**Edu Unity Desk**

****

**By:**

**Sami Ullah**

**23799**

**Danish Rehman**

**22628**

**Muhammad Ikram Aziz**

**2419**

**Supervised by:**

**Dr Jawaid Iqbal**

**Faculty of computing**

**Riphah International University, Islamabad**

**Spring 2024**

**A Dissertation Submitted To**

**Faculty of Computing,**

**Riphah International University, Islamabad**

**As a Partial Fulfillment of the Requirement for the Award of**

**the Degree of**

**Bachelor of Science in Software Engineering**

**Faculty of Computing,**

**Riphah International University, Islamabad**

Date: 15-4-2024

**Final Approval**

This is to certify that we have read the report submitted by ***Sami Ullah (23799), Danish Rehman (22628), Muhammad Ikram Aziz (2419),*** for the partial fulfillment of the requirements for the degree of the Bachelor of Science in Software Engineering (BSSE). It is our judgment that this report is of sufficient standard to warrant its acceptance by Riphah International University, Islamabad for the degree of Bachelor of Science in Software Engineering (BSSE).

**Committee:**

**1**

Dr. Jawaid Iqbal

(Supervisor)

**2**

Dr. Musharraf Ahmad

(Head of Department/chairman)

**Declaration**

We hereby declare that this document **“Edu Unity Desk’’** neither as a whole nor as a part has been copied out from any source. It is further declared that we have done this project with the accompanied report entirely on the basis of our personal efforts, under the proficient guidance of our teachers especially our supervisor **Dr. Jawaid Iqbal**. If any part of the system is proved to be copied out from any source or found to be reproduction of any project from anywhere else, we shall stand by the consequences.

**Sami Ullah**

**23799**

**Danish Rehman**

**22628**

**Muhammad Ikram Aziz**

**2419**

**Dedication**

We dedicate this project to Allah Almighty our creator, our strong pillar, our source of inspiration, wisdom, knowledge and understanding. He has been the source of our strength throughout this program. Also, we dedicate our work to our family, friends and teachers. The unrivalled encouragement from our parents and outstanding support from teachers is what lead to success of this project. We also dedicate our work to our supervisor **Dr. Jawaid Iqbal** and the faculty members.

**Acknowledgement**

First of all, we are obliged to Allah Almighty the Merciful, the Beneficent and the source of all Knowledge, for granting us the courage and knowledge to complete this Project. We are grateful to our supervisor Dr. Jawaid Iqbal for his enthusiasm, patience, insightful comments, helpful information, practical advice, and unceasing ideas that always helped us tremendously in our project. Without his support and guidance, this project would not have been possible. Also, a special thanks to the rest of the faculty members for their unconditional support.

**Sami Ullah**

**23799**

**Danish Rehman**

**22628**

**Muhammad Ikram Aziz**

**2419**

**Abstract**

Edu Unity Desk is a new digital platform that makes education better for students, teachers, and coordinators. It acts like a central place where everyone can communicate and share resources easily. Students can find study materials and track their progress, while teachers can manage their classes and connect with students efficiently. Coordinators have tools to help organize activities and share announcements. As university use more digital tools, Edu Unity Desk is becoming an important resource that helps learning and teaching. With the rising need for such platforms, Edu Unity Desk is ready to help shape the future of education.

**Table of Contents**

|  |  |
| --- | --- |
| List of Figures | 10 |
| List of Tables | 11 |
| Chapter 1: Introduction | 12 |
| 1.1 Opportunity & Stakeholders | 13 |
| 1.2 Solution Overview | 13 |
| 1.3 Report Outline | 15 |
| Chapter 2: Literature / Market Survey | 16 |
| 2.1 Introduction | 17 |
| 2.2 Literature Review | 17 |
| 2.3 Existing System | 19 |
| 2.4 Summary | 20 |
| Chapter 3: Requirement Analysis | 21 |
| 3.1 Introduction | 22 |
| 3.2 Problem Scenarios | 22 |
| 3.3 Functional Requirements | 25 |
| 3.4 Summary | 28 |
| Chapter 4: System Design | 29 |
| 4.1 Introduction | 30 |
| 4.2 Architectural Design | 30 |
| 4.3 Detailed Design | 31 |
| 4.3.1 Use Case Diagram | 31 |
| 4.3.2 Use Case Fully Dressed Format | 32 |
| 4.3.3 Activity Diagram | 42 |
| 4.3.4 Component Diagram | 54 |
| 4.4 Database Design | 55 |
| 4.4.1 Entity Relationship Diagram | 55 |
| 4.5 Summary | 56 |
| Chapter 5: Implementation | 57 |
| 5.1 Endavour | 58 |
| 5.2 Components, Libraries, Web Services and Stubs | 65 |
| 5.3 IDE Tools and Technologies | 65 |
| 5.4 Best Practices / Coding Standards | 65 |
| 5.5 Deployment Diagram | 66 |
| 5.6 Summary | 67 |
| Chapter 6: Testing and Evaluation | 68 |
| 6.1 Introduction | 69 |
| 6.2 List of Test Scenarios | 69 |
| 6.3 Summary |  |
| References |  |

**List of Figures**

|  |  |
| --- | --- |
| Functional Requirements  Architectural Design |  |
| Use Case Diagram |  |
| Activity Diagram |  |
| Component Diagram |  |
| Deployment Diagram |  |

**List of Tables**

|  |  |
| --- | --- |
| Existing Systems |  |
| Problem Scenarios |  |
| Functional Requirements |  |
| Use Case Fully Dressed Format |  |
| Roles and Responsibility Matrix |  |
| List of Table Scenarios |  |

**Chapter 1:**

**Introduction**

**Chapter 1:**

**Introduction**

Edu Unity Desk is a platform dedicated to improving the educational experience by bringing students, teachers, and staff together. We focus on making communication easy and helping everyone work as a team. Our goal is to create a friendly and supportive place for learning, where the students and teacher communicate easily. With Edu Unity Desk, we’re excited to connect people in education and provide the tools needed for success.

* 1. **Opportunity and Stakeholders**

Students are the main focus of the education system, as they need easy and organized ways to manage their studies successfully. Teachers, who share their knowledge, need simple ways to show their skills, share course details, and connect with students. Coordinators are also very important, they help organization run smoothly and need clear ways to communicate and work well with everyone.

* + 1. **Stakeholders**
* Students
* Teachers
* Co-Ordination
  1. **Solution Overview**

Edu Unity Desk offers a simple and helpful platform that makes it easy for students, teachers, and coordinators to connect and work together. For students, Edu Unity Desk provides a clear and organized way to manage their studies, helping them find the resources and information they need quickly. Teachers can easily share their knowledge, upload course information, and interact with students in a friendly environment. Coordinators also benefit from this platform, as it provides smooth communication channels to help keep everything running well. Overall, Edu Unity Desk brings everyone together in one place, making education easier and more effective for all.

* + 1. **Project Scope**

The Edu Unity Desk project will create a simple platform to help students, teachers, and coordinators connect and work together more easily. Here’s what the project will focus on:

* + - 1. **Student Module:**
* Access study materials, assignments, and resources easily.
* Keep track of their progress and due dates.
* Communicate with teachers for help and clarification.
* Participate in discussions and activities.
* Stay informed with updates and announcements.
  + - 1. **Teacher Module**
* Share course materials and resources with students.
* Post updates and important information about classes.
* Communicate directly with students for guidance and support.
* Track student progress and provide feedback.
* Organize and manage classes effectively.
  + - 1. **Co-Ordination Module**
* Manage schedules, events, and school activities.
* Post announcements and important updates for everyone.
* Facilitate communication between students, teachers, and staff
* Students can address queries related to academic
* Students can see the coordinator feedback.
  + - 1. **Course Module**
* There will be course detail
* Course relation with teacher.
* Students can save favorite courses.
  + - 1. **Notice Board**
* A notice board is a central place for important announcements.
* It keeps everyone updated with the latest information.
* It helps share information quickly with all stakeholders.

* 1. **Report Outline**

This report covers the detail of all aspects of the system, for understanding and clarity. This report has been divided into six chapters.

* + 1. **Chapter 1**

This chapter introduces our system by the needs and technical issues addressed by the system. Goals, objectives, and methodologies used for the development of the proposed solution.

* + 1. **Chapter 2**

This chapter focuses on the current situation of the market, and how the system is unique and useful for the users.

* + 1. **Chapter 3**

This chapter is all about need analysis as it examines the real requirements and problem scenarios addressed by the developed system. It also identifies the real user of the system and those affected by the system.

* + 1. **Chapter 4**

This chapter provides all the information related to design factor of the developed system by describing the system architecture design consideration and different diagrams that model the working behavior of the system.

* + 1. **Chapter 5**

This chapter provides information about the environment in which the developed system was used and evaluated for the performance, usability, and other related consideration.

**Chapter 2:**

**Literature / Market Survey**

**Chapter 2:**

**Literature / Market Survey**

* 1. **Introduction**

In this chapter, we are discussing the literature/market survey of our project “Edu Unity Desk”. For our project, we have used interviewing technique to gather the functional requirements of the project. Further, we have created a table in which we have compared the features of the similar existing systems of our project.

* 1. **Literature Review / Market Survey**
     1. **The Interview**

For this project, we used interviews to gather information for Edu Unity Desk. The interview helped us understand what the project is about, its goals, and the problems it wants to solve.

The following portion of this section describes in detail, the interview process which was conducted for requirements elicitation:

* + 1. **Client Profile**
* Name of Organization: Edu Unity Desk
* Industry: Educational Technology
* What are the key responsibilities of your organization?
* What services do you provide?
* For whom are these services?
* How is the success of your services measured?
* Which problems interfere with your success?
  + 1. **Problem Details**
* Why does this problem exist?
* How is the problem currently being solved, if solved?
* How would you like to get the problem solved?
  + 1. **Understanding the User Environment**
* Who are the users of this educational platform?
* What are their geographical locations?
* Do they have any previous experience with similar educational platforms?
* Do you plan to expand the system to more platforms in the future?
  + 1. **Usability of the System**
* Are there any other systems that are to be integrated with this system?
* How usable should the website be for educational activities?
* Do you expect training to help users become familiar with the system?
  + 1. **Analyst’s Input on the Client’s Problem**
* What causes the problems you currently face?
* How is the problem currently being solved?
* What would you like the solution to be?
* How would you rank each problem when compared with others?
  + 1. **Assessing the Solution**
* What if you could have a mobile application for the educational platform?
* Have cloud data storage for everything on the platform?
* How would you rank the importance of these features?
  + 1. **Assessing the Opportunity**
* Who is this project’s targeted user?
* How many users will use it?
* On what basis would you find the solution successful?
  + 1. **Assessing Reliability, Support and Performance Needs**
* How reliable do you expect the system to be?
* What are your expectations for the performance of the system?
* Is there any special licensing that is needed for this type of system?
* What kind of maintenance, if any, do you want for the system?
  + 1. **Legal Limitations**
* Are there any legal and regulatory limitations or standards that we must observe when building the system?  
  1. **Existing Systems**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| System Name | Teacher | Student | Chatting | Courses | Saved Courses | Notice Board | Queries | Voucher Number | Query Status | Total Query | Pending Query |
| Edu Smart | No | Yes | No | Yes | No | Yes | Yes | No | No | No | No |
| Vdesk | No | No | No | No | No | No | Yes | No | No | No | No |
| Moellim | Yes | Yes | Yes | Yes | Yes | No | No | No | No | No | No |
| Edu Pluse | Yes | Yes | No | Yes | No | Yes | No | No | No | No | No |
| UOP Help Desk | No | No | No | No | No | No | Yes | Yes | Yes | No | No |
| Smart Edu | Yes | Yes | No | Yes | No | Yes | Yes | No | No | No | No |

Table 2.3.1 | Existing systems

* 1. **Summary**

From our discussion, we see that there is no such system which we mention above that helps students, teachers, and staff work together easily, as current platforms lack necessary features. That’s why we created “Edu Unity Desk.” This platform aims to improve communication and teamwork in University by offering a notice board for updates, a tool for sharing study materials, and a messaging system for easy communication. Students can ask questions and connect with teachers, while teachers can share lessons and important information. Our goal is to create a friendly learning environment where everyone can work together and succeed.

**Chapter 3:**

**Requirement Analysis**

**Chapter 3:**

**Requirement Analysis**

* 1. **Introduction**

In this chapter, we will discuss the Functional & Non-Functional requirements of our project “Edu Unity Desk”. Prior to that, we will discuss all the problem statements we have found while doing research on the project idea. These functional requirements are gathered from the client using interviewing. The Non-Functional requirements are gathered by observing the functional requirements.

* 1. **Problem Scenarios**

|  |  |
| --- | --- |
| ***Problem Statement – 1*** | |
| **The Problem Of** | Communication is scattered |
| **Affects** | Coordination, Students |
| **The Result of Which** | Improved communication between students and coordinators, creating stronger teamwork and more efficient processes |
| **Benefits Of** | Higher productivity, less frustration, and a more connected learning environment |

Table 3.2.1 | Problem Statement 1

|  |  |
| --- | --- |
| ***Problem Statement – 2*** | |
| **The Problem Of** | Course information is spread out in different places |
| **Affects** | Teachers, Students |
| **The Result of Which** | Centralized course management and easy access to all course information |
| **Benefits Of** | Improved organization, easy-to-find information, and a smoother learning experience, leading to better academic performance for students |

Table 3.2.2 | Problem Statement 2

|  |  |
| --- | --- |
| ***Problem Statement – 3*** | |
| **The Problem Of** | Limited visibility of teachers' skills and expertise |
| **Affects** | Teachers, Students |
| **The Result of Which** | Clear teacher profiles showing qualifications, expertise, and availability |
| **Benefits Of** | Stronger teacher-student connections, better learning support for students, and higher satisfaction with the overall educational experience |

Table 3.2.3 | Problem Statement 3

|  |  |
| --- | --- |
| ***Problem Statement – 4*** | |
| **The Problem Of** | Notices are not shared efficiently |
| **Affects** | Students |
| **The Result of Which** | A centralized notice board for sharing information quickly with everyone |
| **Benefits Of** | Increased awareness, fewer missed opportunities, and better organization across the University community |

Table 3.2.4 | Problem Statement 4

|  |  |
| --- | --- |
| ***Problem Statement – 5*** | |
| **The Problem Of** | Managing multiple websites is difficult |
| **Affects** | Teachers, Students, Coordinators. |
| **The Result of Which** | Combining all functions into one platform to reduce confusion and save time. |
| **Benefits Of** | Easier user experience, smoother workflows, and higher satisfaction for everyone |

Table 3.2.5 | Problem Statement 5

* 1. **Functional Requirements**

**User Authentication and Authorization**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 1 | User Authentication and Authorization | 1.1 | User Login | This ensures that each user has a secure account and can use the platform according to their specific role. |

Table 3.3.1 | FR1– User Login

**Course**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 2 | Course | 2.1 | Course Details | This feature will provide students with information about available courses, including descriptions, schedules, and requirements. |

Table 3.3.2 | FR2– Course Details

**Teacher profile Management**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 3 | Teacher profile Management | 3.1 | Profile Creation | This helps students know more about their teachers and how to contact them. |

Table 3.3.3 | FR3– Profile Creation

**Notice Board**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 4 | Notice Board | 4.1 | Centralized Announcements | This keeps everyone informed about important updates in one easy-to-find location. |

Table 3.3.4 | FR4– Notice Board

**Save Favorite Courses**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 5 | Save Favorite Courses | 4.1 | Save Courses | This allows students to quickly access the courses they are interested in without having to search for them again. |

Table 3.3.5 | FR5– Save Courses

**Feedback and Evaluation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 6 | Feedback and Evaluation | 6.1 | Course Feedback | This helps improve courses by allowing students to share their opinions and get answers to their queries. |

Table 3.3.6 | FR6– Course Feedback

**Communication with Coordination**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 7 | Communication with Coordination | 7.1 | Student Queries | This ensures students can easily reach out for help or clarification on important issues. |

Table 3.3.7 | FR7– Student Queries

**Edit Profile**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 8 | Edit Profile | 8.1 | Edit Account Details | Teachers and Students can edit their profile. |

Table 3.3.8 | FR8– Edit Profile

**View Profile**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No#** | **Functional Requirement** | **Breakdown** | | **Description** |
| **ID** | **Sub-Functionality** |
| 9 | View Profile | 9.1 | View Profile Details | Students can view their Teachers profile |

Table 3.3.9 | FR9– View Profile

* 1. **Summary**

The functional requirements for ‘’Edu Unity Desk’’ include secure user login, detailed course information, teacher profile management, a centralized notice board, and the ability for students to save favorite courses. Additionally, it provides mechanisms for feedback and communication between students and coordinators, enhancing the overall educational experience.

**Chapter 4:**

**System Design**

**Chapter 4:**

**System Design**

* 1. **Introduction**

The system designer is responsible for defining the roles, properties, functions, and connections of different parts of the software. They also decide how each part should be adjusted to fit the environment where it will work. The design is based on the needs gathered from users. In this chapter, we will show all the design diagrams for our project “Edu Unity Desk,” including architectural design, use case design, and activity design. These diagrams will help explain the workflow of the system.

* 1. **Architecture Design**

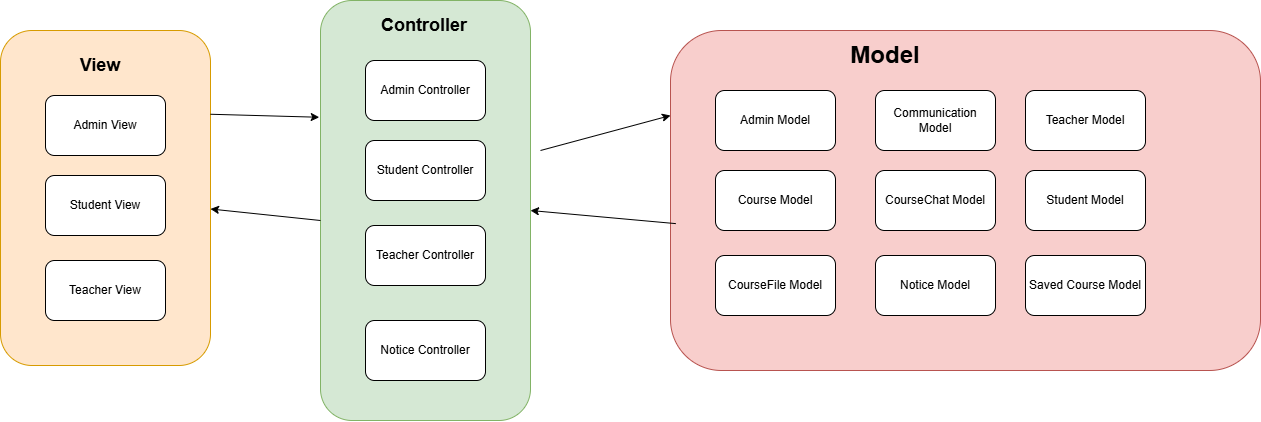
****

Fig 4.2.1 | Architecture Design

* 1. **Detailed Design**

1. **Use case Diagram**

****

Fig 4.3.1 | Use Case Diagram

1. **Use Case Fully Dressed Format**
   1. **Add Teacher**

|  |  |
| --- | --- |
| Use case ID: | UC-001 |
| Use case: | Add Teacher |
| Actors: | Admin, Teacher |
| Preconditions: | Admin is an authorized user with login access to the system. |
| Basic Flow | |
| System Response:   1. **System** displays login page. 2. **System** shows dashboard. 3. **System** displays Teacher Module options. 4. **System** displays the Add Teacher form. 5. **System** validates the information entered. 6. **System** confirms the successful addition of the new teacher by showing a success message. | Actor Actions:   1. Admin logs into the system. 2. Admin navigates to the Teacher Module section. 3. Admin clicks on Add New Teacher. 4. Admin enters the following teacher details: 5. Admin submits the form. |
| Alternative Flow: | 1. If the Admin enters invalid data (e.g., incorrect email format or duplicate email):   6.a.1 System prompts Admin to correct the errors before submission. |
| Post Condition: | The new teacher’s information is successfully stored in the system, making it accessible for further actions. |

Table 4.3.2.1 | UC-1 Add Teacher

* 1. **Update Teacher**

|  |  |
| --- | --- |
| Use case ID: | UC-002 |
| Use case: | Update Teacher |
| Actors: | Admin, Teacher |
| Preconditions: | Admin is an authorized user with login access to the system. |
| Basic Flow | |
| System Response:   1. System displays the main dashboard. 2. **System** displays options within the Teacher Module. 3. System displays the Add Teacher" form with fields to enter teacher details. 4. System validates the entered data and saves the teacher's information if all data is valid. 5. System displays a success message to confirm the addition of the new teacher. | Actor Actions:   1. **Admin** logs into the system. 2. **Admin** navigates to the Teacher Module section. 3. **Admin** clicks on Add New Teacher. 4. **Admin** submits the form. |
| Alternative Flow: | 4.a If the Admin enters invalid data (e.g., incorrect email format or duplicate email):  4.a.1. System prompts Admin to correct errors before submission. |
| Post Condition: | The new teacher’s information is successfully saved in the system. |

Table 4.3.2.2 | UC-2 Update Teacher

* 1. **Add Student**

|  |  |
| --- | --- |
| Use case ID: | UC-003 |
| Use case: | Add Student |
| Actors: | Admin, Student |
| Preconditions: | The Admin and Teacher are authorized users with access to the system.  The Teacher record to be updated exists in the system. |
| Basic Flow | |
| System Response:   1. System displays the main dashboard. 2. **System** shows options within the Student Module. 3. **System** displays a form to add a new student. 4. **System** provides fields for entering student details. 5. **System** validates the entered information. 6. **System** saves the new student's information if all data is valid. 7. **System** displays a success message confirming the student was added. | Actor Actions:   1. **Admin** logs into the system. 2. Admin navigates to the Student Module. 3. Admin clicks on Add New Student. 4. Admin enters the following details: 5. Admin submits the form. |
| Alternative Flow: | 4.a If the Admin enters an invalid email format:  4.a.1 System prompts the Admin to correct the email format before submission. |
| Post Condition: | The new student’s information is successfully saved in the system and can access relevant modules, interact with teachers, and engage with the coordination module. |

Table 4.3.2.3 | UC-3 Add Student

* 1. **Update Student**

|  |  |
| --- | --- |
| Use case ID: | UC-003 |
| Use case: | Update Student |
| Actors: | Admin, Student |
| Preconditions: | The Admin and Student are authorized users with access to the system.  The student record to be updated exists in the system. |
| Basic Flow | |
| System Response:   1. System displays the main dashboard. 2. **System** displays options within the Student Module. 3. **System** shows the student's profile page. 4. **System** allows updates in some fields 5. **System** validates the updated information 6. **System** saves the updated information if all data is valid. 7. **System** displays a success message confirming the student’s profile has been updated. | Actor Actions:   1. **Admin** logs into the system. 2. **Admin** navigates to the Student Module section. 3. Admin go to the student’s profile. 4. Admin clicks on Update" on the student’s profile. 5. Admin updates necessary fields 6. Admin submits the form. |
| Alternative Flow: | 6.a If the Admin enters invalid data (e.g., incorrect email format):  6.a.1 System prompts the Admin to correct errors before submission. |
| Post Condition: | The updated student information is successfully saved in the system.  The updated details are now available for future reference and interactions within the system. |

Table 4.3.2.4 | UC-4 Update Student

* 1. **Communication with Coordination**

|  |  |
| --- | --- |
| Use case ID: | UC-005 |
| Use case: | Communication with Coordination |
| Actors: | Admin, Students |
| Preconditions: | Student is an authorized user with login access to the system. |
| Basic Flow | |
| System Response:   1. **System** displays the main dashboard. 2. **System** shows options for available modules. 3. **System** displays options within the module. 4. **System** automatically generates a voucher number for tracking the query. 5. **System** displays a form for composing a message. 6. **System** saves the message and notifies the coordination team about the new query. 7. **System** displays a success message confirming that the query has been sent. | Actor Actions:   1. Student logs into the system. 2. Student navigates to the Communication with Coordination module. 3. Student clicks on New Message. 4. Student compose the message. 5. Student reviews and clicks on submit query. |
| Post Condition: | The query is successfully logged in the system.  The coordination team is notified and can respond to the student's message. |

Table 4.3.2.5 | UC-5 Communication with Coordination

* 1. **Add Course**

|  |  |
| --- | --- |
| Use case ID: | UC-006 |
| Use case: | Add Course |
| Actors: | Admin |
| Preconditions: | Admin or is an authorized user.  The course to be updated exists in the system. |
| Basic Flow | |
| System Response:   1. **System** displays available modules. 2. **System** shows options within the Course module. 3. **System** prompts the user to enter course details. 4. **System** validates the course information. 5. **System** saves the course details and displays a success message confirming the course was added. | Actor Actions:   1. Admin navigates to the Course module. 2. Admin selects the Add Course" option. 3. Admin begins entering details as prompted. 4. Admin confirms the addition of the course. |
| Alternative Flow: | 4.a If required information is missing:  4.a.1 System prompts the user to complete all fields before proceeding. |
| Post Condition: | The new course is successfully added to the platform.  Course details are now visible to students, teachers, and coordinators.  Authorized users can manage and interact with the course information. |

Table 4.3.2.6 | UC-6 Add Course

* 1. **Update Course**

|  |  |
| --- | --- |
| Use case ID: | UC-007 |
| Use case: | Update Course |
| Actors: | Admin |
| Preconditions: | Admin is an authorized user.  The course to be updated exists in the system. |
| Basic Flow | |
| System Response:   1. **System** displays available modules. 2. **System** shows options within the Course Management module. 3. **System** displays a list of existing courses. 4. **System** displays the current course details. 5. **System** provides editable fields for course details. 6. **System** validates the updated course information. 7. **System** saves the updated information and displays a success message confirming that the course details have been updated. | Actor Actions:   1. Admin navigates to the Course module. 2. Admin selects the Update Course option. 3. Admin searches for and selects the course to be updated. 4. Admin reviews the existing course information. 5. Admin updates course details as needed. 6. Admin confirms the updates. |
| Alternative Flow: | **5.a** If required information is missing:  **5.a.1** **System** prompts the user to complete all necessary fields before proceeding. Admin fills in any missing information and reattempts submission. |
| Post Condition: | The new course is successfully added to the platform.  Course details are now visible to students, teachers, and coordinators.  Authorized users can manage and interact with the course information. |

Table 4.3.2.7 | UC-7 Update Course

* 1. **Notice Board**

|  |  |
| --- | --- |
| Use case ID: | UC-008 |
| Use case: | Notice Board |
| Actors: | Admin, Teachers, Students |
| Preconditions: | The admin must be logged into the system with the appropriate permissions.  The students and teachers must also be logged in to view the notices. |
| Basic Flow | |
| Admin Actions:   1. Logging in 2. Posting messages 3. Reviewing messages 4. Managing attachments. | Student Actions:   1. Logging in 2. Viewing messages 3. Clicking on messages to view details. |
| Alternative Flow: | 4.1. If the admin does not fill in the title field, the system shows an error message stating, "Please fill in the title field."  4.2. If the admin does not fill in the content field, the system shows an error message stating, Please fill in the content field. |
| Post Condition: | The details of the posted message will be displayed on the Notice Board of the system. |

Table 4.3.2.8 | UC-8 Notice Board

* 1. **Allocate Voucher Number**

|  |  |
| --- | --- |
| Use case ID: | UC-009 |
| Use case: | Allocate Voucher Number |
| Actors: | Admin, Students |
| Preconditions: | The admin and the student must be authorized and logged into the system.  The student must post a query before a voucher number can be allocated. |
| Basic Flow | |
| Admin Actions:   1. Logging in 2. Viewing and managing queries 3. Marking queries as seen 4. Responding to queries | Student Actions:   1. Logging in 2. Posting queries 3. Viewing voucher numbers. |
| Post Condition: | The system sends the generated voucher number to the student.  The query is visible in the system with its status. |

Table 4.3.2.9 | UC-9 Allocate Voucher Number

* 1. **Favorite Course**

|  |  |
| --- | --- |
| Use case ID: | UC-010 |
| Use case: | Favorite Course |
| Actors: | Students |
| Preconditions: | The student is authenticated and logged into the system.  The student has access to the course management module.  The course is already added to the system. |
| Basic Flow | |
| System Response:   1. The system checks if the course is already in the student's favorites list. 2. If not, the system adds the selected course to the student's favorites. 3. The system updates the student's favorite courses list. 4. The system displays a confirmation message that the course has been added to the favorites. 5. The updated favorites list is now visible to the student. | Student Actions:   1. The student navigates to the Courses" module. 2. The student selects a course from the available list. 3. The student clicks Add to Favorites. |
| Post Condition: | The selected course is added to the student's list of favorite courses.  The updated favorite courses list is visible to the student. |

Table 4.3.2.10 | UC-10 Favorite Course

* + 1. **Activity diagrams**

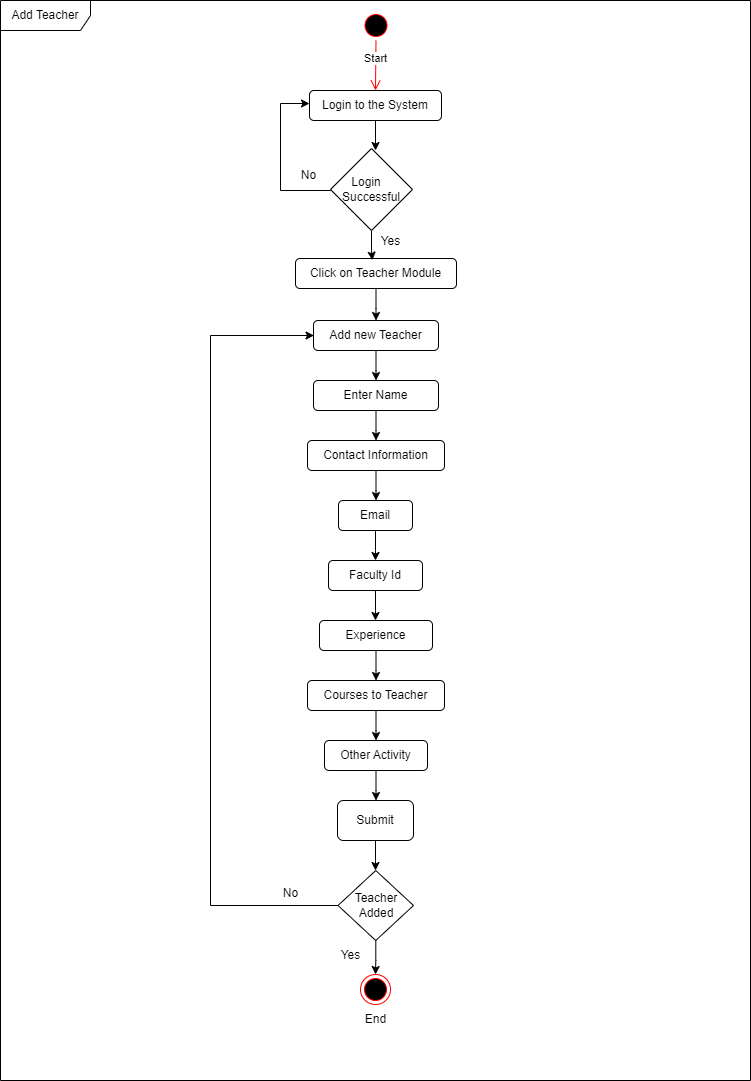
****

Fig 4.3.3.1: Add Teacher

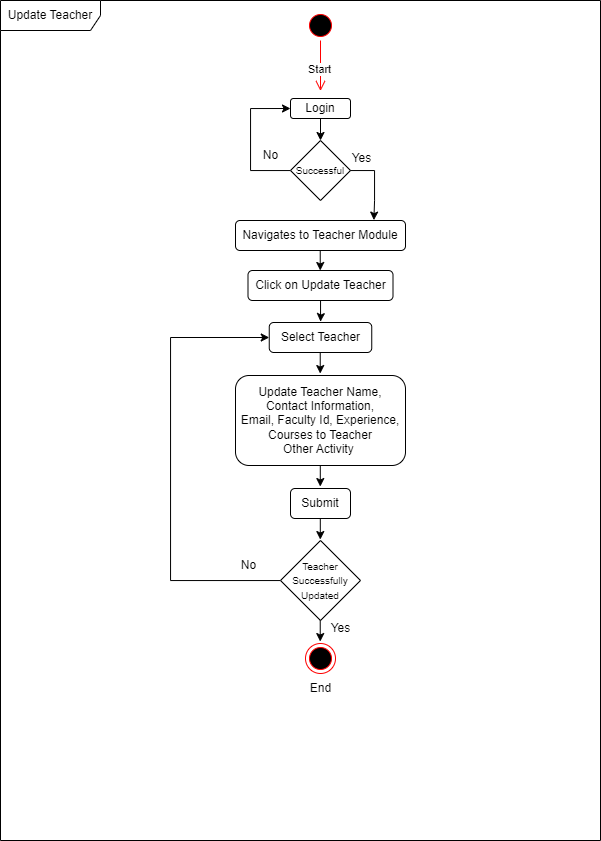
****

Fig 4.3.3.2: Update Teacher

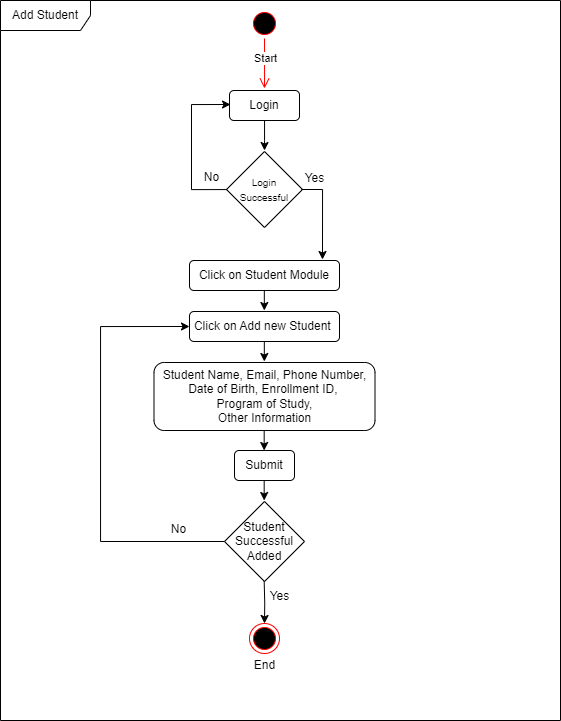


Fig 4.3.3.3: Add Student

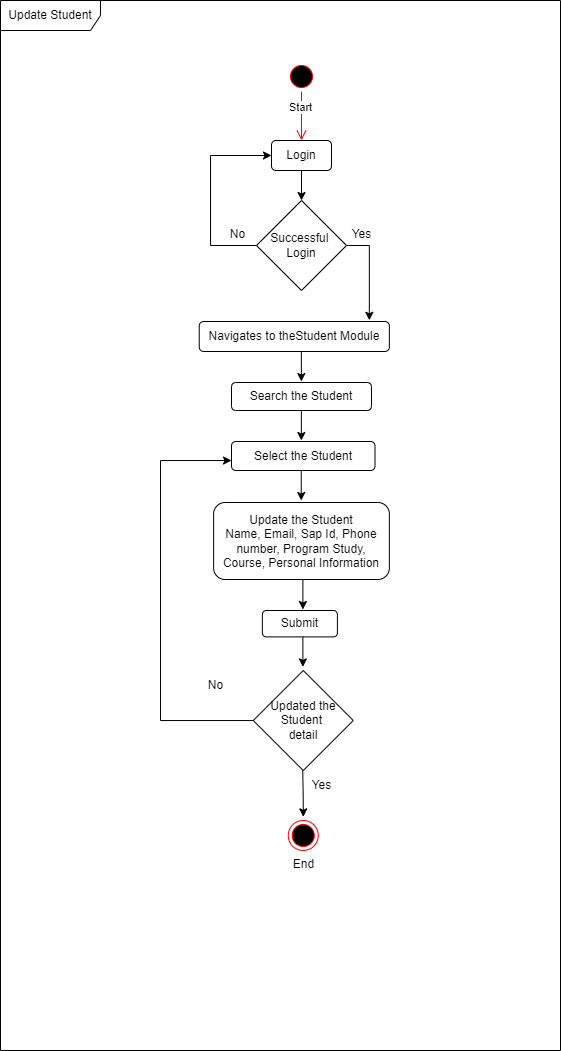
****

Fig 4.3.3.4: Update Student

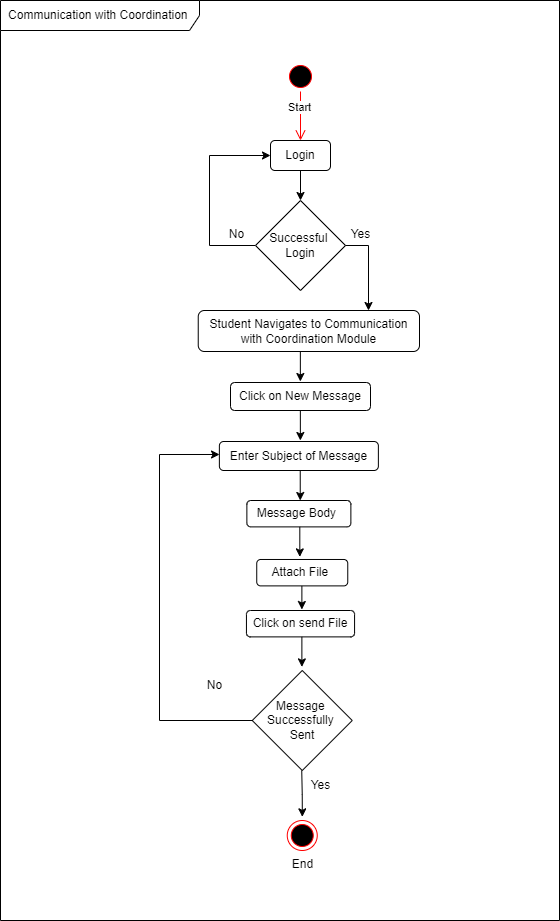
****

Fig 4.3.3.5: Communication with Coordination

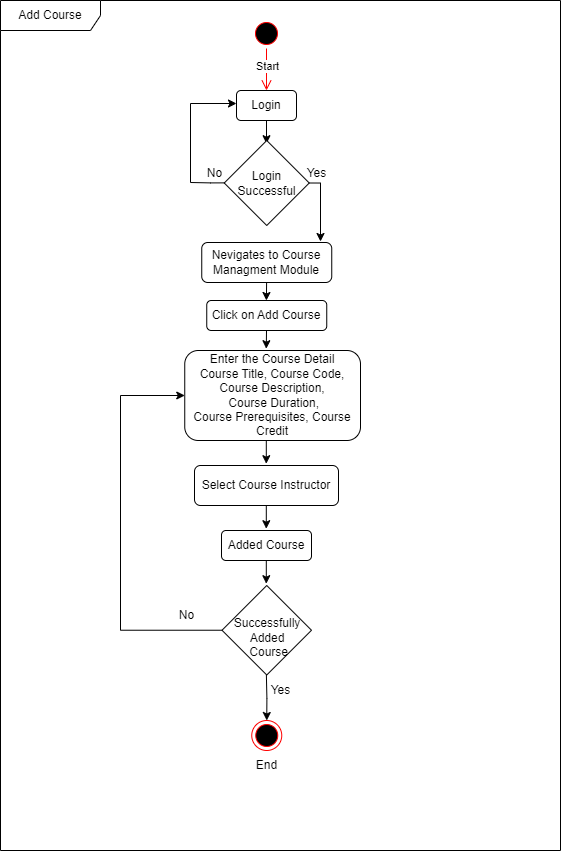
****

Fig 4.3.3.6: Add Course

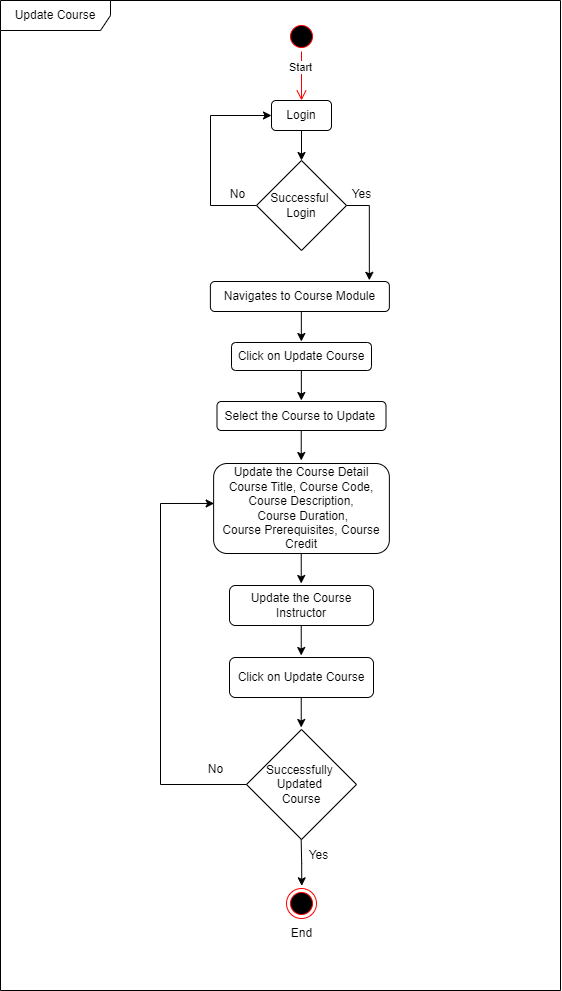
****

Fig 4.3.3.7: Update Course

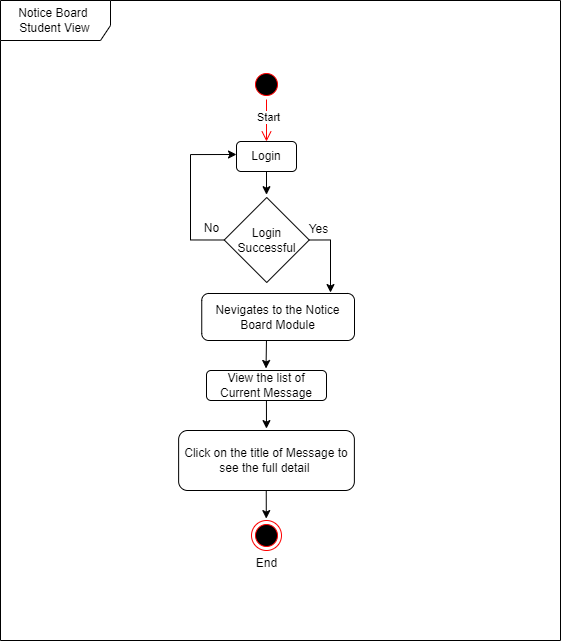


Fig 4.3.3.8: Notice Board Student View

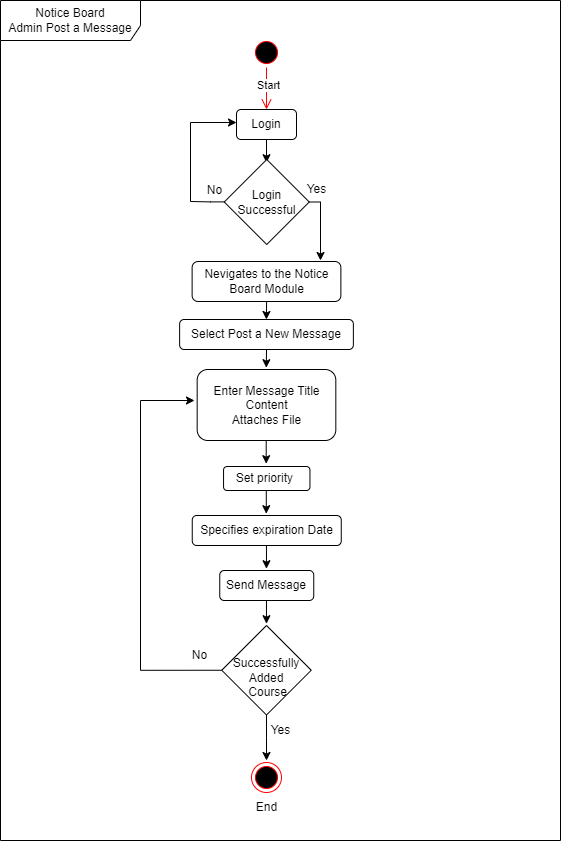
****

Fig 4.3.3.9: Notice Board Admin post a Message

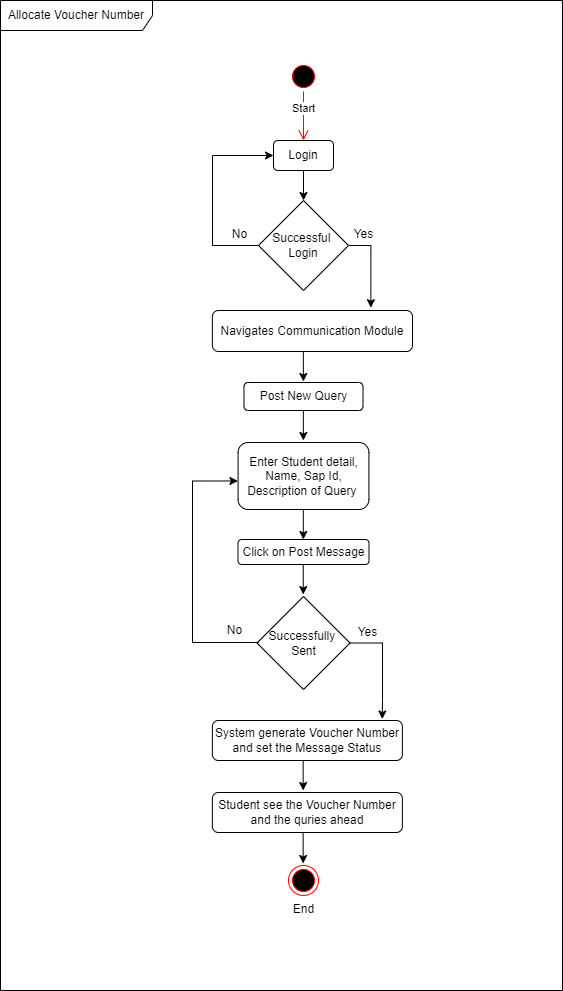
****

Fig 4.3.3.10: Allocate Voucher Number

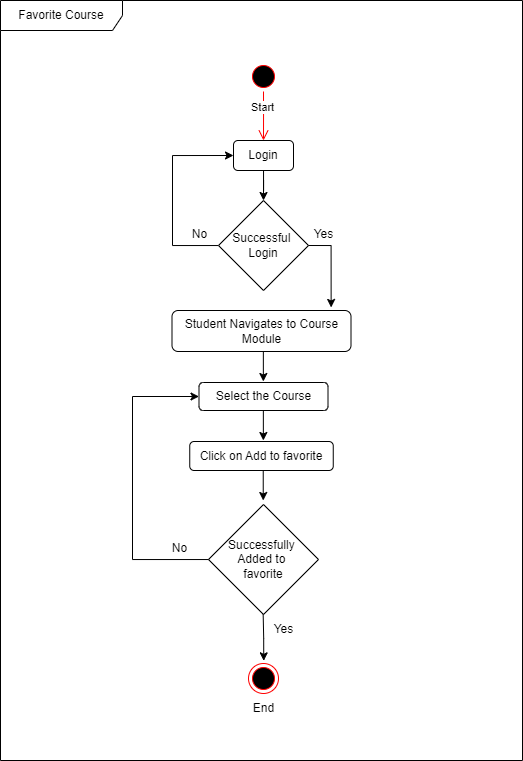
****

Fig 4.3.3.11: Favorite Course

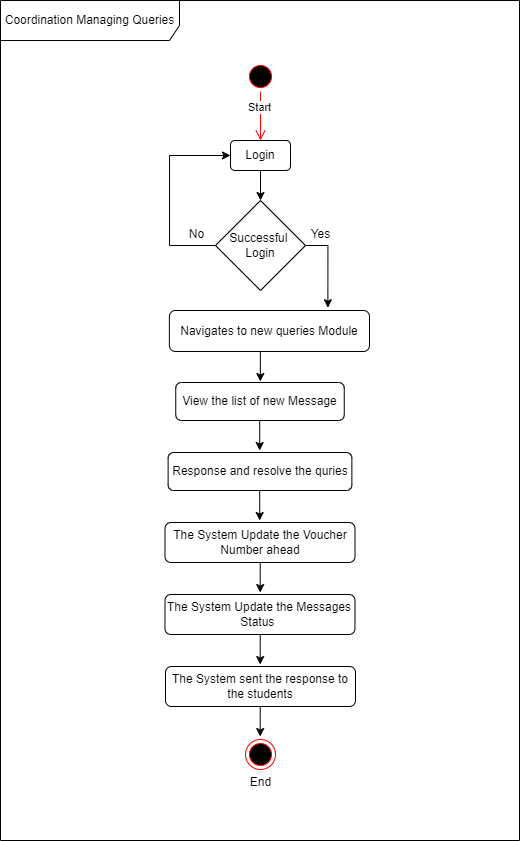
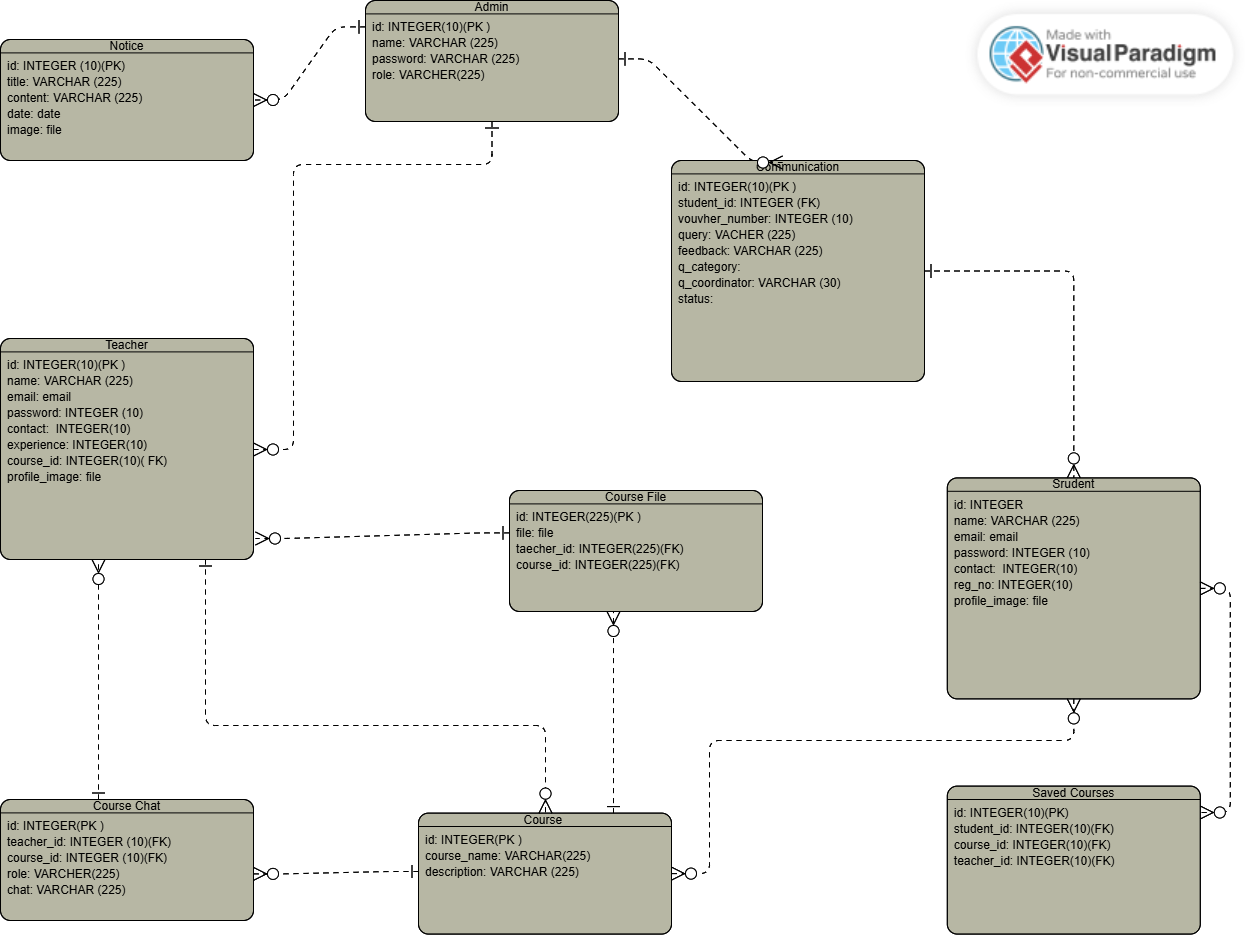
****

Fig 4.3.3.12: Coordination Managing Courses

* 1. **Database Design**
     1. **Entity Relationship Diagram**

****

* 1. **Summary**

In this chapter, we showed design the diagrams like architectural design, use case design, activity design & entity relationship diagram of our project “Edu Unity Desk”. In all the diagrams we tried to draw out the workflow and technical design of our system “Edu Unity Desk”. This system design helped us in elaborating & understanding the refined aspects of our system.

**Chapter 5:**

**Implementation**

**Chapter 5:**

**Implementation**

* 1. **Endeavour**
     1. **Team**
* Sami Ullah
* Danish Rehman
* Muhammad Ikram Aziz
  + 1. **Work Breakdown Structure**

1. **Project Management**
   1. Work Breakdown Structure
   2. Roles and Responsibility Matrix
   3. Change Control System
2. **Reports / Documentation**
   1. Team Members and Project Proposal
   2. Project Proposal Document
      1. Opportunities and Stakeholders
      2. Existing Systems
      3. Problem Statement
      4. Proposed Solution
      5. Project Scope
         1. Admin
         2. Teachers
         3. Students
      6. Technologies
   3. Proposal Plan
      1. Change Record
      2. List of faculties proposed changes
      3. Work Breakdown Structure
   4. Process Document
      1. Estimate Documentation
   5. Planning Document
      1. Problem the Software will solve
      2. The development approach the team will use
      3. The Primary Function of the Software
      4. The order of development
      5. Leadership Roles of the Project
      6. Each Team Members Responsibilities
   6. Scheduling Documentation
   7. Technical Documentation
   8. Final Documentation Introduction
   9. Literature / Market survey
      1. Surveys
      2. Interviews
      3. Literature review as section within a document
   10. Requirement Analysis
       1. Elicited Requirements
       2. Functional Requirements
       3. Stakeholder Requirements
   11. System Design
       1. Interface Design
          1. Graphical User Interface
          2. Form Based User Interface
       2. Architectural Design
       3. Use Cases
       4. Component Diagram
       5. Deployment Diagram
       6. Activity Diagrams
   12. Implementation
   13. Testing
       1. system Testing
   14. Conclusion and Outlook
3. **System**
   1. Development Environment
      1. IDE (VS Code)
      2. Server
      3. Database
   2. Website
      1. Front End
         1. Login
         2. Registration
         3. Home
         4. Communication
         5. Course Category
         6. Notice Board
         7. Post Query
         8. Students
         9. Teachers
      2. Backend
         1. Admin Panel
         2. Add, Update, Delete and View Students
         3. Add, Update, Delete and View Teachers
         4. PHP API
      3. **Roles and Responsibility Matrix**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **WBS #** | **WBS**  **Deliverable** | **Activity #** | **Activity to Complete the Deliverable** | **Duration (# of Days)** | **Responsible Team Member(s) & Role(s)** |
| 2 | Documentation | 2.1 | Team Members and Project Proposal | 1 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2 | Documentation | 2.2 | Project Proposal Document | 7 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.2 | Project  Proposal  Document | 2.2.1 | Opportunity & Stakeholders | 1 | Sami Ullah  Danish Rehman |
| 2.2 | Project  Proposal  Document | 2.2.2 | Existing Systems | 3 | Danish Rehman  Ikram Aziz |
| 2.2 | Project  Proposal  Document | 2.2.3 | Problem Statement | 1 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.2 | Project  Proposal  Document | 2.2.4 | Proposed Solution | 3 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.2 | Project  Proposal  Document | 2.2.5 | Project Scope | 1 | Sami Ullah  Danish Rehman |
| 2.3 | Proposal Plan | 2.3.1 | List of faculties  proposed changes | 1 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.3 | Proposal Plan | 2.3.2 | Work Breakdown  Structure | 2 | Sami Ullah |
| 2.5 | Planning Document | 2.5.1 | Problem the Software will solve | 2 | Sami Ullah  Danish Rehman |
| 2.5 | Planning Document | 2.5.2 | The development approach the team will use | 2 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.5 | Planning Document | 2.5.3 | The Primary  Function the Software | 1 | Danish Rehman |
| 2.5 | Planning Document | 2.5.4 | The Order of Development | 1 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 2.5 | Planning  Document | 2.5.5 | Leadership Roles  of the Project | 1 | Danish Rehman |
| 2.5 | Planning Document | 2.5.6 | Each Team Member’s  Responsibilities | 1 | Danish Rehman |
| 2 | Documentation | 2.6 | Scheduling Documentation | 1 | Danish Rehman  Ikram Aziz |
| 2 | Documentation | 2.7 | Technical Documentation | 2 | Danish Rehman  Ikram Aziz |
| 2 | Documentation | 2.8 | Final Documentation  Introduction | 1 | Danish Rehman |
| 2 | Documentation | 2.9 | Literature / Market Survey | 7 | Danish Rehman |
| 2.9 | Literature / Market Survey | 2.9.1 | Survey | 2 | Danish Rehman  Ikram Aziz |
| 2.9 | Literature / Market Survey | 2.9.2 | Interviews | 3 | Danish Rehman |
| 2.9 | Literature / Market Survey | 2.9.3 | Focus groups | 1 | Danish Rehman |
| 2.9 | Literature / Market Survey | 2.9.4 | Customer Observation | 1 | Danish Rehman  Ikram Aziz |
| 2.10 | Requirement  Analysis | 2.10.1 | Functional  Requirements | 4 | Danish Rehman |
| 2.10 | Requirement  Analysis | 2.10.3 | Stakeholder  Requirements | 1 | Danish Rehman |
| 2. | Documentation | 2.11 | System Design | 10 | Sami Ullah  Danish Rehman |
| 2.11 | System Design | 2.11.1 | Interface Design | 8 | Sami Ullah  Danish Rehman |
| 2.11.1 | Interface Design | 2.11.1.1 | Graphical User  Interface (GUI) | 4 | Sami Ullah  Danish Rehman |
| 2.11.1 | Interface Design | 2.11.1.3 | Form-Based User  Interface | 3 | Sami Ullah  Danish Rehman |
| 2.11 | System Design | 2.11.2 | Architectural  Design | 3 | Sami Ullah  Danish Rehman |
| 2.11.2 | Architectural  Design | 2.11.2.1 | Architecture  Design | 1 | Sami Ullah  Danish Rehman |
| 2.11.3 | Use Cases | 2.11.3 | Use Cases | 3 | Sami Ullah  Danish Rehman |
| 2.11.4 | Component  Diagram | 2.11.4 | Component  Diagram | 1 | Sami Ullah  Danish Rehman |
| 2.11.5 | Deployment  Diagram | 2.11.5 | Deployment  Diagram | 1 | Sami Ullah  Danish Rehman |
| 2.11.6 | Activity  Diagrams | 2.11.6 | Activity Diagrams | 7 | Sami Ullah  Danish Rehman |
| 2.12 | Implementation | 2.12.1 | Development | 40 | Sami Ullah  Danish Rehman |
| 2.13 | Testing & Performance  Evaluation | 2.13 | Testing & Performance  Evaluation | 12 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 3 | System | 3.1 | Development  Environment | 10 | Sami Ullah  Danish Rehman  Ikram Aziz |
| 3 | System | 3.2 | Website | 25 | Sami Ullah  Danish Rehman |
| 3.2 | Website | 3.2.1 | Front End | 20 | Sami Ullah  Danish Rehman |
| 3.2.1 | Frontend | 3.2.1.1.1 | Login Page | 2 | Sami Ullah |
| 3.2.1 | Frontend | 3.2.1.1.2 | Registration Page | 2 | Sami Ullah |
| 3.2.1 | Frontend | 3.2.1.1.3 | Home | 7 | Sami Ullah |
| 3.2.1 | Frontend | 3.2.1.1.4 | Communication | 5 | Sami Ullah  Danish Rehman |
| 3.2.1 | Frontend | 3.2.1.1.5 | Course Category | 4 | Sami Ullah  Danish Rehman |
| 3.2.1 | Frontend | 3.2.1.1.6 | Notice Board | 2 | Sami Ullah |
| 3.2.1 | Frontend | 3.2.1.1.7 | Post Query | 4 | Sami Ullah  Danish Rehman |
| 3.2.1 | Frontend | 3.2.1.1.8 | Students | 3 | Sami Ullah |
| 3.2.1 | Frontend | 3.2.1.1.9 | Teachers | 2 | Sami Ullah |
| 3.2 | Website | 3.2.2 | Back End | 15 | Sami Ullah  Danish Rehman |
| 3.2.2 | Back End | 3.4.2.1 | PHP | 10 | Sami Ullah  Danish Rehman |

* 1. **Components, Libraries, Web Services and stubs**
* HTML
* CSS
* Bootstrap
* JavaScript
* PHP
* Font awesome

**Database**

* Xampp
* MySQL
  1. **IDE, Tools and Technologies**
* Visual Studio Code
* Visual Paradigm
* MS Word
* Adobe Illustrator
* Canva
  1. **Best Practices / Coding Standards**
     1. **Software Engineering Methodologies**

In our project we have used Scrum Agile Methodology because it has a greater adaptability to frequently changing scope. Here is how we managed our project is given below:

* Meeting was conducted 3 to 4 times in a week.
* A meeting with the supervisor was conducted at least 1 to 2 times a week to discuss progress.
* In the meeting tasks was created.
* Then these tasks were performed by the group members.
* Progress was discuss in meeting.
  1. **Summary**

In this chapter we have provided list of components and libraries that we have used in our project for better user experience. We have mentioned Work break down structure WBS. We have also mentioned tools and IDEs and best practices and coding standards of software engineering.

**Chapter 6:**

**Testing and Evaluation**

**Chapter 6:**

**Testing and Evaluation**

* 1. **Introduction**

For assuring the quality of the system, testing is the most essential step for assuring the quality of the system. The purpose of testing is to find out the system errors and bugs in the system. So, in this chapter we will discuss testing of our web application Edu unity Desk. We will use the black box testing. In black box testing we will examine functionality of an application from requirements.

* 1. **List of Test Scenarios**
     1. **Admin Login**

**Input:** Username

**Valid:**

* + {A, B, C, …}
  + {a, b, c, …}
* {1, 2, 3, 4, …}
* {!, @, #, $, %, …}

**Invalid:**

* { }

**Input:** Password

**Valid:**

* + {A, B, C, …}
  + {a, b, c, …}
* {1, 2, 3, 4, …}
* {!, @, #, $, %, …}

**Invalid:**

* { }

**Test Case Design through Use Case**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **ECP** | | **Expected Output** |
| **Username** | **Password** |  |
| 1 | Admin login successful | Sami19 | Sami@1919 | Admin logged in |
| 2 | Admin login successful | Sami@19 | Sami@1919 | Admin logged in |
| 3 | Unsuccessful Admin login |  | Sami@1919 | Please fill the admin name field |
| 4 | Unsuccessful Admin login | Sami@19 |  | Please fill the password field |

Table 6.2.1.1 | Admin Login

* + 1. **Add Teacher**

**Input:** Full Name

**Valid:**

* + {A, B, C, …}
  + {a, b, c, …}
* {!, @, #, $, %, …}
* {1, 2, 3, 4, …}

**Invalid:**

* { }

**Input:** Experience

**Valid:**

* + {1, 2, 3, …}

**Invalid:**

* {!, @, #, $, %, …}
* {A, B, C, …}
* {a, b, c, …}

**Input:** Email

**Valid:**

{A, B, C, ...}

{a, b, c, …}

{!, @, #, $, %, …}

{1, 2, 3, 4, …}

**Invalid:**

* {!!, @@, ##, $$, %%, …}
* { }

**Input:** Contact

**Valid:**

{1, 2, 3, 4, …}

{A, B, C, ...}

{a, b, c, …}

{!, @, #, $, %, …}

**Invalid:**

{ }

**Test Case Design through Use Case**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **ECP** | | | | **Expected Output** |
| **Full Name** | **Email** | **Experience** | **Contact** |  |
| 1 | Teacher Added Successfully | Danish | 22628@students.riphah.edu.pk | 7 | 03307985412 | Teacher Added Successfully |
| 2 | Unsuccessful Add Teacher |  | 22628@students.riphah.edu.pk | 7 | 0330\*798541 | Invalid full name entered. Please enter the field. |
| 3 | Unsuccessful Add Teacher | @Danish19 | 22628.riphah.edu.pk | 22628 | 0330\*798541 | Invalid email entered. Please enter with a valid  email. |
| 4 | Unsuccessful Add Teacher | Sami | 22628@students.riphah.edu.pk | 7 | 03307985412 | Email already exists. |

Table 6.2.2.1 | Add Teacher

* + 1. **Add Student**

**Input:** Full Name

**Valid:**

* + {A, B, C, …}
  + {a, b, c, …}
* {!, @, #, $, %, …}
* {1, 2, 3, 4, …}

**Invalid:**

* **{ }**

**Input:** Registration No

**Valid:**

* + {1, 2, 3, …}
* {!, @, #, $, %, …}
* {A, B, C, …}
* {a, b, c, …}

**Invalid:**

* { }

**Input:** Email

**Valid:**

{A, B, C, ...}

{a, b, c, …}

{!, @, #, $, %, …}

{1, 2, 3, 4, …}

**Invalid:**

* {!!, @@, ##, $$, %%, …}

**Input:** Contact

**Valid:**

{1, 2, 3, 4, …}

{A, B, C, ...}

{a, b, c, …}

{!, @, #, $, %, …}

**Invalid:**

* **{ }**

**Test Case Design through Use Case**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **ECP** | | | | **Expected Output** |
| **Full Name** | **Email** | **Registration No** | **Contact** |  |
| 1 | Student Added Successfully | Sami | 23799@students.riphah.edu.pk | 23799 | 03328541255 | Student Added Successfully |
| 2 | Unsuccessful Add Student |  | 23799@students.riphah.edu.pk | 23456 | 0332/854125 | Invalid full name Please fill the field. |
| 3 | Unsuccessful Add Student | Sami | 23799@students.riphah.edu.pk | 67890 | 0332/854125 | Email already exists |
| 4 | Unsuccessful Add Student | Sami | 23769@students.riphah.edu.pk | 23799 | 0332/854125 | Registration No already exists |

Table 6.2.3.1 | Add Student

* + 1. **Post Query**

**Input:** Category

**Valid:**

* + {Select from the given}

**Invalid:**

* { }

**Input:** Coordinator

**Valid:**

* + {Select from the given}

**Invalid:**

* { }

**Input:** Query

**Valid:**

* + {A, B, C, …}
  + {a, b, c, …}

{!, @, #, $, %, …}

{1, 2, 3, 4, …}

**Invalid:**

* { }

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **ECP** | | | **Expected Output** | |
| **Category** | **Coordinator** | **Query** | |  |
| 1 | Query Post successfully | Student Service | Admin | Here is the message related to university course registration of fall 2024. | | Query Post successfully |
| 2 | Unsuccessful Post query |  | Admin | Here is the message related to university course registration of fall 2024. | | Please select the category. |
| 3 | Unsuccessful Post query | Admission department |  | Here is the message related to university course registration of fall 2024. | | Please select the coordinator. |
| 4 | Query Post unsuccessfully | Student Service | Admin |  | | Please fill the Post query field. |

Table 6.2.4.1 | Post Query