails, it will be done by developing the functionality and running the test again. If the test is successful we will proceed to the next part of functionality.

In the case of CCS some of the inclusion in the TDD would involve creating tests within the system such as the tests in aspects such as the registration of courses, payment for fees and even the timetable. All the functions are made in a manner where their corresponding test pass implying that the system is operating as expected from the onset. This strategy helps to identify the error at its preliminary stage thus improving reliability of the system.

c. Release Testing:

When there is a need to decide if the Centralized Course Registration System is prepared for release, then testing comes in as a way of evaluating its readiness. The process starts with identification of the specific Acceptance criteria that the system must fulfill, for instance, functional, performance and usability acceptance criteria. These criteria apply in

the planning phase which entails the testing strategy that features the scope, roles and resources, and timeliness.

Once the plan has been set in view, concrete test cases are created to investigate each functionality outlined in the acceptance criteria. For instance, analyzing the registration process could involve such cases as when a student tries to register for a full course, or logs in, and so on. These tests are performed in test beds which provide them close resemblance to the conditions in which they are used.

After all tests are conducted, the data is then analyzed thoroughly in a bid to determine the possible faults or things that need to be changed. When the system of the requirements is achieved and the testing is done, it is called ready for implementation. However if major defects are identified then the system might need to be fine tuned before retesting is done. This strenuous approach ensures that the system is optimized for use; conforms to the user expectations and has no critical defects or deficiencies when launched to the market; thus limiting the likelihood of post implementation complaints.

d. User testing

User testing implies a process through which the customers are asked to contribute with their suggestions during the system testing phase. A user's working environment plays a significant role in relating to the dependability, performance, usability, and robustness of the system.

Alpha testing is done where the users are invited in together with the development team where they use the program on the developer's premises.

Beta testing A release of the program is made available to users to enable them analyze it and in the process give feedback to the system developers.

Acceptance testing Customers analyse a system to determine whether it is fit for acceptance by system developers and implementation in the customer setting.

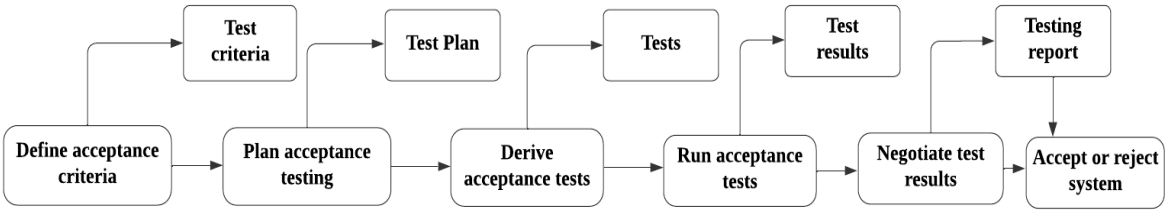


Figure 2. 1: acceptance testing process

Six stages in the acceptance testing process:

Define Acceptance needs: Setting of specific requirements for basic yet highly sensitive processes like course search, registration, and login. For instance, the system should not record any mistake with regards to registrations such as students registering for classes.

Acceptance Testing Strategy: In a detailed plan with regards to functional, usability and performance test, it is necessary to identify the scope, the participants, resources as well as the schedule when executing the tests.

Acceptance Tests: Some of the test cases can be developed straight from the acceptance criteria such as a test case of a student trying to register for a full course.

Run Acceptance Tests: Perform the tests and observe the reaction then take the record of the outcomes. Try it on as many user personas as possible, be it students or faculty members.

Negotiate Test Results: Review and report tests' results to the relevant stakeholders for consideration. It is here that one is determine whether the given system meets these criteria or not, or whether there is need to change something.

Accept or Reject System: Based on the test results, make a final decision to accept the system or identify necessary improvements.

9. ROLES AND RESPONSIBILITIES OF PROJECT MEMBERS

Name	Student ID	Rules
MD FARMAN ALI	**AIU20092185**	**Requirement Analysis & Software Process Model**
AREYAN MUHEMED ALI HUSSEIN	**AIU21102179**	**System Architecture & System Models Software Testing**
FUAD HIYBO HUSSIN MOHAMED	**AIU21102174**	**Component-Based Development**
AROUA TALHI	**AIU22102123**	**Project Manager & Intro, System Project Objectives & Software Process Model, Conclusion & Recommendation**

10. CONCLUSION AND RECOMMENDATION

The proof that this efficiency has been accomplished is the final stage of the Centralised Course Registration System (CCRS) project which successful implementation shows that it is possible to design a user-friendly environment for course registration to solve the issues that the educational institution meets. Some key functions consist of current course enrollment, course timetable management, personalized courses, and course plans as well as academic achievement tracking, which makes task easier between students and staffs.

In generating a system that was developed repeatedly based on the user feedback, the use of Incremental Development paradigm was significant. With this method developers were able to introduce incremental innovations concerning the functionality and usability of the applications, as well as their overall performance. In addition, the use of a layered architecture made the system over improbably flexible and scalable to handle future conditions and to introduce other new functionality.

This work reveals that the CCRS is a great improvement to the manner in which course registration is done in the academic circles because: For the similar reasons, the security measures are highly impressive in the system and that it is laid down for compatibility with existing academic systems and scalability in the context of promoting long term institutional development and to address the challenges posed by technology. Last of all, the use of the CCRS facilitates improvement of everyone's course management efficacy, safety, and reliability.

RECOMMENDATION

As it has been mentioned earlier, there are seven modifications that can enhance the performance of CCRS. This would entail identifying technical aspects of the current interface that are complex to users and trying to redesign them to make the interface easier to use especially for the non-technical computer users; thereby increasing user happiness and interaction. It would potentially be useful to the students and the administrators because it would foster updated data analysis in a learning institution, for instance, by choosing courses and monitoring academic outcomes. The option to better adapt the system for mobile devices would add more convenience and the integration with other LMSs would expand the scope of the system's use.

Security is another area for sustained reinvention with frequent changes of encryption the security measures and the compliance with different regulations to guard information that is private. The following are administrative operations that would benefit from a current switch to efficiency: Scheduling of classes, conflicts arising from the calendar and enrollment changes among others. Thus, making certain that the system will continuously remain scalable for use as institutions expand will be another crucial way of sustaining the viability and effectiveness of this system in the long run. These would further improve the CCRS and therefore would be of immense benefit in the student and administration systems.

11. REFERENCE

1. Singh, R., Singh, R., Kaur, H., & Gupta, O. P. (2016, June 20). *Development of online student course registration system*. Oriental Journal of Computer Science and Technology. https://www.computerscijournal.org/vol9no2/development-of-online-student-course-registration-system/
2. Chung, L., Nixon, B. A., Yu, E., & Mylopoulos, J. (2012). *Non-functional requirements in software engineering* (Vol. 5). Springer Science & Business Media.
3. Kabir, M. (2022). *Web-Based Student Registration and Exam Form Fill-up Management System for Educational Institute*. ResearchGate. https://www.researchgate.net/publication/360118139_Web_Based_Student_Registration_and_Exam_Form_Fill-up_Management_System_for_Educational_Institut
4. Slyman, M. M. A. (n.d.). *Course Registration and Result Processing System*. Retrieved from https://d1wqtxts1xzle7.cloudfront.net/56190588/slyman.pdf
5. Masoumi, R. (2000). *Web-based student registration system* (Master's thesis, Concordia University). Spectrum Research Repository. https://spectrum.library.concordia.ca/id/eprint/879/1/MQ43671.pdf
6. Ekpenyong, M. E. (2008). A real-time IKBS for students' results computation. *International Journal of Physical Sciences (Ultra Scientist of Physical Sciences)*, 20(3M), 2.
7. Team, A. (2023) *Functional and non-functional requirements: Specification an*, *AltexSoft*. Available at: https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/ (Accessed: 28 January 2024).
8. Grandhiprasuna. (2021, December 26). *Course Registration System-Problem Statement, SRS,ERD, DFD, Structured Chart* [Slide show]. SlideShare. https://www.slideshare.net/slideshow/course-registration-systemsrs-dfd-structured-chart/25090
9. Author(s). (Year). *Title of the Paper*. Journal/Publisher, URL.

10. Prem, D. (2023, August 30). *Academic Student course registration system*. https://www.linkedin.com/pulse/academic-student-course-registration-system-dasari-prem/
11. Seyam, H. (2019, December 11). *Course registration system* [Slide show]. SlideShare. https://www.slideshare.net/slideshow/course-registration-system-204518411/204518411