WACHEMO UNIVERSITY

COLLEGE OF ENGINEERING & TECHNOLOGY

DEPARTMENT OF SOFTWARE ENGINEERING

Project Document

on

Student Feedback Management System

Submitted in Partial Fulfillment of the Requirements for the

Requirement Engineering Course

Submitted by:

Name ID

1.Sirawdink Abayneh……..….....16D4888
2.Desalegn worku…….1600119
3.Solomon shukura…….…..…...1600337
4.Hailiye Abiti……..……..…1600915
5.Beimnet Mesfin…….……..…...1600061
6.Rediet Nigussie……1600756
3.Hassen Shefik……..1600204

Mr: Mesay Aschalew

Submission:date 02/01/2026

Hosanna Ethiopia

Acknowledgement

We would like to express our sincere gratitude to our instructor, **Mr. Mesay Aschalew**, for his continuous guidance, support, and valuable feedback throughout the development of the Student Feedback Management System project. His constructive advice, technical insights, and encouragement played a vital role in helping us understand software engineering concepts and successfully design and document this system.

We also extend our heartfelt appreciation to Wachemo University and the Department of Software Engineering for providing the necessary academic environment, resources, and facilities that supported the completion of this project. The knowledge and skills gained through the program greatly contributed to the successful execution of this work.

Special thanks go to all project team members for their strong commitment, cooperation, and teamwork during every phase of the project, from requirement analysis to system design and documentation. The collaborative effort and shared responsibility among team members made this project both productive and rewarding.
This project reflects the dedication, teamwork, and learning experience of everyone involved, and we are proud of the final outcome.

List of Tables

List of Figures

List of Acronyms and Abbreviations

SFMS – Student Feedback Management System
SRS – Software Requirements Specification
WCU – Wachemo University
UML – Unified Modeling Language
ERD – Entity Relationship Diagram
UI – User Interface
OTP – One-Time Password
HTML – HyperText Markup Language
CSS – Cascading Style Sheets
JS – JavaScript
PHP – Hypertext Preprocessor
DB – Database
DBMS – Database Management System
MySQL – My Structured Query Language
CRUD – Create, Read, Update, Delete
API – Application Programming Interface
LAN – Local Area Network
IDE – Integrated Development Environment
XAMPP – Cross-Platform Apache MySQL PHP Perl
HTTP – HyperText Transfer Protocol
HTTPS – HyperText Transfer Protocol Secure
QA – Quality Assurance
FR – Functional Requirement
NFR – Non-Functional Requirement
V&V – Validation and Verification
SDLC – Software Development Life Cycle
MVC – Model View Controller
PDF – Portable Document Format
ETB – Ethiopian Birr

Chapter One

1.1.Introduction

This document presents the design and development of the Student Feedback Management System (SFMS), a web-based system developed by Software Engineering students at Wachemo University. The main purpose of this project is to provide a centralized and structured platform where students can submit feedback about courses, instructors, and university services, and where authorized staff can review and analyze this feedback for quality improvement.

The Student Feedback Management System is designed to connect students, administrators, and responsible staff within one integrated system. Students can register, log in, submit feedback, and view their submission history. Administrators manage user accounts, monitor feedback submissions, and generate reports. Responsible staff, such as department heads and service managers, can access assigned feedback and analyze reports related to their areas.

The system was developed following standard software engineering practices, including requirement analysis, use case modeling, UML diagram design (such as use case diagrams, sequence diagrams, deployment diagrams, subsystem decomposition diagrams, ER diagrams, and UI flow diagrams), system architecture design using a three-tier layered architecture, and implementation and testing planning. These activities helped ensure the system is well-structured, secure, scalable, and easy to maintain.

Tools such as Visual Paradigm were used for system modeling, PHP, HTML, CSS, and JavaScript for implementation planning, MySQL for database design, and WAMP(Apache Server) for local testing and execution. Documentation and testing artifacts were prepared to ensure correctness, reliability, and alignment with university requirements.

This document explains each phase of the system design, implementation, and testing process. It serves as a guide for understanding how the Student Feedback Management System was designed, how it operates, and how it can support continuous improvement at Wachemo University in the future

1.2 Background

1.2.1 Background of the Organization (Wachemo University)

Wachemo University is a public higher education institution established in 2009 in Hossana town, about 230 km southwest of Addis Ababa. The university started teaching in 2012 with 538 students in 12 departments and has now grown into a large institution with more than 30,000 students across 73 undergraduate and 73 postgraduate programs.

The university operates through seven major colleges, supported by over 4,000 academic and administrative staff. Over the years, WCU has expanded its teaching, research, and community service reach and has been recognized as a top-performing university among its generation. Its continuous growth brings increasing responsibility to improve service delivery and student engagement especially in areas like feedback collection and academic quality improvement.

1.2.2 Background of the System (Student Feedback Management System)

The current feedback collection process at WCU depends mainly on manual forms or inconsistent department-level practices, which makes the data hard to organize, slow to analyze, and difficult to use for decision-making. Because of the university's large student population and diverse academic programs, managing feedback manually creates delays, errors, and limited transparency.

The proposed web-based Student Feedback Management System is designed to centralize and digitize this process. It enables students to submit feedback online, allows staff to review organized reports, and helps the university make faster, evidence-based decisions. By using PHP, MySQL, and a simple browser-based interface, the system supports efficient data management, improves accessibility, and strengthens the university's overall quality assurance efforts.

1.3 Statement of the Problem

1.3.1 Problems Faced by Wachemo University

Although Wachemo University has grown very fast, it still faces many problems in collecting and using student feedback effectively. These problems affect the quality of teaching and university services. The main problems are listed below:

1. Manual and Unorganized Feedback Collection
   Student feedback is mainly collected using paper forms or different methods in each department. This makes the feedback scattered and difficult to collect, store, and manage in one place.
2. Slow Feedback Processing and Analysis
   Because feedback is handled manually, it takes a long time to enter, organize, and analyze the data. This causes delays in using feedback for improvement.
3. Risk of Data Loss and Mistakes
   Paper-based feedback and unstructured files can easily be lost or damaged. Manual handling also increases the chance of errors, which reduces the accuracy of the feedback.
4. Lack of Transparency and Responsibility
   Students often do not know whether their feedback is reviewed or used. At the same time, administrators cannot easily track actions taken based on feedback, which reduces trust in the system.

5. Low Student Participation
   Current feedback methods are inconvenient and time-consuming. Many students also worry about being identified, which discourages them from giving honest feedback.
6. Difficulty Managing Feedback from Many Units
   The university has many colleges, departments, cafeterias, and dormitories. Managing feedback from all these units manually is complex and inefficient.
7. No Central Reporting System
   There is no single system to create summary reports, identify patterns, or support data-based decisions at the university level.

1.3.2 Problems Addressed by the Proposed System

The proposed Student Feedback Management System is designed to solve the following problems identified above:

1. Lack of a Central Feedback System
   The system provides one web-based platform to collect, store, and manage all student feedback in one place.
2. Inefficient Feedback Handling and Reporting
   The system automates feedback submission and report generation, making the process faster and more efficient.
3. Unstructured Feedback Data
   The system uses standard feedback forms and rating scales, which makes feedback consistent and easier to compare across departments and services.
4. Weak Privacy and Security
   The system supports anonymous feedback and uses role-based access to protect student data and ensure confidentiality.
5. Limited Support for Decision-Making
   The system generates summarized reports that help administrators and department heads identify problems and improvement areas.

1.4 Objective

1.4.1 General Objective

The general objective of this project is to design and implement a Student Feedback Management System for Wachemo University to collect structured student feedback and support improvement in teaching and academic services.

1.4.2 Specific Objectives

To achieve the general objective, the project will be carried out through the following specific objectives:

1. Examine the current student feedback collection methods at Wachemo University to identify weaknesses related to accessibility, unknown, and effectiveness.
2. Identify key stakeholders involved in the student feedback process, including students, college deans, department heads, cafeteria manager and dormitory protectors.

3. Gather and document functional requirements related to feedback submission, review, analysis, and reporting.
4. Gather and document non-functional requirements such as security, usability, performance, and scalability.
5. Define system requirements that ensure fair, and honest student feedback.
6. Design system use case diagrams to model interactions between students, instructors, and administrative users.
7. Design database structures to manage feedback records, academic units, users, and reports.
8. Design workflow and activity diagrams to represent the student feedback lifecycle from submission to response.
9. Design user-friendly interfaces that encourage student participation and simplify administrative review.
10. Develop frontend components using HTML, CSS, and JavaScript to support feedback submission and system navigation.
11. Implement backend functionalities using PHP to manage authentication, feedback handling, and role-based access control.
12. Implement a secure MySQL database to store feedback data, user roles, academic information, and system logs.
13. Integrate system components and manage version control using Git and GitHub for organized development and maintenance.
14. Implement feedback categorization and rating mechanisms to allow structured evaluation of courses, instructors, and services.
15. Develop reporting and visualization features to help academic units analyze feedback trends and identify improvement areas.
16. Ensure data privacy and confidentiality by applying appropriate security controls and access restrictions.
17. Provide system documentation and basic user guidelines to support effective system use and future maintenance.

1.5 Proposed System

The proposed system is a web-based Student Feedback Management System (SFMS) designed for Wachemo University to improve how student feedback is collected, managed, and used. The system provides a centralized and secure platform where students can submit feedback on courses, instructors, and selected university services through an easy-to-use web interface.

The SFMS replaces the existing manual and fragmented feedback methods with a structured digital process. Students can submit feedback anytime using a browser, while the system ensures privacy and supports anonymous responses when required. This encourages honest and meaningful feedback.

By using widely supported web technologies, the system is reliable, easy to maintain, and accessible. Overall, the proposed system improves efficiency, transparency, and academic quality by turning student feedback into useful and actionable information.

1.6 Scope and Limitations

1.6.1 Scope of the System

The scope of the Student Feedback Management System describes the main functions and features the system is designed to support at Wachemo University. The scope is defined to focus on essential feedback activities while keeping the system practical and manageable within the project timeline.

1. Student Feedback Submission
   The system allows students to submit feedback on courses, instructors, cafeteria services, and dormitory services using structured online forms. This ensures feedback is collected in a consistent and organized manner across the university.

2. User Authentication and Role-Based Access
   The system provides secure login and assigns specific roles such as student, instructor, department head, and service manager. This helps protect sensitive information and ensures users can access only the data related to their responsibilities.

3. Feedback Without Revealing Student Identity
   The system allows students to submit feedback without showing their identity. This encourages students to share honest opinions without fear of negative consequences.

4. Centralized Feedback Storage
   All feedback data is stored in a single, centralized database. This improves data accuracy, reduces the risk of data loss, and makes feedback easier to manage and retrieve.

5. Feedback Review and Basic Reporting
   Authorized users can view summarized feedback reports for courses, departments, and services. These reports support academic and service improvement by highlighting common issues and strengths.

6. Web-Based System Access
   The system can be accessed through standard web browsers on computers and mobile devices. This allows users to use the system without installing additional software.

1.6.2 Limitations of the System

The limitations describe the areas the system does not cover and the restrictions on its capabilities. These limitations exist due to technical constraints, limited development time, and available resources.

1. Dependence on Honest Student Participation
   The system cannot guarantee that all feedback provided by students is honest or constructive because the quality of feedback depends on individual user behavior, which cannot be fully controlled by the system.

2. Limited Feedback Detail
   Feedback is mainly collected using structured forms and rating scales, which makes data easier to organize and analyze but may not capture very detailed personal opinions or explanations.

3. Basic Reporting and Analytics
   The system provides only basic summaries and simple visual reports, as advanced analytics and predictive analysis require additional time, specialized expertise, and higher computing resources that are not available within the scope of this project.

4. Limited Scope of Feedback Areas
   The system focuses only on academic feedback and selected services such as cafeteria and dormitory services, since including all administrative and non-academic services would significantly increase system complexity beyond the defined project scope.

5. No Direct Measurement of Improvement Impact
   The system does not track how feedback leads to actual improvements in teaching or services over time, as measuring such impact requires long-term monitoring and external evaluation beyond a semester-based project.

6. Security Risks Due to Credential Sharing
   The system cannot fully prevent users from sharing their login credentials, as user behavior remains outside the system's direct technical control even when authentication mechanisms are in place

1. 7 Methods and Tools

This section describes the methods and tools used to gather system requirements, analyze and design the system, validate and verify requirements, implement the system, and support the overall development of the Student Feedback Management System for Wachemo University.

1.7.1 Requirements Gathering Techniques / Methods

Accurate requirements are essential for developing a system that meets real user needs. To achieve this, multiple requirement gathering techniques are used

1. Interviews with Academic and Service Staff
   Interviews are conducted with instructors, department heads, cafeteria managers, and dormitory staff. These interviews help understand how feedback is currently collected, reviewed, and used, as well as the problems faced with existing methods.
2. Observation of Existing Processes
   The current feedback collection process is observed to understand real workflows and identify inefficiencies such as delays, data loss, and lack of follow-up.

3. Review of Existing Documents and Practices
   Existing feedback forms, reports, and university guidelines are reviewed to identify gaps and areas that need improvement in the proposed system.
4. Team Discussions and Idea Sharing
   Group discussions among project members are used to share ideas, analyze findings, and agree on realistic system features based on collected information.

1.7.2 System Analysis and Design Methods

System analysis and design methods are used to transform requirements into a clear and structured system design. The following techniques and diagrams are applied:

1. Use Case Analysis
   Use case diagrams identify system users and describe how they interact with the system. This helps define core functions such as submitting feedback, viewing reports, and managing users.
2. Process and Activity Modeling
   Activity diagrams represent the flow of feedback from submission to review. This ensures that system processes are logical and easy to understand.
3. Sequence Diagrams
   Sequence diagrams show detailed interactions between system components over time, illustrating how operations like feedback submission, processing, and reporting occur step by step.
4. Database Design (ER Diagrams)
   Entity-Relationship diagrams are used to design the database structure for managing users, feedback records, academic units, and reports efficiently.
5. Class Diagrams
   Class diagrams model the structural design of the system, showing classes, attributes, methods, and relationships. They help define the backend architecture and guide the implementation in an object-oriented way.

6. Interface Prototyping
   Simple interface prototypes visualize system screens and navigation, helping improve usability before system implementation begins.

1.7.3 Requirement Validation and Verification

Requirement validation and verification ensure that the system requirements are correct, complete, and meet stakeholder expectations.

1. Stakeholder Requirement Review
   Collected requirements are reviewed with students, instructors, and administrators to confirm accuracy and relevance.
2. Consistency and Feasibility Checks
   Requirements are checked to ensure they are clear, non-conflicting, and achievable within the project timeline and available resources.
3. Traceability of Requirements
   Each requirement is linked to system use cases, diagrams, and planned features to ensure full coverage.

4. Testing-Based Verification
   Functional testing is performed during development to verify that implemented features meet the specified requirements.

1.7.4 System Implementation Methods

The system implementation follows an Agile development approach, which supports flexibility, continuous improvement, and regular feedback during development. This approach is suitable for an academic project where requirements may be refined as development progresses.

1. Agile and Iterative Development
   The system is developed in small iterations, where core features such as user authentication and feedback submission are implemented first. Additional features are added gradually based on progress and feedback from stakeholders.

2. Modular System Development
   The system is divided into modules such as user management, feedback submission, reporting, and administration. Each module is developed, tested, and improved independently, which simplifies development and maintenance.

3. Separation of Frontend and Backend
   The frontend is responsible for user interaction and input validation, while the backend handles system logic, database operations, and access control. This separation improves system organization and supports parallel development.
4. Continuous Testing and Improvement
   Testing is performed during each development iteration to identify issues early and ensure that implemented features meet system requirements.

5. Version Control and Team Collaboration
   Git and GitHub are used to manage source code versions, track changes, and support collaboration among team members throughout the development process

1.7.5 Development Environment and Programming Tools

The following tools and technologies are used to develop, test, and support the Student Feedback Management System.

A. Software Tools

1. HTML, CSS, and JavaScript
   Used to design and develop user-friendly and responsive web interfaces for system interaction.
2. PHP
   Used for server-side processing, user authentication, and handling system logic.
3. MySQL
   Used as the database management system to store feedback data, user information, and system records securely.

4. Visual Studio Code
   Used as the main code editor for writing, editing, and managing frontend and backend source code.
5. WAMP Server
   Used as a local development environment that provides Apache, PHP, and MySQL services for testing and running the system before deployment.
6. Visual Paradigm
   Used for system modeling and creating diagrams such as use case, activity, and ER diagrams.
7. Figma
   Used to design and prototype user interfaces before implementation.
8. Git and GitHub
   Used for version control, source code management, and collaboration among project members.
9. Web Browser
   Used for testing, debugging, and validating system functionality and user interface behavior across different devices.

B. Hardware Environment

1. Desktop or Laptop Computers
   Used for system development, testing, and documentation.
2. Server (University Server and local server)
   Used to host the system and manage database operations.
3. Mobile Devices
   Used to test system accessibility and usability on different screen sizes.
4. Internet Connection
   Used for system access, collaboration, and version control.

1.8 Significance of the Project

The Student Feedback Management System (SFMS) is important because it improves the way Wachemo University collects, manages, and uses feedback, while also providing practical experience for the developers. The significance can be categorized into benefits for the organization and benefits for the developers:

A. For Wachemo University

1. Improved Feedback Collection
   The system replaces manual forms with an online platform, allowing students to submit feedback quickly and efficiently.
2. Enhanced Data Organization and Analysis
   Feedback data is stored digitally in a structured format, making it easier to analyze trends, generate reports, and identify areas for improvement.
3. Evidence-Based Decision Making
   Academic staff and administrators can use accurate feedback to make informed decisions related to teaching quality, course management, and service improvements.

4. Transparency and Accountability
   The system ensures that feedback is recorded and managed properly, increasing transparency in academic and service evaluations.
5. Time and Resource Saving
   Automated collection, storage, and basic reporting reduce the workload for staff compared to traditional manual methods.
6. Support for Quality Improvement
   Regular feedback helps the university monitor and improve academic programs and services, contributing to better student satisfaction and institutional reputation.

B. For Developers

1. Practical Experience in Web-Based System Development
   Developers gain hands-on experience in designing, coding, testing, and deploying a real-world web system using PHP, MySQL, and web technologies.
2. Understanding of System Analysis and Design
   Working on the SFMS allows developers to practice requirement gathering, UML modeling, database design, and interface prototyping.
3. Teamwork and Project Management Skills
   Using Agile methodology and version control (Git/GitHub) helps developers improve collaboration, planning, and iterative development skills.
4. Problem-Solving and Critical Thinking
   Developers learn to identify system problems, propose solutions, and implement features that meet real user needs

1.9 Beneficiaries of the System or Application

The SFMS benefits multiple stakeholders at Wachemo University. The beneficiaries and their benefits are described below:

- ➢ Students
  - Can submit feedback easily through a web interface.
  - Feedback can be provided privately, encouraging honesty and constructive input.
  - Students can see that their opinions are recorded and valued, improving engagement and satisfaction.
- ➢ Instructors
  - Receive structured feedback on teaching performance.
  - Identify strengths and areas for improvement in their courses.
  - Use feedback to adjust teaching methods and improve student learning outcomes.
- ➢ Department Heads
  - Can monitor instructor performance and departmental service quality.
  - Receive summarized reports that support decisions related to promotions, training, or interventions.
  - Improve departmental planning based on real feedback data.
- ➢ Cafeteria and Dormitory Management
  - Receive feedback on services, facilities, cleanliness, and staff behavior.

- Identify issues that need immediate attention and improve service quality.
- Ensure student satisfaction in non-academic areas, supporting overall campus life quality.

➢ University Administration

- Can monitor the overall effectiveness of academic and support services.
- Generate reports for policy making, quality assurance, and institutional planning.
- Strengthens accountability and ensures that feedback leads to actionable improvements.

1.10 Feasibility Study

1. Technical Feasibility

The SFMS can be implemented using the university's existing IT infrastructure. PHP, MySQL, and web technologies are compatible with Wachemo University's servers. Students and staff can access the system via standard computers and mobile devices, and low-bandwidth considerations are addressed by lightweight front-end design. Tools like Visual Paradigm and Figma support modeling and interface prototyping without additional infrastructure.

2. Economic Feasibility

The system reduces costs associated with paper-based feedback, manual processing, and reporting delays. Open-source technologies minimize software expenses. Development, deployment, and maintenance fit within the university's semester-based project budget without requiring additional funding.

1.10.2 Economic Feasibility Analysis

This section demonstrates the economic feasibility of the proposed Student Feedback Management System by comparing the cost of the current manual feedback process with the cost of the proposed digital system over time.

Assumptions Used for Calculation

To make the analysis realistic, the following assumptions are made based on the university structure:

➢ Number of departments: 73
➢ Number of lecturers per department: 5
➢ Total lecturers:
➢ $73 \times 5 = 365$ lecturers
➢ Number of feedback forms per lecturer per year: 100
➢ Cost of one paper (one form): 3 ETB
➢ Analysis period: 5 years

These values represent a reasonable average for academic feedback activities.

A. Cost of the Manual Feedback System

1. Annual Paper Consumption

- ➢ Total forms per year =365 lecturers × 100 forms = 36, 500 forms
- ➢ Annual paper cost = 36, 500 × 3 ETB = 109, 500 ETB

Table 1: Annual Cost of Manual Feedback

Cost item	Unit cost(ETB)	Quantity per Year	Total Annual Cost(ETB)
Paper (per sheet)	3 ETB	36, 500 sheets	109, 500
Printing ink	—-	Estimated	2,000
Printer maintenance	—--	Annual	2,500
Total Annual Cost			114,000 ETB

Table 2: Manual System Cost For 5 years

Year	Total Cost (ETB)
1	114,000
2	114,000
3	114,000
4	114,000
5	114,000
Total(5 Years)	570, 000 ETB

Proposed System Cost Table

Cost item	Annual Cost (ETB)	5-Year Cost (ETB)
Web hosting	—-----	—-------
Domain name	2,000	10,000
System maintenance	3,000	15,000
System payment		100,000
Total Cost	5,000	125,000 ETB

Table 3: Cost of Proposed System

C. Cost Comparison

Table 4: cost comparison between Manual and Proposed System (5 Years)

System Type	Total Cost (ETB)
Manual Feedback System	570,000
Proposed Digital System	125,000
Total Cost Saved	445,000ETB

D. Economic Feasibility Conclusion

The economic analysis shows that the proposed Student Feedback Management System is financially feasible. Over five years, the manual feedback process costs about 570, 000 ETB, while the proposed system costs only 125,000 ETB, resulting in a savings of 445,000ETB mainly from reduced paper use, printing, and manual work. Beyond these direct savings, the system also enables structured feedback for cafeteria and dormitory services, which were not formally evaluated under the manual process, adding value without significant extra cost. In addition, this analysis does not include intangible benefits such as improved student satisfaction, better service quality, faster decision-making, and increased transparency, which further increase the system's overall economic value. Therefore, the proposed system is economically justified and provides clear long-term benefits to the university.

3. Operational Feasibility

The SFMS integrates smoothly into current academic and service processes. Students, instructors, and service managers (cafeteria and dormitory) can submit and review feedback with minimal training. Workflows for feedback collection, validation, and reporting are streamlined, and multilingual support ensures accessibility for all users.

4. Legal Feasibility

The system respects university policies and Ethiopian regulations regarding student data and privacy. Feedback can be collected anonymously while maintaining secure storage. Data access is role-based, ensuring legal and policy compliance for record management and reporting.

5. Political Feasibility

The project has support from university leadership, department heads, and relevant service units. It provides transparent, neutral, and actionable feedback for decision-making, aligning with institutional goals of improving teaching and service quality. Political resistance is unlikely due to its objective and administrative benefits.

1.11.1 Time Schedule (time estimation, part chart)

Table 5: Time Schedule(Time Estimation,pert chart)

Phase	Duration	Description
Requirement & Specification	2 weeks	Collecting requirements from students, instructors, departments, and university administration
System Design	2 weeks	Database design, system architecture, feedback forms, user roles (admin, student, instructor)
Project Management & Planning	2 weeks (parallel)	Scheduling, monitoring progress, documentation, and coordination
Implementation & Coding	3 weeks	Developing modules: login, feedback forms, data storage, reports, admin dashboard
System Testing	1 week	Functional testing, integration testing, fixing major errors
Quality Assurance	2 weeks	Data accuracy checks, usability testing, security validation
Deployment & Maintenance	2 weeks	System installation, user training, minor updates, and support

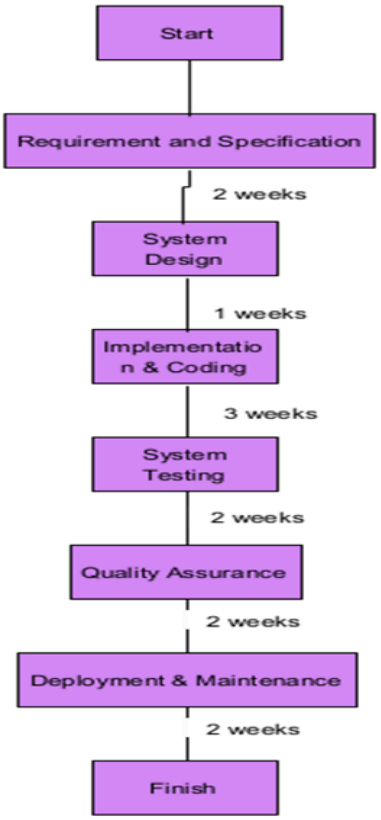


Figure 1: Pert Chart Diagram

1.11.2 Budget Plan (Cost estimation techniques bottom up approach)

In the Bottom-Up approach, the total project cost is estimated by calculating the cost of each individual activity involved in developing the Student Feedback Management System and then summing them to obtain the overall budget. This approach is suitable for academic systems where project tasks and scope are clearly defined.

Table 6: Budget Plan (Cost estimation techniques bottom up approach)

Activity	Description	Estimated Cost (ETB)
Requirement analysis& system Design	Gathering requirements from students, instructors, and service units (cafeteria, dormitory, library), and designing system architecture and database	20,000
Implementation & Coding	Development of system modules such as login, feedback submission,	25,000

	report generation, admin dashboard, and database integration	
Testing & Quality Assurance	Functional testing, bug fixing, usability testing, and data validation	10,000
Deployment & Training	System installation, basic user training for administrators, and initial maintenance support	8,000
Total Estimated Cost		63,000

Chapter Two

2. DESCRIPTION OF THE EXISTING SYSTEM OR APPLICATION

2.1 Business Rules and Constraints

Business Rules

The existing feedback collection procedure at Wachemo University follows several organizational rules:
B1: Only Wachemo University students are allowed to give feedback.
B2: Students can only evaluate instructors who teach their registered courses within their department and academic year.
B3: Academic officers distribute printed feedback forms manually at the end of each semester.
B4: Each student must submit one completed feedback form per instructor/course they are taking.
B5: All forms must be submitted within the deadline, typically during the final examination period.
B6: Completed forms are handled only by authorized academic staff to maintain confidentiality.
B7: Instructors do not see raw individual forms; they receive only summarized results from department heads.
B8: All collected feedback forms must be kept as physical records for future reference.
B9: The results from feedback are used for evaluating instructor performance, training, and promotion decisions.

Constraints

The current process faces several limitations that reduce efficiency:

- The feedback process depends heavily on paper and manual effort.
- The university often faces form shortages and printing delays due to increasing paper costs.
- Manual counting and data entry are prone to human errors.
  The entire process from form distribution to result compilation — takes a long time.
- Physical storage space is required to archive feedback forms for several years.
  Forms may be lost, damaged, or duplicated, making the data unreliable.
- Students may feel uncomfortable when submitting forms because the process does not fully protect their identity.
- Feedback is currently limited to teaching-related services only there is no structured evaluation for: Dormitory services, Cafeteria services,Library services, Administrative services
- Accessing old feedback for decision-making is difficult since manual records are not indexed or searchable.
- Invalid, incomplete, or duplicate submissions cannot be automatically detected, reducing data quality.

2.3 Functions or Main Activities of the Existing Manual Process

The current feedback collection and processing workflow at Wachemo University involves the following major activities:

1. Preparation of Feedback Forms
   Academic officers print and prepare forms at the end of each semester.
2. Distribution to Students
   Forms are manually distributed during classes or examination sessions.
3. Form Completion
   Students fill out forms for each instructor they take a course with.
4. Collection of Completed Forms
   Academic officers gather all forms within the given submission deadline.
5. Manual Checking and Sorting
   Forms are reviewed and grouped by instructor, course, and academic year.
6. Data Summarization and Scoring
   Department staff manually count and calculate feedback results for each instructor.
7. Report Preparation
   Summarized data is written into a performance report format.
8. Result Delivery
   Department heads share the final summary with university management and instructors.
9. Physical Storage of Records
   All completed forms and summary reports are stored in document files for future reference.

2.4 Players of the Existing System or Application

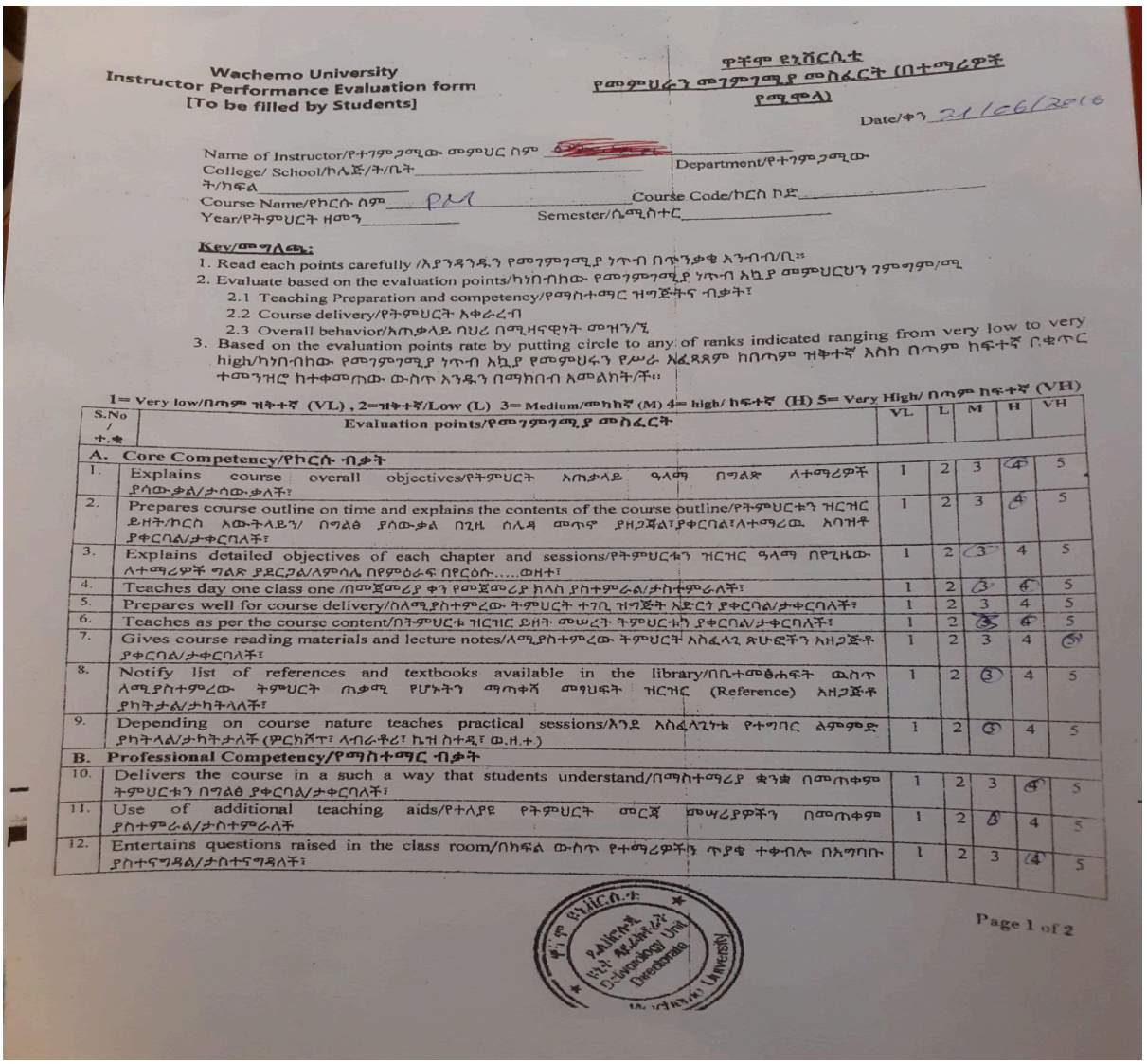
The existing feedback process involves several key participants who perform different roles to complete the manual workflow:

1. Students
   Provide evaluations by filling out feedback forms for each instructor and course.
2. Instructors/Lecturers
   Receive performance results after the evaluation process is completed, but do not directly handle raw forms.
3. Academic Officers
   Distribute feedback forms to students, collect them back, and ensure correct handling.
4. Department Heads
   Review summarized results, prepare official reports, and share them with university management.
5. University Management
   Uses feedback results for decision-making related to instructor improvement, training, and evaluation.

2.5 Organization Structure

2.6.Documents used in the Existing System or Application

The documents Used to collect from the Student about Instruction



Wachemo University
Instructor Performance Evaluation form
[To be filled by Students]

ቋቱሞ ዩኒቨርሲቲ
የመምህራን መገምገሚያ መስፈርት (በተማሪዎች የሚሞላ)

Date/ቀን 21/06/2016

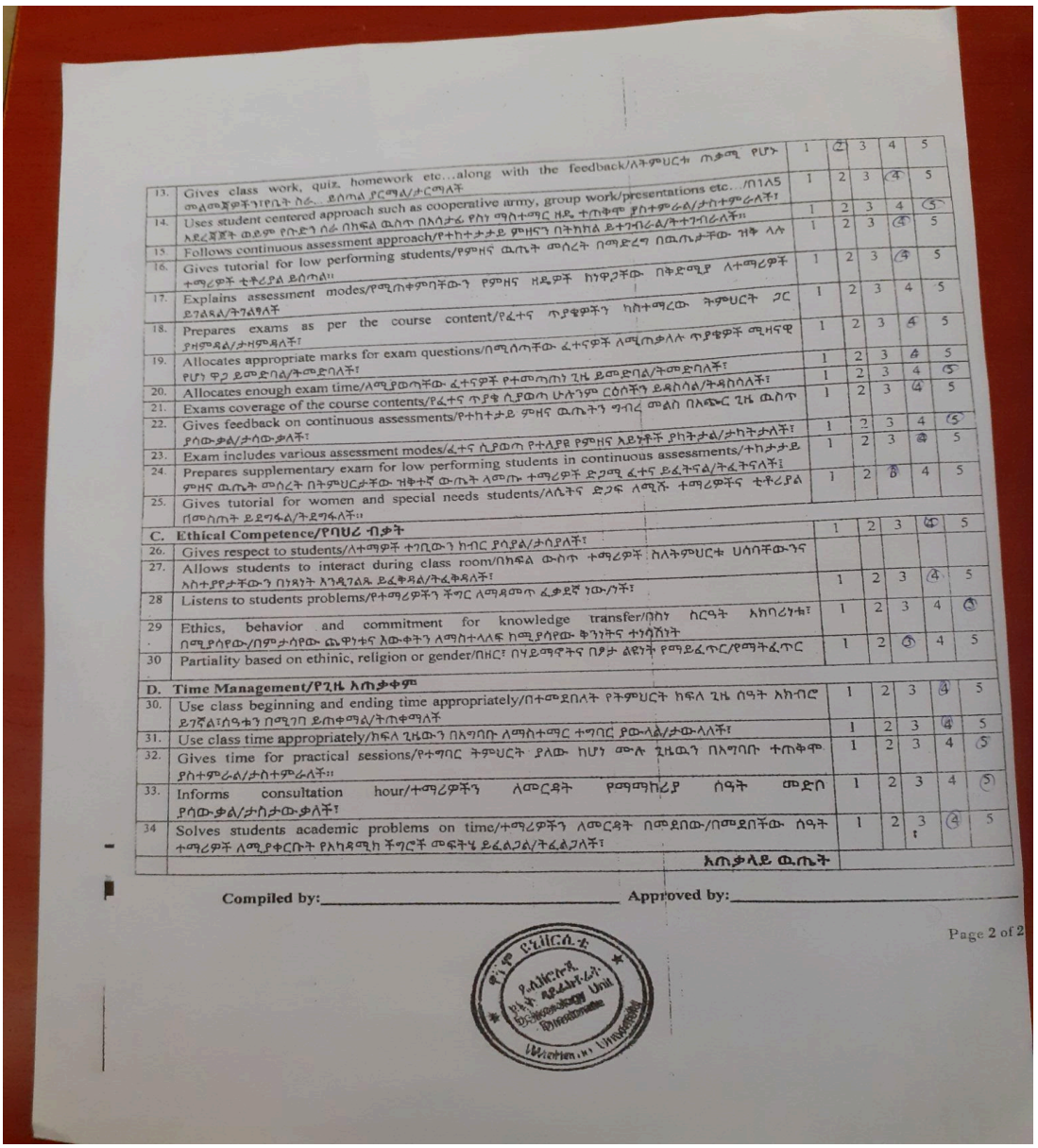
Name of Instructor/የተገመጋሚው መምህር ስም [redacted]
College/ School/ክሊጅ/ት/ቤት _____ Department/የተገመጋሚው ት/ክፍል _____
Course Name/የኮርሱ ስም PM Course Code/ኮርስ ኮድ _____
Year/የትምህርት ዘመን _____ Semester/ሴሚስተር _____

Key/መግለጫ:

1. Read each points carefully /እያንዳንዱን የመገምገሚያ ነጥብ በጥንቃቄ አንብብ/ቢ።
2. Evaluate based on the evaluation points/ካነበብከው የመገምገሚያ ነጥብ አኳያ መምህርህን ገምግም/ሚ
   2.1 Teaching Preparation and competency/የማስተማር ዝግጅትና ብቃት፣
   2.2 Course delivery/የትምህርት አቀራረብ
   2.3 Overall behavior/አጠቃላይ ባህሪ በሚዛናዊነት መዝን/ኚ
3. Based on the evaluation points rate by putting circle to any of ranks indicated ranging from very low to very high/ካነበብከው የመገምገሚያ ነጥብ አኳያ የመምህሩን የሥራ አፈጻጸም ከበጣም ዝቅተኛ እስከ በጣም ከፍተኛ ቦቁጥር ተመንዝሮ ከተቀመጠው ውስጥ አንዱን በማክበብ አመልክት/ች።

1= Very low/በጣም ዝቅተኛ (VL) , 2=ዝቅተኛ/Low (L) 3= Medium/መካከኛ (M) 4= high/ ከፍተኛ (H) 5= Very High/ በጣም ከፍተኛ (VH)

S.No / ተ.ቁ	Evaluation points/የመገምገሚያ መስፈርት	VL	L	M	H	VH
A.	**Core Competency/የኮርሱ ብቃት**					
1.	Explains course overall objectives/የትምህርት አጠቃላይ ዓላማ በግልጽ ለተማሪዎች ያሳውቃል/ታሳውቃለች፣	1	2	3	(4)	5
2.	Prepares course outline on time and explains the contents of the course outline/የትምህርቱን ዝርዝር ይዘት/ኮርስ አውትላይን/ በግልፅ ያሳውቃል በጊዜ ሰሌዳ መጥኖ ያዘጋጃል፣ያቀርባል፣ለተማሪዉ አባዝቶ ያቀርባል/ታቀርባለች፣	1	2	3	(4)	5
3.	Explains detailed objectives of each chapter and sessions/የትምህርቱን ዝርዝር ዓላማ በየጊዜው ለተማሪዎች ግልጽ ያደርጋል/ለምሳሌ በየምዕራፍ በየርዕሱ.....ወዘተ፣	1	2	(3)	4	5
4.	Teaches day one class one /በመጀመሪያ ቀን የመጀመሪያ ክላስ ያስተምራል/ታስተምራለች፣	1	2	(3)	(4)	5
5.	Prepares well for course delivery/ስለሚያስተምረው ትምህርት ተገቢ ዝግጅት አድርጎ ያቀርባል/ታቀርባለች፣	1	2	3	4	5
6.	Teaches as per the course content/በትምህርቱ ዝርዝር ይዘት መሠረት ትምህርቱን ያቀርባል/ታቀርባለች፣	1	2	(3)	(4)	5
7.	Gives course reading materials and lecture notes/ለሚያስተምረው ትምህርት አስፈላጊ ጽሁፎችን አዘጋጅቶ ያቀርባል/ታቀርባለች፣	1	2	3	4	(5)
8.	Notify list of references and textbooks available in the library/በቤተመፅሐፍት ዉስጥ ለሚያስተምረው ትምህርት ጠቃሚ የሆኑትን ማጣቀሻ መፃህፍት ዝርዝር (Reference) አዘጋጅቶ ያከትታል/ታካትላለች፣	1	2	(3)	4	5
9.	Depending on course nature teaches practical sessions/እንደ አስፈላጊነቱ የተግባር ልምምድ ያካትላል/ታካትታለች (ምርክሾፕ፣ ላብራቶሪ፣ ኬዝ ስተዲ፣ ወ.ዘ.ተ)	1	2	(3)	4	5
B.	**Professional Competency/የማስተማር ብቃት**					
10.	Delivers the course in a such a way that students understand/በማስተማሪያ ቋንቋ በመጠቀም ትምህርቱን በግልፅ ያቀርባል/ታቀርባለች፣	1	2	3	(4)	5
11.	Use of additional teaching aids/የተለያዩ የትምህርት መርጃ መሣሪያዎችን በመጠቀም ያስተምራል/ታስተምራለች	1	2	(3)	4	5
12.	Entertains questions raised in the class room/በክፍል ውስጥ የተማሪዎችን ጥያቄ ተቀብሎ በአግባቡ ያስተናግዳል/ታስተናግዳለች፣	1	2	3	(4)	5



13.	Gives class work, quiz, homework etc...along with the feedback/ለትምህርቱ ጠቃሚ የሆኑ መልመጃዎችን፣የቡድን ስራ ይሰጣል ያርማል/ታርማለች	1	2	3	4	5
14.	Uses student centered approach such as cooperative army, group work/presentations etc. /በ1ለ5 አደረጃጀት ወይም የጉድን ሰራ በክፍል ውስጥ በእሳታፊ የስነ ማስተማር ዘዴ ተጠቅሞ ያስተምራል/ታስተምራለች፡፡	1	2	3	4	5
15.	Follows continuous assessment approach/የተከተታታይ ምዘናን በትክክል ይተገብራል/ትተገብራለች፡፡	1	2	3	4	5
16.	Gives tutorial for low performing students/የምዘና ውጤት መሰረት በማድረግ በዉጤታቸው ዝቅ ላሉ ተማሪዎች ቲቶሪያል ይሰጣል፡፡	1	2	3	4	5
17.	Explains assessment modes/የሚጠቀምባቸውን የምዘና ዘዴዎች ከነዋጋቸው በቅድሚያ ለተማሪዎች ይገልጻል/ትገልፃለች	1	2	3	4	5
18.	Prepares exams as per the course content/የፈተና ጥያቄዎችን ከስተማረው ትምህርት ጋር ያዘምዳል/ታዘምዳለች፣	1	2	3	4	5
19.	Allocates appropriate marks for exam questions/በሚሰጣቸው ፈተናዎች ለሚጠይላሉ ጥያቄዎች ሚዛናዊ የሆነ ዋጋ ይመድባል/ትመድባለች፣	1	2	3	4	5
20.	Allocates enough exam time/ለሚያወጣቸው ፈተናዎች የተመጣጠነ ጊዜ ይመድባል/ትመድባለች፣	1	2	3	4	5
21.	Exams coverage of the course contents/የፈተና ጥያቄ ሲያወጣ ሁሉንም ርዕሰችን ይዳስሳል/ትዳስሳለች፣	1	2	3	4	5
22.	Gives feedback on continuous assessments/የተከተታይ ምዘና ዉጤትን ግብረ መልስ በለጮር ጊዜ ዉስጥ ያሳውቃል/ታሳውቃለች፣	1	2	3	4	5
23.	Exam includes various assessment modes/ፈተና ሲያወጣ የተለያዩ የምዘና አይነቶች ያካትታል/ታካትታለች፣	1	2	3	4	5
24.	Prepares supplementary exam for low performing students in continuous assessments/ተከታታይ ምዘና ዉጤት መሰረት በትምህርታቸው ዝቅተኛ ውጤት ላመጡ ተማሪዎች ድጋሚ ፈተና ይፈትናል/ትፈትናለች፤	1	2	3	4	5
25.	Gives tutorial for women and special needs students/ለሴትና ድጋፍ ለሚሹ ተማሪዎችና ቲቶሪያል ሰመስጠት ይደግፋል/ትደግፋለች፡፡	1	2	3	4	5
C.	**Ethical Competence/የባህሪ ብቃት**	1	2	3	4	5
26.	Gives respect to students/ለተማዎች ተገቢውን ክብር ያሳያል/ታሳያለች፣					
27.	Allows students to interact during class room/በክፍል ውስጥ ተማሪዎች ስለትምህርቱ ሀሳባቸውንና አስተያየታቸውን በነጻነት እንዲገልጹ ይፈቅዳል/ትፈቅዳለች፣	1	2	3	4	5
28	Listens to students problems/የተማሪዎችን ችግር ለማዳመጥ ፈቃደኛ ነው/ነች፣	1	2	3	4	5
29	Ethics, behavior and commitment for knowledge transfer/በስነ ስርዓት አክባሪነቱ፣ በሚያሳየው/በምታሳየው ጨዋነቱና እውቀትን ለማስተላለፍ ከሚያሳየው ቅንነትና ተነሳሽነት	1	2	3	4	5
30	Partiality based on ethinic, religion or gender/በዘር፣ በሃይማኖትና በፆታ ልዩነት የማይፈጥር/የማትፈጥር	1	2	3	4	5
D.	**Time Management/የጊዜ አጠቃቀም**	1	2	3	4	5
30.	Use class beginning and ending time appropriately/በተመደበለት የትምህርት ክፍለ ጊዜ ሰዓት አክብሮ ይገኛል፣ሰዓቱን በሚገባ ይጠቀማል/ትጠቀማለች					
31.	Use class time appropriately/ክፍለ ጊዜውን በአግባቡ ለማስተማር ተግባር ያውላል/ታውላለች፣	1	2	3	4	5
32.	Gives time for practical sessions/የተግባር ትምህርት ያለው ከሆነ ሙሉ ጊዜዉን በአግባቡ ተጠቅሞ ያስተምራል/ታስተምራለች፡፡	1	2	3	4	5
33.	Informs consultation hour/ተማሪዎችን ለመርዳት የማማከሪያ ሰዓት መድቦ ያሳውቃል/ታስታውቃለች፣	1	2	3	4	5
34	Solves students academic problems on time/ተማሪዎችን ለመርዳት በመደበው/በመደበችው ሰዓት ተማሪዎች ለሚያቀርቡት የአካዳሚክ ችግሮች መፍትሄ ይፈልጋል/ትፈልጋለች፣	1	2	3	4	5
	አጠቃላይ ዉጤት					

Compiled by:_____ **Approved by:**_____

2.7 Strength and Weakness of the Existing System or Application

2.7.1 Strengths of the Existing System

The current paper-based feedback process has the following strengths:

1. Low Initial Setup Requirement
   The university does not need computers or advanced technical infrastructure to conduct feedback.
2. Familiar to Users
   Both students and staff are already used to the paper evaluation process, which reduces confusion.
3. Official Documentation
   Physical forms serve as tangible records that can be stored for auditing when needed.
4. No Need for Technical Skills
   Students and staff with limited digital literacy can still participate easily.

2.7.2 Weaknesses of the Existing System

Despite its functionality, the current approach has several major weaknesses:

1. High Cost of Paper and Printing
   The process requires thousands of forms every semester, leading to unnecessary financial burden.
2. Time-Consuming Process
   Feedback collection, sorting, and summarizing take a long time, delaying decision-making.
3. Risk of Data Loss and Damage
   Paper forms can be misplaced, torn, or destroyed by accident, reducing information reliability.
4. Limited Privacy and Honesty
   Students may hesitate to give true feedback due to fear of being recognized, affecting quality of results.
5. Low Efficiency in Data Analysis
   Manual summarization increases workload and can result in errors during result compilation.
6. Lack of Service Coverage
   Feedback is mainly for academic evaluation — services like cafeteria and dormitory are not included.
7. Storage Space Requirement
   Physical documents must be stored for years, causing space management issues.
8. Not Easily Searchable
   Retrieving old feedback for reference or analysis is difficult and time-consuming.
9. Human Error Vulnerability
   Mistakes may occur during distribution, collection, counting, and recording of forms.

2.8 Alternative Solutions

Before introducing the proposed web-based Student Feedback Management System, several alternative solutions can be considered to improve the existing manual feedback process. These alternatives aim to reduce delays, improve data handling, and increase accessibility, but each has its own limitations.

1. Improving the Existing Paper-Based System

The university could improve the current paper-based process by using standardized and well-designed feedback forms across all departments. Additional staff could be assigned to collect, sort, and summarize feedback to reduce delays. Better storage and filing practices, such as labeled folders and secure cabinets, could help organize documents more effectively. Secure drop boxes could also be placed in departments to increase student privacy during submission.

Limitation: Despite these improvements, the process would remain costly, time-consuming, and highly dependent on manual work, making it prone to human errors and delays.

2. Excel-Based Digital Data Entry

Another option is to continue collecting feedback on paper but enter the results into Microsoft Excel or similar spreadsheet tools. This allows faster calculations, automatic charts, and basic summary reports. Digital storage would also reduce the need for large physical storage spaces.

Limitation: This approach still depends on paper forms and manual data entry, which is time-consuming and can introduce typing errors and inconsistencies.

3. Online Survey Tools (Google Forms / Microsoft Forms)

The university could use online survey tools such as Google Forms or Microsoft Forms to collect student feedback. These tools are easy to set up, support basic data collection, and automatically generate summary charts. They also reduce paper usage and allow online access.

Limitation: These tools offer limited customization, weak role-based access control, and limited integration with university systems. They do not fully support features such as secure authentication, feedback assignment to responsible staff, detailed reporting, or long-term structured data management.

Chapter Three

REQUIREMENT SPECIFICATION AND ANALYSIS

3.1 Description of the Proposed System

The proposed Student Feedback Management System is a web-based application designed to digitalize and improve the entire feedback process at Wachemo University. The system allows students to provide feedback on instructors, courses, dormitory, cafeteria, and basic student services through a secure online platform.

It eliminates paper usage, reduces delays, increases data accuracy, ensures privacy, and enables administrators to easily generate result reports for decision-making. The system provides role-based access for students, instructors, department heads, academic officers, and university management.

3.1.1 User Characteristics

Table7: The system users include individuals with different technical and administrative capabilities:

User Type	Characteristics	Role in System
Students	Basic computer and smartphone knowledge	Submit feedback for assigned courses and services
Instructors	Able to view digital reports	View summarized feedback results for performance improvement
Department Heads & Academic Officers	Familiar with academic administration	Monitor feedback participation, manage instructors and academic units

University Management	Decision-makers, limited system interaction time	Access high-level reports for policy and improvement planning
System Administrator	Technical expertise	Manage users, system configuration, security, and database

3.1.2 Constraints

The proposed Student Feedback Management System will operate under several limitations to ensure secure, reliable, and controlled usage:

Technical Constraints:
The system requires a stable network connection and the availability of Wachemo University's infrastructure, including computers, local servers, and internet access, to function effectively.

Authentication Constraints:
Only registered WCU students and authorized staff can log in using valid university-issued credentials. This ensures that feedback is submitted and managed only by legitimate users.

Resource Constraints:
The system's hosting capacity is limited, which may affect performance if a sudden increase in users occurs, such as during peak feedback submission periods at the end of the semester.

Device Accessibility:
Students who do not have personal devices or internet access must use university computer labs to submit feedback, which may limit convenience for some users.

Time Constraints:
Feedback submission is allowed only during evaluation periods defined by the university calendar. Submissions outside these periods will not be accepted, ensuring orderly and consistent collection of feedback.

3.1.3 Assumptions and Dependencies

The development and successful operation of the Student Feedback Management System rely on several assumptions and dependent conditions:

User Participation Assumption:
It is assumed that students will actively submit their feedback within the deadlines when the system is open. Without timely participation, the feedback data may be incomplete or less representative.

Administrative Support Assumption:
Academic officers, department heads, and Quality Assurance staff are expected to properly manage feedback cycles, assign responsibilities, and review reports to ensure the system's effectiveness.

System Maintenance Dependence:
Continuous IT support is required for tasks such as backups, security updates, and troubleshooting. This ensures the system remains reliable, secure, and operational at all times.

University Policy Dependence:
The system depends on university management integrating feedback results into performance evaluation, training programs, and service improvement plans. Without policy support, the feedback collected may not lead to actionable changes.

Infrastructure Assumption:
It is assumed that the university will provide adequate network access, server resources, and devices for students and staff when needed. This infrastructure is essential for smooth system operation and accessibility.

3.2 Requirement Specifications

3.2.1 Functional Requirements

The following functional requirements describe what the system shall do in order to support the digital feedback process at Wachemo University.

A. Student Functional Requirements

FR1:The system shall allow students to register an account using their university ID.
FR2:The system shall allow students to log in securely using a username and password.
FR3:The system shall allow students to reset their password if forgotten.
FR4:The system shall display available feedback categories including courses, instructors, cafeteria, library, and dormitory services.
FR5:The system shall allow students to submit course feedback for the courses they are enrolled in.
FR6:The system shall allow students to submit instructor feedback for instructors assigned to their department and year level.
FR7:The system shall allow students to submit feedback for cafeteria services.
FR8:The system shall allow students to submit feedback for library services.
FR9:The system shall allow students to view their previously submitted feedback history.
FR10:The system shall allow students to update their personal profile information.
FR11:The system shall allow students to log out of the system securely.

FR12:The system shall allow a logged-in student to submit feedback or complaints related to dormitory facilities and store the feedback securely in the database

B. Administrative and Management Functional Requirements

FR13:The system shall allow the system administrator to manage student accounts including approving, blocking, and deleting accounts.
FR14:The system shall allow administrators to view all submitted feedback.
FR15:The system shall allow administrators to filter feedback by category such as course, instructor, cafeteria, library, or department.
FR16:The system shall allow administrators to generate summarized feedback reports with graphs and statistics.
FR17:The system shall allow administrators to assign feedback-related tasks to responsible staff such as department heads or quality assurance officers.

FR18: The system shall allow responsible staff to view only the feedback that has been officially assigned to them for review
FR19:The system shall allow authorized staff to analyze aggregated student feedback and view summary reports to support decision-making.

3.2.2 Non-Functional Requirements

Security:
Our system is secured using PHP authentication, password hashing, and MySQL encryption to protect sensitive student feedback and account information. Role-based access is implemented in PHP, ensuring that only authorized staff can view reports or manage data. Input validation through JavaScript prevents unauthorized scripts or SQL injection, keeping feedback data safe and confidential.

Performance:
The system performs efficiently, with pages loading within 3 seconds and support for multiple students submitting feedback simultaneously. This is achieved through optimized PHP backend queries, indexed MySQL tables, and lightweight HTML/CSS/JS frontend design, which reduces server load and ensures smooth user experience even during peak submission times.

Usability:
The system is simple and intuitive, with a clean interface built using HTML and CSS, and interactive features handled by JavaScript. Form validations, tooltips, and step-by-step feedback submission processes ensure that students and staff with varying technical skills can navigate the system easily and complete tasks without errors.

Reliability:
The system ensures data reliability by storing feedback in real-time using MySQL, with PHP scripts handling transaction control to avoid data loss. Automatic error handling and logging capture failures during submission or server interruptions. This ensures that feedback records are preserved even if unexpected interruptions occur.

Maintainability:
The system is modular, separating frontend (HTML, CSS, JS) from backend (PHP, MySQL). PHP functions are organized by feature modules, and database tables are normalized to reduce redundancy. This structure allows developers to update features, fix bugs, or add new feedback types with minimal risk to existing functionalities.

Portability:
The system runs on all major browsers (Chrome, Firefox, Edge) due to standard HTML, CSS, and JS usage. PHP backend and MySQL database are compatible with any server environment supporting the LAMP stack, ensuring that the platform can be deployed across different university servers or accessed from multiple devices without compatibility issues.

Scalability:
The system is designed to grow with increasing users, feedback submissions, and university departments. Its Three-Tier Architecture allows each layer to scale independently — web servers can be expanded to handle more traffic, and the database server can grow to store larger datasets. This structure ensures new feedback categories, services, or user roles can be added without redesigning the system.

Privacy:
Student feedback is confidential, with PHP enforcing role-based access so only authorized personnel view aggregated reports. MySQL stores feedback without exposing personal identifiers, and JavaScript ensures secure client-side input handling. This encourages honest and constructive feedback while protecting student identity.

3.3.System Modeling

3.5.1 Actor Identification

- ➢ Student
- ➢ Admin
- ➢ Responsible Staff
  - Department Head
  - Cafeteria Head
  - Dormitory Head
  - Library Manager

3.5.2 Use-Case Identification

Student Use Cases

1. Register Account
2. Login
3. Reset Password
4. View Available Feedback Categories
5. Submit Course Feedback
6. Submit Instructor Feedback
7. Submit Cafeteria Feedback
8. Submit Library Feedback
9. Submit Dormitory Feedback
10. View Feedback Submission History
11. Update Profile

12. Logout

Administrator Use Cases

13. Manage Student Accounts (Approve, Block, Delete)
14. View All Submitted Feedback
15. Filter Feedback by Category
16. Generate Feedback Reports
17. Assign Feedback to Responsible Staff

Responsible Staff Use Cases

18. View Assigned Feedback for Review
19. Analyze Feedback Reports for Courses, Instructors, or Services

3.5.3 Use-Case Diagram

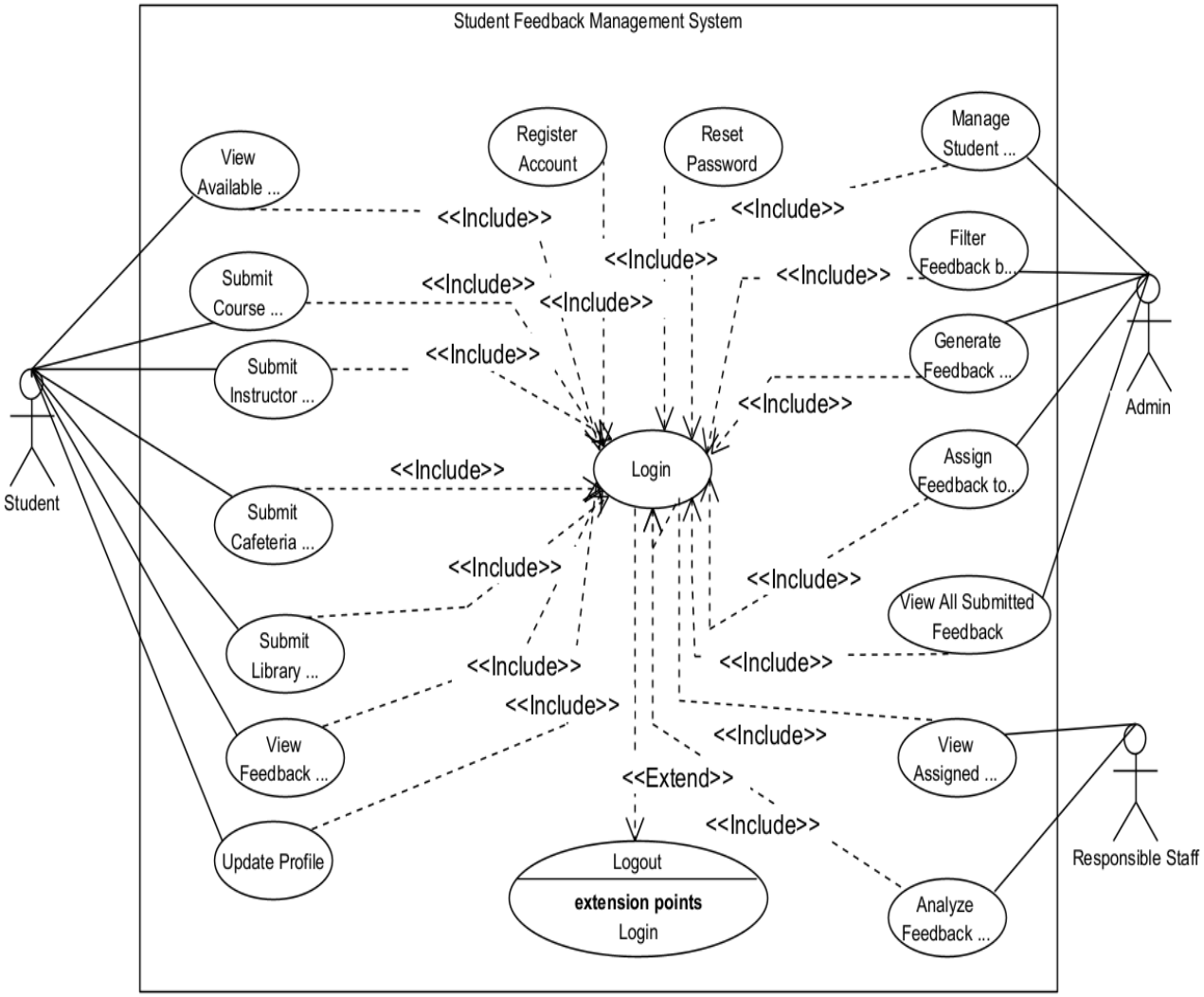


Figure 2: Use case diagram

3.5.4 Description of Use-Case

Use Case ID	Uc-01	
Use Case Name	Register Account	
Actor	Head of Department,Instructor,Student,Responsible staff	
Description	This use case describes how a new user (student, staff, or cafeteria head) at Wachemo University registers an account in the Student Feedback Management System.	
Precondition	1. The user has not previously registered in the system. 2. The system is available and connected to Wachemo University's network. 3. The user has basic information required for registration (ID, department, batch, phone/email)	
Basic Scenario	Actor Action	Actor Action
	Step1: The user opens the website. **Step3:** The actor clicks Create Account **Step5:** The actor fills out all required fields and clicks Submit **Step10:** The actor enters the OTP.	**Step 2:** The system displays the Home Page with options: "Create Account", and " **Step 4:** The system displays the Registration Page with required fields **Step 6:** The system validates that all required fields are completed and properly formatted. **Step 7:** The system requests to check for duplicate phone numbers and The system validates the response information against stored user data. **Step 8:** The system sends a One-Time Password (OTP) via email to the provided phone number. **Step 9:** The system displays the OTP Verification Page requesting the 6-digit OTP. **Step 11:** The system verifies OTP accuracy and expiration. **Step 12:** The system creates the new account, stores all user data, and displays the Account Successfully Created. **Step 13:** The system redirects to the Login page **Step 14**: use case end
Alternative	1. **Invalid Field Input:** At Step 3 or 4, if any field format is invalid, the system highlights the field, displays an appropriate error message, and returns to the same step. 2. **Duplicate Student ID / Email / Phone:** At Step 6, if the Student ID, email, or phone number already exists, the system shows: "This account already exists" and returns to Step 3. 3. **OTP Not Delivered:** At Step 7, if OTP delivery fails, the system displays: "OTP	

	could not be sent. Please try again" and returns to Step 7. 4. **Invalid OTP:** At Step 8, if OTP is incorrect or expired, the system shows: "Invalid OTP. Please re-enter" and return to Step 8.
Postcondition	1. A new account is successfully created and ready for login. 2. User information is securely stored in the system database.

Table8: Use case Description for Register Account

Use Case ID	Uc-02	
Use Case Name	Login	
Actor	,Instructor, Student,Admin,responsible staff	
Description	This use case describes how a login user at **Wachemo University,**logs into the **Student Feedback Management System (SFMS)**.	
Precondition	1. The user has a registered and verified account in the system. 2. The system is online and connected to Wachemo University servers. 3. The user knows their valid login credentials (Student ID / Staff ID / Email / Phone number and password).	
Basic Scenario	Actor Action	System Response
	Step1: The user opens the website. **Step3:The user clicks the login button.** **Step5:** The user enters their username and password. **Step6:** The user clicks the **Click submit**.	**Step2:** The system displays the HomePage. **Step4:**The system displays the Login Page. **Step7:** The system validates that all required fields are filled. **Step8:** The system verifies the user against the database. **Step9**: The system redirects to the users dashboard. **Step10:** use case end.
Alternative	**1.**If the username or password is incorrect, the system displays **"Invalid username or password"** and returns to **Step 4**. **2.**If required fields are empty, the system displays **"Please fill in all required fields"** and returns to **Step 4**.	
Postcondition	1. The user is successfully authenticated. 2. A secure user session is created and remains active until logout or session timeout. 3. The user gains access to system features based on their assigned role.	

Table9 Use case Description for Login

Use Case ID	Uc-03	
Use Case Name	Reset Password	
Actor	Student, Instructor,,Admin,student	
Description	This use case describes how a registered user of the Wachemo University Student Feedback Management System resets a forgotten password. The system allows users to securely reset their password using a verification code (OTP) sent via registered email or phone number,	
Precondition	1. The user is already registered in the system. 2. The user has access to the registered email address or phone number. 3. The system is online and operational.	
Basic Scenario	Actor Action	System Response
	Step1: The user opens the dashboard. **Step3:** The user clicks on the "Forgot Password?" **Step6:** User enters their registered email address or phone number. **Step7:** The user clicks the "Send OTP" button. **Step10:** User enters the received OTP code in the OTP Verification page**.** **Step12:** The user enters a new password and confirms it in the reset password form.	**Step2:** System displays the Login Page with a "Forgot Password?" l **Step4:** System displays the Reset Password Page. **Step5:** System displays the reset password form. **Step8:** System validates the entered information. **Step9:** The system generates an OTP and sends it to the user's registered phone number. **Step11:** System verifies the OTP. **Step13:** System checks password strength and matching. **Step14:** System updates the password successfully in the database. **Step16:** System displays a success message: "Password reset successfully. Please log in." **Step18:** use case end.
Alternative	1. If the email or phone number is not registered, the system displays an error message and returns to Step 4. 2. If the OTP is invalid or expired, the system displays "Invalid or expired OTP" and returns to Step 5. 3. If the new password and confirmation do not match, the system shows an error and returns to Step 12.	

postcondition	1. The user's password is successfully updated in the system database. 2. The user can log in using the new password. 3. System security and data integrity are maintained.

Table 10: Use case Description for Reset Password

Use Case ID	Uc-04	
Use Case Name	View Available Feedback Categories	
Actor	Student	
Description	This use case describes how a student at Wachemo University, views the list of available feedback categories (such as Course Content, instructor	
Precondition	1. The student is registered in Wachemo University. 2. The student has a valid system account. 3. The student is logged into the Student Feedback Management System. 4. Feedback categories have been created by the Admin or relevant authority.	
Basic Scenario	Actor Action	System Response
	Step1: Student opens a web browser or campus system and accesses the Student Feedback Management System login page. **Step3**: Student enters valid username and password and clicks Login. **Step5:** Student selects Feedback from the main menu. **Step7:** Student clicks on View Feedback Categories. **Step9:** Student views the list of feedback categories.	**Step2:** System displays the Login Page **Step4:** System authenticates the student and displays the Student Dashboard. **Step6**: System displays the Feedback Menu Page. **Step8:** System retrieves available feedback categories from the database. **Step10:** System displays the Feedback Categories Page showing category names and brief descriptions**.** **Step11:** use case end
Alternative	1. If the student session has expired, the system displays a session timeout message and redirects to the Login Page, returning to Step 1. 2. If no feedback categories are available, the system displays: *"No feedback categories are currently available"* and remains on the Feedback Categories Page, returning to Step 5.	
Postcondition	1. The student successfully views the available feedback categories. 2. The system remains ready for the student to select a category and proceed with submitting feedback.	

Table 11: Use case Description for View Available Feedback Categories

Use Case ID	Uc-05	
Use Case Name	Submit Course Feedback	
Actor	Student	
Description	A student at Wachemo University submits feedback about a course using the Student Feedback Management System.	
Precondition	1. The student is registered at Wachemo University. 2. Student is enrolled in the course. 3. Course feedback period is open.	
Basic Scenario	Basic Scenario	Actor Action
	Step1: Student logs into the system. **Step3;** Student selects **Course Feedback**. **Step5:**Student selects course and semester. **Step7:**Student rates course and enters comments**.** **Step9**: Student submits feedback	**Step2:** System validates credentials and shows dashboard. **Step4:** System displays the Course Feedback page. **Step5;** The system displays the Instructor feedback form. **Step8:** System loads course feedback form. **Step9:**System validates required fields. **Step10:** system save feedback and shows confirmation **Step11:** use case end.
Alternative	1. If the student submits the form without rating the course or entering required comments, the system displays *"Please complete all mandatory fields"* and returns to **Step 4**.	
Postcondition	1. Course feedback is stored in the system. 2. Feedback is available to the department/instructor. 3. Student receives confirmation message	

Table12:Use case Description for Submit Course Feedback

Use Case ID	Uc-06
Use Case Name	Submit Instructor Feedback
Actor	Student

Description	This use case describes how a student at Wachemo University, submits digital feedback about an instructor through the Student Feedback Management System.	
Precondition	1. The student is registered at Wachemo University. 2. The student has a valid system account. 3. The student is logged into the Student Feedback Management System. 4. The student is enrolled in at least one course with an assigned instructor. 5. The instructor feedback submission period is open.	
Basic Scenario	Actor Action	System Response
	Step1: The student opens the dashboard. **Step3;** The student selects "Submit Instructor Feedback" from the dashboard menu. **Step6:The student fills the form** and writes comments. **Step7:** The student clicks the "Submit Feedback" button.	**Step2:** The system displays the Student Dashboard. **Step4:** The system displays the Instructor Feedback Page. **Step5;** The system displays the Instructor feedback form. **Step8:** The system validates that the feedback field is not empty. **Step9:The system saves the feedback in the database.** **Step10:** The system displays: "Instructor feedback submitted successfully" **in the** Instructor **feedback form.** **Step11:** use case end.
Alternative	1. If the feedback text area is empty, the system displays **"Display error messages"** and returns to **Step 5**.	
Postcondition	1. The instructor feedback is securely stored in the system database. 2. Feedback becomes accessible to authorized academic management for evaluation and improvement. 3. The student is prevented from submitting duplicate feedback for the same instructor.	

Table 13: Use case Description for Submit Instructor Feedback

Use case ID	UC-07
Use case Name	Submit cafeteria feedback
Actor	Student

Description	This use case describes how a student at Wachemo University submits digital feedback about cafeteria services through the Student Feedback Management System.	
Precondition	1. The student is registered at Wachemo University. 2. The student has a valid system account. 3. The student is logged into the Student Feedback Management System. 4. The cafeteria feedback submission period is open.	
Basic Scenario	Actor Action	System Response
	Step1: The student opens the dashboard. **Step3;** The student selects "Submit Cafeteria Feedback" from the dashboard menu. **Step6:The student fills the form** and writes comments. **Step7:** The student clicks the "Submit Feedback" button.	**Step2:** The system displays the Student Dashboard. **Step4:** The system displays the Cafeteria Feedback Page. **Step5;** The system displays the cafeteria feedback form. **Step8:** The system validates that the feedback field is not empty. **Step9:The system saves the feedback in the database.** **Step10:** The system displays: "Cafeteria feedback submitted successfully" **in the cafeteria feedback form.** **Step11:** use case end.
Alternative	1. If the cafeteria text area is empty,the system displays "Display error messages" and returns to Step 5.	
Postcondition	1. The cafeteria feedback is securely saved in the system database. 2. Feedback becomes available to authorized cafeteria management and university administrators. 3. The student is prevented from submitting duplicate cafeteria feedback for the same period.	

Table 14:Use case Description for Submit cafeteria feedback

Use case ID	Uc-08
Use case Name	Submit Library Feedback

Actor	Student	
Description	This use case describes how a **student at Wachemo University, Hossana** submits feedback about library facilities (e.g., book availability, seating, internet, cleanliness) through the **Student Feedback Management System (SFMS)**.	
Precondition	1. The student is registered and logged into the SFMS. 2. The student is enrolled at Wachemo University. 3. The system is online and operational.	
Basic Scenario	Actor Action	System Response
	Step1: The student opens the dashboard **Step3:** The student selects "Library Feedback" from the dashboard menu. **Step6:** The student fills the form and writes comments. **Step7:** The student clicks the Submit Feedback button**.**	**Step2:** The system displays the Student Dashboard. **Step4:** The system opens the Library Feedback Page. **Step5:** The system displays the library feedback form. **Step8:** The system validates that the feedback field is not empty. **Step9:** The system saves the feedback, assigns a unique feedback ID, records submission date and time in the database. Step10:The system displays feedback submitted successfully in the library feedback form. **Step11:** use case end.
Alternative	1. If the feedback text area is empty, the system displays **"Display error messages"** and returns to **Step 5**.	
Postcondition	1. The library feedback is stored securely in the SFMS database. 2. Authorized staff (Library Admin / Responsible Staff / Admin) can access and respond to the feedback. 3. The student can track the feedback status under **My Feedback**.	

Table 15:Use case Description for Submit Library Feedback

Use case ID	Uc -09

Use case Name	Submit Dormitory Feedback	
actor	Student	
Description	This use case describes how a student at Wachemo University submits feedback or complaints regarding dormitory facilities (e.g., water, electricity, cleanliness, maintenance, security) through the Student Feedback Management System (SFMS)	
Precondition	1. The student is registered and logged into the SFMS. 2. The student is officially assigned to a dormitory block and room in the university database. 3. The system is online and operational	
Basic Scenario	Actor Action	System Response
	Step1: The student opens the dashboard **Step3:** The student selects "Dormitory Feedback" from the dashboard menu. **Step6:the student fills the form and writes comments.** **Step7:the student clicks the sub,its feedback button.**	**Step2:** The system displays the Student Dashboard. **Step4:** The system opens the Dormitory Feedback Page. **Step5:** The system displays the dormitory feedback form. **Step8:** The system validates that the comments field is not empty **Step9:** The system saves the feedback, assigns a unique feedback ID, records submission date and time, **Step10:**displays "Feedback submitted successfully". **Step11:** use case end.
Alternative	1. If the feedback text area is empty,the system displays"Display error message"and returns to step5.	
Postcondition	1. The dormitory feedback is securely stored in the SFMS database. 2. Authorized staff (Admin / Head of Department / Dormitory Management) can access and respond to the feedback. 3. The student can track the status of submitted feedback under **My Feedback**	

Table 16:Use case Description for Submit Dormitory Feedback

Use case ID	Uc-10

Use case Name	Update profile	
actor	Student,admin , responsible staff	
Description	This use case describes how a registered user at Wachemo University, updates their personal information (e.g., name, phone number, email, department, or password) in the Student Feedback Management System (SFMS).	
Precondition	1. The user is registered and logged into the SFMS. 2. The system is online and operational. 3. The user has valid credentials and access to their profile	
Basic Scenario	Actor action	System response
	Step1: The user opens the dashboard. **Step3:** The user navigates to "Update Profile" from the dashboard menu. **Step6:** The user updates the necessary fields (e.g., phone, email, department, password). **Step7**: The user clicks the **Save Change** button.	**Step2:** The system displays the User Dashboard. **Step4**: The system opens the Profile Information Page displaying current user details. Step5:The system displays the edit profile page. **Step8**: The system validates that all required fields are filled and checks for valid formats (e.g., email, phone number). **Step9:** The system updates the user information in the database and displays "Profile updated successfully". **Step10:** use case end.:System displays a success message "Profile updated successfully " messages in the Edit profile page. **Step11:** use case end.
Alternative	1. If a required field is empty and system or network error occurs the system displays "Display error messages" and returns to Step 5.	
Postcondition	1. The user's profile information is updated securely in the SFMS database. 2. Updated information is immediately available for system operations. 3. The user receives confirmation of the successful update.	

Table 17:Use case Description for Update profile

Use case ID	Uc-11

Use case Name	Manage Student Accounts	
actor	Admin	
Description	This use case describes how an Admin at Wachemo University manages student accounts within the Student Feedback Management System (SFMS).	
Precondition	1. The admin is registered and logged into the SFMS. 2. The system is online and operational. 3. Student data exists in the university database or is ready to be added.	
Basic Scenario	Actor action	System response
	Step1: Admin open dashboard. **Step3:** Admin navigates the student management page from the dashboard menu. **Step5:** Admin selects a student account to manage. **Step7:** Admin choose an action: Create, Update, Deactivate, or Delete student account. **Step7:** Admin enters or modifies the student details (ID, name, email, department, role, status) **Step8:** Admin confirms the action.	**Step2:** The system admin displays the Admin Dashboard. **Step4:** The system displays the Student Management Page. **Step6:** The system displays the Student account details Page. **Step9:** The system validates all required fields and checks for duplicates. **Step10:** The system updates the account status in the database. **Step11:System displays success messages:"Account updated successfully".** **Step12:** use case end.
Alternative	1. If required fields are missing, the system displays **"Display error messages"** and returns to **Step 6**.	
Postcondition	1. Student accounts are updated, created, deactivated, or deleted securely in the SFMS database. 2. Changes are immediately reflected in system operations. 3. Admin receives confirmation of completed actions	

Table 18:Use case Description for Manage Student Accounts

Use case ID	Uc-12

Use case Name	logout	
actor	student , responsible staff , instructor	
Description	This use case describes how a logged-in user at Wachemo University, securely logs out from the Student Feedback Management System (SFMS).	
Precondition	1. The user is registered and currently logged into the SFMS. 2. The system is online and operational	
Basic Scenario	Actor Action	System Response
	Step1:open dashboard Step3: The user clicks the Logout option from the top navigation menu **Step5:** The user confirms logout by clicking Yes / No	**Step2:** System displays user dashboard. **Step4: System validates the logout request** **Step 6:**The system prompts for confirmation: “Are you sure you want to logout? **Step7**: The system terminates the current session, clears session data, and redirects the user to the Login Page. **Step8**:End Basic Scenario.
	1. If the user clicks No at the confirmation prompt, the system aborts the logout process and returns to Step 2. 2. If a system or network error occurs during logout, the system displays “Logout failed. Please try again and return to Step 2.”	
Postcondition	1. The user session is terminated securely. 2. All sensitive session data is cleared from the system. 3. The user is redirected to the Login Page and cannot access protected pages without logging in again.	

Table 19:Use case Description for logout

Use case ID	Uc-13

Use case Name	Filter Feedback by Category	
actor	Admin,Responsible Staff	
Description	This use case describes how authorized staff at Wachemo University filter submitted student feedback by category (e.g., Library, Dormitory, Cafeteria, etc) using the Student Feedback Management System (SFMS).	
Precondition	1. The user is registered and logged into the SFMS. 2. The user has authorization to view submitted feedback. 3. Feedback records exist in the system database. 4. The system is online and operational.	
Basic Scenario	Actor Action	System Response
	Step1: The user opens the dashboard. **Step3**: The user navigates to feedback management from the dashboard menu. **Step5:** The user clicks a feedback category from the Category Filter dropdown**.** **Step7:** The user selects a feedback category from the filtered feedback result page.	**Step2**: The system displays the user Dashboard. **Step4**: The system displays a Feedback management Page. **Step6**: The system displays a filtered feedback result page. **Step8:** The system validates the filtered feedback. **Step9**: The system displays a filtered feedback list with student name,date,feedback details and rating. **Step10**: use case end
Alternative	1. If no feedback exists for the selected category and the system fails to apply the filter, the system displays “Display error messages” and returns to Step 4.	
Postcondition	1. Feedback is successfully filtered based on the selected category. 2. No feedback data is modified. 3. The user gains improved visibility and faster access to relevant feedback.	

Table 20:Use case Description for Filter Feedback by Category

Use case ID	Uc-14

Use case Name	Generate Feedback Reports	
actor	Admin, Responsible Staff	
Description	This use case describes how authorized staff at Wachemo University, Hossana generate feedback reports from the Student Feedback Management System (SFMS).	
Precondition	1. The user is registered and logged into the SFMS. 2. The user has permission to generate reports. 3. Feedback records exist in the system database. 4. The system is online and operational.	
Basic Scenario	Actor Action	System Response
	Step1: The user logs into the SFMS. **Step3**: The user navigates**"Reports Generate page"** from the dashboard menu. **Step5**: The user selects report criteria (feedback category, date range, status, department). **Step6**: The user clicks the Generate Report button. **Step11:The user views the generated report on the feedback generation page.**	**Step2**: The system displays the Admin/Staff Dashboard. **Step4**: The system displays the Report Generation Page. **Step7**: The system validates the selected criteria **Step8**: The system processes the data and generates a feedback report. **Step10**: The system displays the report. **Step11**: use case end.
Alternative	1. If no feedback matches the selected criteria, the system displays **"No data available for the selected criteria"** and returns to **Step 3**.	
Postcondition	1. Feedback reports are successfully generated. 2. The report is available for viewing, downloading, or printing. 3. No feedback data is modified during report generation.	

Table 21: Use case Description for Generate Feedback Reports

Use case ID	Uc-15	
Use case Name	Assign Feedback for Responsible Staff	
actor	Responsive Staff	
Description	This use case describes how the Head of Department or Admin assigns submitted student feedback to the appropriate responsible staff member (e.g., instructor, cafeteria head, service office) through the digital Student Feedback Management System at Wachemo University, Hossana.	
Precondition	1. The actor is authenticated and logged into the system. 2. Student feedback has already been submitted and stored in the system. 3. Responsible staff members are registered in the system.	
Basic Scenario	Actor Action	System Response
	Step1: Users open dashboard. **Step3:**Actor navigates to feedback management page from the dashboard menu **Step5:**The user selects a feedback item to assign. **Step7:**The user selects the responsible staff member from the dropdown list. **Step8:**The user clicks the "Assign Feedback" button.	**Step2:** System displays the user Dashboard. **Step4:**System displays the Feedback Management Page. **Step6:**System displays the Assign Feedback Assignment Page with a staff list. **Step9:**System validates staff availability **Step10:**System saves the assignment and updates feedback status to "Assigned" in the database. **Step11:**System displays the Feedback successfully assigned to Users. **Step12:**End of Basic Scenario
Alternative	1. If no staff member is selected, the system displays "**Display error messages**" and **returns to Step 6**.	
Postcondition	1. Feedback is successfully assigned to a responsible staff member. 2. Feedback status is updated to Assigned. 3. Assigned staff receives a system notification for action.	

Table 22:Use case Description for Assign Feedback to Responsible Staff

Use Case ID	UC-16	
Use Case Name	View Assigned Feedback for review	
Actor	Responsible staff	
Description	This use case describes how an instructor at Wachemo University, Hossana views student feedback that has been officially assigned to them for review through the Student Feedback Management System. The system replaces the former manual paper-based feedback process, reducing delays, loss of feedback, and human errors.	
Precondition	1. The Responsible staff is registered in the system. 2. The Responsible staff has a valid username and password. 3. The Responsible staff is logged into the Student Feedback Management System. 4. Feedback has already been submitted by students and assigned to the Responsible staff by the system or Head of Department	
Basic Scenario	Actor Action	System Response

	Step1: Responsible staff opens the dashboard. **Step3**: Responsible staff selects Assigned Feedback list page from the dashboard menu. **Step5**: Responsible staff selects an academic year and semester.	**Step2**: System displays the dashboard. **Step4**: System opens the Assigned Feedback Page. **Step6**: System validates the request data. **Step7**: System filters and displays feedback assigned for the selected period. **Step8**: System opens the Feedback Detail Page. **Step9**: System displays full feedback details (course, category, comments, date). **Step10**:scenario end
Alternative Scenario	1.If no feedback is assigned to the Responsible staff and the selected academic year or semester has no records, the system displays “Display error messages” and remains on Step 4.	
Post Condition	1.Assigned feedback is successfully viewed by the Responsible staff. 2.The system records that the feedback has been accessed for review. 3.No data is modified during this use case	

Table 23:Use case Description for View Assigned Feedback for Review

Use case ID	UC – 17
Use Case Name	Analyze Feedback Reports
Actor	Responsive Staff

Description	This use case describes how authorized users analyze feedback reports related to courses, instructors, or university services (such as cafeteria or dormitory) in the Student Feedback Management System at Wachemo University.The system automatically aggregates and analyzes student feedback, replacing the previous manual report analysis process that was time-consuming, error-prone, and inefficient. The digital reports support data-driven decision-making and quality improvement.	
Precondition	1. The actor must be logged into the system. 2. The actor must have permission to view and analyze feedback reports. 3. Feedback data must already exist in the system. 4. The system must be online and connected to the university database.	
Basic Scenario	Actor Action	System Response
	Step1: Responsive staff logs into the student management system. **Step3**: Responsive staff selects Reports and Analytics from the main menu. **Step5**: Responsive staff chooses analysis criteria. **Step6**: Actor clicks Generate Report.	**Step2**: System displays the Responsive staff dashboard. **Step4**: System displays the feedback report page. Step7:System displays the analysis results page. **Step8**: System validates choose analysis criteria.. **Step9**: System display responsive staff views the generated report analysis. **Step10**: End of Basic Scenario
Alternative	1.If no feedback data exists for the selected criteria, the system shows: “Display error messages” and returns to Step 4.	
Post Condition	1. Feedback reports are successfully analyzed and displayed. 2.The actor gains insights into performance, strengths, and areas for improvement. 3.Reports may be exported or used for decision-making and quality improvement.	

Table 24:Use case Description for Analyze Feedback Reports

Use Case ID	UC – 18	
Use Case Name	View Feedback Submission History	
Actor	Student, Head of Department, Cafeteria Head, Admin	
Description	This use case allows a logged-in user to view a chronological list of all feedback or complaints they have previously submitted to the system, including the current status (e.g., Pending, Under Review, or Resolved).	
Precondition	1 The user is successfully logged into the system. 2 The user has submitted at least one feedback entry previously (to see data).	
Basic Scenario	Actor Action	System Response
	Step1: The user opens the dashboard. **Step3**: The user navigates to the feedback history page. **Step5**: The user selects the feedback category to view. **Step7**: The user clicks on any feedback item to view details.	**Step2**: The system displays a dashboard for users. **Step4**: The system displays the Feedback History Page **Step6**: The system displays a feedback detail page. **Step8**: The system validates the clicked data. **Step9**: The system displays the feedback detail page with complete information. **Step10**: End of Basic Scenario.
Alternative Scenario	1. No History Found: If the user has never submitted feedback, the system displays a message: "Display error messages" and provides a link to the feedback submission form. Return to **Step6**.	
Post Condition	1 The user is informed about the progress of their previous feedback. 2 The user can confirm if their issues are being addressed by the relevant department (e.g., Department Head or Campus Maintenance).	

Table 25:Use case Description for View Feedback Submission History

Use Case ID	UC-19	
Use Case Name	View All Submitted Feedback	
Actor	admin	
Description	This use case describes how authorized staff at Wachemo University view all student feedback that has been submitted through the digital Student Feedback Management System	
Precondition	1. Actors are registered and logged into the system. 2. Student feedback has already been submitted and stored in the system. 3. The actor has permission to view submitted feedback.	
Basic Scenario	Actor Action	System Response
	Step1: User logs into System. **Step3**: User selects View Submitted Feedback from the menu. **Step5**: User selects feedback category (course, instructor, cafeteria, department). **Step7**: User clicks to List.	**Step2**: System displays the Dashboard Page. **Step4**: System display the Submitted Feedback List Page. **Step6**: System displays the Feedback Detail Page. **Step8**: System validates the list. **Step9**: System display all Submitted Feedback in Feedback Detail Page. **Step10**: End of Basic Scenario.
Alternative Scenario	1. If no feedback exists in the system, the system displays: “Display error messages” and returns to Step4.	
Post Condition	1. The actor successfully views all available submitted feedback. 2. Feedback data remains unchanged.	

	3. Actors gain insight to improve academic or service quality.

Table 26:Use case Description for View All Submitted Feedback

3.4. Requirement Analysis

3.5.1. Activity Diagram

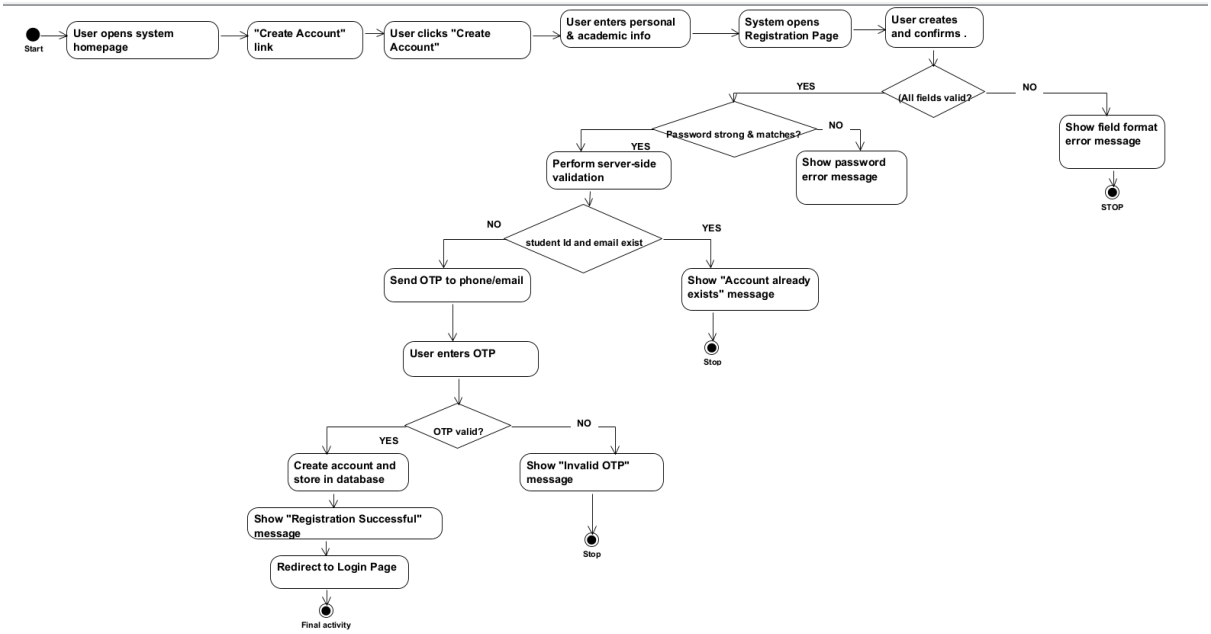


Figure 3: Diagram 1:Activity Registration

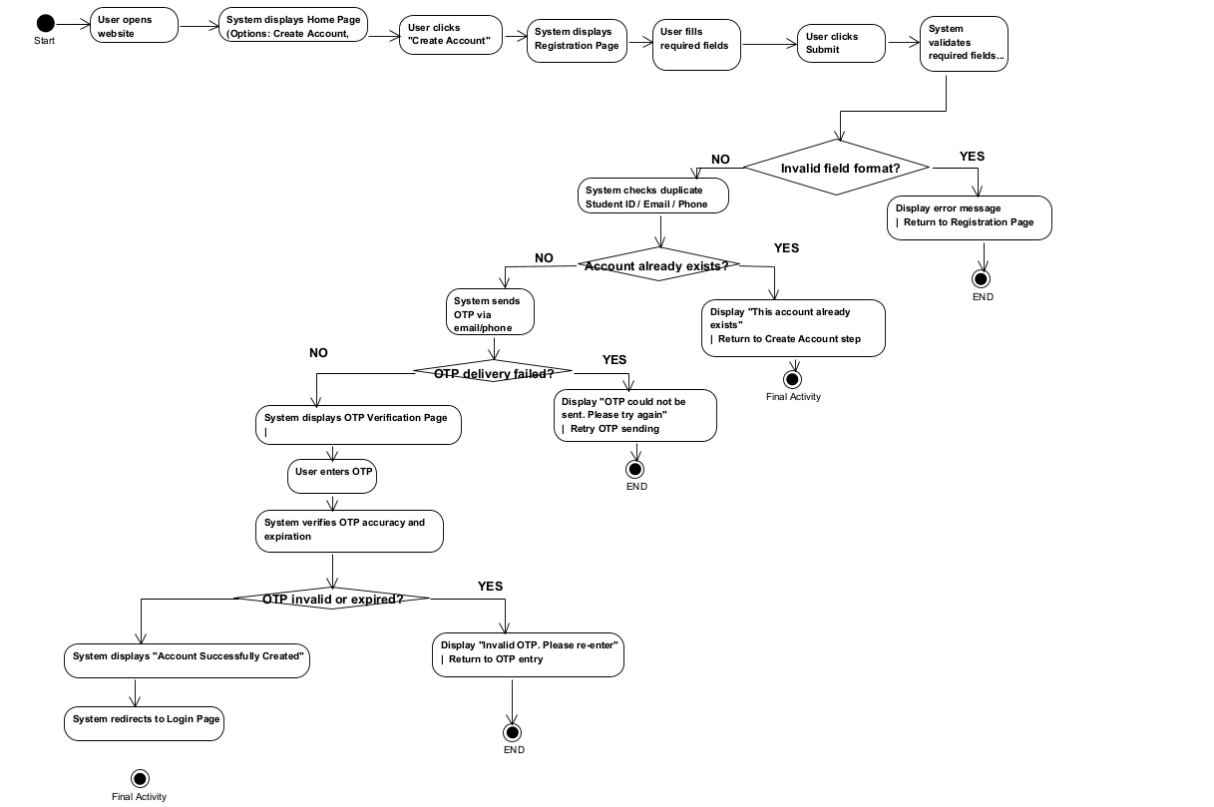


Figure 4: Diagram 2:Activity login

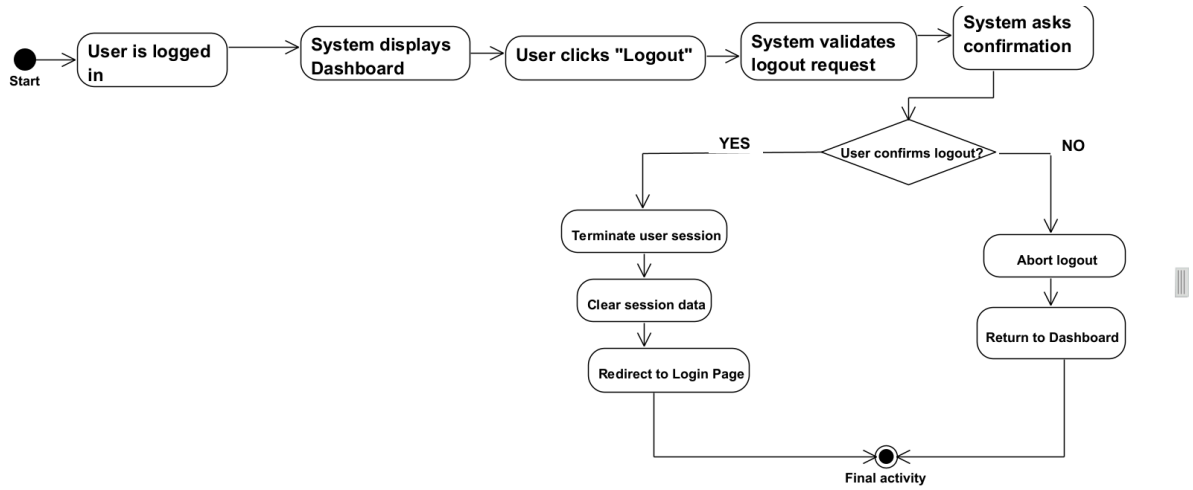


Figure 5: Diagram 3:Activity logout

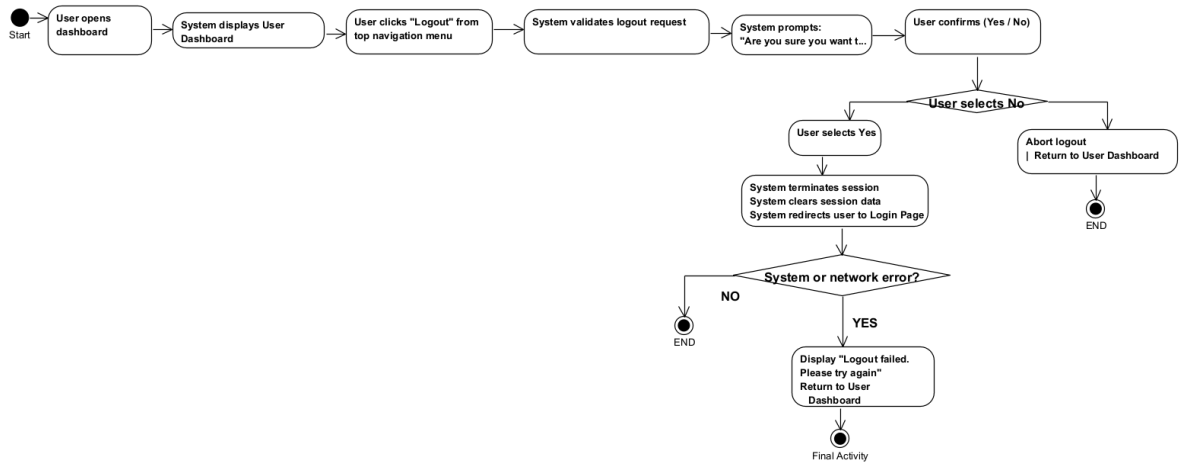


Figure 6: Diagram 4:Activity Submit Library Feedback

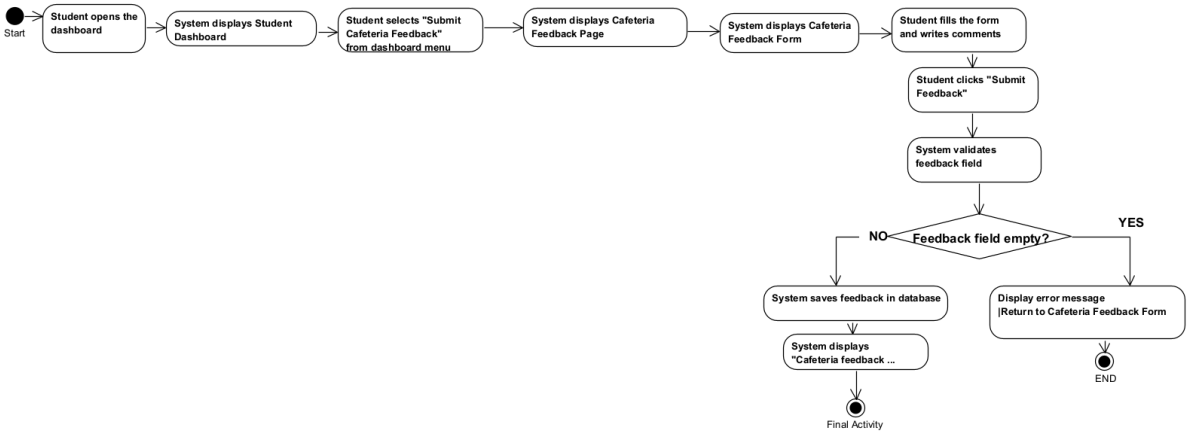


Figure 7: Diagram 5:Activity for Submit cafeteria

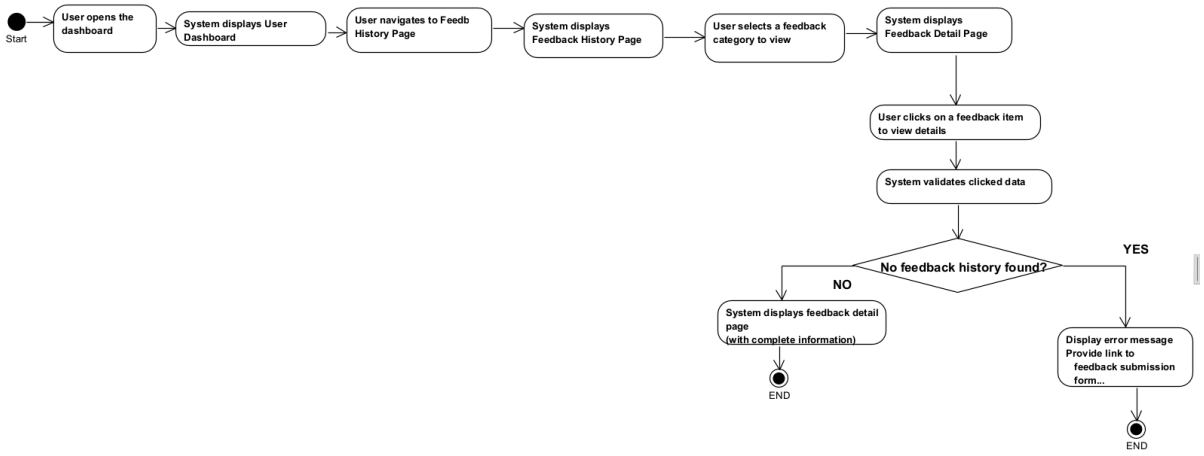


Figure 8: Diagram 6:View Feedback Submission History

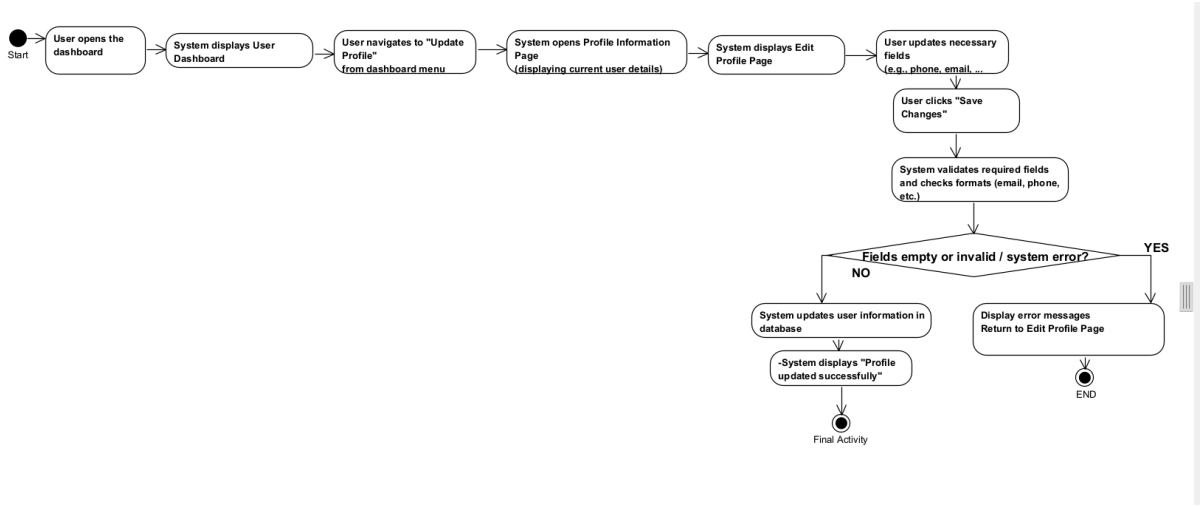


Figure 9: Diagram 7:Update Profile

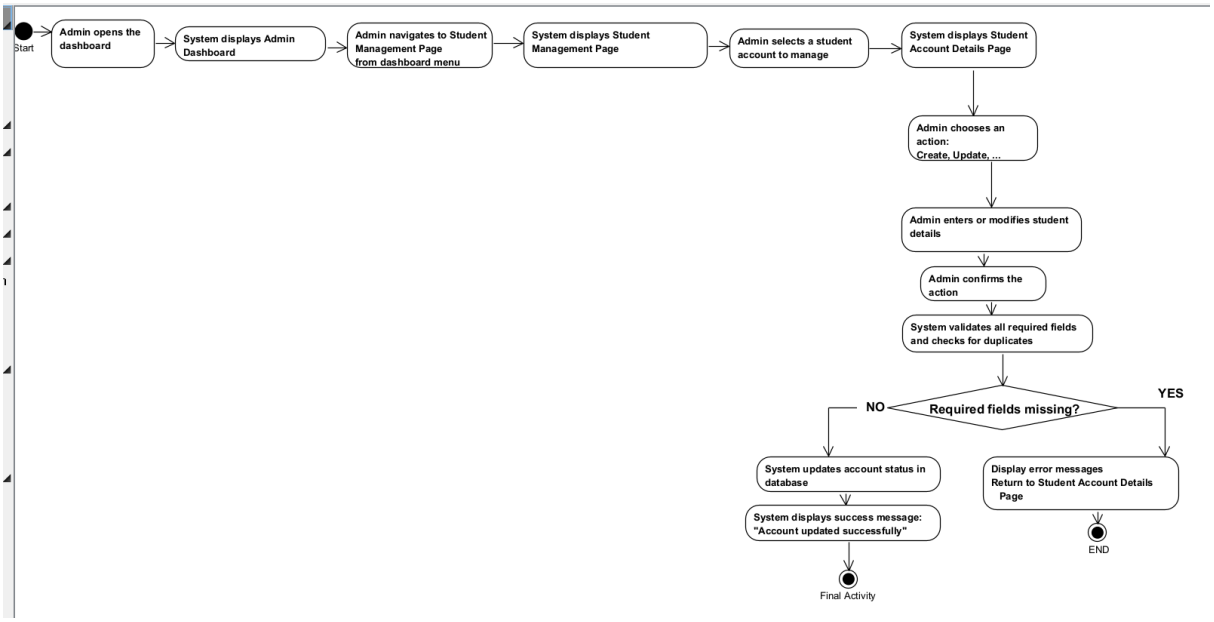


Figure 10: Diagram 8:Manage Student Accounts

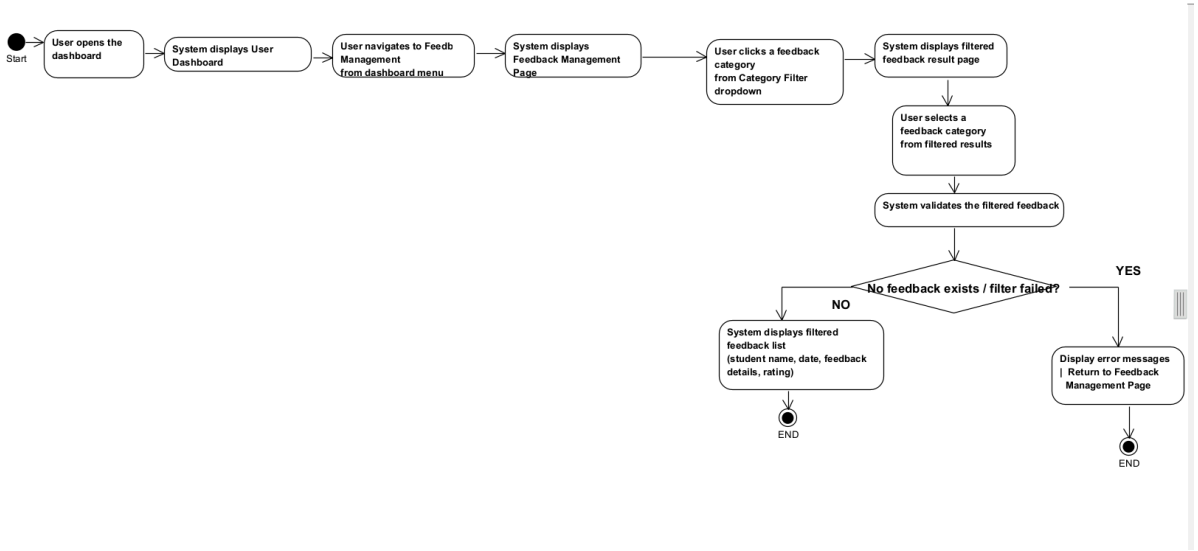


Figure 11: Diagram 9:Filter Feedback by Category

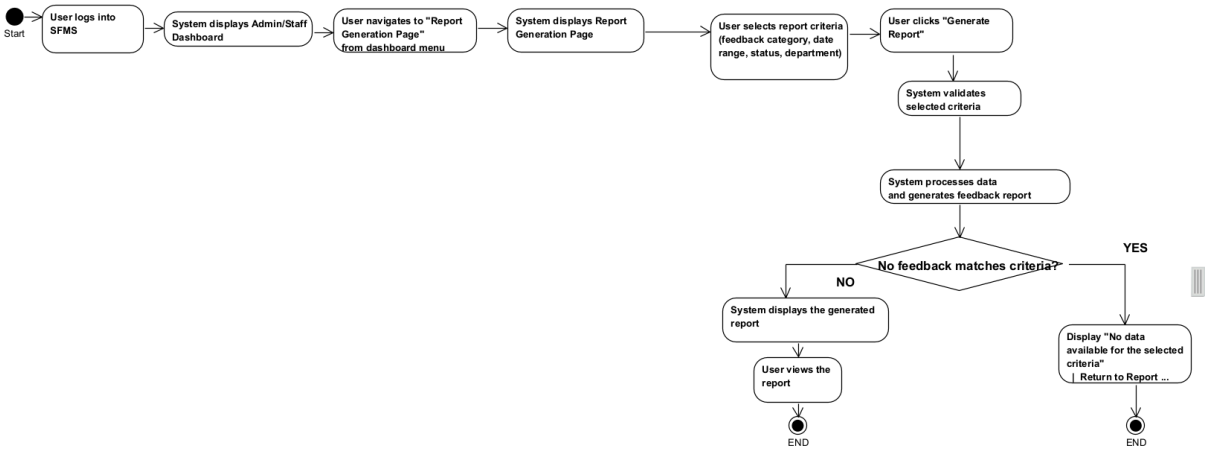


Figure 12: Diagram 10:Generate Feedback Reports

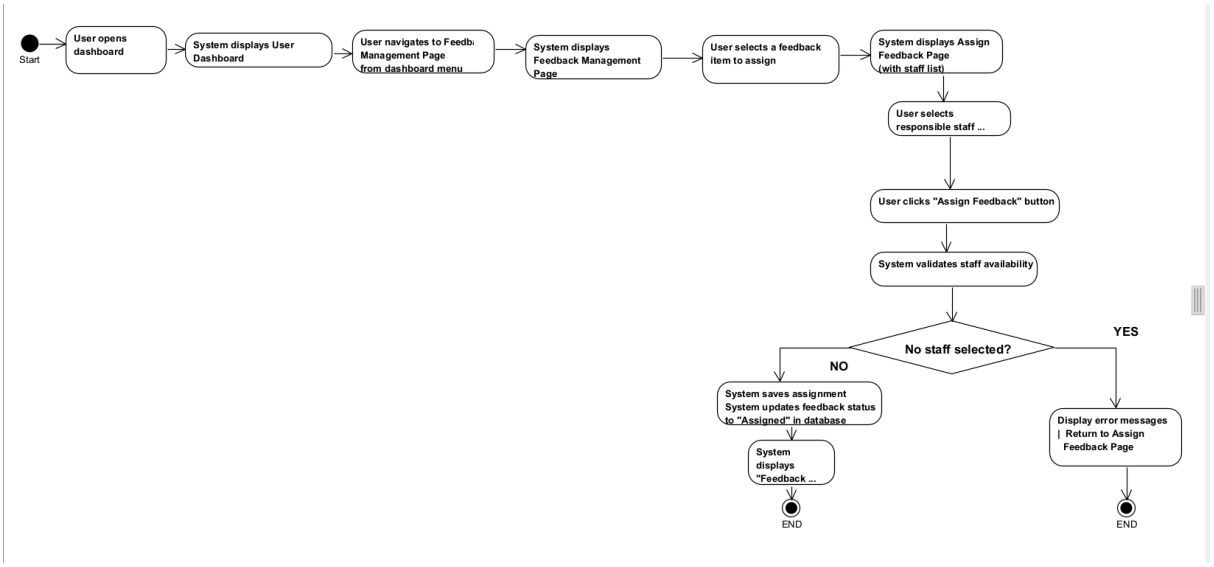


Figure 13: Diagram 11:Assign Feedback to Responsible Staff

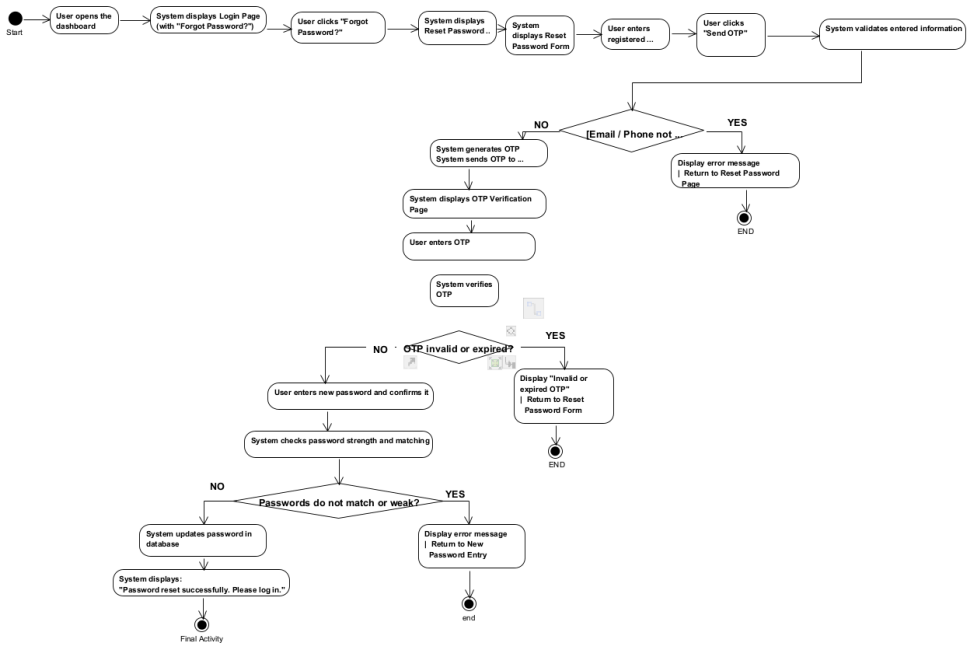


Figure 14: Diagram 12:Reset password

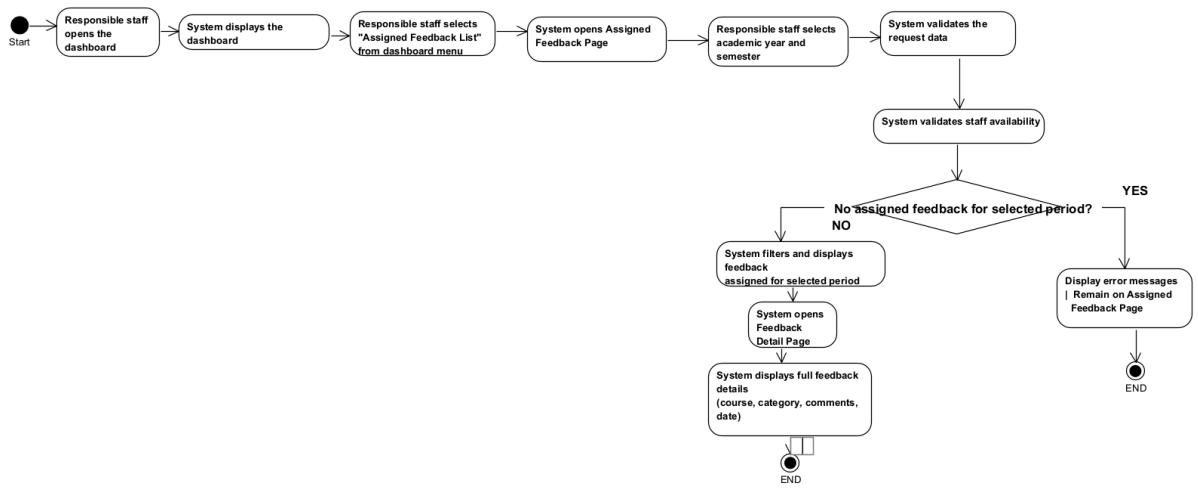


Figure 15:Diagram 13:View Assigned Feedback for Review

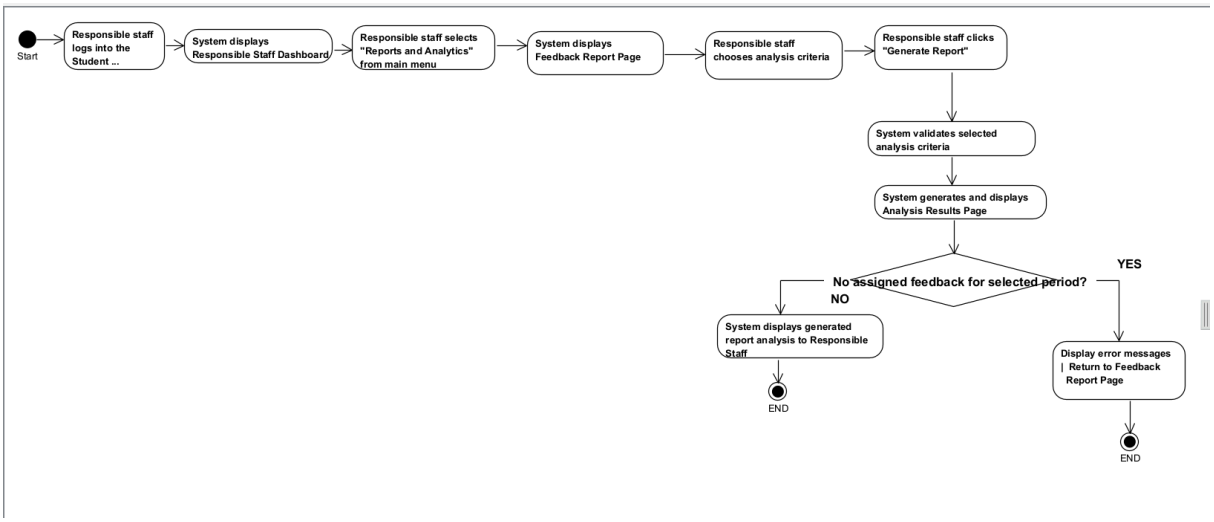


Figure 16:Diagram 14:Analyze Feedback Reports

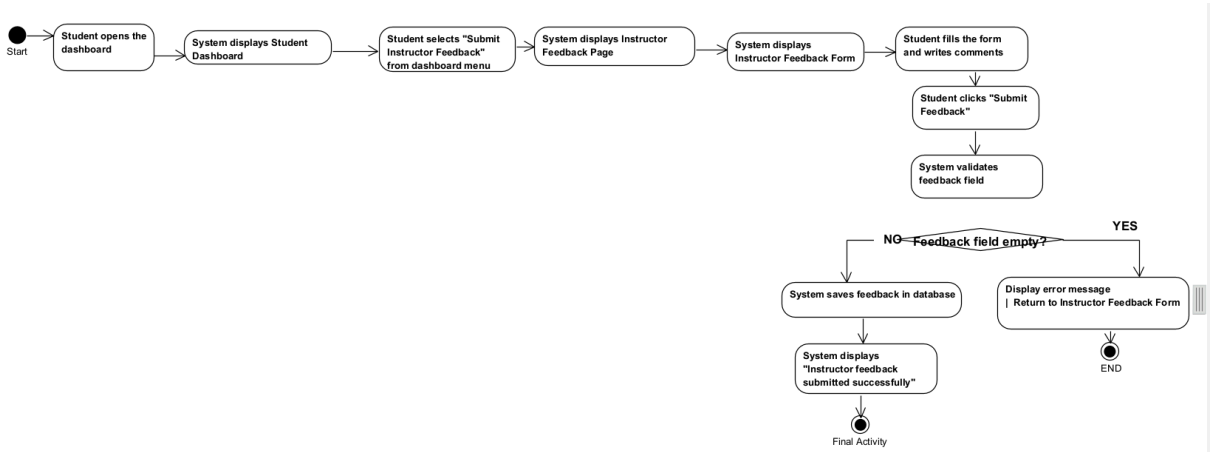


Figure 17: Diagram 15:Submit Instructor Feedback

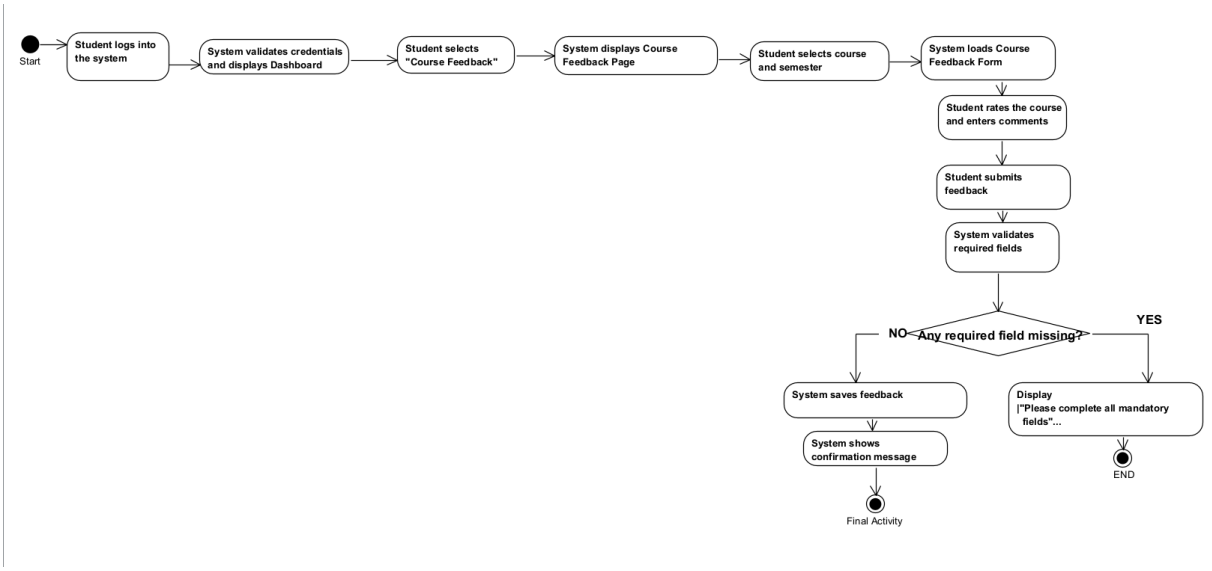


Figure 18:Diagram 16:Submit Course Feedback

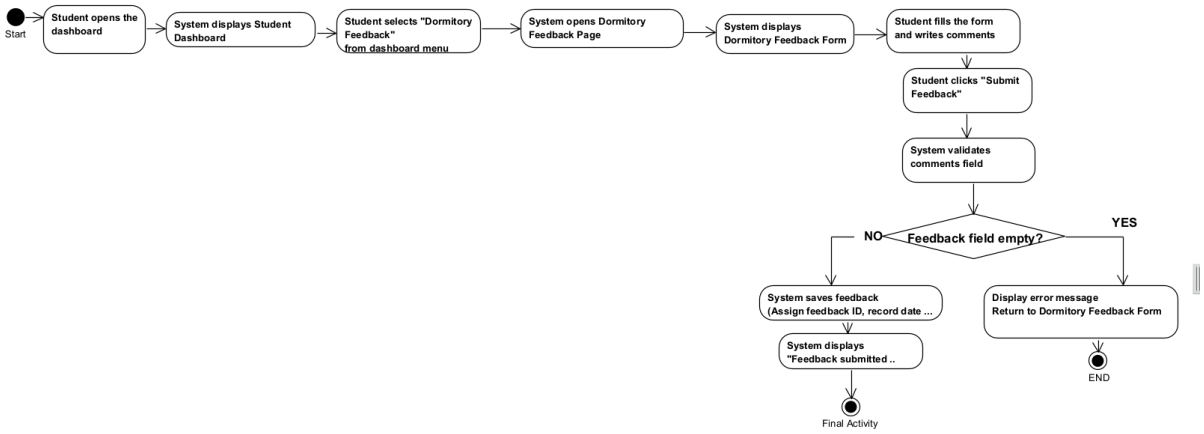


Figure 19:Diagram 17:Submit Dormitory Feedback

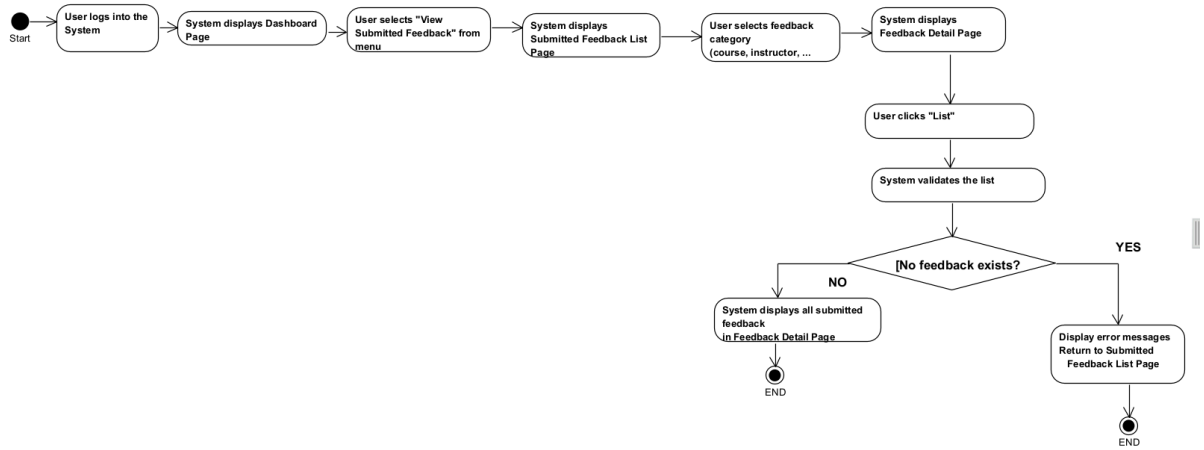


Figure 20:Diagram 18:View All Submitted Feedback

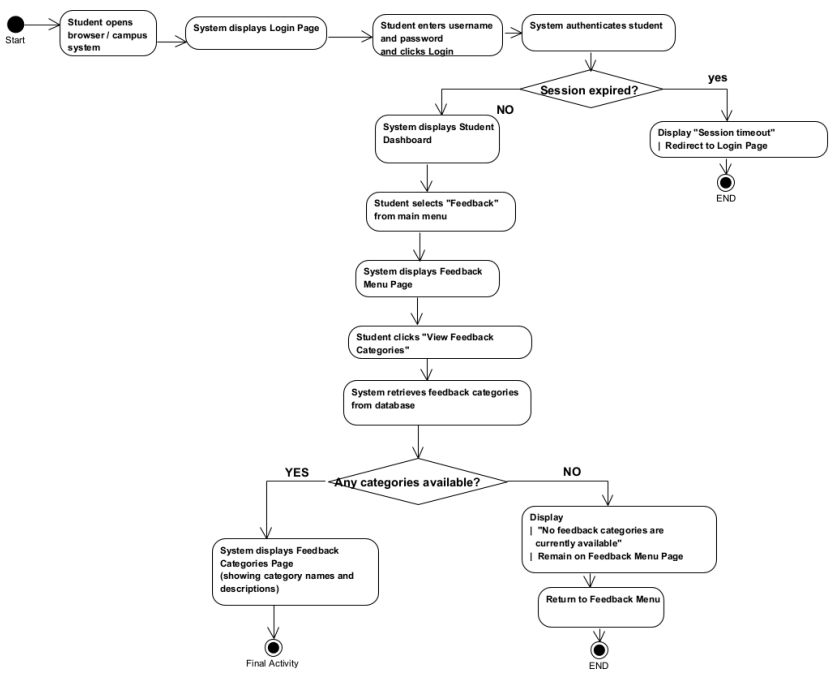


Figure 21:Diagram 19:View Available Feedback Categories

3.5.2. Sequence Diagram

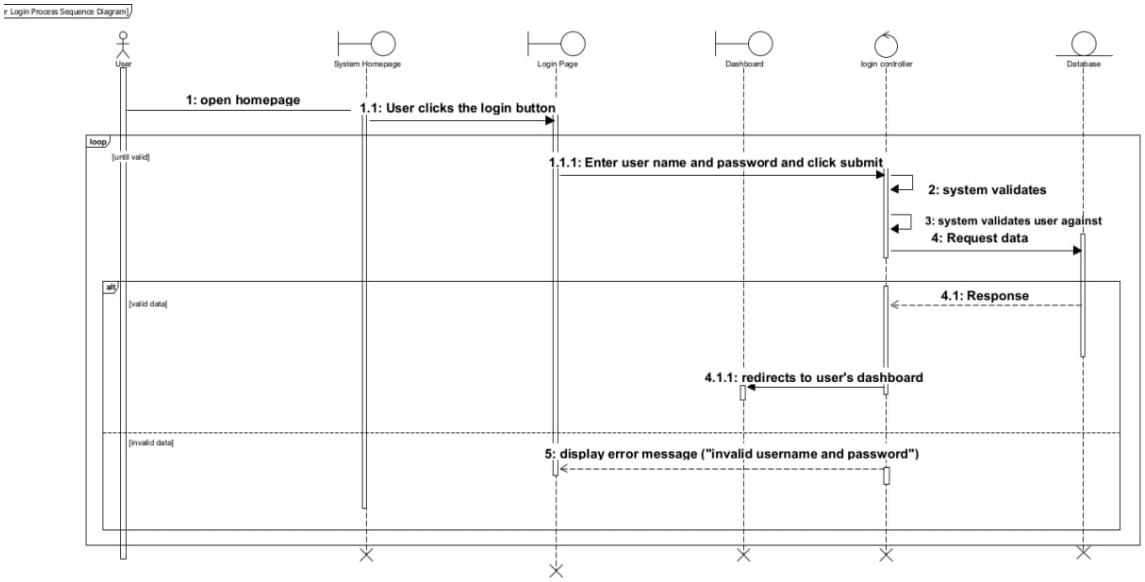


Figure 21: Sequence diagram for login

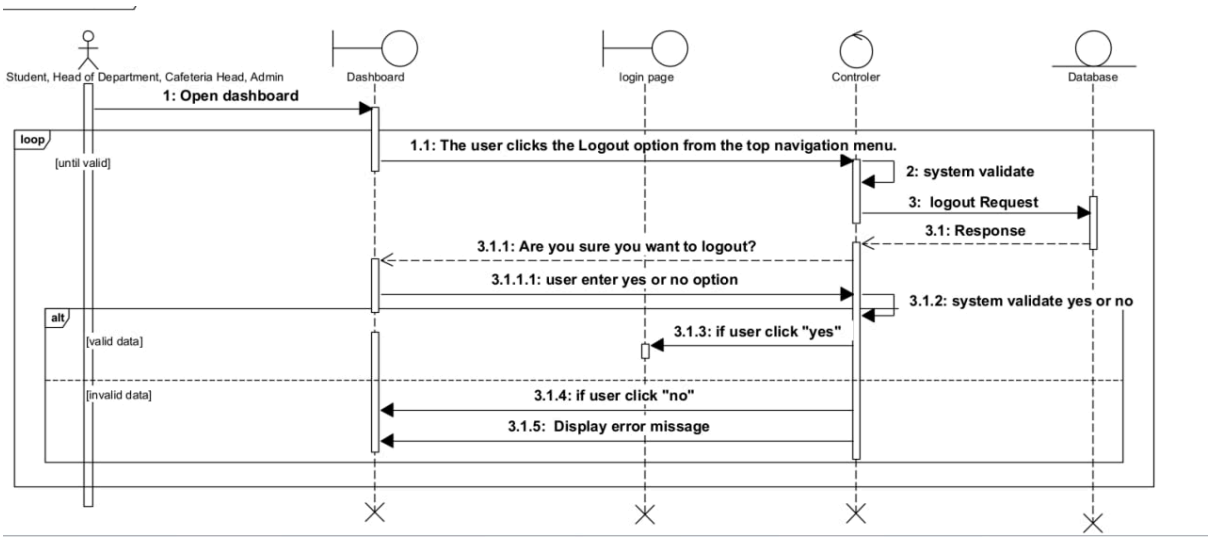


Figure 22:Sequence diagram for logout

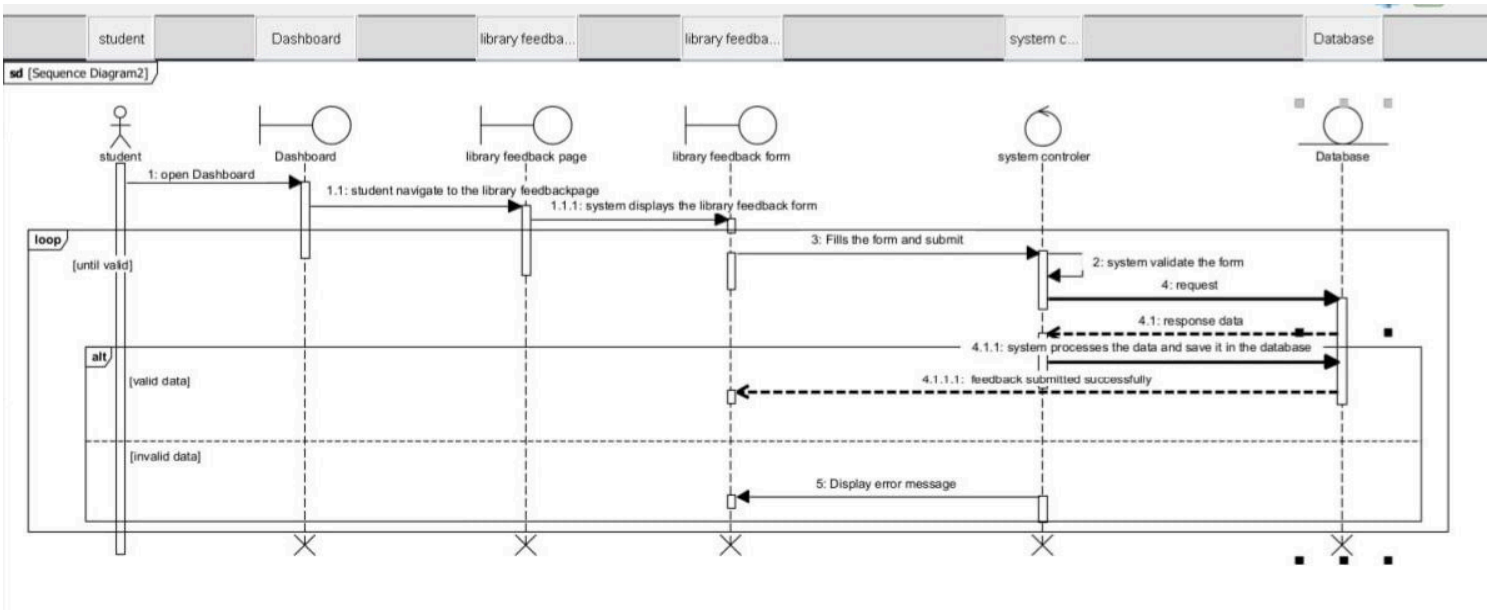


Figure 23:Sequence diagram for submit library feedback

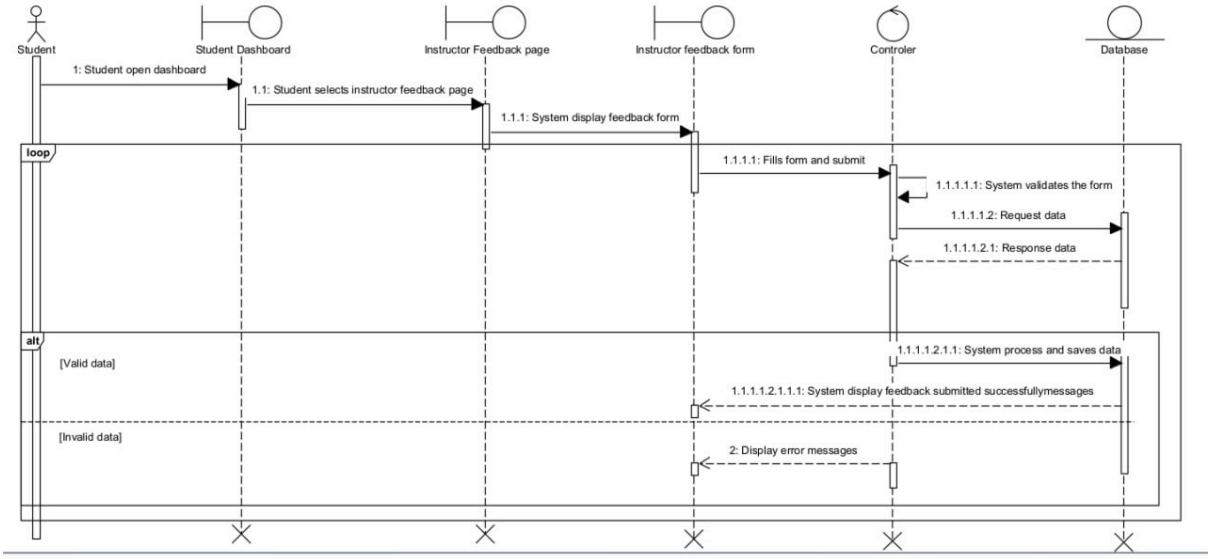


Figure 24:Sequence diagram for submit instructor feedback

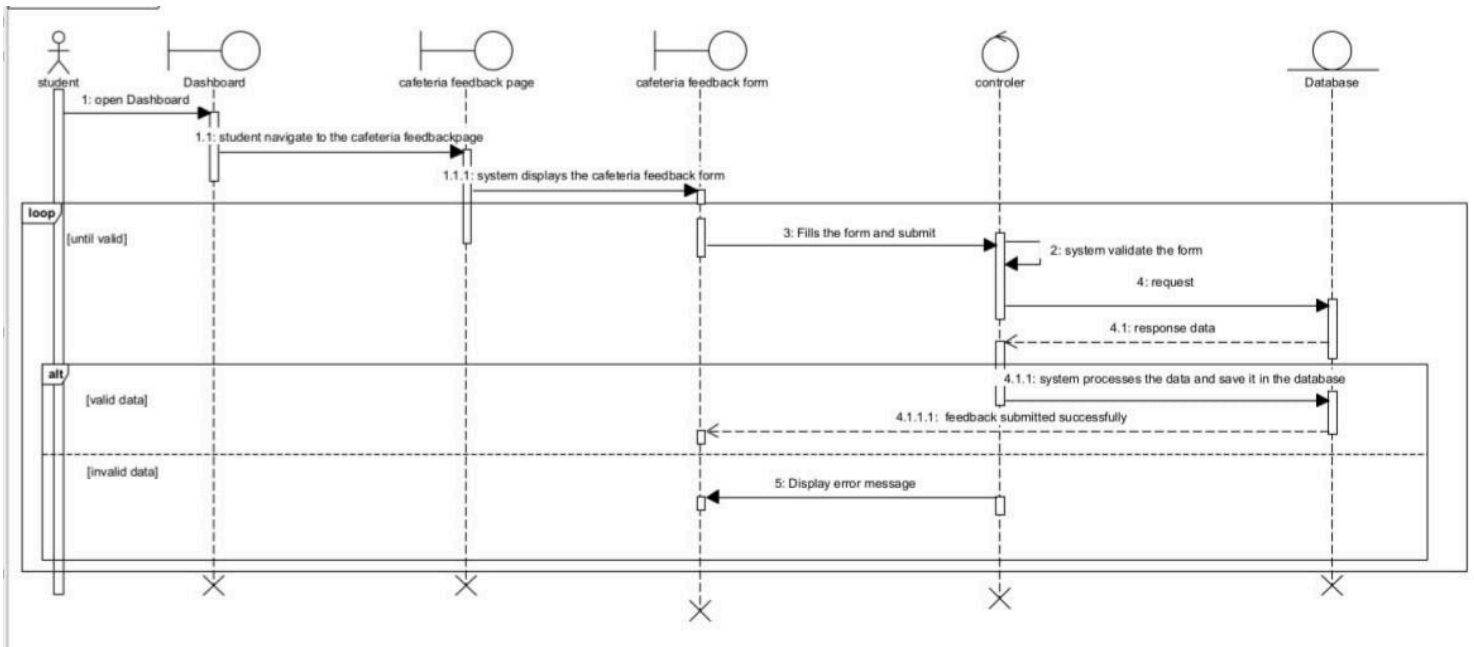


Figure 25:Sequence diagram for submit cafeteria feedback

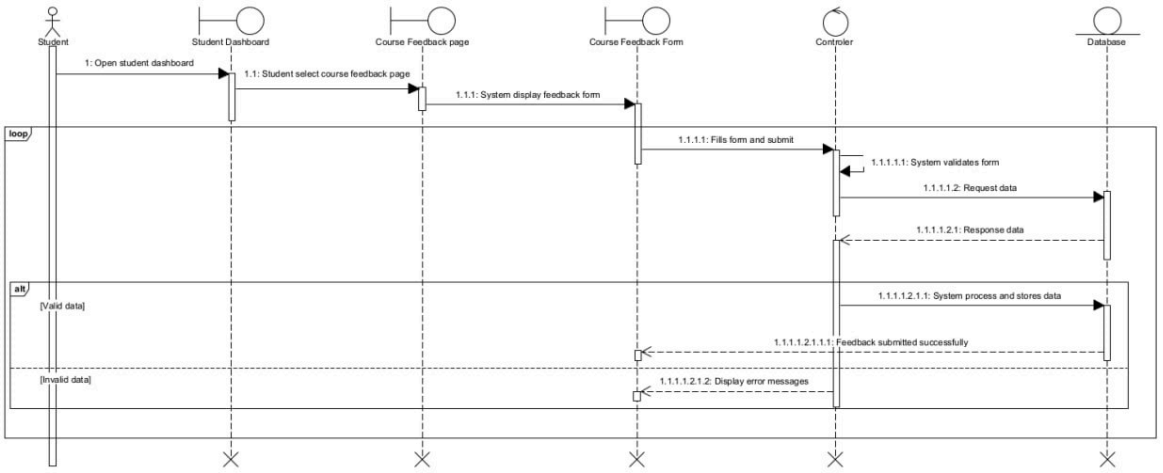


Figure 26:Sequence diagram for submit course feedback

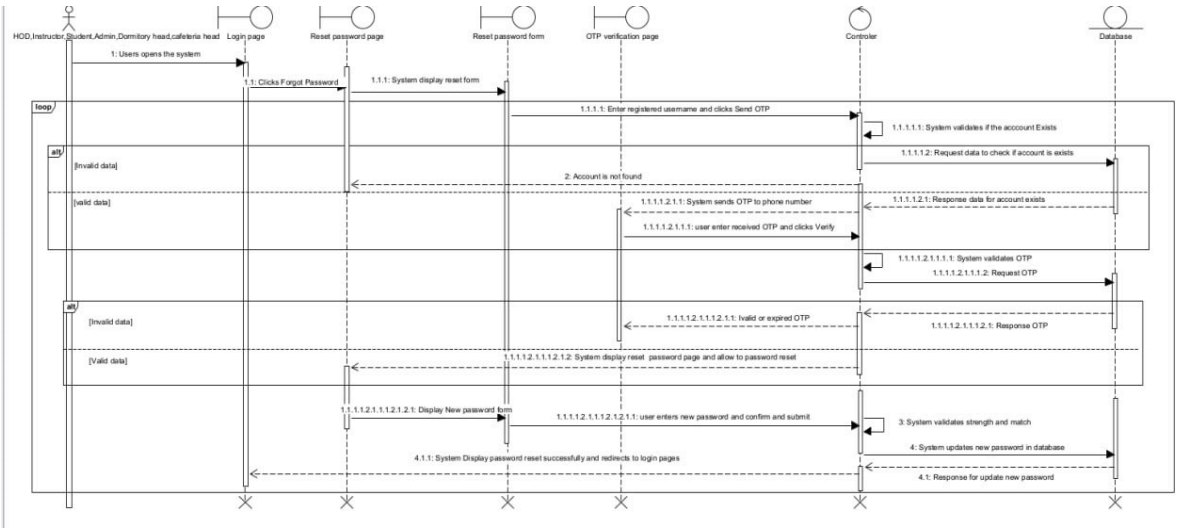


Figure 27:Sequence diagram for reset password

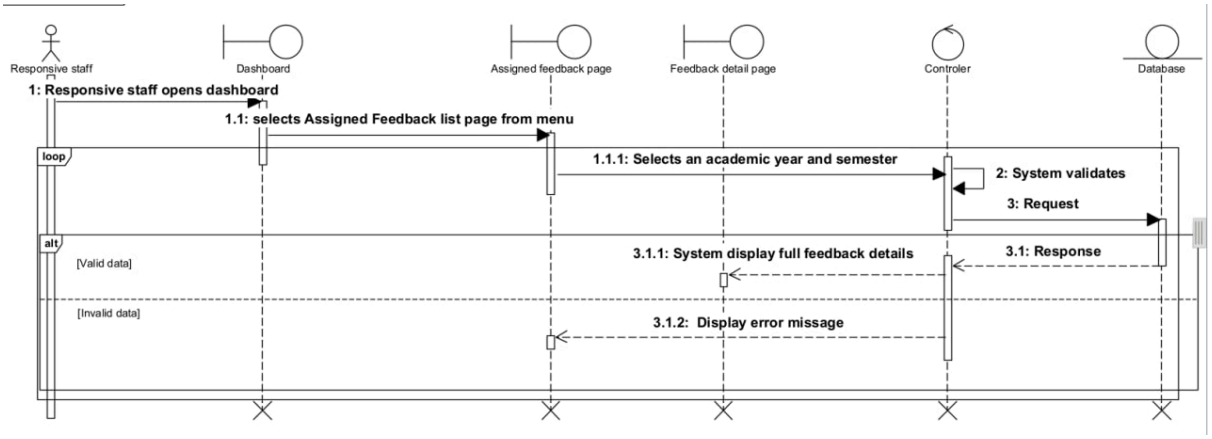


Figure 28:Sequence diagram for view assigned feedback

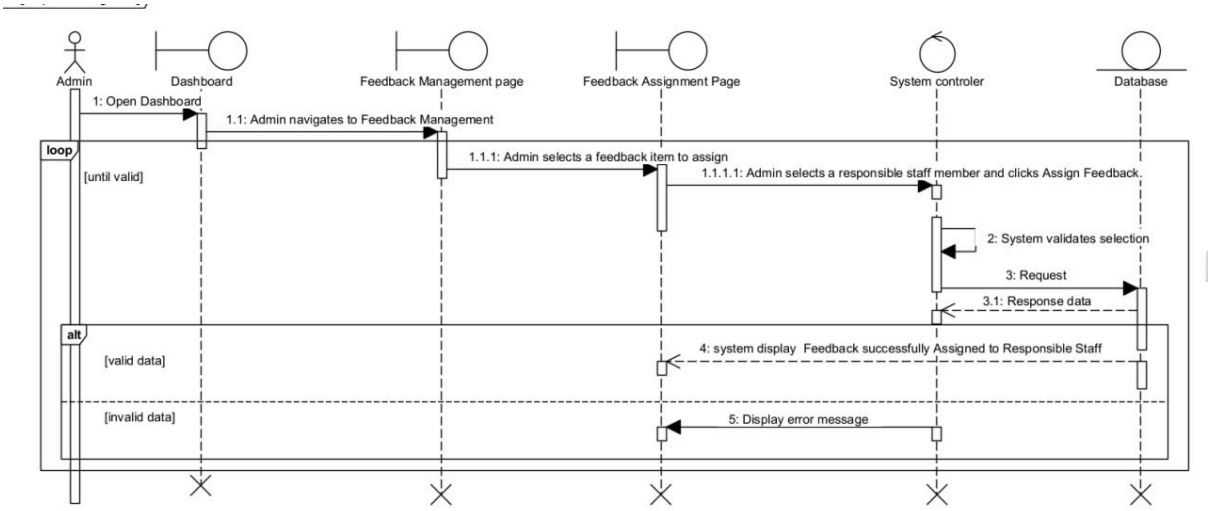


Figure 29:Sequence diagram for assign feedback for responsible staff

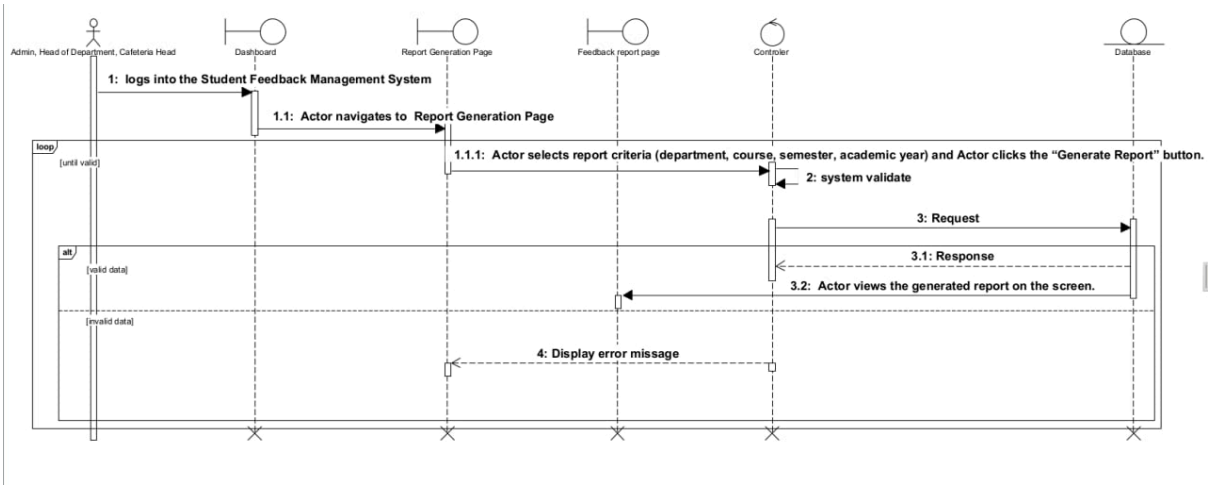


Figure 30:Sequence diagram for generate feedback report

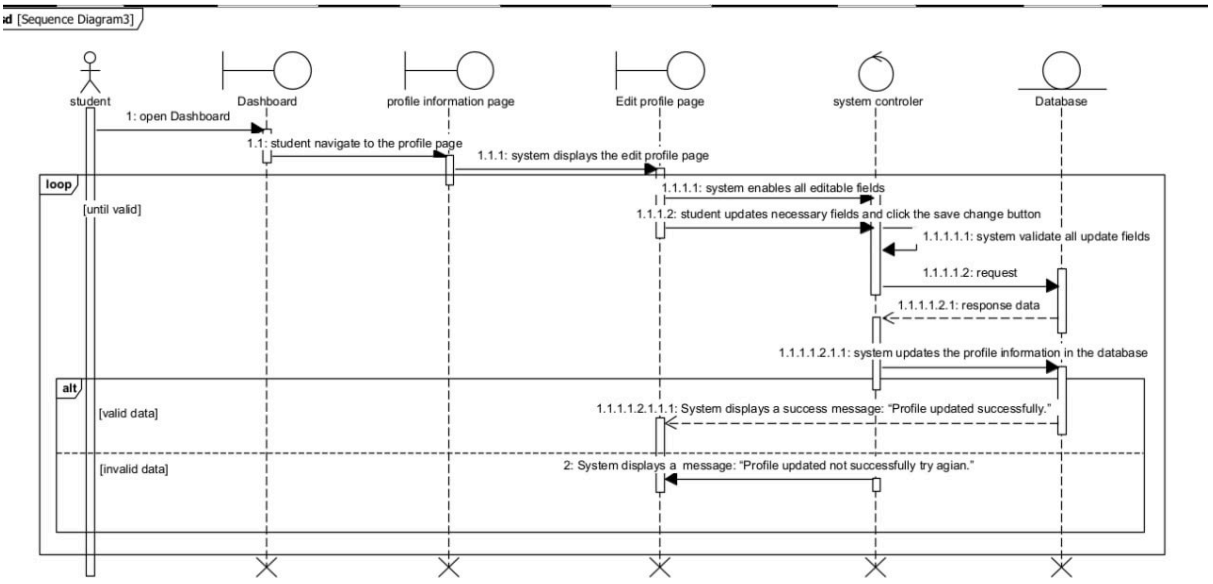


Figure 31:Sequence diagram for update profile

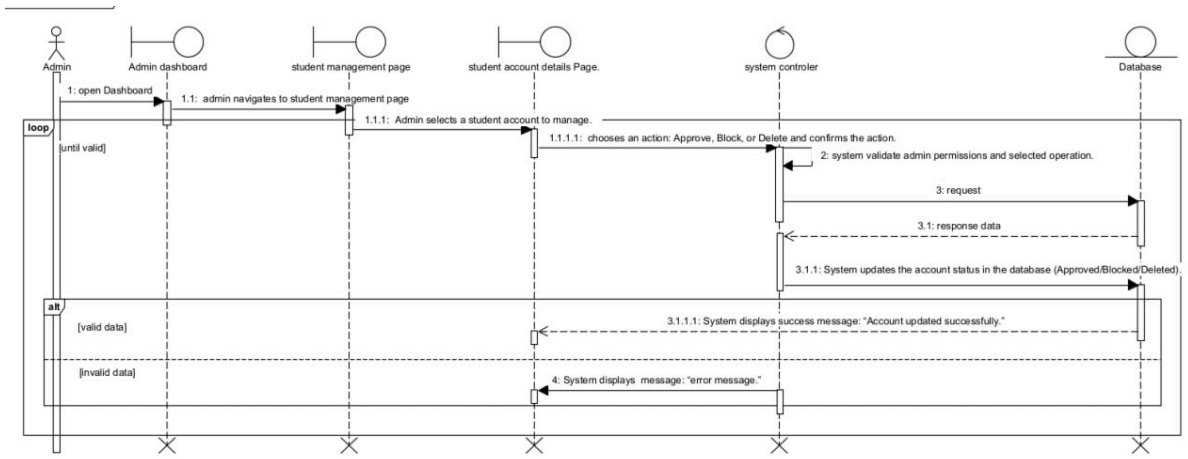


Figure 32:Sequence diagram for manage account

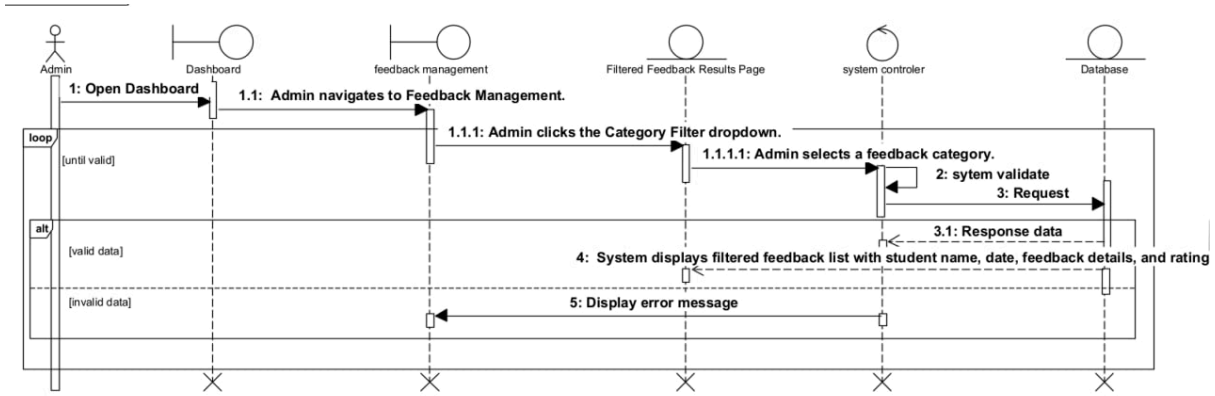


Figure 33:Sequence diagram for filter feedback by category

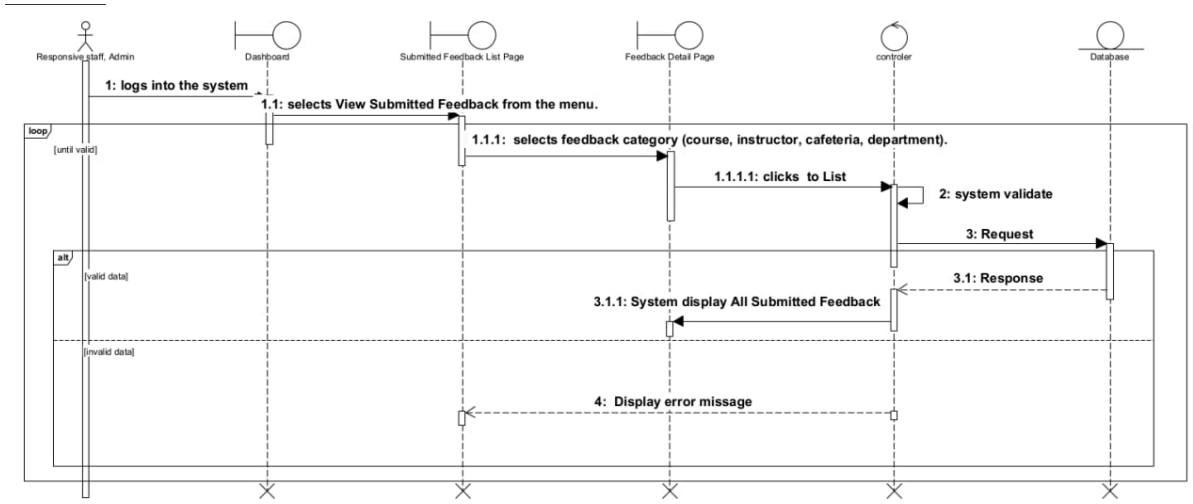


Figure 34:Sequence diagram for view all submitted feedback

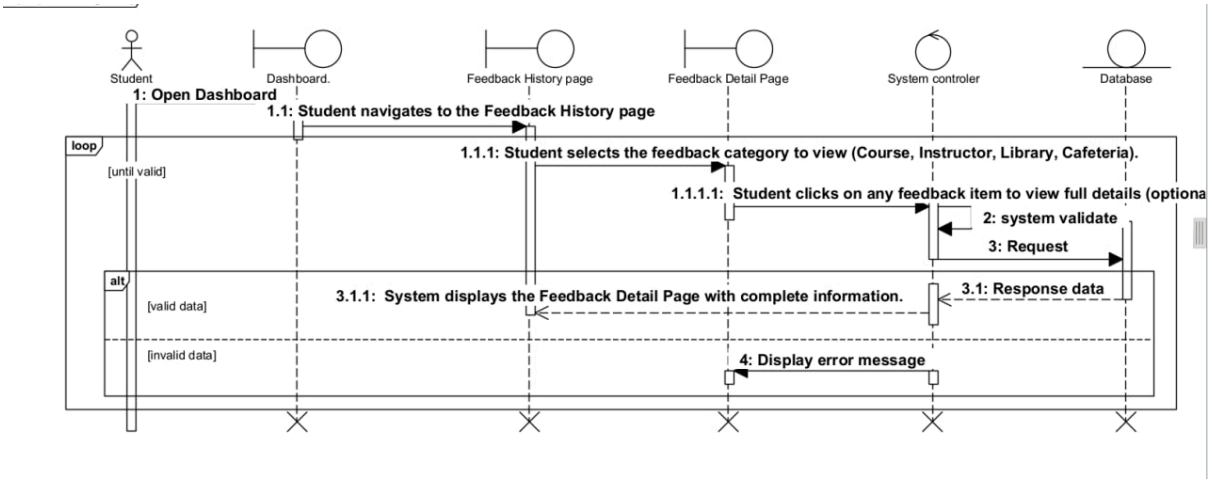


Figure 35:Sequence diagram for view feedback submission history

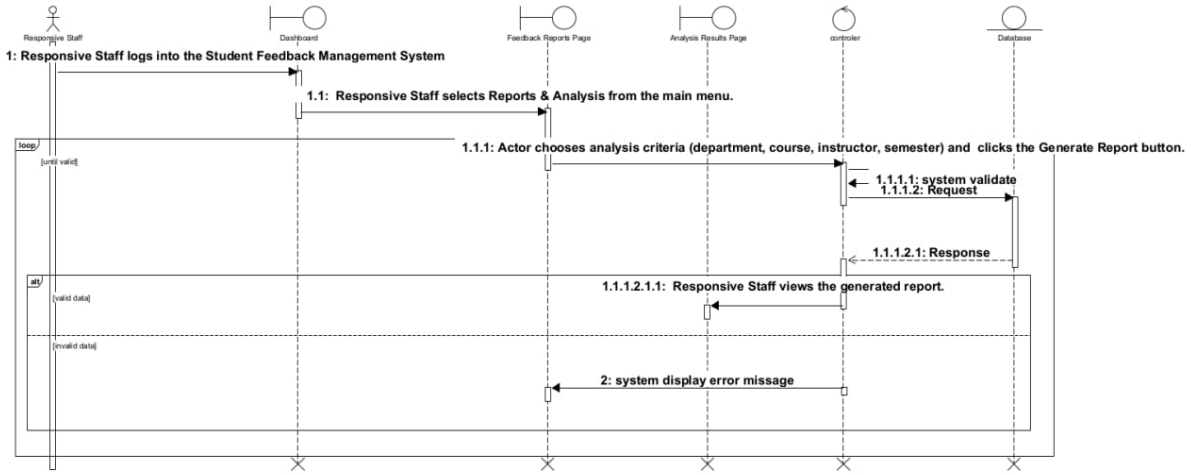


Figure 36:Sequence diagram for analysis feedback report

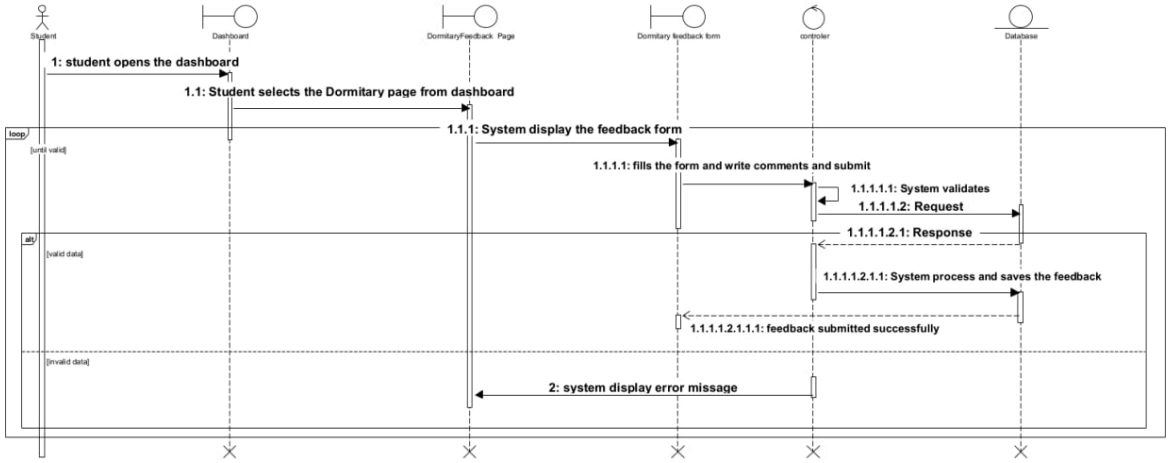


Figure 37:Sequence diagram for Submit Dormitory Feedback

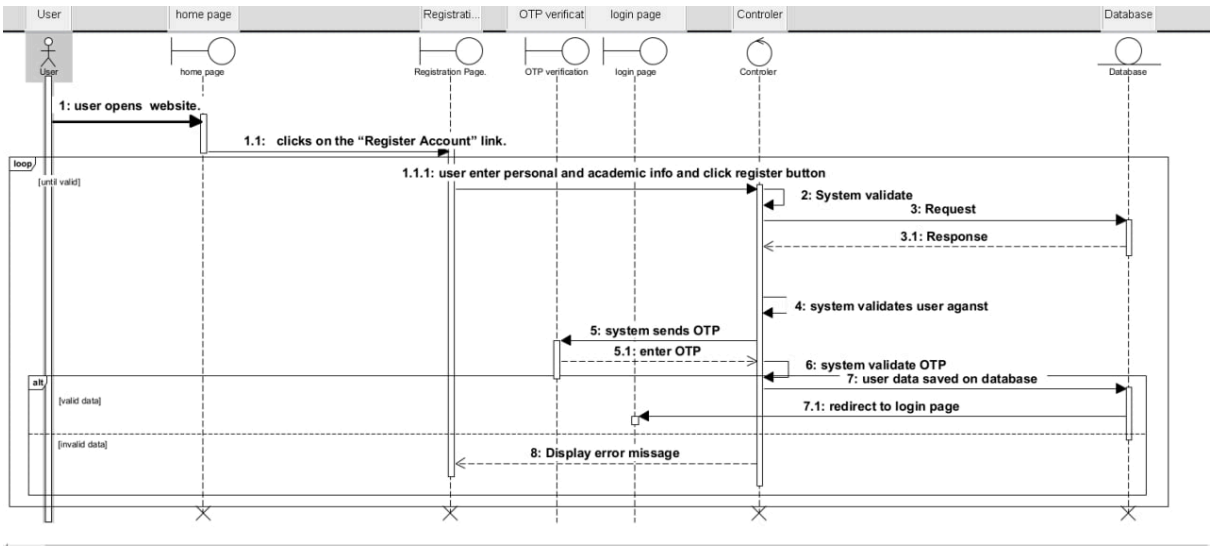


Figure 38:Sequence diagram for Register account

3.5.3 Requirement Validation & Verification

3.5.3.1 Validity Checks

Validity checks ensure that the documented requirements accurately reflect the real-world needs of stakeholders and align with the objectives of the Student Feedback Management System (SFMS). For SFMS, the following steps are applied to verify the validity of requirements:

1. Stakeholder Alignment:
   Conduct iterative reviews with students, instructors, department heads, and system administrators to confirm that the requirements capture their needs and expectations. Ensure that no conflicting requirements arise between different user roles, such as feedback anonymity for students versus reporting needs for instructors and administrators.

2. Alignment with Project Objectives:
   Cross-reference requirements with the general and specific objectives of SFMS to ensure alignment. Validate that the system supports key functionalities such as anonymous feedback submission, secure authentication, role-based dashboards, feedback summarization, and reporting for academic and service units (e.g., library, cafeteria).

3. Feasibility Testing:
   Assess whether the requirements can be implemented within the technical, financial, and operational constraints of the project. Avoid over-promising features such as mobile apps or automated instructor scoring, which may exceed the course project scope or available resources.

4. Traceability Checks:
   Verify that each requirement is traceable to a corresponding user need or use case scenario. For example, the “Submit Feedback” functionality must have clear user stories for students, while the “Generate Report” functionality must correspond to department head needs.

5. Review by Domain Experts:
   Involve faculty, IT staff, and software development experts to validate the practicality and relevance of the requirements. Address any gaps or mismatches identified by experts, such as data security concerns or usability issues for student dashboards.

6. Legal and Regulatory Compliance:
   Confirm that all requirements, especially those related to authentication, data storage, and anonymity, adhere to Ethiopian educational regulations and data protection standards. Ensure compliance with privacy and confidentiality rules for student information.

3.5.3.2 Consistency Checks

Consistency checks are essential to ensure that there are no conflicts, ambiguities, or contradictions within the requirements outlined in the SFMS SRS document. The following steps and measures are applied to verify consistency:

1. Terminology Consistency:
   Use uniform and clearly defined terms throughout the SRS document, such as “student,” “instructor,” “department head,” “feedback,” and “administrator.” Ensure that all system entities, roles, permissions, and authentication mechanisms are consistently referred to in every section of the document.
2. Inter-Requirement Consistency:
   Check for logical consistency among requirements. For example:
   - If a feature specifies that students can submit feedback anonymously, ensure that related reporting functionalities for instructors and department heads do not compromise anonymity.
   - Validate that feedback summary generation integrates seamlessly with both course and service feedback modules without contradictions.
3. Alignment Across Views:
   Ensure consistency between textual requirements and diagrams (e.g., ERD, UML diagrams). For instance, the relationship between students, courses, instructors, and feedback described in the SRS must align with the database model (ERD).
4. Functional and Non-Functional Consistency:
   Confirm that functional requirements do not conflict with non-functional requirements. Example: If the system is required to handle multiple simultaneous feedback submissions (non-functional), ensure that no functional requirement imposes unrealistic restrictions that would prevent this.
5. Conflict Resolution:
   Identify and resolve any conflicts between stakeholder priorities. For example: If students need guaranteed anonymity while submitting feedback, ensure that this does not conflict with the department head’s need for detailed analytical reports.
6. Compliance Consistency:
   Cross-check compliance requirements for data privacy, security, and authentication to ensure uniform adherence across all system features. For instance, both student login and feedback storage must consistently comply with Ethiopian data protection guidelines and university policies.
7. Version Control and Traceability:
   Maintain a well-defined version control system to avoid conflicts between updated and previous requirements. Ensure that changes in one requirement (e.g., feedback form structure) are consistently reflected across all dependent requirements, diagrams, and UI prototypes.

3.5.3.3 Completeness Checks

Completeness checks ensure that all necessary requirements for the Student Feedback Management System (SFMS) are included and sufficiently described. The following measures are applied:

1. Functional Coverage:
   Verify that all core functionalities are thoroughly documented. Examples include:
   - Feedback submission for courses, cafeteria, library, and other university services
   - Feedback summary generation for instructors
   - Report generation for service units and department heads
   - Dashboard and analytics features

2. Boundary Cases:
   Ensure requirements cover edge cases, such as:
   - Handling incomplete feedback forms or missing ratings
   - Preventing duplicate submissions
   - Invalid login attempts or unauthorized access

3. Non-Functional Requirements:
   Confirm that all performance, scalability, and reliability requirements are specified. Example: the system must support multiple simultaneous feedback submissions without affecting performance or data integrity.

4. External Interactions:
   Verify integration points, such as:
   - Email notifications for feedback updates
   - Exporting reports to PDF or Excel formats
   - Any potential future integration with university portals

5. Traceability:
   Ensure each requirement can be traced to its corresponding design element, use case, or project objective. For example, the “Submit Feedback” requirement should correspond to the use case diagram, ERD, and UI prototype.

6. User Roles and Permissions:
   Confirm that all roles and their permissions are covered. All users managing or interacting with the system are referred to collectively as Responsible Staff, ensuring proper access control and handling of feedback.

7. Gaps Identification:
   Identify and document any undefined or missing requirements, such as potential mobile app support or integration with additional university services, for consideration in future system iterations.

3.5.3.4 Realism Checks

Realism checks evaluate whether the documented requirements for the Student Feedback Management System (SFMS) are achievable within the constraints of time, budget, technology, and resources. The following considerations are applied:

1. Technical Feasibility:
   Verify that the system's requirements (e.g., feedback submission, report generation, dashboards) are implementable using the available technologies: HTML, CSS, JavaScript for the front-end, PHP for the back-end, and MySQL for data storage.

2. Resource Availability:
   Ensure that the development team possesses the required expertise in web development, database design, and UI/UX design to implement the proposed system.

3. Timeline Assessment:
   Validate that all requirements can be realistically implemented within the project timeline without compromising quality or completeness.

4. Cost Analysis:
   Evaluate whether the requirements (e.g., implementing dashboards, exporting reports, and maintaining a secure database) are achievable within the allocated budget for the project.

5. Hardware and Infrastructure:
   Ensure the system requirements (e.g., real-time feedback submission and report generation) are compatible with the university’s available hardware, server infrastructure, and network capabilities.

6. Scalability Expectations:
   Assess whether the system can accommodate future expansions, such as increased student participation, additional feedback categories (e.g., new services), or integration with other university systems, without significant rework.

7. Regulatory Compliance:
   Confirm that the requirements adhere to relevant legal and regulatory standards, particularly for data privacy, anonymity, and secure handling of student feedback

3.5.3.5 Verifiability

Verifiability checks ensure that every requirement documented in the SFMS SRS can be tested or validated through measurable criteria. This step guarantees that the system will meet the specified requirements upon completion.

1. Clarity of Requirements:
   Confirm that each requirement is stated in clear, unambiguous terms, ensuring it can be easily interpreted for testing purposes.

    - Example: “The system shall allow Responsible Staff to generate feedback reports for their units” is measurable and testable.

2. Test Case Mapping:
   Verify that all requirements are linked to corresponding test cases, ensuring complete coverage during the testing phase.

    - Example: The requirement to submit and store student feedback should have a test case to verify successful submission, storage in the database, and accurate reporting.

3. Acceptance Criteria:
   Validate that each requirement includes specific acceptance criteria that define success.

    - Example:

        - Requirement: Generate feedback summary reports.
        - Acceptance Criteria: Reports must be viewable on the dashboard within 5 seconds of request and reflect all submitted feedback accurately.

4. Objective Measurability:
   Ensure that requirements can be evaluated using objective metrics such as performance, accuracy, and reliability.

    - Example: A requirement for feedback submission must include validation steps to ensure feedback is stored correctly in the database and cannot be duplicated.

5. Traceability:
   Confirm that every requirement is traceable from its initial documentation to its corresponding implementation and validation in the testing phase.

6. Documentation of Constraints:
   Verify that any system limitations or constraints are explicitly stated and testable.

    - Example: The system may have a maximum number of simultaneous feedback submissions it can handle, which should be defined and tested

Chapter Four

4.1.Design issues for your system(like reuse, future change, refactoring concept of components

The Student Feedback Management System is designed with several key software design considerations to ensure long-term effectiveness, maintainability, and scalability.

- **Modularity & Component Reuse**
  The system is divided into independent modules such as:
  - Authentication
  - Feedback Submission
  - User Management
  - Report Generation

  Shared features (e.g., login, profile management) are reused across Students, Administrators, and Responsible Staff.
  Improves development efficiency and reduces duplication.
- **Support for Future Change (Extensibility)**
  The architecture allows adding:
  - New feedback categories (e.g., transportation, classroom facilities)
  - New user roles or privileges
  - Additional reporting features

  Changes can be applied without modifying existing stable components, reducing risk.
- **Maintainability & Refactoring**
  Code follows clean coding standards and Object-Oriented principles such as:
  - Encapsulation to hide internal details of modules
  - Abstraction to simplify complex operations

  Refactoring can be performed easily to improve performance or structure without affecting functionality.
- **Security Considerations**
  User credentials are securely stored using password hashing.
  Session management ensures protected authentication flow.
  Role-based access control prevents unauthorized access to feedback data and administrative functions.
  Input validation prevents attacks such as SQL Injection.
- **Performance & Scalability**
  Efficient database indexing improves query speed for large numbers of students and feedback records.
  Three-tier architecture enables scaling:
  - More users then scale the web server
  - More data then scale the database server

  Optimized data retrieval ensures fast display of reports and feedback lists.

- **Usability & Accessibility**
  User interface designed to be:
  - Simple and intuitive for students
  - Mobile and desktop responsive

  Helps ensure higher feedback participation.
- **Reliability & Data Integrity**
  Proper validation ensures complete and correct feedback submissions.
  Database constraints maintain consistency between:
  - Students, courses, instructors, and services records

  Backup and recovery strategies prevent data loss.

4.2.Design patterns

Category	Design Pattern	Description /use	Reason for Choosing or Not
Creational	Factory Method	Creates objects without specifying the exact class, useful for dynamic object creation.	Chosen: Our system may create different types of feedback (course, instructor, service) dynamically, so Factory is flexible and maintainable.
	Singleton	Ensures only one instance of a class exists, commonly used for database connections.	Not chosen: While useful for DB connections, our PHP setup naturally handles single connections; less critical here.
	Builder	Helps construct complex objects step by step.	Not chosen: Feedback forms and reports are relatively simple; stepwise construction is unnecessary.
	Prototype	Creates new objects by cloning existing ones, useful for repeated templates.	Not chosen: Feedback and user objects are not frequently cloned; unnecessary overhead
	Abstract Factory	Provides an interface to create families of related objects without specifying their concrete classes.	Not chosen: Our system has few object types; full abstract factories are overly complex.
Structural	Decorator	Allows adding new behavior to objects dynamically without modifying their structure.	Chosen: Can add features like optional notifications or custom report formatting without changing core logic.
	Facade	Provides a simplified interface to complex subsystems	Not chosen: Our system is simple enough; frontend-backend interactions are straightforward.
	Adapter	Allows incompatible interfaces to work together.	Not chosen: All system components are compatible; no interface mismatch exists.
	Composite	Treats individual objects and groups uniformly, useful for hierarchical structures.	Not chosen: Feedback system does not have hierarchical object structures.

Behavioral	Observer	Defines a one-to-many dependency: when one object changes, all dependents are notified.	Chosen: Automatically notifies responsible staff or admin when new feedback is submitted
	Strategy	Defines a family of algorithms and makes them interchangeable.	Not chosen: Feedback handling is straightforward; multiple algorithm choices are not required.
	command	Encapsulates requests as objects, useful for undo/redo operations.	Not chosen: Our system does not require undo/redo for feedback submissions.
	Mediator	Centralizes complex communications between objects, reducing dependencies	Not chosen: Components are loosely coupled already; mediator adds unnecessary complexity

Table27: Design Pattern Comparison

4.4 Class Diagram

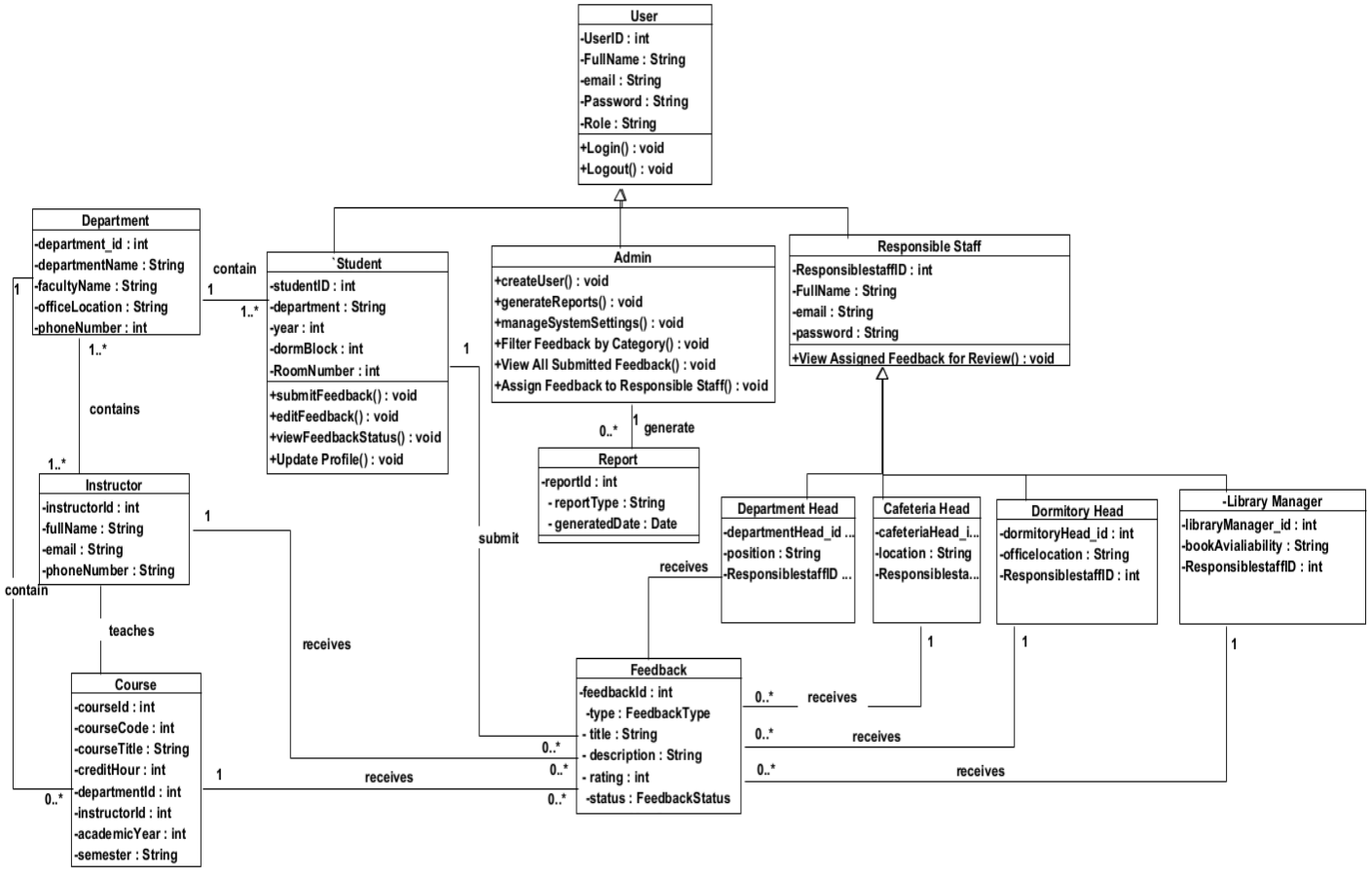


Figure 39: Class diagram for SFMS

4.4.Database Model

4.4.5 Entity Relationship Diagram (ERD)

A. Summary Paragraph of Problem Description

The Student Feedback Management System (SFMS) is designed to manage and analyze student feedback for courses, instructors, and university service units in a centralized system. It supports anonymous feedback submission and structured reporting for Responsible Staff. The system stores user, course, service unit, and feedback data while ensuring data integrity and privacy. Relationships among these entities are modeled in the ERD to accurately represent real-world interactions and support effective decision-making.

B. Entity Analysis

Entity analysis identifies the main entities required to store and manage data in the Student Feedback Management System (SFMS). These entities are derived from the system requirements, use cases, and problem description. Each entity represents a real-world concept with multiple instances, unique attributes, and a significant role in the system.

Table28: below presents the key entities identified for SFMS along with brief descriptions:

Entity Name	Description
User	Represents individuals interacting with the system, such as students, admin, staff and Responsible Staff.
Role	Defines access levels and responsibilities within the system.
Department	Represents academic departments responsible for courses and services
Course	Represents academic courses for which feedback is collected.
ServiceUnit	Represents non-academic services such as cafeteria, library, and other units.
Feedback	Stores feedback details including ratings, comments, and submission date.
FeedbackCategory	Classifies feedback types (e.g., course feedback, service feedback).
Report	Represents summarized feedback data generated for analysis and decision-making.

C Analysis of Cardinality and Participation Constraints

To define the cardinality and participation constraints in the Student Feedback Management System (SFMS) ERD, we analyze the relationships from the perspective of each entity, focusing on their interactions and dependencies within the university's feedback processes. Cardinality specifies the maximum number of instances one entity can be associated with another, while participation indicates whether an entity's involvement in a relationship is mandatory (total) or optional (partial).

This analysis examines each relationship and assesses the dependencies between entities based on the system's use cases, such as student feedback submission, course enrollment, dormitory assignments, and service evaluations

.The table below summarizes the cardinality and for all identified relationships among entities in the SFMS

Table29: cardinality and relationship

Relationship.	Cardinality
Student ↔ Department	Many-to-One
Instructor ↔ Department	Many-to-One
Course ↔ Department	Many-to-One
Instructor ↔ Course	One-to-Many
Student ↔ Course	Many-to-Many
Student ↔ Feedback	One-to-Many
Course ↔ Feedback	One-to-Many
Instructor ↔ Feedback	One-to-Many
Responsible Staff ↔ Feedback	One-to-Many

D. Attributes of each Entity Types

Identifying attributes in the Student Feedback Management System (SFMS) focuses on selecting only the essential data required to manage users, feedback, and reports at Wachemo University. Attributes represent core properties of entities, not actions or relationships. Redundant and unnecessary details are avoided to ensure a clean, accurate, and maintainable database design.

4.4.1 Entity Relationship Diagram (ERD)

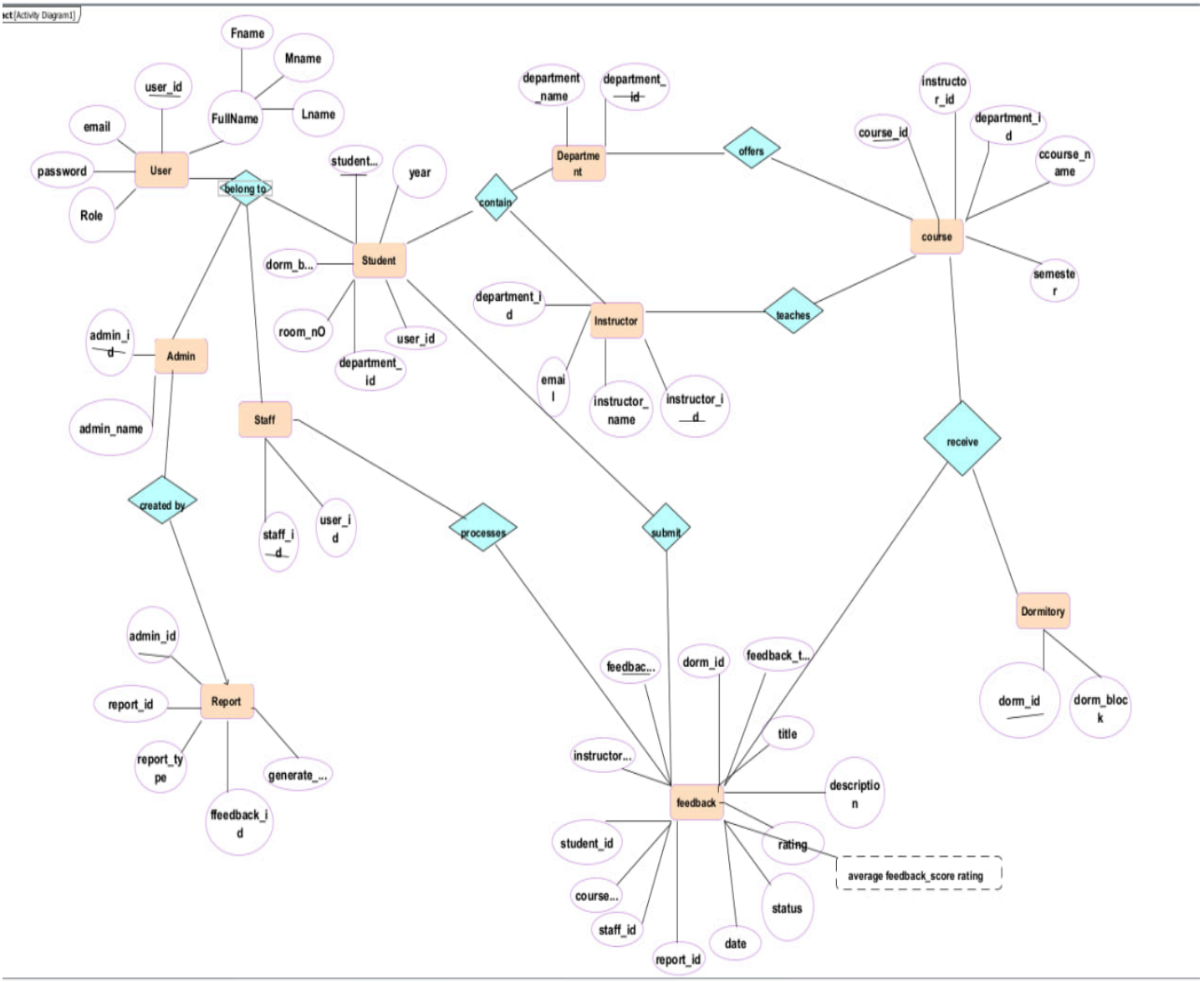
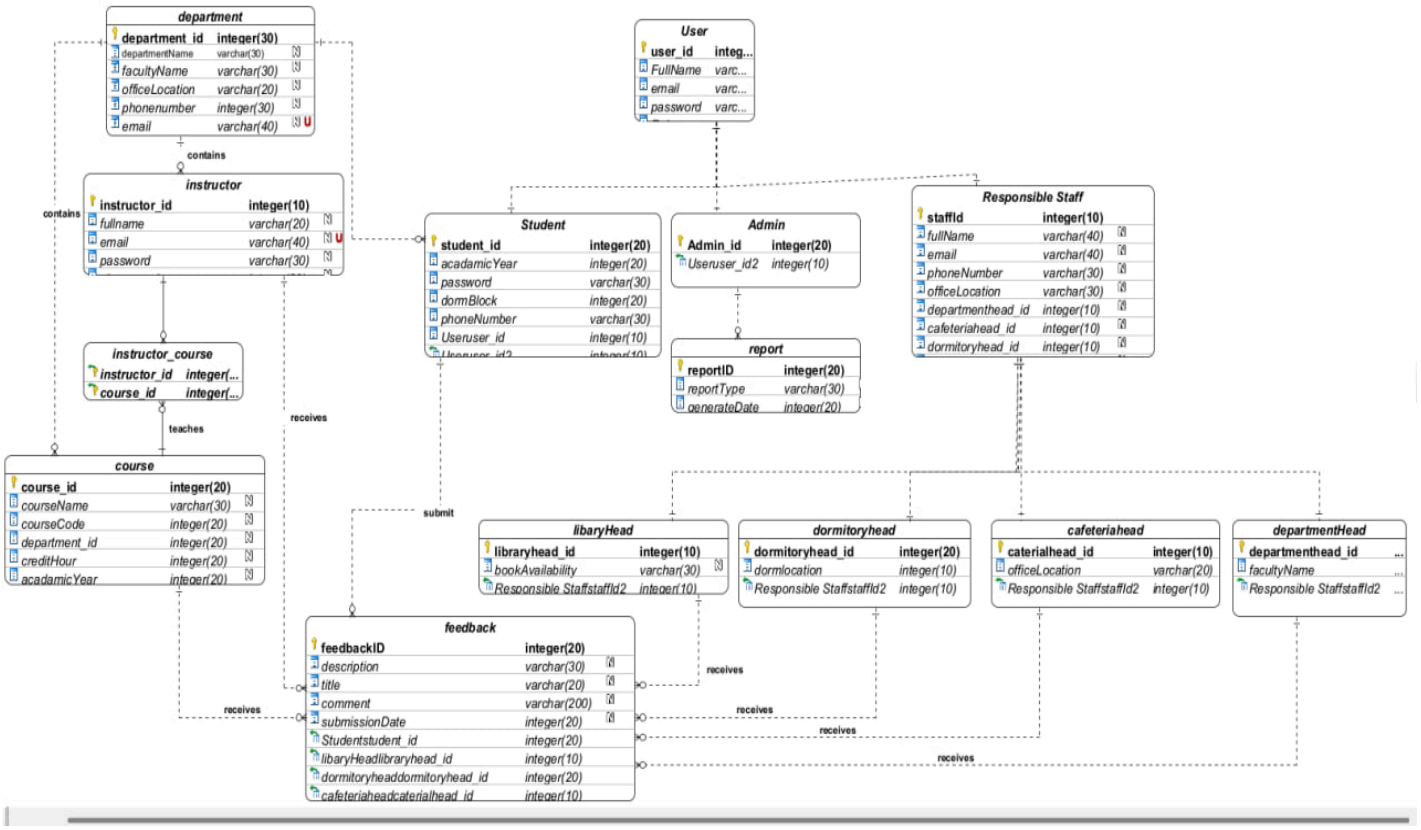


Figure 40:Conceptual ERD


department
department_id integer(30)
departmentName varchar(30)
facultyName varchar(30)
officeLocation varchar(20)
phonenumber integer(30)
email varchar(40)
contains
instructor
instructor_id integer(10)
fullname varchar(20)
email varchar(40)
password varchar(30)
contains
instructor_course
instructor_id integer(...
course_id integer(...
teaches
course
course_id integer(20)
courseName varchar(30)
courseCode integer(20)
department_id integer(20)
creditHour integer(20)
acadamicYear integer(20)
receives
User
user_id integ...
FullName varc...
email varc...
password varc...
Student
student_id integer(20)
acadamicYear integer(20)
password varchar(30)
dormBlock integer(20)
phoneNumber varchar(30)
Useruser_id integer(10)
Admin
Admin_id integer(20)
Useruser_id2 integer(10)
report
reportID integer(20)
reportType varchar(30)
generateDate integer(20)
Responsible Staff
staffId integer(10)
fullName varchar(40)
email varchar(40)
phoneNumber varchar(30)
officeLocation varchar(30)
departmenthead_id integer(10)
cafeteriahead_id integer(10)
dormitoryhead_id integer(10)
submit
libaryHead
libraryhead_id integer(10)
bookAvailability varchar(30)
Responsible Staffstaffld2 integer(10)
dormitoryhead
dormitoryhead_id integer(20)
dormlocation integer(10)
Responsible Staffstaffld2 integer(10)
cafeteriahead
caterialhead_id integer(10)
officeLocation varchar(20)
Responsible Staffstaffld2 integer(10)
departmentHead
departmenthead_id ...
facultyName ...
Responsible Staffstaffld2 ...
feedback
feedbackID integer(20)
description varchar(30)
title varchar(20)
comment varchar(200)
submissionDate integer(20)
Studentstudent_id integer(20)
libaryHeadlibraryhead_id integer(10)
dormitoryheaddormitoryhead_id integer(20)
cafeteriaheadcaterialhead_id integer(10)
receives
receives
receives
receives
receives

4.4.6 Persistence Modelling

Persistence modelling in the **Student Feedback Management System (SFMS)** defines how system data is permanently stored in the database. It specifies database tables, attributes, relationships, and constraints to ensure **data consistency, efficient storage, and reliable retrieval**. This process translates the conceptual and logical data models into a **physical database schema** that supports long-term feedback management at Wachemo University

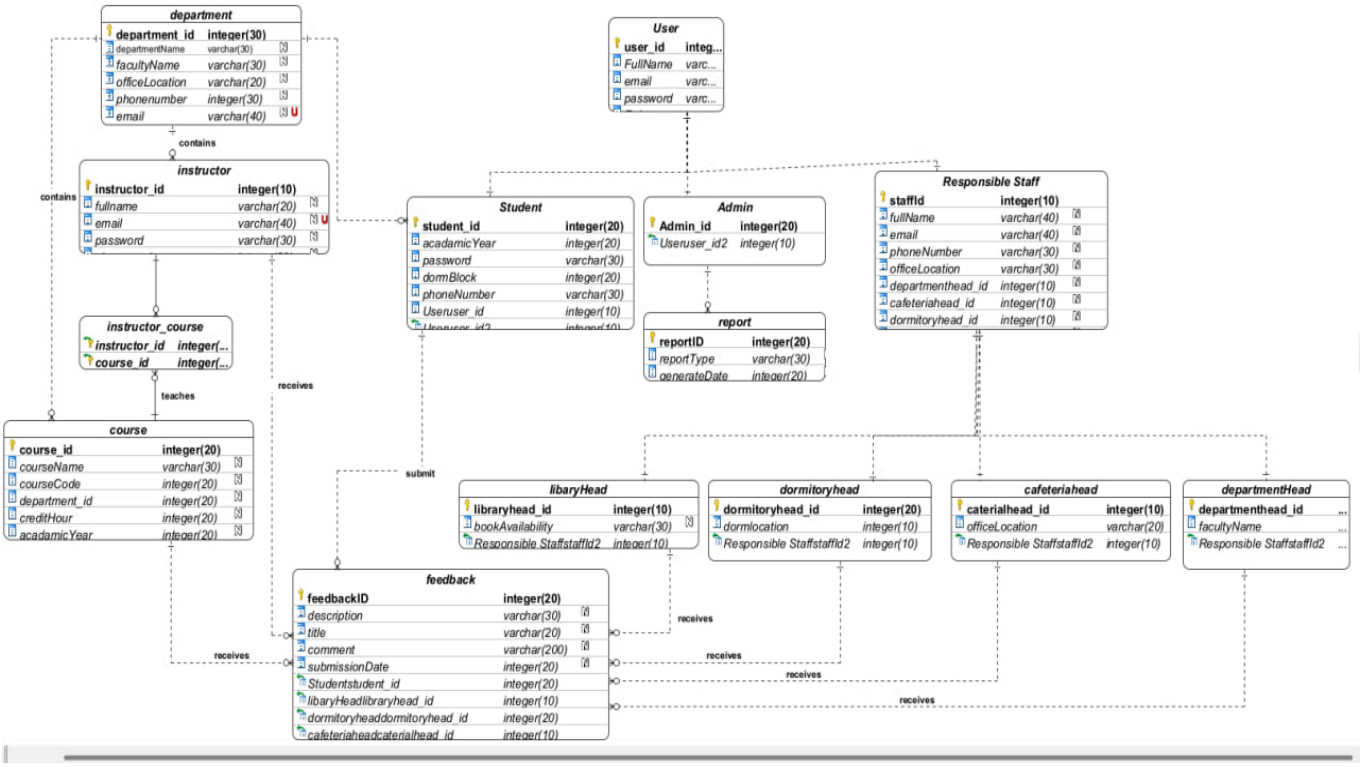


Figure 41:Persistence Modelling

4.4.3 Mapping with Normalization

The mapping and normalization process in the Student Feedback Management System (SFMS) organizes data to reduce redundancy and maintain data integrity. This ensures efficient storage, accurate updates, and reliable querying.

- **First Normal Form (1NF):** All attributes contain atomic values, and there are no repeating groups. Each feedback record stores one rating, one comment, and one submission date.
- **Second Normal Form (2NF):** All non-key attributes fully depend on the entire primary key, especially in tables with composite keys such as feedback submissions.
- **Third Normal Form (3NF):** Transitive dependencies are removed by separating related data into distinct tables (e.g., departments stored separately from students)

Table30: Normalisation

User entity

user_id	fullname	email	password	role
u123	Desalegn worku	des@gmail.com	1234	student
u021	Solomon shukura	solo@gmail.com	1123	instructor

- First Normal Form 1NF

  Already normalized

  - Second Normal Form 2NF
    1NF satisfied

user_id	fullname	email	password	role

    user_id->{ fullname, email, password, role}

· No partial dependency

  - Third Normal Form 3NF

    Must be in 2NF

    User_id->role

User table

user_id	fullname	email	password
u123	Desalegn worku	des@gmail.com	1234
u021	Solomon shukura	solo@gmail.com	1123

Role table

Role_id (pk)	Role_name
1	student
2	instructor

User_role table (support multiple roles)

User_id	Role_id
u123	1
u032	2

Final normalization form (3NF)

user_id	fullname	email	password	Role_name	role_id
u123	Desalegn worku	des@gmail.com	1234	student	1
u021	Solomon shukura	solo@gmail.com	1123	instructor	2

2. student

Student_id	year	Dorm_block	Room_number	Department_id	User_id
Wcu160011	2016	205	007	D01	Wcu1122
Wcu150119	2015	158	012	D02	Wcu0011
Wcu140890	2014	246	032	D03	Wcu0101

- First Normal Form 1NF

All values are atomic or single values

Primary key=student_id

1NF satisfied

- Second Normal Form 2NF

Student_id	year	Dorm_block	Room_number	Department_id	User_id

· Primary key =student_id(single key) all attribute fully depend on it

· No partial dependency

- Third Normal Form 3NF

·

Student table (3NF)

Student_id	year	Department_id	User_id
Wcu160011	2016	D01	Wcu1122
Wcu150119	2015	D02	Wcu0011
Wcu140890	2014	D03	Wcu0101

· Dormitory table 3NF

dorm_id	Dorm_block	Room_no
1	205	007
2	158	012
3	246	032

· Final normalization form (3NF)

Student_id	year	Department_id	User_id	dorm_id	Dorm_block	Room_no
Wcu160011	2016	D01	Wcu1122	1	205	007
Wcu150119	2015	D02	Wcu0011	2	158	012

4.5.Subsystem Decomposition

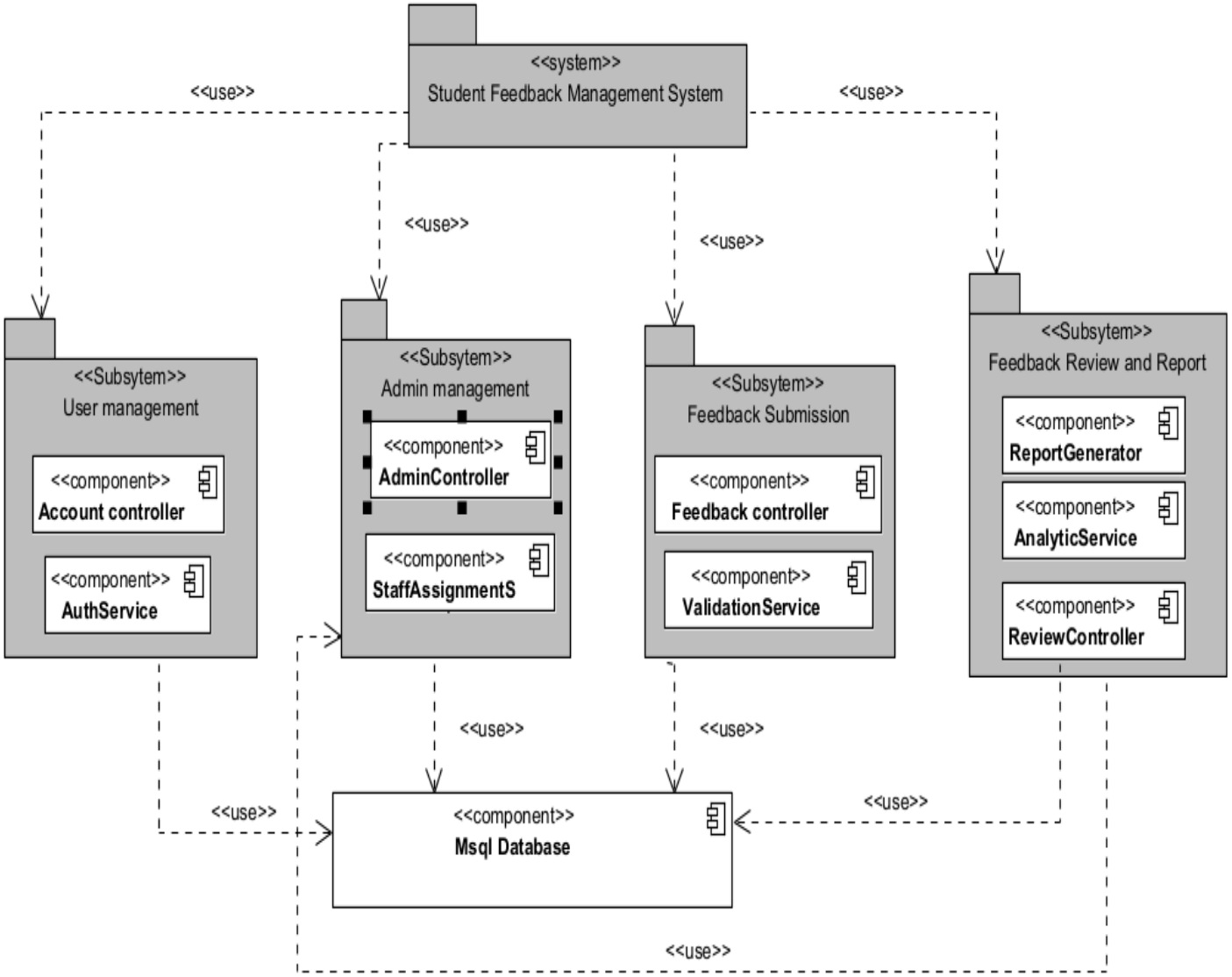


Figure 42: subsystem decomposition

4.6.Deployment Diagram

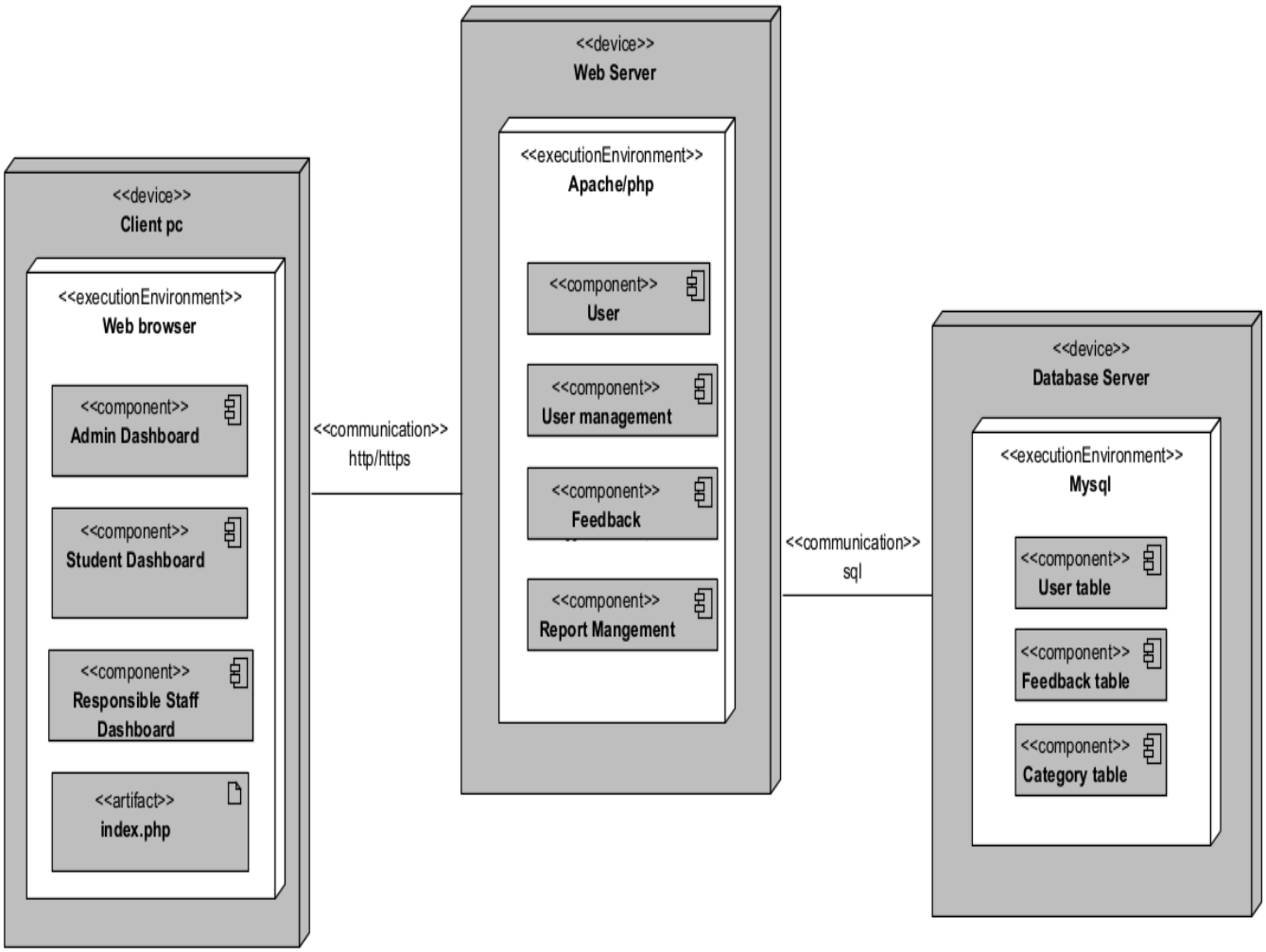


Figure 43: Deployment Diagram

4.6 System Architecture (Layered Architecture of the System)

4.6.1 Overall System Architecture

The Student Feedback Management System uses a **Three-Tier Layered Architecture**, which separates the system into three main layers to improve maintainability, scalability, and security:

1. **Presentation Layer (Client Side)**
   This layer contains the user interfaces where students, administrators, and responsible staff interact with the system.
2. **Application / Business Logic Layer (Web Server)**
   Responsible for implementing system rules, processing input, and managing authentication.
3. **Data Layer (Database Server)**
   Stores and manages all system data

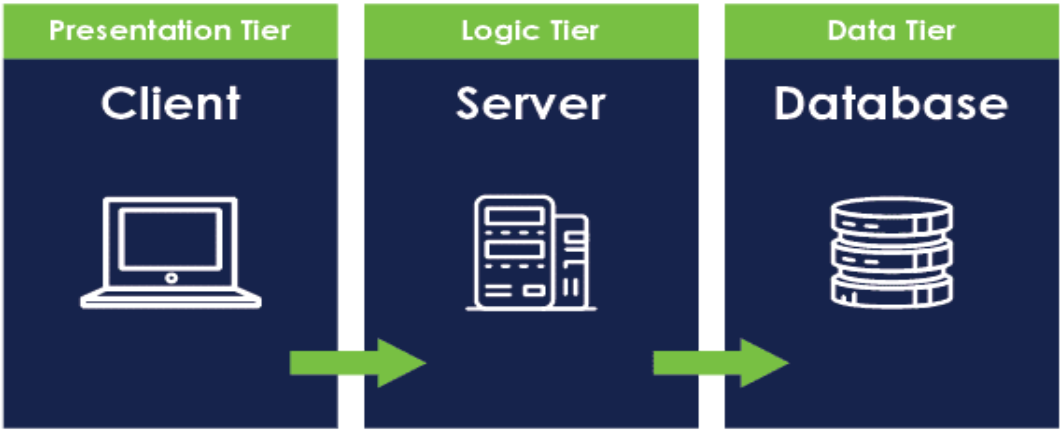


Figure 44:Overall System Architecture

4.6.2 Detail of Architectural Patterns and Styles Used (Three-Tier Architecture)

The system follows a Three-Tier Architecture because it enhances security, simplifies maintenance, and supports future expansion. Each layer has clear responsibilities and works together through structured communication.

1. **Presentation Layer (Client Layer)**
   Technologies: HTML, CSS, JavaScript
   Execution Environment: Web browsers (Chrome, Edge, Firefox…)
   Main Functions:
   - Displays user-friendly interfaces
   - Performs basic validation before sending data to server
   - Sends user requests through HTTP/HTTPS to the server

Users access the system from any device without installing additional software, improving accessibility and usability.

2. **Application / Business Logic Layer (Web Server Layer)**

   Technology: PHP

   Execution Environment: Apache/Php

   Main Functions:

   Manages authentication, session control, and security checks

   Validates and processes all feedback operations

   Communicates data securely with the database using SQL.

   Feedback reports

Controls all sensitive processing securely at the server side, preventing direct access to system data and ensuring correct system behavior.

3. **Data Layer (Database Server Layer)**

   Technology: MySQL DBMS

   Execution Environment: MySQL phpMyAdmin

   Main Functions:

   - Stores and organizes all essential data (student records, feedback, courses, departments)
   - Ensures data consistency, protection, and controlled access
   - Provides reliable data for reporting and decision-making

Centralized data management enables secure storage, easier analysis, and long-term records required by the university.

Table31: Benefits of Three-Tier Architecture for Our System

Key Benefit	How it Helps Our Project
Security	Database is protected since it is only accessible via business logic layer
Easy Maintenance	Database is protected since it is only accessible via business logic layer
Scalability	More services (library, dormitory, etc.) can be added without redesigning the whole system
Performance	Workload is distributed across layers, reducing server strain
Reliability	Each layer can be upgraded or replaced without system failure

The Three-Tier Architecture is selected because it provides a secure, scalable, and maintainable structure where the Presentation Layer handles user interaction, the Business Logic Layer enforces system rules, and the Database Layer ensures reliable data storage. This architecture makes the system suitable for long-term use and future growth at Wachemo University.

4.8.User-Interface (UI) Design

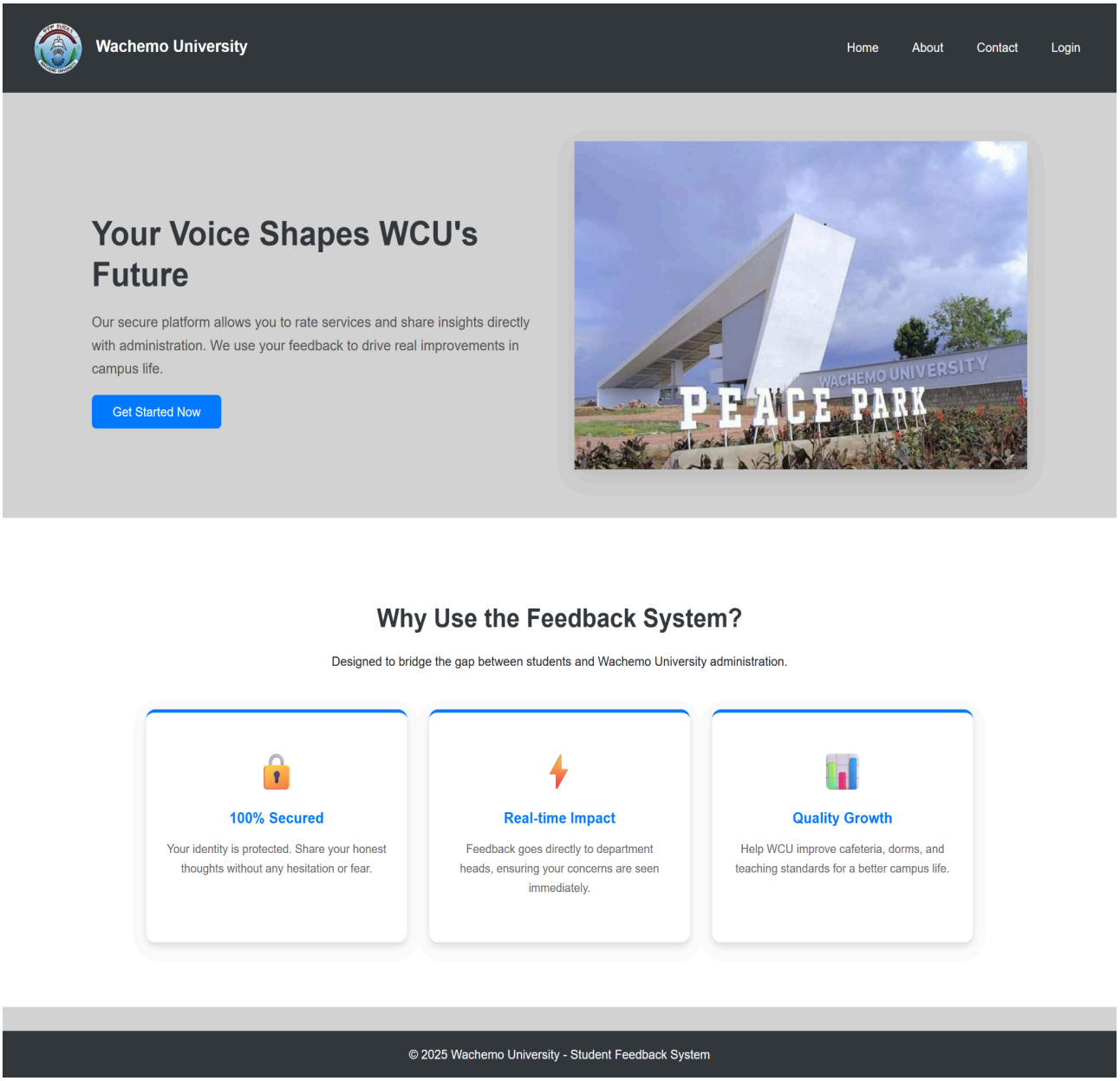


Figure 45: Landing Page Design

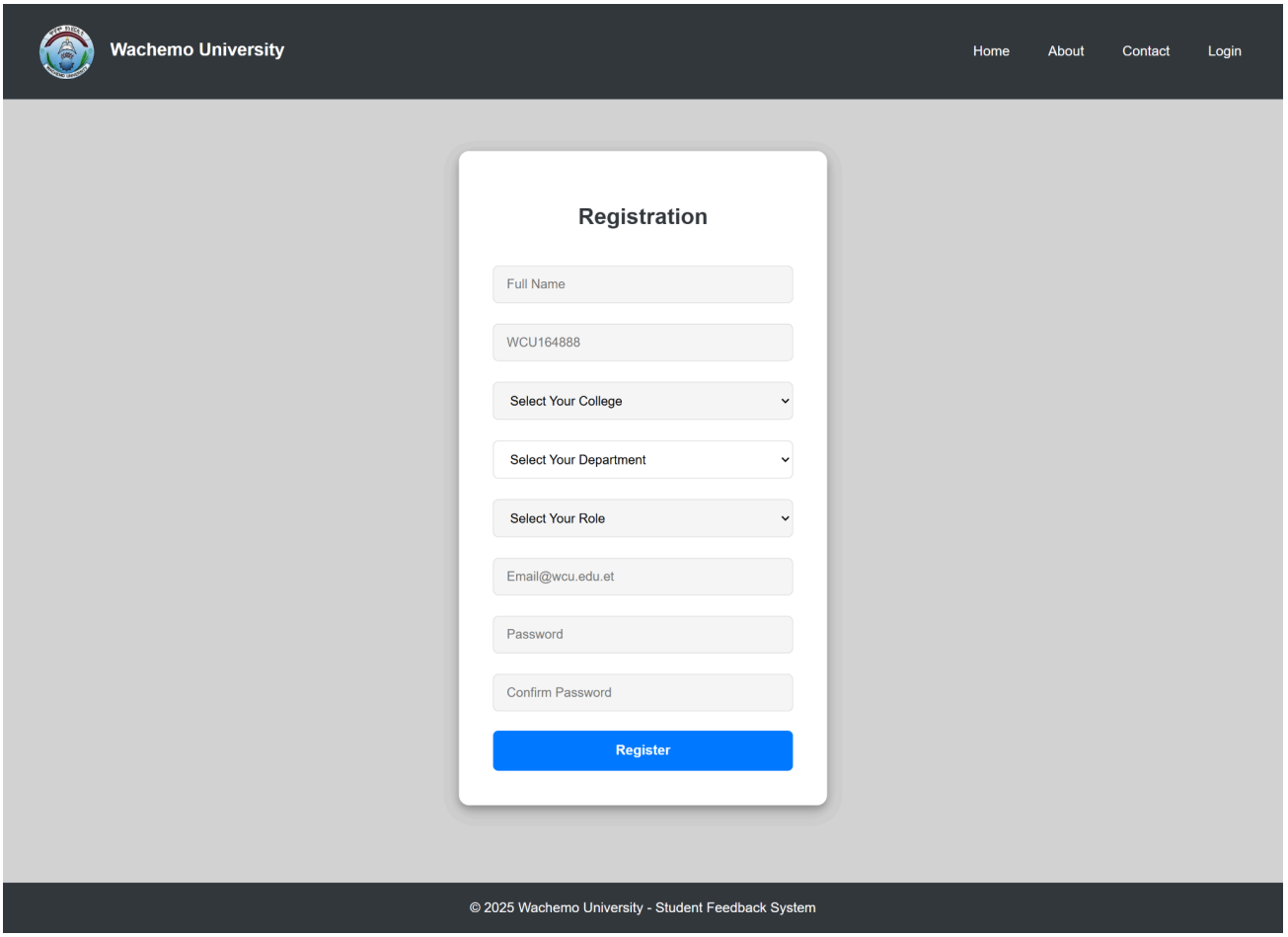


Figure 46: Registration page Design

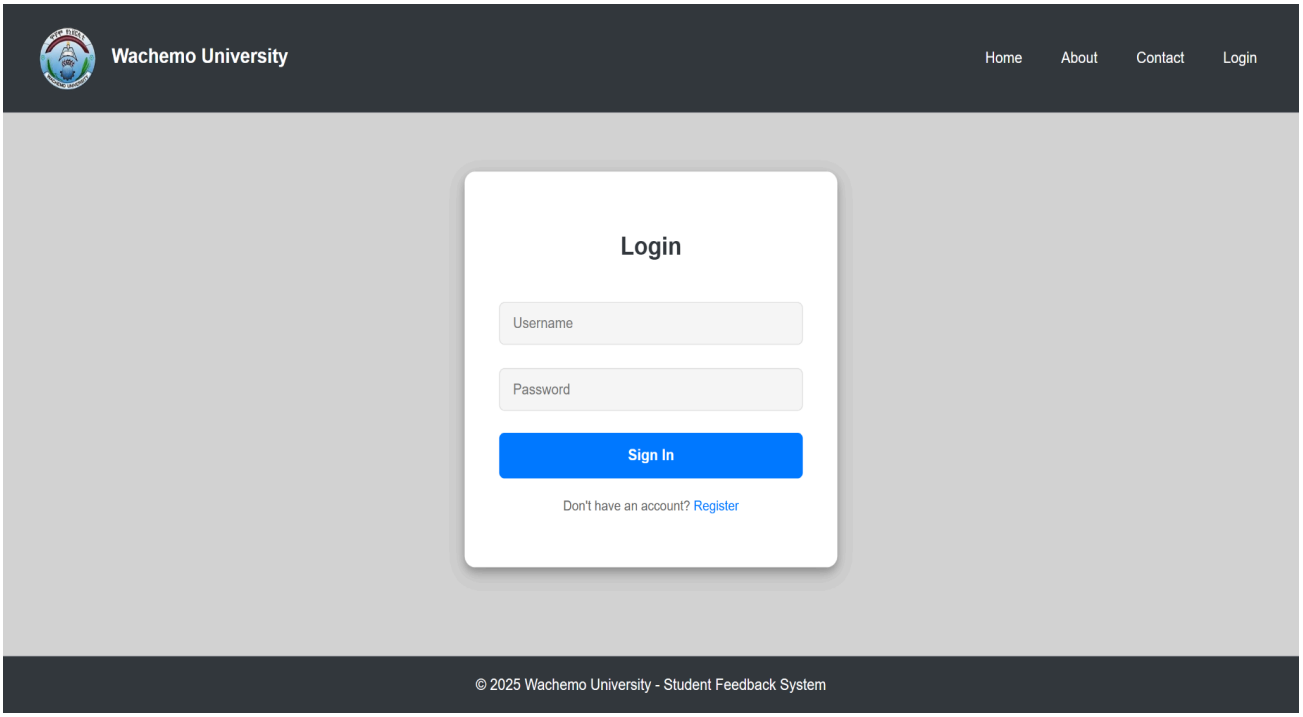


Figure 47: Login Page Design

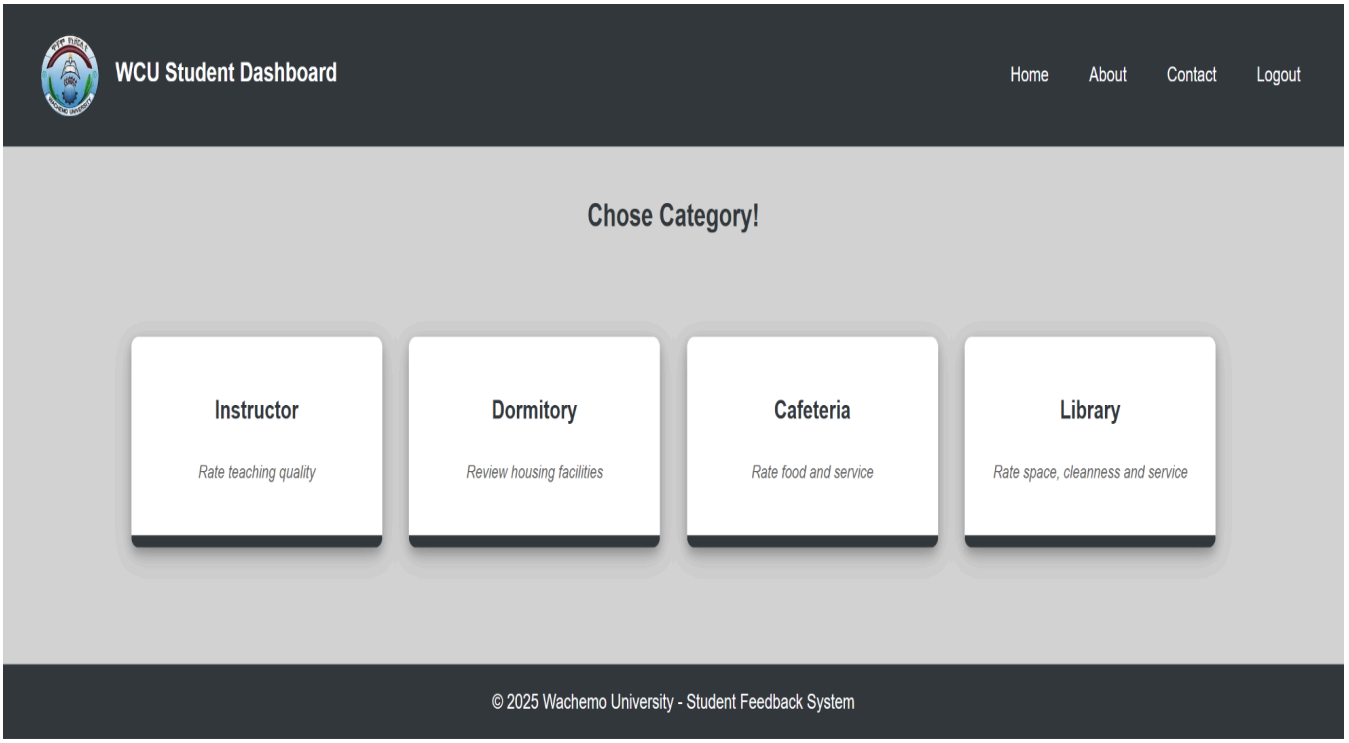


Figure 48: Student Dashboard Design

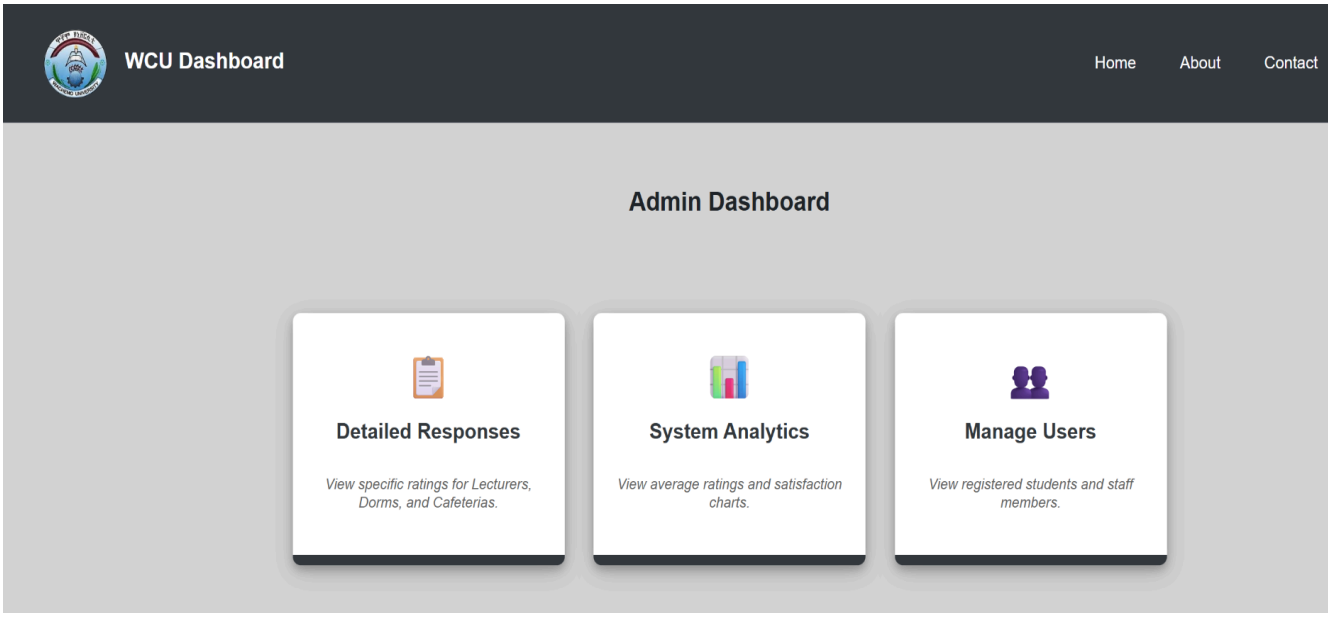


Figure 49: Adim Design Dashboard

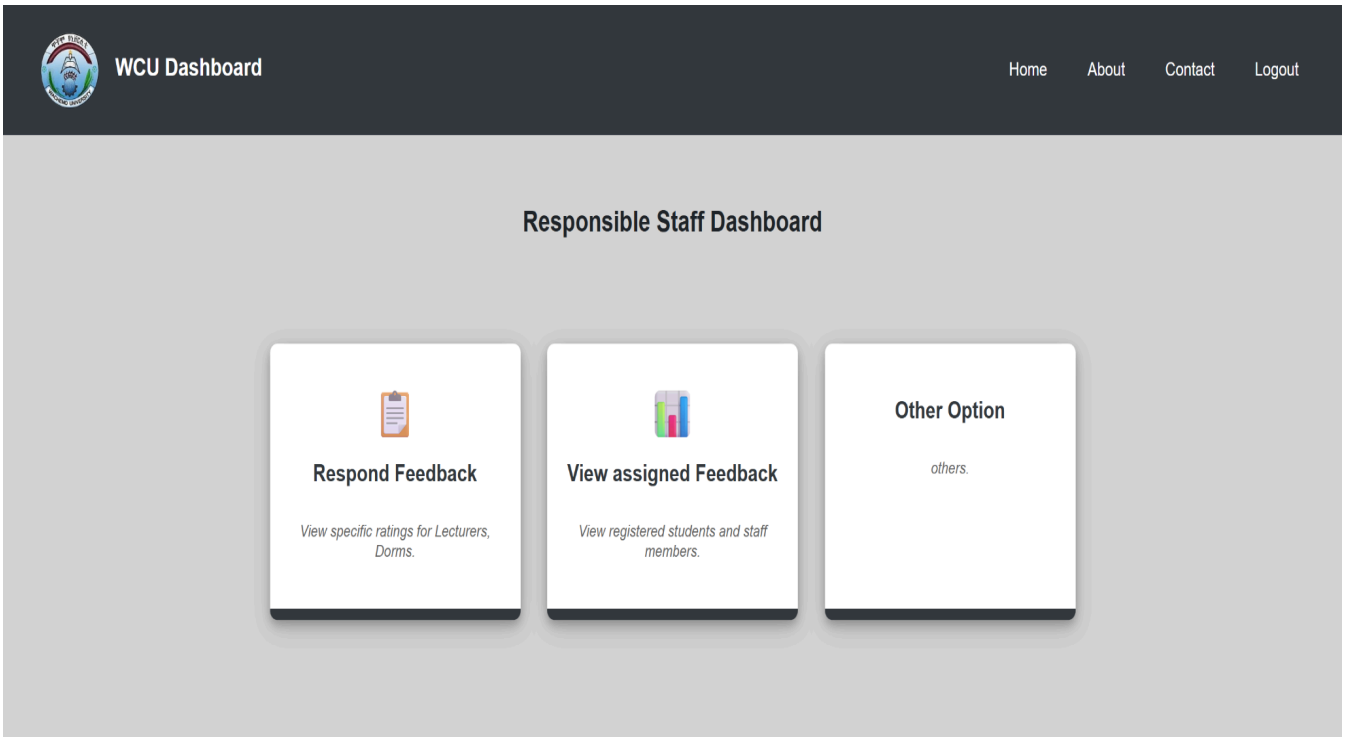


Figure 50: Responsible Staff Dashboard

4.9. UI Flow Diagramming

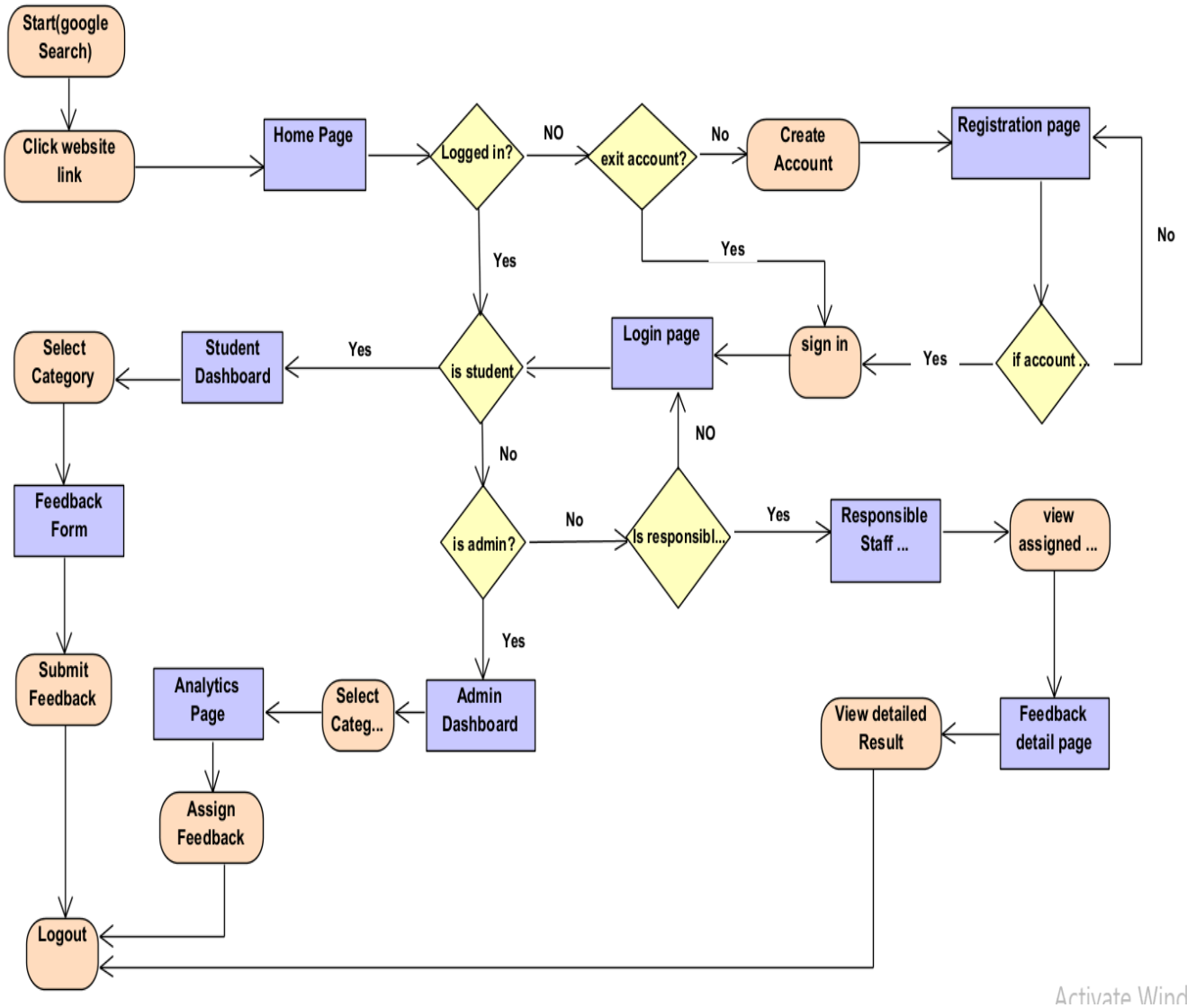


Figure 51: User Interface Design

Chapter Five

5. IMPLEMENTATION AND TESTING

5.1.Algorithm Design

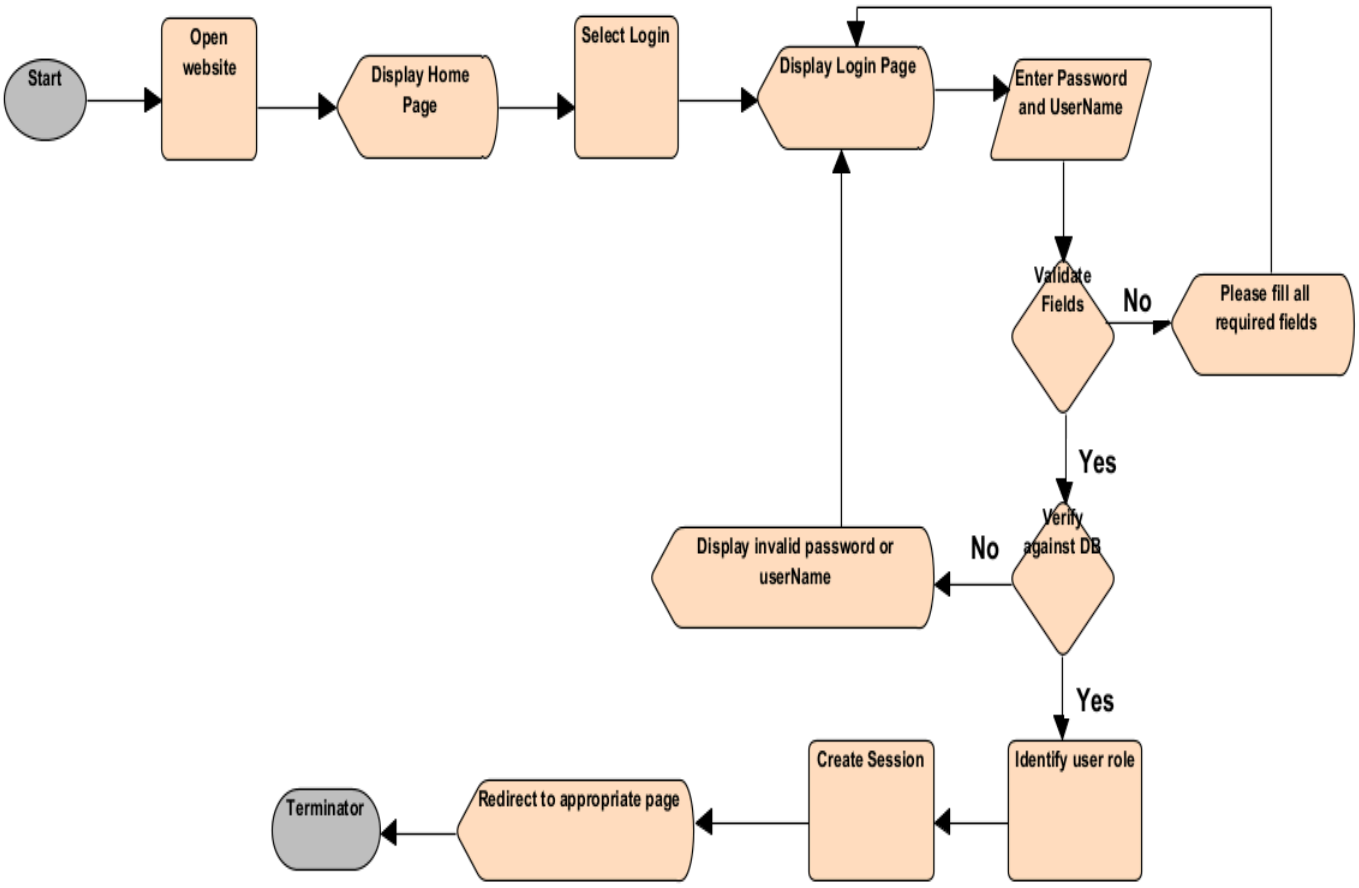


Figure 52: Flowchart for Login

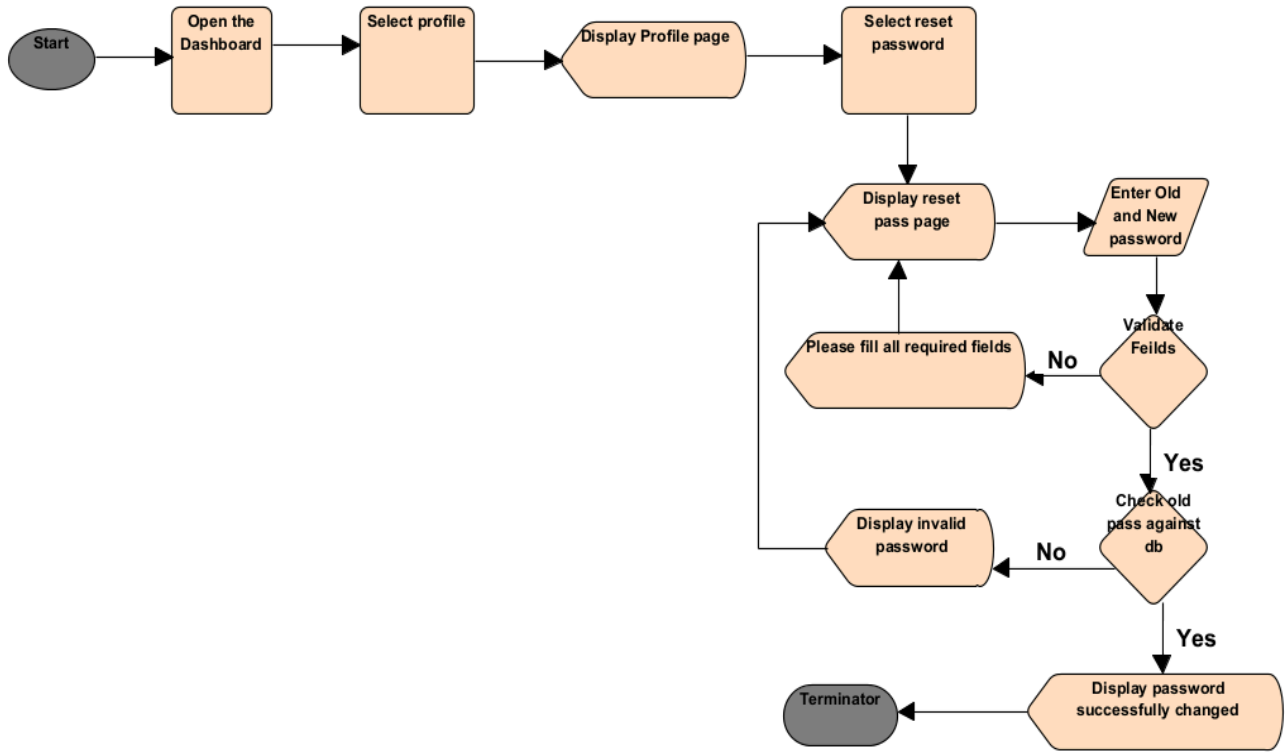


Figure 53: Flowchart for reset password

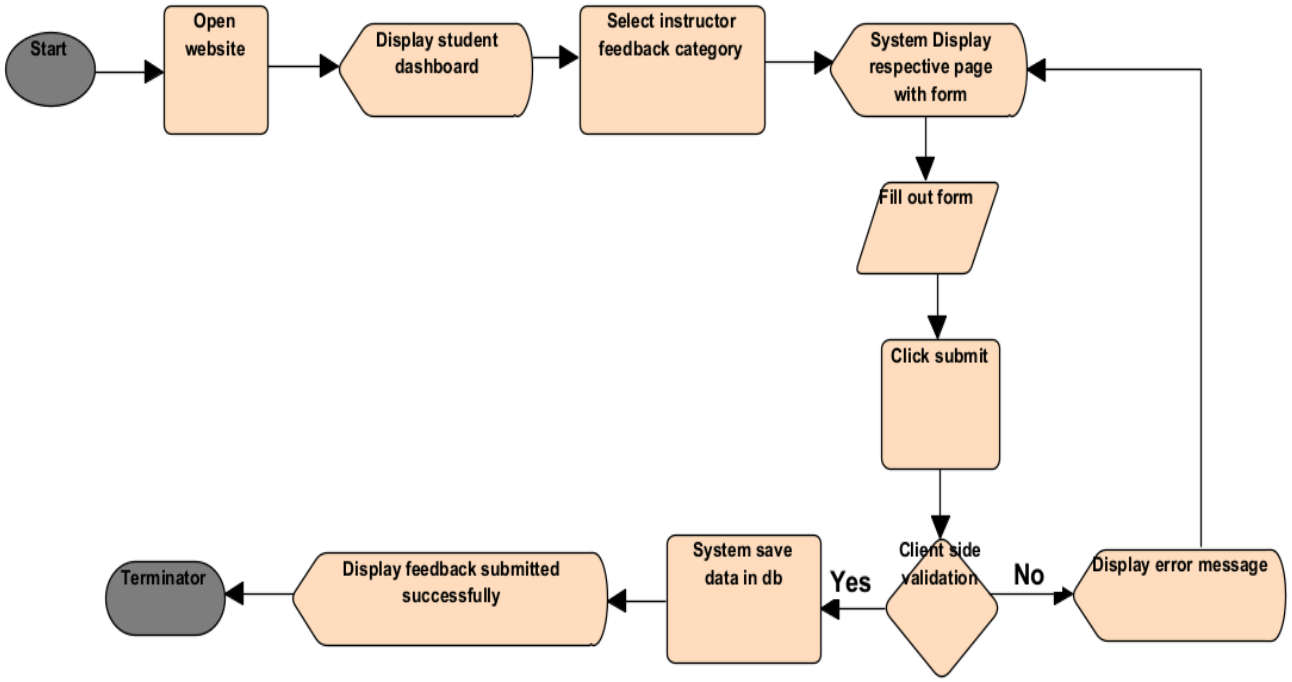


Figure 54: Flowchart for submit course feedback

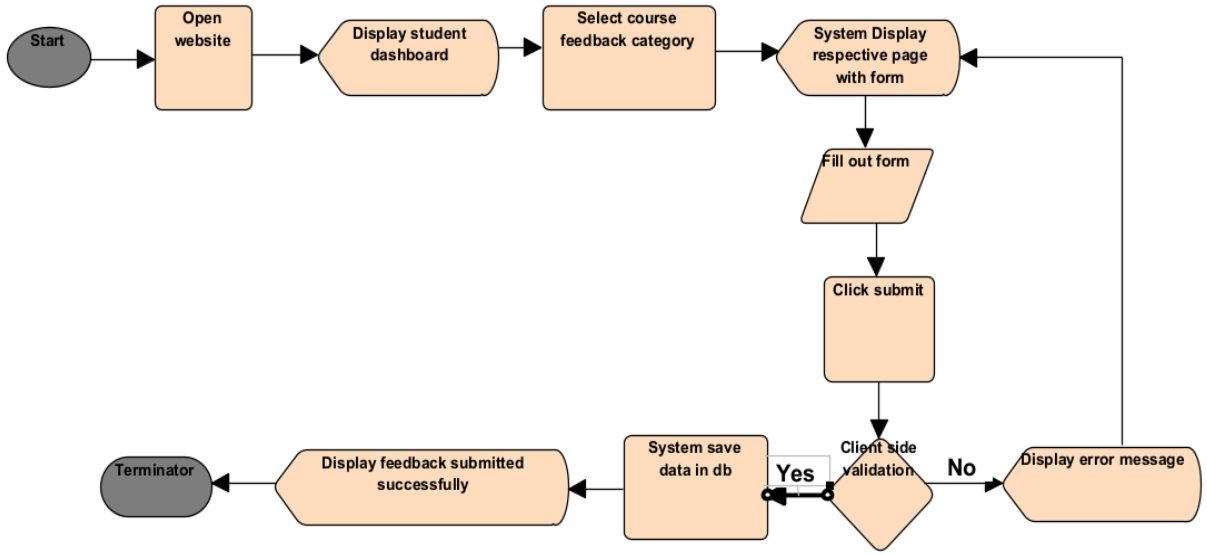


Figure 55: Flowchart for submit course feedback

5.2.Sample Code

```php
<?php
    include "../../config/db.php";
    session_start();

    $userName = mysqli_real_escape_string($conn, $_POST['username']);
    $pass = $_POST['password'];

    $result = mysqli_query($conn, "SELECT * FROM users WHERE username='$userName'");

    if ($userData = mysqli_fetch_assoc($result)) {
        if (password_verify($pass, $userData['password'])) {
            $_SESSION['user_id'] = $userData['id'];
            $_SESSION['role'] = $userData['role'];
            $_SESSION['fullname'] = $userData['fullname'];

            header("Location: ../dashboard/" . ($userData['role']=='admin' ? 'adminDash.php' : 'studentDash.php'));
            exit;
        }
    }


    $_SESSION['error'] = "Invalid username or password. Please try again.";
    header("Location: login.php");
    exit;
?>
```

Figure 56: Sample code for loginHandler.php

```
function validateRegister(event) {
    const passwordField = document.getElementsByName("password")[0];
    const passwordValue = passwordField.value;

    const oldMessage = document.getElementById("pw-error");
    if (oldMessage) {
        oldMessage.remove();
    }

    if (passwordValue.length < 6) {
        event.preventDefault();
        let errorMsg = document.createElement("p");
        errorMsg.id = "pw-error";
        errorMsg.innerText = "Password must be 6+ characters.";
        errorMsg.style.color = "red";
        errorMsg.style.fontSize = "13px";
        errorMsg.style.marginBottom = "5px";

        passwordField.parentNode.insertBefore(errorMsg, passwordField);
        passwordField.style.border = "2px solid red";      You, last week • Fix
    } else {
        passwordField.style.border = "1px solid #ccc";
    }
}
```

Figure 57: Sample code for Client-Side Validation

```
<?php
session_start();      You, last week • First Verision of The system …

function requireLogin() {
    if (!isset($_SESSION['user_id'])) {
        header("Location: ../login.php");
        exit;
    }
}

function requireRole($role) {
    if ($_SESSION['role'] !== $role) {
        die("Access denied");
    }
}
```

Figure 58: Sample code auth.php

5.3 Test Strategy

5.3.1 Objectives, scope, testing levels (unit, integration, system, acceptance)

Objectives

- Ensure each module of the Student Feedback Management System functions correctly.
- Identify and fix defects early at the module level before system integration.
- Verify that the system satisfies Wachemo University's requirements for students, administrators, and responsible staff.
- Improve system reliability, security, and data accuracy before deployment.

Scope

The testing scope covers the core functionalities of the system, including user registration, login, and password reset; feedback submission for courses, instructors, cafeteria, library, and dormitory services; role-based access control for different users; and feedback viewing and report generation. Testing does not include external systems such as SMS or email gateways, nor unrelated university services that are outside the project boundary

Testing Levels

1. Unit Testing

Unit testing focuses on testing individual functions and classes in isolation. It covers PHP functions for login validation, password hashing and verification, OTP generation and verification, feedback insertion functions, and report calculation methods. The tools used include PHPUnit, which tests PHP classes and methods such as login(), resetPassword(), and submitFeedback() independently; PHP's built-in assertion functions for simple logic verification during early testing; and JavaScript console assertions (console.assert) to verify frontend validation logic, such as checking empty fields or rating ranges. These tools ensure that the smallest units of code in PHP and JavaScript work correctly, aligning with our project's implementation style

2. Integration Testing

Integration testing checks how different modules work together. It verifies complete functional flows such as Login ,Dashboard , Feedback Submission , Database storage, Password reset , OTP verification ,Password update, and Admin login , Report generation ,Data retrieval from the database. This testing uses PHPUnit integration test cases to examine interactions between PHP classes and database queries, phpMyAdmin to confirm that data is correctly stored or updated after each action, and the Apache + MySQL environment provided by WAMPto simulate real execution flow across system modules. These tools ensure that backend logic, database operations, and user actions are properly connected..

3. System Testing

System testing evaluates the entire application as a complete system. It focuses on verifying full user workflows such as Register , Login , Submit Feedback ,Logout, checking role-based dashboards for students, administrators, and responsible staff, and validating error handling and system messages. This testing is performed using web browsers such as Chrome, Firefox, and Edge to observe system behavior across different browsers, along with browser developer tools to inspect network requests, session handling, and user interface behavior. System testing relies on real user interaction, which browsers accurately provide..

4. Acceptance Testing
Acceptance testing ensures that the system meets the expectations and requirements of Wachemo University stakeholders. During this phase, the feedback submission process is evaluated to confirm it follows university evaluation rules, generated reports are reviewed to ensure they are clear, understandable, and usable by responsible staff, and access control is verified to comply with WCU policies. Acceptance testing is mainly performed using manual test scenarios and checklists, while spreadsheets such as Excel or Google Sheets are used to document acceptance criteria, test results, and approval status. This level of testing focuses on real-world system usage rather than internal technical implementation details.

5.3.2 Test Environment

- Server: Apache Web Server (WAMP)
- Backend Language: PHP
- Database: MySQL
- Client: Web browsers on personal devices and university lab computers
- Network: University LAN or Wi-Fi

5.3.3 Hardware and Software Setup

Software	**Hardware**
Apache Server (WAMP)	Desktop or Laptop computer
PHP 8+	Minimum 4 GB RAM
MySQL	Stable internet or local network access
Web Browsers	
Spreadsheet software for test documentation	

Table32 : Hardware and Software Setup for test

5.3.4 Tools Required

PHPUnit:	Used for unit testing PHP classes and functions such as login validation and password reset logic.
phpMyAdmin:	Used to verify database operations such as record insertion and updates during testing
Browser Developer Tools:	Used to test form validation, UI behavior, and error handling
Google Sheets:	Used to document test cases, expected outputs, and actual results.

Table33: Tools Required for test

5.3.5 Test Cases

Input Conditions and Expected Outputs

Valid login credentials	User redirected to correct dashboard
Invalid password	Error message displayed
Valid feedback submission	Feedback saved in database
Empty feedback field	Validation error shown

Table34: Input Conditions and Expected Outputs

Boundary Cases and Error Handling
Empty input fields during login or registration
Duplicate student ID or email during registration
Invalid or expired OTP during password reset
Attempt to resubmit feedback for the same course

5.3.6 Validation and Verification

Verification ensures that the Student Feedback Management System is developed according to the specified requirements. This is achieved through systematic testing of individual modules, integration flows, and overall system behavior, as well as code reviews to confirm adherence to coding standards and design patterns. Validation, on the other hand, ensures that the system meets the actual needs of Wachemo University students, administrators, and responsible staff by testing real-world scenarios, such as submitting feedback, generating reports, and managing accounts. To support this process, a **traceability matrix** is maintained, linking each functional requirement to one or more corresponding test cases. This guarantees that all requirements are tested and no essential functionality is overlooked. Tools like **Excel or Google Sheets** are used to create and manage the traceability matrix, providing an organized overview of requirements, test cases, and verification status.

5.3.7 Results Documentation

Test results will be systematically recorded for each test case. Both expected and actual outcomes will be documented, and cases will be clearly marked as passed or failed. Observed issues, system responses, and deviations from expected behavior will be noted for reference and future improvements. This documentation will serve as evidence of proper testing and help in reviewing the system's reliability and completeness. Tools like Google Sheets, test tracking templates will be used to maintain clear and organized records

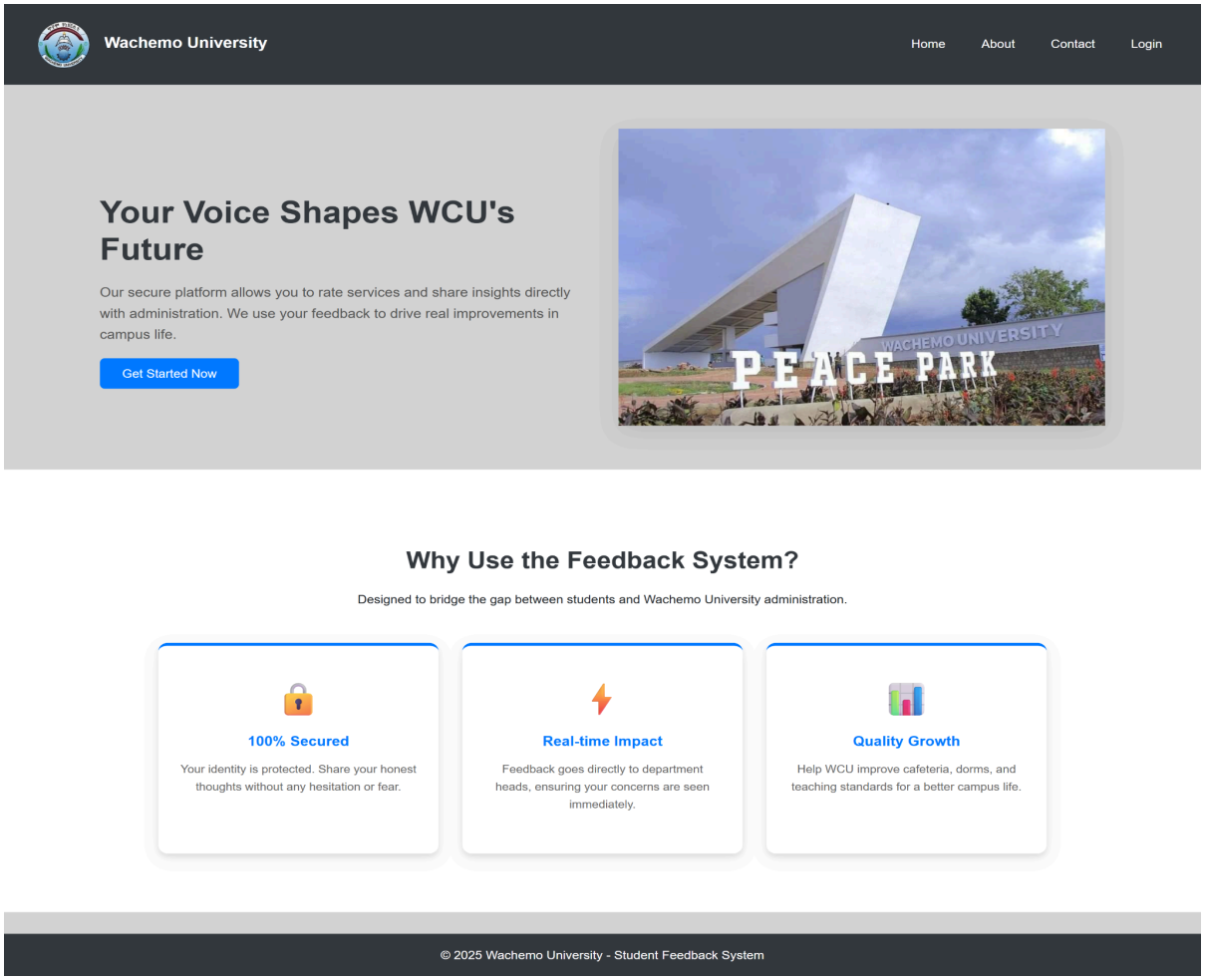
5.3.8 Bug Tracking

Any problems found during testing will be recorded in defect reports. Each report will include a clear description of the problem, steps to reproduce it, how serious it is, and which part of the system is affected. Every bug will be followed until it is fixed, and we will check again to make sure the problem is solved. Tools like GitHub Issues, Trello, or simple spreadsheets will be used to keep track

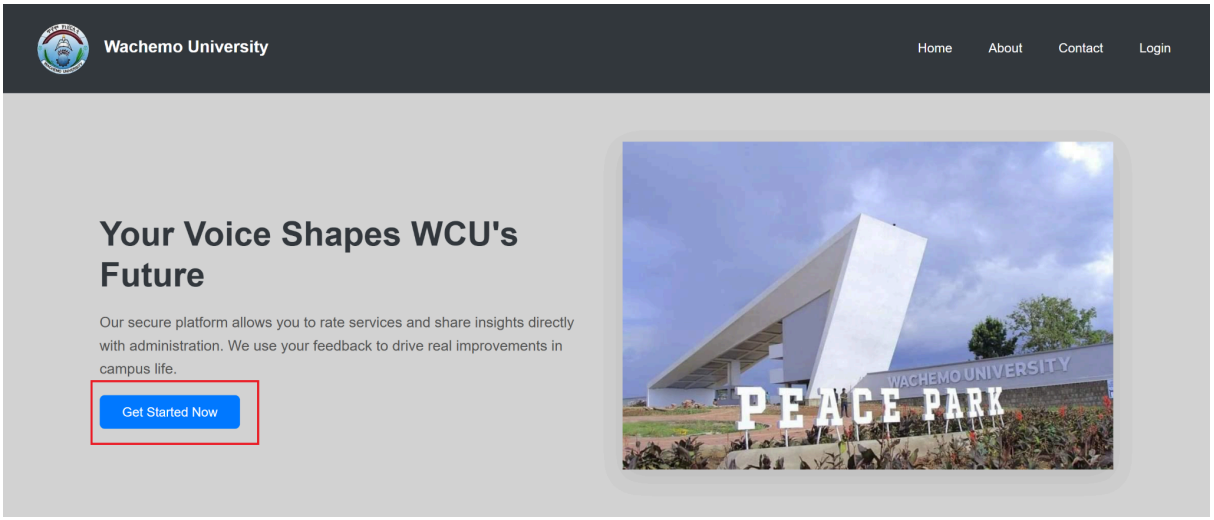
of bugs. This helps make sure all problems are fixed before the system is submitted, keeping the Student Feedback Management System for Wachemo University reliable and high quality.

5.4.User-Manual

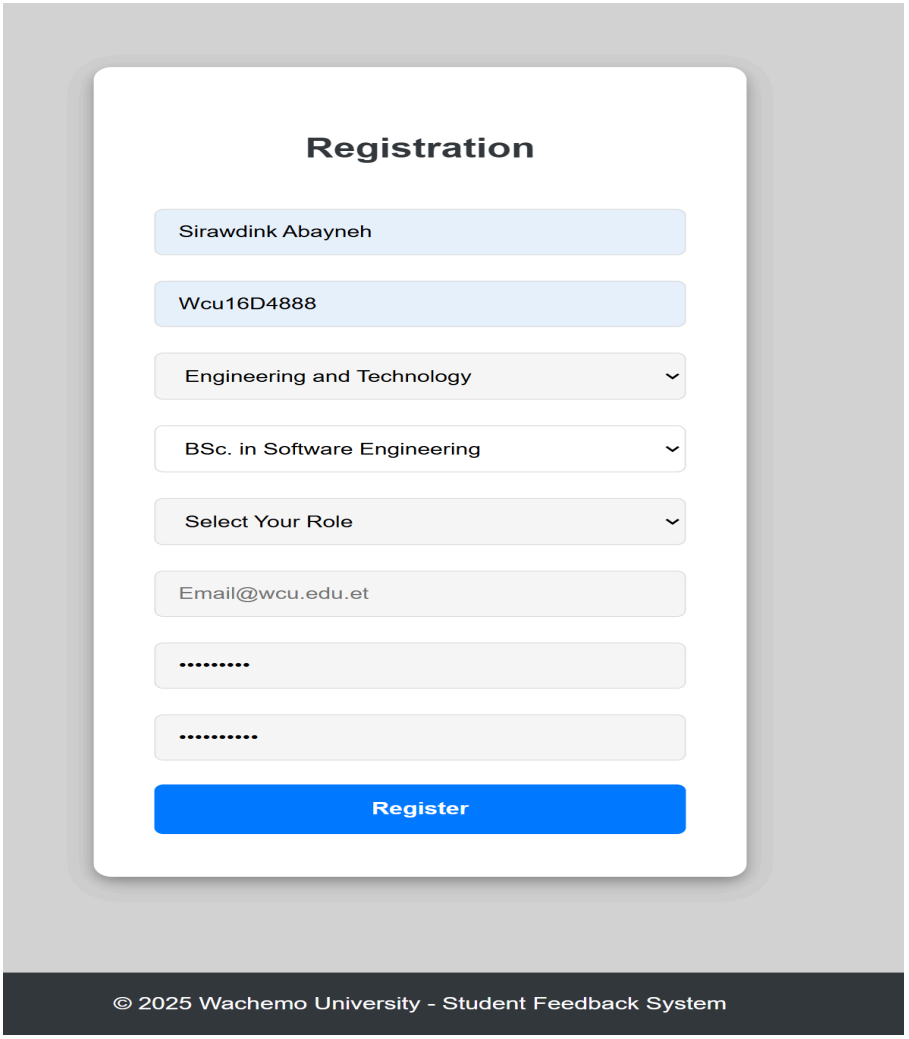
1. How to Register Account
   Step1 Open Website



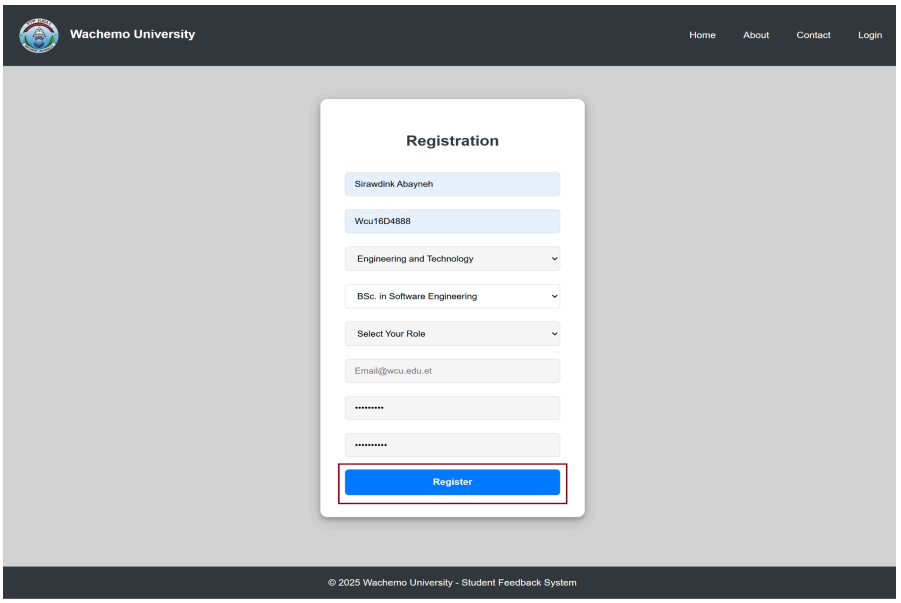
Step2: Click get started



Step 3 Fill out personal and academic details



Step4 Click Register

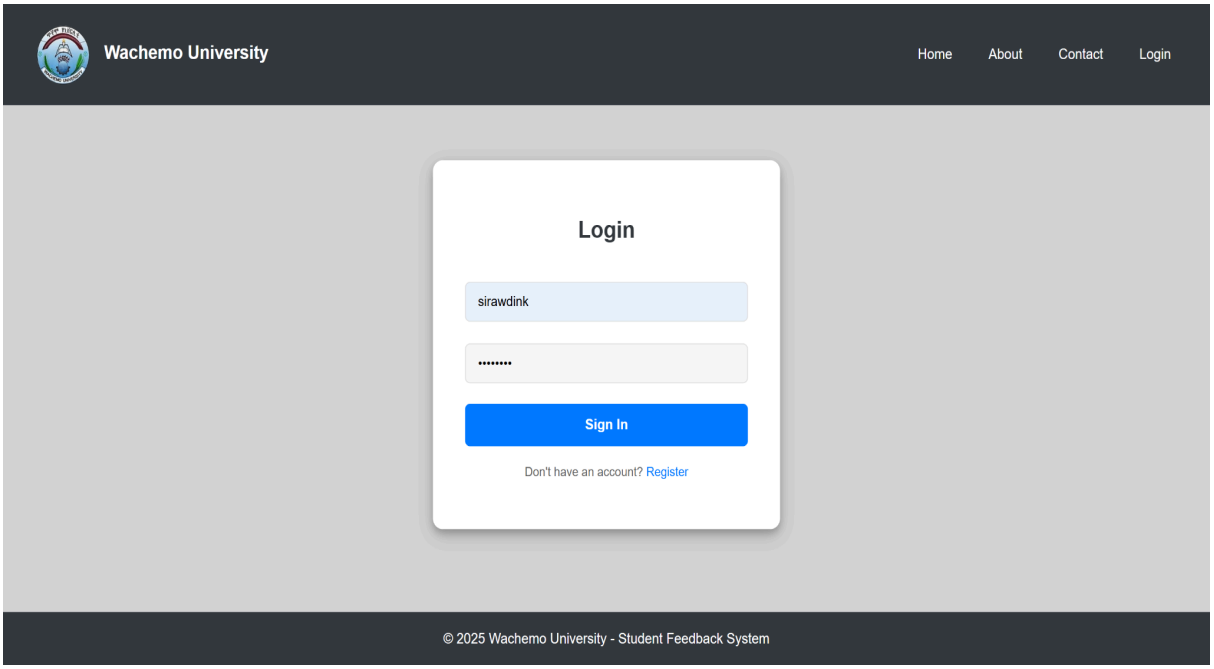


Step3 Upon successful verification, you will redirect to Login page

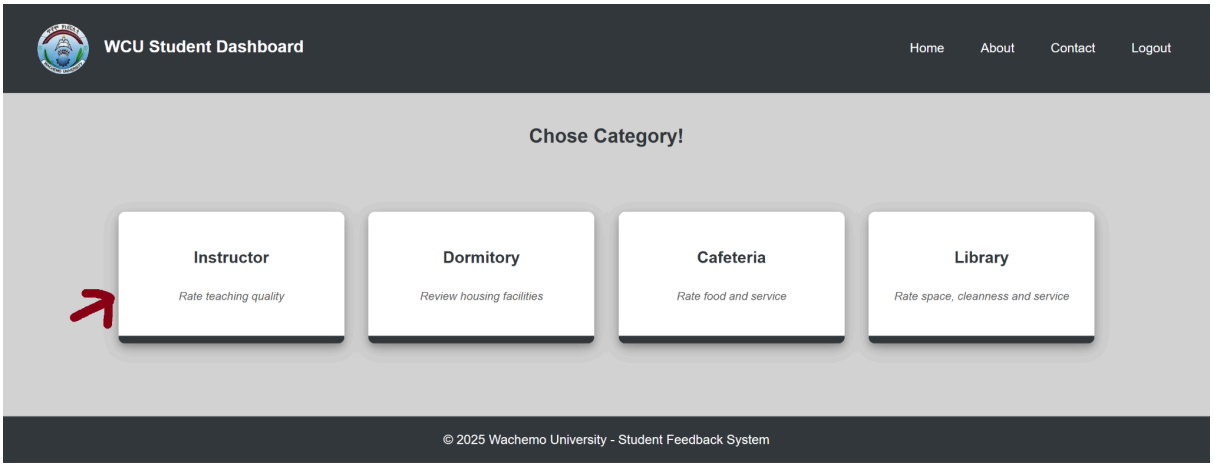
Note: Ensure all fields are correctly filled

2. Steps to submit a feedback for Instructor

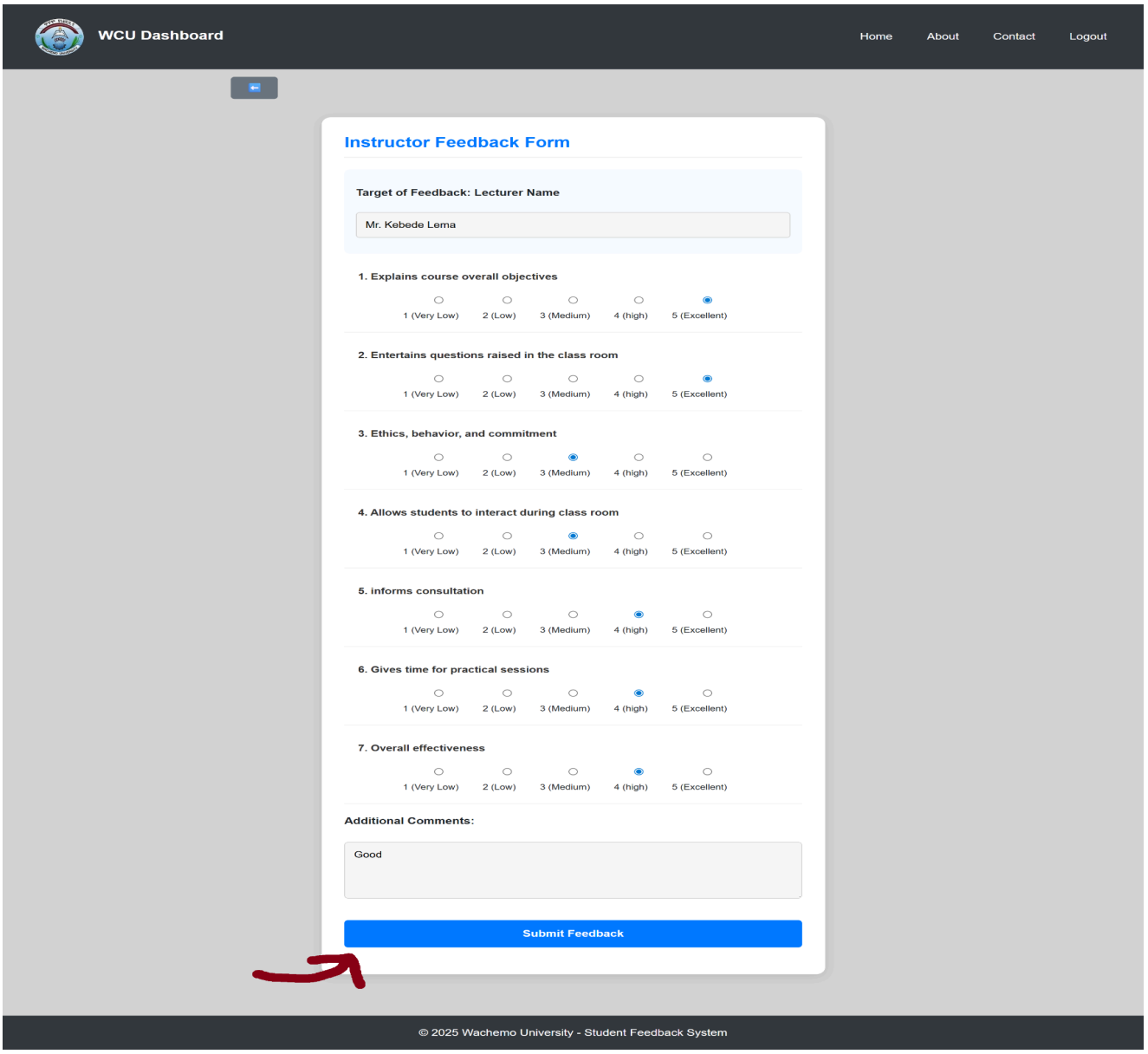
Step1: Login in the system by filling required credential



Step2: Navigate Category on Dashboard and Select



Step3 Fill out the feedback for and Click Submit



Step 4 Upon successful verification , You will be redirect to Student dashboard

5.5 User Training

User training is important to ensure that students, administrators, and responsible staff can use the Student Feedback Management System (SFMS) effectively and confidently. The system is designed to be simple and user-friendly, so only minimal training is required.

For **students**, basic guidance will be provided through short instructions on the login page and within the system interface. A user manual with screenshots will explain how to register, log in, submit feedback, view submission history, update profiles, and log out. This helps students understand the process without needing formal training sessions.

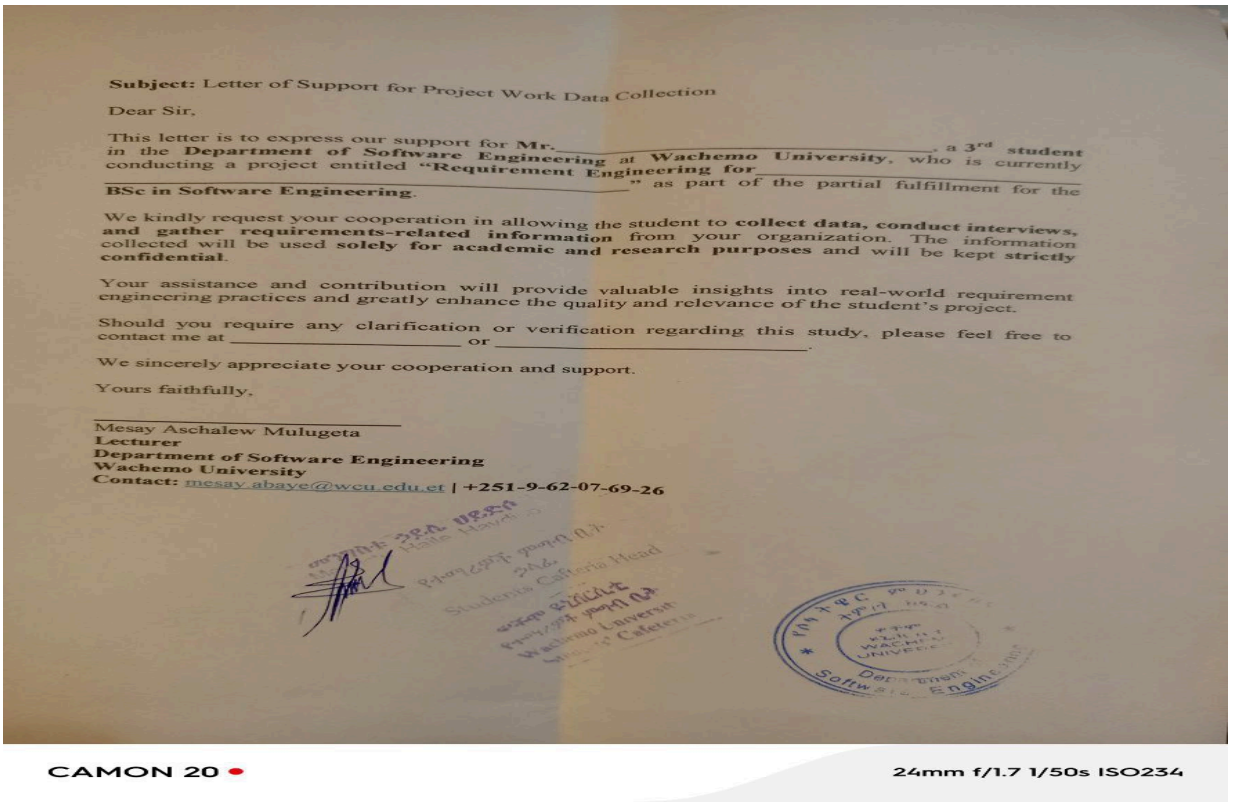
For **administrators and responsible staff**, brief orientation sessions can be conducted to explain account management, feedback review, report generation, and feedback assignment. Written guidelines and sample screenshots will also be provided to support daily system use.

In addition, basic help messages and clear labels within the system guide users during each action. This training approach reduces confusion, supports smooth system adoption, and ensures that all users can effectively participate in the feedback process at Wachemo University.

Appendix I: Interview Conducted

This appendix documents the interviews conducted during the requirement gathering phase of the Student Feedback Management System (SFMS). The interviews helped us understand the current feedback process, identify existing problems, and gather user expectations for the proposed system.

Figure :interview conducted



Subject: Letter of Support for Project Work Data Collection

Dear Sir,

This letter is to express our support for Mr._____, a 3rd student in the **Department of Software Engineering** at **Wachemo University**, who is currently conducting a project entitled "**Requirement Engineering for**_____" as part of the partial fulfillment for the **BSc in Software Engineering**.

We kindly request your cooperation in allowing the student to **collect data, conduct interviews, and gather requirements-related information** from your organization. The information collected will be used **solely for academic and research purposes** and will be kept **strictly confidential**.

Your assistance and contribution will provide valuable insights into real-world requirement engineering practices and greatly enhance the quality and relevance of the student's project.

Should you require any clarification or verification regarding this study, please feel free to contact me at _____ or _____.

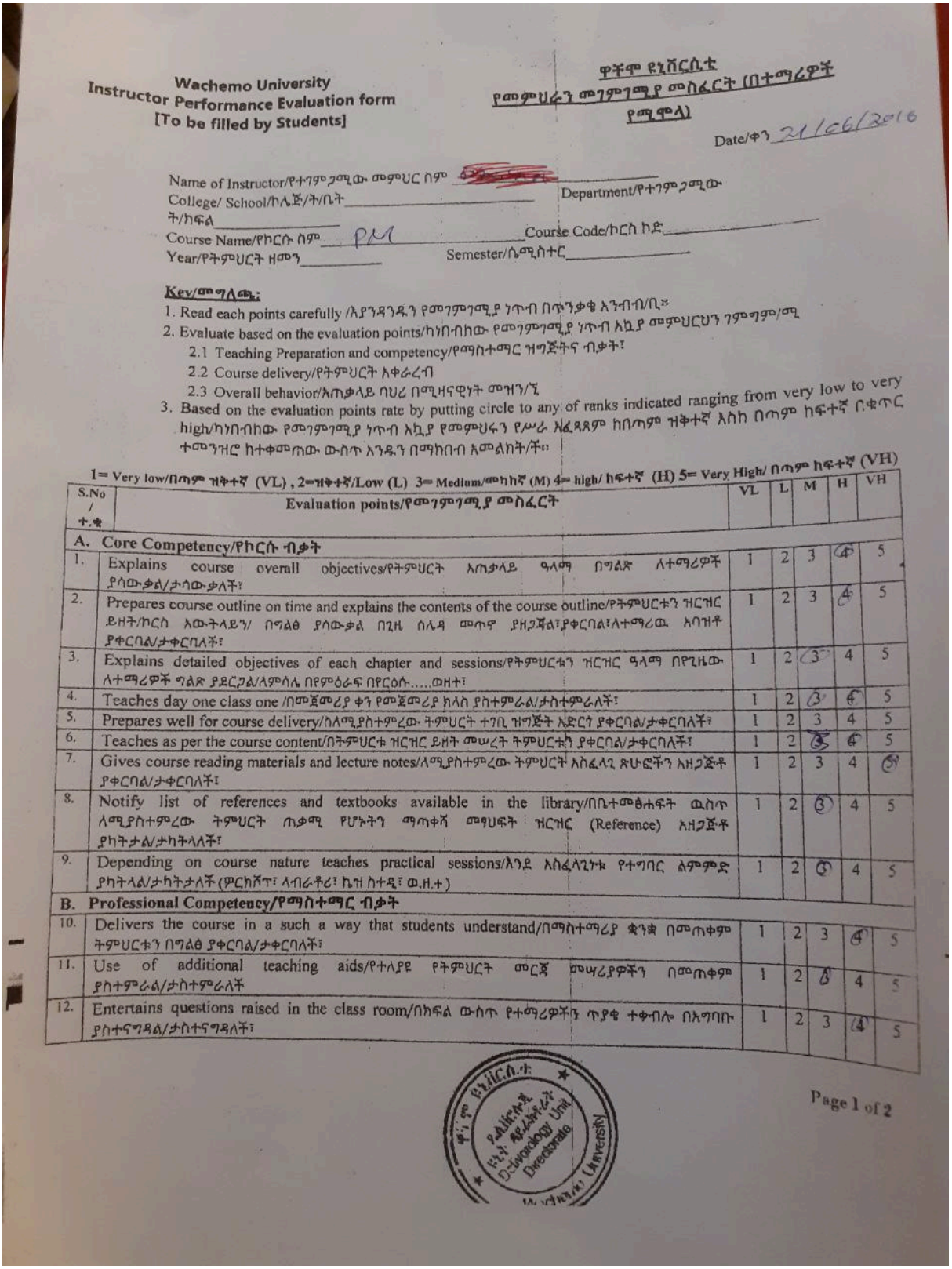
We sincerely appreciate your cooperation and support.

Yours faithfully,

_____
Mesay Aschalew Mulugeta
Lecturer
Department of Software Engineering
Wachemo University
Contact: mesay.abaye@wcu.edu.et | **+251-9-62-07-69-26**

Appendix II: Sample Paper-Based Documents

This appendix includes examples of traditional paper documents that inspired the digital system design



Wachemo University
Instructor Performance Evaluation form
[To be filled by Students]

ዋቸሞ ዩኒቨርሲቲ
የመምህራን መገምገሚያ መስፈርት (በተማሪዎች የሚሞላ)

Date/ቀን 21/06/2016

Name of Instructor/የተገመጋሚው መምህር ስም _____
College/ School/ኮሌጅ/ት/ቤት _____ Department/የተገመጋሚው ት/ክፍል _____
Course Name/የኮርሱ ስም PM Course Code/ኮርስ ኮድ _____
Year/የትምህርት ዘመን _____ Semester/ሴሚስተር _____

Key/መግለጫ:

1. Read each points carefully /እያንዳንዱን የመገምገሚያ ነጥብ በጥንቃቄ እንብብ/ቢ።
2. Evaluate based on the evaluation points/ካነበብከው የመገምገሚያ ነጥብ አኳያ መምህርህን ገምግም/ሚ
   2.1 Teaching Preparation and competency/የማስተማር ዝግጅትና ብቃት፣
   2.2 Course delivery/የትምህርት አቀራረብ
   2.3 Overall behavior/አጠቃላይ ባህሪ በሚዛናዊነት መዝን/ኚ
3. Based on the evaluation points rate by putting circle to any of ranks indicated ranging from very low to very high/ካነበብከው የመገምገሚያ ነጥብ አኳያ የመምህሩን የሥራ አፈጻጸም ከበጣም ዝቅተኛ እስከ በጣም ከፍተኛ ቦቁጥር ተመንዝሮ ከተቀመጠው ውስጥ አንዱን በማክበብ አመልክት/ች።

1= Very low/በጣም ዝቅተኛ (VL), 2=ዝቅተኛ/Low (L) 3= Medium/መካከኛ (M) 4= high/ ከፍተኛ (H) 5= Very High/ በጣም ከፍተኛ (VH)

S.No / ተ.ቁ	Evaluation points/የመገምገሚያ መስፈርት	VL	L	M	H	VH
A.	**Core Competency/የኮርሱ ብቃት**					
1.	Explains course overall objectives/የትምህርት አጠቃላይ ዓላማ በግልጽ ለተማሪዎች ያሳውቃል/ታሳውቃለች፣	1	2	3	(4)	5
2.	Prepares course outline on time and explains the contents of the course outline/የትምህርቱን ዝርዝር ይዘት/ኮርስ አውትላይን/ በግልፅ ያሳውቃል በጊዜ ሰሌዳ መጥኖ ያዘጋጃል፣ያቀርባል፣ለተማሪዉ አባዝቶ ያቀርባል/ታቀርባለች፣	1	2	3	(4)	5
3.	Explains detailed objectives of each chapter and sessions/የትምህርቱን ዝርዝር ዓላማ በየጊዜው ለተማሪዎች ግልጽ ያደርጋል/ለምሳሌ በየምዕራፍ በየርዕሱ.....ወዘተ፣	1	2	(3)	4	5
4.	Teaches day one class one /በመጀመሪያ ቀን የመጀመሪያ ክላስ ያስተምራል/ታስተምራለች፣	1	2	(3)	(4)	5
5.	Prepares well for course delivery/ስለሚያስተምረው ትምህርት ተገቢ ዝግጅት አድርጎ ያቀርባል/ታቀርባለች፣	1	2	3	4	5
6.	Teaches as per the course content/በትምህርቱ ዝርዝር ይዘት መሠረት ትምህርቱን ያቀርባል/ታቀርባለች፣	1	2	(3)	(4)	5
7.	Gives course reading materials and lecture notes/ለሚያስተምረው ትምህርት አስፈላጊ ጽሁፎችን አዘጋጅቶ ያቀርባል/ታቀርባለች፣	1	2	3	4	(5)
8.	Notify list of references and textbooks available in the library/በቤተመፅሐፍት ዉስጥ ለሚያስተምረው ትምህርት ጠቃሚ የሆኑትን ማጣቀሻ መፃህፍት ዝርዝር (Reference) አዘጋጅቶ ያካትታል/ታካትላለች፣	1	2	(3)	4	5
9.	Depending on course nature teaches practical sessions/እንደ አስፈላጊነቱ የተግባር ልምምድ ያካትላል/ታካትታለች (ዎርክሸፕ፣ ላብራቶሪ፣ ኬዝ ስተዲ፣ ወ.ዘ.ተ)	1	2	(3)	4	5
B.	**Professional Competency/የማስተማር ብቃት**					
10.	Delivers the course in a such a way that students understand/በማስተማሪያ ቋንቋ በመጠቀም ትምህርቱን በግልፅ ያቀርባል/ታቀርባለች፣	1	2	3	(4)	5
11.	Use of additional teaching aids/የተለያዩ የትምህርት መርጃ መሣሪያዎችን በመጠቀም ያስተምራል/ታስተምራለች	1	2	(3)	4	5
12.	Entertains questions raised in the class room/በክፍል ውስጥ የተማሪዎችን ጥያቄ ተቀብሎ በአግባቡ ያስተናግዳል/ታስተናግዳለች፣	1	2	3	(4)	5

Page 1 of 2

.

References

1. Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson Education.
2. Pressman, R. S., & Maxim, B. R. (2015). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill.
3. Wachemo University.Official Website https://wcu.edu.et/
4. PHP Documentation. https://www.php.net/docs.php
5. MySQL Documentation. https://dev.mysql.com/doc/
6. Apache Software Foundation. *Apache HTTP Server Documentation*. https://httpd.apache.org/docs/
7. PHPUnit Documentation. https://phpunit.de/documentation.html
8. Visual Paradigm. *UML Modeling Guide*. https://www.visual-paradigm.com/guide/
9. Figma Documentation. https://help.figma.com/
10. Mozilla Developer Network (MDN). *Web Development Documentation*. https://developer.mozilla.org/
11. GitHub Documentation. https://docs.github.com/
12. Module materials provide from course instructor.