# ACADEMIC ANALYSIS SYSTEM (AAS)

By

**Muhammad Hamza Rafiq**

**Hamayun Mushtaq**

**Muhammad Nawaz**

Submitted to the Faculty of Computer Science, Military College of Signals National University of Sciences and Technology, Islamabad in partial fulfilment for the requirements of a B.E Degree in Computer Software Engineering

JUNE 2018

## CERTIFICATE OF CORRECTNESS AND APPROVAL

*This is to officially state that the thesis work contained in this report*

**"Academic Analysis System (AAS)"**

*is carried out by*

**Muhammad Hamza Rafiq, Hamayun Mushtaq, and Muhammad Nawaz**

*under my supervision and that in my judgement, it is fully ample, in scope and excellence, for the degree of Bachelors of Computer Software Engineering from National University of Sciences and Technology (NUST), Islamabad.*

Approved By:

Signature: ______________________________

Supervisor: **Lec Ayesha Naseer**

MCS, Rawalpindi

# ABSTRACT

As every semester passes in MCS, a whole procedure of manual academic management begins. The students try finding their results in between the phenomenal hype of features that CMS provides, and then move on to estimate what their future plan should be to do better in their degree. The faculty and staff try to manage this heap of data by generating results, but the output is still ‘data’, not ‘information’.

Considering the requirements of MCS CSE Dept, Academic Analysis System (AAS) is a web application that provides its three categories of users: students, faculty and staff; a simplified academic monitoring system. Its extensive scope features tasks like managing academic penalties, graduation criteria lists, past academic records, F-grades, subjects for improvement, future GPA calculation etc. AAS is looked forward to as an app that can work side by side with platforms like CMS, by providing its users with the services which they need daily. With a simple and elegant user interface and constant online availability, AAS is a successful replacement for the daily academic analysis.

## DECLARATION OF ORIGINALITY

We hereby declare that the work contained in this report and the intellectual content of this report are the product of the sole effort of our group, comprising of Muhammad Hamza Rafiq, Hamayun Mushtaq, and Muhammad Nawaz. No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere, nor does it include any verbatim of the published resources which could be treated as a violation of the international copyright decree. We also affirm that we do recognize the terms 'plagiarism' and 'copyright' and that in case of any copyright infringement or plagiarism established in this thesis, we will be held fully accountable of the consequences of any such violation.

*Dedicated to all those*

*who lead us on the journey*

*from ignorance to knowledge.*

# ACKNOWLEDGEMENTS

All of our gratitude is extended to Almighty Allah, the most beneficent, whose blessings empowered us to tread this journey.

We feel indebted to the benign faculty of Department of Computer Software Engineering, who raised us to these standards of knowledge; and especially to Lec Ayesha Naseer, whose unending support, motivation and guidance made us capable of turning the concept of this project into reality.

We also extend our thankfulness to our families and friends, who kept our moral spirits high in the times of need.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# KEY TO SYMBOLS OR ABBREVIATIONS

| | |
|---|---|
| **AAS** | Academic Analysis System |
| **App** | Application |
| **AS** | Assumption |
| **CGPA** | Cumulative Grade Points Average |
| **CMS** | Content Management System http://cms.nust.edu.pk/ |
| **CNIC** | Computerized National Identity Card |
| **CO** | Constraints |
| **CSE Dept** | Computer Software Engineering Department |
| **D** | Dependency |
| **EE Dept** | Electrical Engineering Department |
| **GPA** | Grade Point Average |
| **HTML** | Hypertext Markup Language |
| **HTTP** | Hypertext Transfer Protocol |
| **HTTPS** | Hypertext Transfer Protocol Secure |
| **JS** | JavaScript |
| **LMS** | Learning Management System https://lms.nust.edu.pk |
| **Mac** | Macintosh |
| **MCS** | Military College of Signals |
| **MIT** | Massachusetts Institute of Technology |
| **NUST** | National University of Sciences and Technology |

| | |
|---|---|
| **OE** | Operating Environment |
| **PHP** | Hypertext Preprocessor |
| **REQ** | Functional Requirement |
| **SDS** | Software Design Specification |
| **SE** | Security Requirements |
| **SF** | Safety Requirements |
| **SQL** | Structured Query Language |
| **SRS** | Software Requirements Specification |
| **STP** | Software Test Plan |
| **UD** | User Documentation |
| **UI** | User Interface |
| **URL** | Uniform Resource Locator |

CHAPTER 1

# Introduction

# 1. Introduction

## 1.1 Motivation

Web Application of Academic Analysis System is projected as a platform for aiding the UG students and faculty of MCS CSE Dept to view the evaluation of academic performances. The system will advance the problem–resolution process which is previously being done manually, by automating it. This would prove to befit all the categories of individuals that interact with the Department for academic reasons.

The Dept has dedicated certain individuals that perform these heavy responsibilities to facilitate the students and as a matter of fact, they do a good job at displaying individual statistics, but the information that can be accessed by analyzing and evaluating those results is lost between the pages.

It, therefore, urges the use of modern and systematic technology to develop a system that can analyze the academic data, the Academic Analysis System.

## 1.2 Problem Statement

The problem statement for this project can be broken down into three phases, each of them meeting one category of its users. The main user audience of AAS are the students. CMS is currently providing too many features that are of no good use to the students. Similarly, there is no platform where they can keep an eye on their academic records while managing their past weak grades and penalties, and estimating their way into the GPA Progression.

The second phase encompasses the issues that the faculty, particularly those working as Head of Department or Course Advisors face. They have no platform available where they can keep an eye on their subordinate students by checking their academic data easily. Similarly, the faculty has no idea about the subjects which they should offer for improvement so the largest amount of students can benefit from it. Moreover, the academic penalties are currently initiated and managed manually.

As for the third category of users i.e. staff, they manage all the results and course data either manually or on software like MS Excel. This process is tedious and causes issues because of being operated separately at various platforms.

AAS looks forward to addressing these issues in the form of a web application that clearly meets the needs of its users.

## 1.3 Project Scope

AAS will be a Web-based Application that shall provide a platform for users to view an evaluation of academic performances. The Department Faculty can use the system to see which students should be penalized with probation, suspension or withdrawal. It will offer the users to view which subjects would be most appropriate to be offered for improvement. To automate the entire procedure, which is currently being performed manually, the system is being developed.

### 1.3.1 External Scope

The scope of the project can be increased to be implemented for PG and PhD courses as well. Similarly, it can be used by other departments after slight modifications according to their own requirements.

## 1.4 Project Objectives

### 1.4.1 Academic Objectives

- To understand comprehensively and implement the concepts of Database Management
- To understand the concepts of basic Web development
- To understand the concept of programming on the server side
- To go through the process of professional software development

### 1.4.2 Application/End-Goal Objectives

The Application Objectives are as under:

- The results of every semester will be fed into the system.
- AAS will generate the list of students who will be awarded probation, relegation, suspension or withdrawal.
- Students can view their academic progress, with respect to Credit Hours, along with a list of subjects most suitable for improvement.
- The Department can check which subjects should be offered as Repeat Courses so the maximum number of students can benefit from it.
- It will generate a list of students who fulfil the graduation criteria academically, and according to Credit Hours requirements, at the end of every degree and vice versa.
- A web interface will be provided for the execution of above steps.

- A database will be developed to store the results at the backhand.

## 1.5 Document Organization

The document looks at all the aspects of development and usage of AAS since its initial stage to its completion. Starting with Chapter 1 and 2 covering an introduction and literature review of the topic, Chapter 3 and 4 further discusses the process of requirement elicitation and design development. After completion of these stages, Chapter 5 and 6 of the document covers details regarding implementation and testing of the system.

## 1.6 Intended Audience and Reading Suggestions

The document is meant for the following stakeholders.

- **Project Supervisor:** to assist in project supervision and guiding the team in a better way.
- **Development Team:** to help in the development of product and trace-back of functional requirements.
- **Testing Team:** to help the testers to understand the applicable constraints.
- **Users:** The potential stakeholders of the system, including the UG Students and faculty of MCS CSE Dept.
- **UG Project Evaluation Team:** to assist the evaluation committee in evaluating the progress of UG Projects.
- **Staff:** to understand the process of data entry into the system's database.

## 1.7 Summary

This chapter gives a brief overview of the system and its functionalities, as well as this document, which covers all of the essential details regarding AAS. This chapter also covers details regarding the scope and objectives of the system.

CHAPTER 2

# Literature Review

# 2. Literature Review

## 2.1 Introduction

At the completion of every semester, a whole process of academic assessment on individual, course, departmental and institutional level is performed. This involves analysis of the results, their categorization and implementation. Keeping in view the requirements of MCS CSE Dept and the NUST Academic Statutes, AAS will automate the entire academic evaluation procedure for UG courses. Every semester, the results will be fed into a database. These will be analyzed and the system will develop detailed lists of students for probations, improvements etc. These lists could be accessed by the faculty and administration as well as the students. The project will be implemented in the form of a Web Application.

## 2.2 Product Perspective

Currently, when a student wants to access his academic results, the individual can log in to CMS, but it only allows the user to see subject grades, semester GPAs and CGPA. There is no way, except manual calculations, to view which subjects are most feasible for improvement. Similarly, after every semester, the clerical staff of Department has to generate lists for academic penalties after going through loads of paperwork. Moreover, when Department has to offer a course for improvement, they have no method to evaluate the list of subjects that befit the students the most.

This is where the AAS comes in to play. It aims to eliminate the need for the manual analysis of these academic results. The semantics of the project will make it easier to keep an eye on the academic progression of the students. Furthermore, it is a new, standalone product.

It follows the Client-Server architecture technique as mentioned in the diagram below:

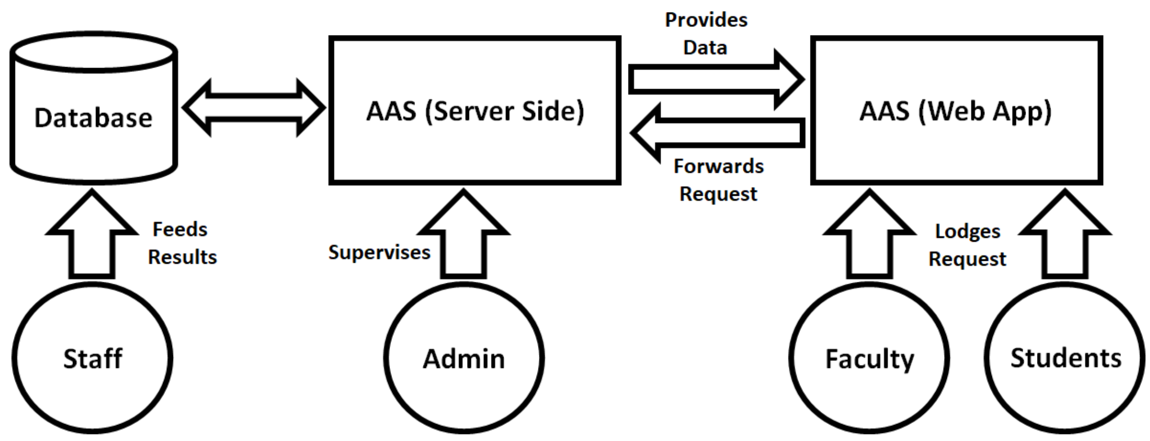


*Figure 2-1 - Client-Server Architecture*

## 2.3 Product Features

Following are the major functions of the Academic Analysis System (AAS):

- Every user will be registered with AAS, and get classified as Faculty, Student or Staff.
- The results and subjects will be fed into the system database by Staff, and it will be updated before and after every semester.
- The system will generate a list of students that fall below certain criteria for penalties like relegation, probation, withdrawal etc.
- The system will display the current academic progress to students, along with the list of subjects with the most margin of improvement in grade, if taken as repeat course.

- Lists will be generated showing Dept Faculty the subjects which should be offered in Summer Semester as repeat course, such that largest number of students get the benefit.
- When a course completes their 4 years of study, a list will be generated displaying names of students who have fulfilled the graduation criteria.
- AAS will allow the registered users to log in. Each user will have different features visible, based on the categories: Staff, Faculty and Students.

## 2.4 User Classes and Characteristics

The software has three types of users: Faculty, Students and Staff. All the three types of users have different access level to the system and its data and can perform functions assigned to their respective roles.

### 2.4.1 Faculty

The faculty will use the web-based interface of the AAS in order to view and approve the Academic Penalties. They can also use it to check for students fulfilling graduation criteria and to see which subjects should be offered as repeat courses.

### 2.4.2 Staff

The Staff can update the subjects and results of students. They can also create new data entities for new students and remove the access right of students who have graduated.

### 2.4.3 Students

This will form the majority of the users of the system. They can view their personal info, GPA Progression, academic penalties they have received including F Grades (and list of subjects they can’t study unless they clear those subjects). They can also check which subjects will be most beneficial for them if they take them as repeat course.

## 2.5 Operating Environment

### 2.5.1 OE-1

AAS back-end utility i.e. the online server that can be bought and database can be maintained there. All the data will be accessed on the server and data manipulation can be done on the same server.

### 2.5.2 OE-2

The Web-based system of AAS shall run on the computer system with following specifications:

- Pentium 4 or Higher CPU
- At least 512MB of RAM
- At least 1 GB of free disk space
- Windows 7 or later operating system
- Chrome, Firefox or Internet Explorer
- Colour monitor and a working internet connection.

### 2.5.3 OE-3

AAS should be managed with MySQL database management system.

### 2.5.4 OE-4

AAS will be able to run on any smartphone with a working internet connection using its Browser Application.

## 2.6 Design and Implementation Constraints

CO-1: Web compatible platform, PHP based, is needed for the end-user.

CO-2: All HTML code should conform to the HTML 5 standard.

CO-3: Lack of user-expertise in using the applications on the internet.

CO-4: Internet connection needed.

CO-5: Use of English language as the only means of communication in the system.

CO-6: System must adhere to NUST Academic Regulations.

CO-7: MCS CSE Dept will be responsible for maintaining the delivered software.

## 2.7 User Documentation

UD-1: Final release will be accompanied with an online user guide to inform users how to use Academic Analysis System (AAS). User documentation that would be delivered along with the final product

- Online User Manual
- SRS Document
- SDS Document
- Test Plan Document

- Final Report

## 2.8 Assumptions and Dependencies

AS-1: Basic assumption for development of AAS is that system should be available 24x7 since a user can access the academic data at any time.

AS-2: The faculty and staff will be honest and not alter the academic records.

AS-3: The faculty and staff are willing to take the time to keep the database and records updated.

AS-4: The server will be able to handle a large number of requests especially at times likes the announcement of results.

AS-5: Users of Academic Analysis System (AAS) should be assumed to have access to a computer with internet access.

D-1: There will be a permanent dependency on the internet, as without this, the requests can’t be processed.

D-2: System is dependent on a server for full access 24x7 as the database will reside on that server.

CHAPTER 3

# Requirements

# 3. Requirements

## 3.1 Introduction

The purpose of this chapter is to present a detailed description of the Academic Analysis System. It will explain the purpose, features, interfaces, functionality, entire process, constraints and the application's reaction to external stimuli. It is intended for stakeholders and the system developers.

## 3.2 External Interface Requirements

### 3.2.1 User Interfaces

AAS consists of three major interfaces, including web-based interfaces that will be used by the faculty, students or staff. The learning curve for these interfaces will be gradual, so as to make the users of the app feel at ease while learning about the options available to them.

### 3.2.2 Hardware Interfaces

#### 3.2.2.1 Computer System

The system shall have:

- Keyboard input.
- Mouse input.
- A monitor.
- A working internet connection and the hardware requirements that come with it (Network card, Ethernet Port, Modem etc.)

#### 3.2.2.2 Web and Database Server

- To process requests and retrieve/store data.

### 3.2.3 Software Interfaces

- AAS should be able to run on any version of the following Web browsers: Google Chrome, Mozilla Firefox, Internet Explorer, Safari, Microsoft Edge and Opera.
- Primary Operating System supported by AAS Interface will be Windows 7.
- AAS should be able to run on Web server configured in a stable Linux/Unix/MAC/Windows machine.
- AAS should be work with MySQL database management system.

### 3.2.4 Communications Interfaces

- The system shall be connected to the web services that we will create.
- Communication between the Web Interface and the server will be through HTTP over a web browser.

## 3.3 Functional Requirements

### 3.3.1 Account creation

#### 3.3.1.1 Description and Priority

The system will enable the staff to create the user accounts for all members of CSE Dept while providing them with the access rights based on their hierarchy in the system. The priority of this system feature is high.

#### 3.3.1.2 Stimulus/Response Sequences

Input: The staff will provide credentials to create a user account.

Output: The account with specific rights to each person will be created.

#### 3.3.1.3 Functional Requirements

REQ-1: The AAS shall allow the staff to create and manage accounts for users.

REQ-2: The AAS shall enable the staff to remove access rights of students who've graduated.

### 3.3.2 Login / access rights

#### 3.3.2.1 Description and Priority

The system will enable the users to log in. The users will provide their unique ID and password to access their account. Faculty and Staff will be able to manage particular things based on their hierarchy and give rights accordingly.

#### 3.3.2.2 Stimulus/Response Sequences

Input: User will enter login credentials to the system.

Output: The system will grant the valid user, the access to the features according to his access rights.

#### 3.3.2.3 Functional Requirements

REQ-3: The AAS shall allow all the users to log in to their account.

REQ-4: The AAS shall grant the logged in user, the access to the specific system features.

### 3.3.3 Update Subjects and Results

#### 3.3.3.1 Description and Priority

The application will allow the staff to update the subjects and results before and after every regular semester.

#### 3.3.3.2 Stimulus/Response Sequences

Input: The staff will enter the data into the database.

Output: The records for students will be updated in the database.

#### 3.3.3.3 Functional Requirements

REQ-5: The AAS will allow the Staff to enter the subject and grades data in the database.

REQ-6: The AAS will allow Staff to update the subject and grades data in the database.

### 3.3.4 Academic Records

#### 3.3.4.1 Description and Priority

The system will allow the students to view their academic records (including previous GPAs and CGPA).

#### 3.3.4.2 Stimulus/Response Sequences

Input: The student will log in the AAS App.

Output: The system will access the records from the database and display them to students.

#### 3.3.4.3 Functional Requirements

REQ-7: The AAS will display the academic records to the students including Current Semester, CGPA and Credit Hours Completed etc.

### 3.3.5 Academic Penalties

#### 3.3.5.1 Description and Priority

The system will get academic penalties approved from certain members of the faculty and display them to students and faculty.

#### 3.3.5.2 Stimulus/Response Sequence

Input: Faculty and students will request to display the list of academic penalties.

Output: The list of penalties will be displayed for either approval or viewing, depending upon access rights.

#### 3.3.5.3 Functional Requirements

REQ-8: The system will allow specific members of the faculty to approve the academic penalties of students.

REQ-9: The Faculty can view the list of academic penalties awarded to students.

REQ-10: The AAS will let the students view the academic penalties they have been awarded.

REQ-11: The system will display the subjects which a student can't study if they have scored an F grade in a pre-requisite subject.

### 3.3.6 GPA Progression

#### 3.3.6.1 Description and Priority

The entire GPA progression of students will be displayed to them in form of double line graph.

#### 3.3.6.2 Stimulus/Response Sequences

Input: The students will log in to their profile.

Output: The graph will be displayed on their profile.

#### 3.3.6.3 Functional Requirements

REQ-12: Students can view their GPA and CGPA progression in a graphical form.

### 3.3.7 Subjects for Improvement

#### 3.3.7.1 Description and Priority

After performing data analysis of students' records, list of subjects that prove to be most beneficial will be displayed to students. This will be done on an individual basis for students and departmental level for faculty.

#### 3.3.7.2 Stimulus/Response Sequences

Input: User will request to view the list of subjects suitable for improvement.

Output: The system will display the subjects with the most margin of improvement for students, and most number of students benefitting, to faculty.

#### 3.3.7.3 Functional Requirements

REQ-13: AAS will display the list of subjects, with the greatest number of students benefitting from them, to faculty.

REQ-14: AAS will display the list of top five subjects, with the greatest margin of improvement, to students.

REQ-15: The system will then display the projected CGPA to students when they select a specific subject for improvement.

## 3.3.8 Graduation Criteria Lists

### 3.3.8.1 Description and Priority

The system will generate graduation criteria lists when a course completes four years of study. This data will be forwarded to the faculty and the students.

### 3.3.8.2 Stimulus/Response Sequences

Input: The staff will mark the completion of the 8th semester.

Output: The System will generate a list of students who haven't fulfilled the graduation criteria.

### 3.3.8.3 Functional Requirements

REQ-16: The AAS will allow Faculty to view the list of students who couldn't fulfil the graduation criteria.

REQ-17: The AAS will allow Students to view if they have fulfilled the graduation criteria.

### 3.3.9 Sign out

#### 3.3.9.1 Description and Priority

The system will generate graduation criteria lists when a course completes four years of study. This data will be forwarded to the faculty and the students.

#### 3.3.9.2 Stimulus/Response Sequences

| | |
|---|---|
| Input: | The user will click the sign out button. |
| Output: | The system will log them out and update their login status in the database. |

#### 3.3.9.3 Functional Requirements

| | |
|---|---|
| REQ-18: | The AAS will allow the users to sign out from the application. |

## 3.4 Other Nonfunctional Requirements

### 3.4.1 Performance Requirements

Certain functionalities will be required, based on the performance and response of AAS. AAS will take less than 15 seconds to send data to the server. Up to 50 users can lodge the requests simultaneously.

### 3.4.2 Safety Requirements

SF-1: In case of data loss, the system will back up the data and will restore it as per demand.

SF-2: System will be deployed on online cloud server platform so they've their inherent fault-tolerance capabilities.

SF-3: Students can lodge a manual complaint to Dept in case of wrong data.

SF-4: Identity of Staff managing the Database should not be disclosed to the Students.

### 3.4.3 Security Requirements

SE-1: Users shall be required to log in to the AAS for their own credential information.

SE-2: The system shall permit only authorized faculty members or staffs to do administrator's task.

SE-3: The system shall permit users to view only their own profile and data that are intended for them.

SE-4: The system must perform an encoding technique such as hashing to save all passwords securely.

SE-5: The System will provide confidentiality and integrity.

### 3.4.4 Software Quality Attributes

Quality attributes of AAS are described below. By following these attributes, the quality of AAS will be improved.

#### 3.4.4.1 Runtime System Qualities

At runtime, AAS has to provide its users with functionalities so that they can publish and search for the desired services. Some of the runtime qualities that should be considered in the development of AAS are described here.

##### *3.4.4.1.1 Functionality*

AAS must provide functions to publish and search the different services. AAS must provide the functions of authentication of a user.

###### *3.4.4.1.2 Availability*

AAS should be available 24/7 since the complaint can be lodged at any time. If at all, the system is down so the servers will take about 15 minutes to start the AAS again.

###### *3.4.4.1.3 Usability*

Usability is an important criterion in the development of AAS. The system should present all functionalities in such a way that nothing is missed by the user. The graphical user interface of the app is to be designed with usability as the first priority. The app will be presented and organized in a manner that is both visually appealing and easy for the user to navigate.

#### 3.4.4.2 Non-Runtime System Qualities

These are qualities of AAS which are required to make this software useful for further enhancements. It will also be helpful for future development as well as extending the system to different environments.

##### *3.4.4.2.1 Modifiability*

AAS must support modifiability so any further improvements or features are easy to incorporate.

##### *3.4.4.2.2 Portability*

AAS should be able to run in different computer environments. The AAS server should be a platform-independent and should support interoperability.

##### *3.4.4.2.3 Testability*

Different quality tests should be performed so that AAS is free from faults and perform according to requirements.

# CHAPTER 4

# Design

# 4. Design

## 4.1 Introduction

This chapter covers all the functional requirements and demonstrates how they interrelate with each other abstractly. The low-level design also illustrates as to how all of these requirements have been implemented. This low-level design does not address any non-functional requirements that the system has and that has been mentioned in the SRS Document.

## 4.2 Overview of the Modules

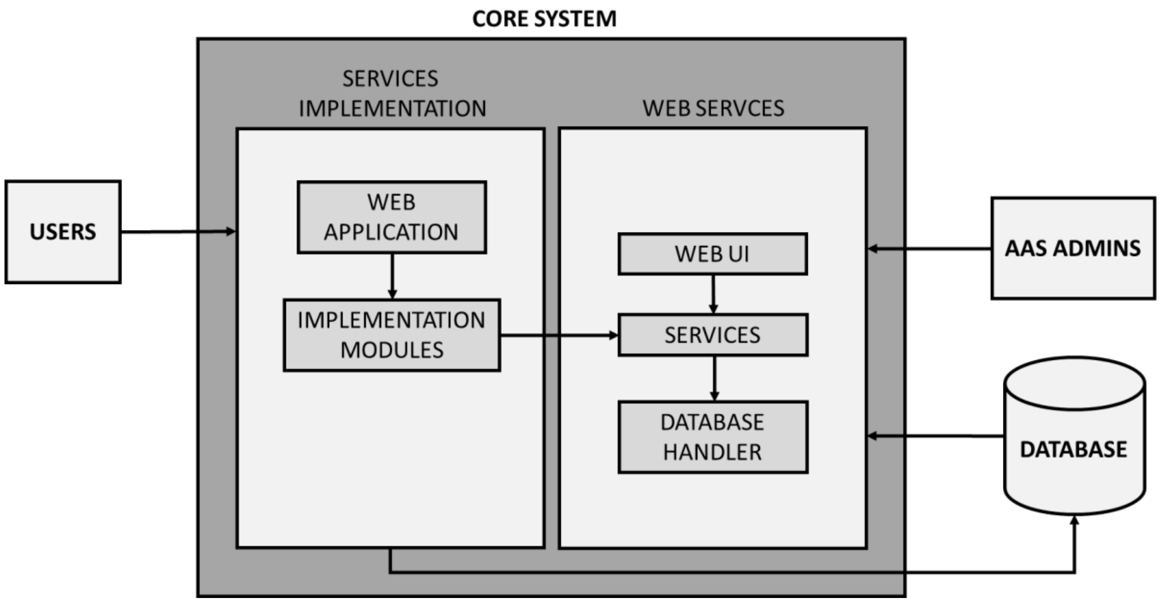


*Figure 4-1 - Abstract Diagram*

### 4.2.1 Explanation of Abstract Diagram

The system will be architected mainly in four fundamental modules "Users", "Core system", "AAS Administration", and the "Database". It will further be having sub-modules as shown in the Figure 4-1. The abstract diagram provides an overview of the system, from users accessing the system until the processing of databases. The sub-modules of the Abstract diagram are further elaborated below.

#### 4.2.1.1 Users

Users of the AAS will access the web app and then choose functionality according to the requirement. Users consist of students, faculty or any staff. The user interacts with the app that further accesses the services provided by the app.

#### 4.2.1.2 Web Application

The web app is the platform to access the system and having different functionalities like viewing GPA Progression, viewing and approving academic penalties, changing user status etc.

#### 4.2.1.3 Application Modules

It consists of the modules that will be responsible for all the processes including GPA Progression and academic penalties for the users.

#### 4.2.1.4 Services

These are the functionalities provided by the AAS. The services for different kind of users are different. As a student, the services provided by the AAS are viewing GPA Progression, improvements and academic penalties. But more privileges are for system administrators, concerned staff and faculty.

#### 4.2.1.5 Web User Interface

The web interface provides services to the students and faculty. It provides a platform to handle the students' requests and deal with backend processing related to user management, academic evaluation etc. Web user interface is dependent upon the services that are further handled by database handler which in turn interacts with the actual database where all the data about users, courses is stored.

#### 4.2.1.6 Administration

System Administration interacts with the web user interface for managing user accounts, dealing with the whole functionality of the system by handling all the provided services by AAS.

#### 4.2.1.7 Database Handler

Database handler provides a connection between the services (that are displayed on the web interface) and the databases where all the data of courses regarding provided services is stored. Database hander basically handles inputs and outputs of some action that needs database access. It’s a gateway to the actual databases.

#### 4.2.1.8 Databases

A database stores all the data about the web application, students’ records, academic penalties and all the related processing happens there.

## 4.3 Structure and Relationships

Layered Architecture **3-Tiers** will be used to implement AAS. From a high-level perspective, a service-based solution can be seen to implement the Web Interface. This will be composed of multiple services, each communicating with the others by passing messages. Conceptually, the services can be seen as components of the overall solution. However, internally, each service is made up of software components, just like any other application, and these components can be logically grouped into the presentation, business, and data layers. Other applications can make use of the services without being aware of the way they are implemented.

### 4.3.1 Layers Details (Web Interface)

#### 4.3.1.1 Presentation Layer

It provides a platform for the interaction of the user with the system. It displays data to the user and accepts input from the user. This is the part which receives the HTTP request and returns the HTML response. The Presentation layer can only receive requests from, and return responses to, an outside agent. This is usually a person, but maybe another piece of software.

It can only send requests to, and receive responses from, the Business layer. It cannot have direct access to either the database or the Data Access layer.

#### 4.3.1.2 Business Logic (Web Service)

When an application must provide services to other applications, as well as implementing features to support clients directly, a common approach is to use a service layer that exposes the business functionality of the application. The services layer effectively provides an alternative view that allows clients to use a different channel to access the application.

#### 4.3.1.3 Data Access Layer

This layer receives a request from the Service Layer and sends back data after querying it from the database server.

## 4.3.2 Architecture Diagram

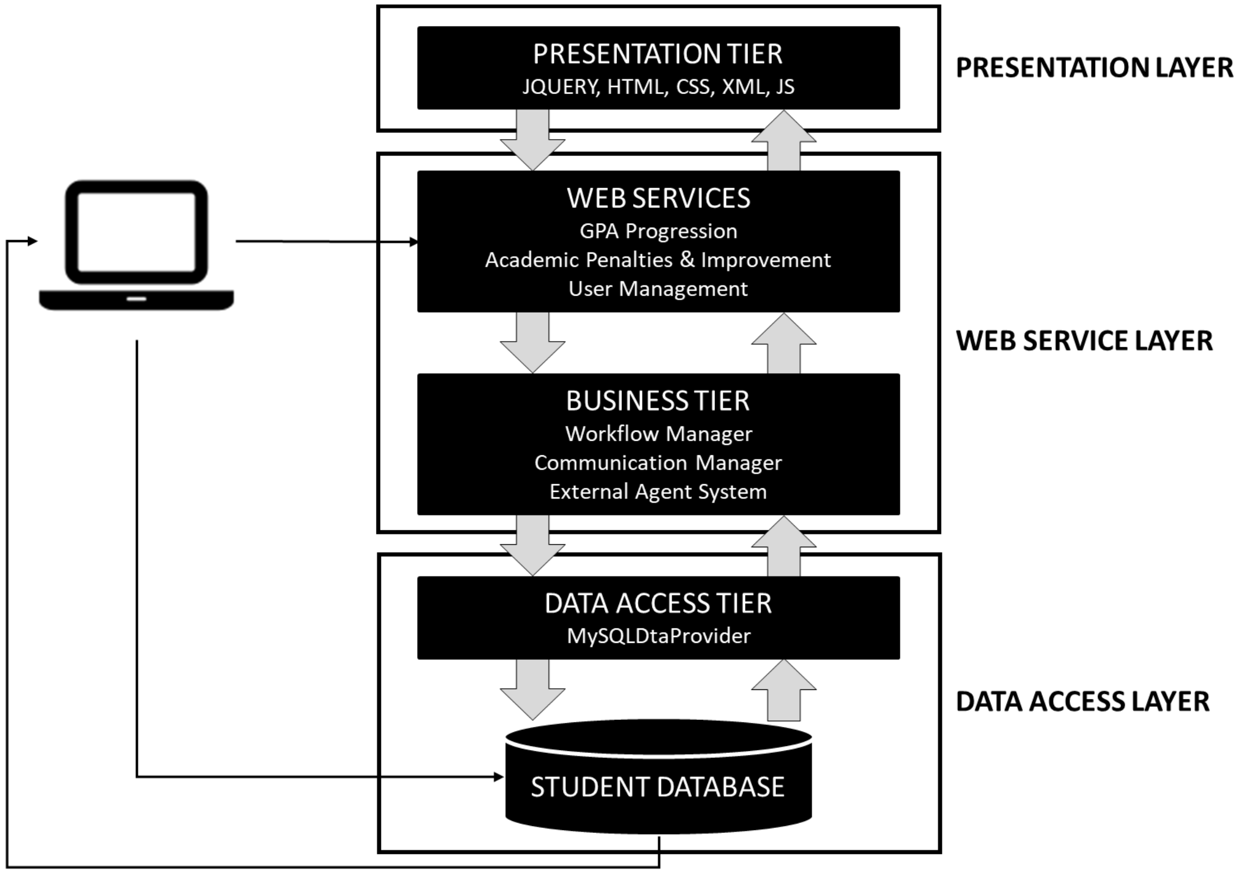


*Figure 4-2 - Architecture Diagram*

Once a user logs in the system, he is granted access according to the category (students, faculty etc.). Every category of users is allowed access to specific features of the system. These features access their data from a central database. This database will be managed by a staff, who will be supervising the entry of all academic results and students.

Following is a detail of the tiers and their components.

### 4.3.2.1 GPA Progression Service

The students' data is accessed from the database and CGPA and projected GPA is displayed to the students. This service is available for students only.

#### 4.3.2.2 Academic Penalties and Improvements

The students are displayed their academic penalties and the subjects which will benefit them the most in improvement. A list of all academic penalties being awarded, as well the improvement subjects that can be offered, is displayed to the faculty.

#### 4.3.2.3 User Management Service

User Management Service is a web service which provides functionality for system login/access so that only authorized personnel of the AAS can access the interface.

#### 4.3.2.4 Workflow Manager

This component contains classes which generate a workflow for the project. The workflow tasks are then assigned resources and forwarded to respective individuals.

#### 4.3.2.5 Communication Manager

This component contains classes which handle communication between different categories of individuals which are dealing with the same project. It provides functions to send notifications to the students.

#### 4.3.2.6 External Agency System

This component contains classes which handle project related information. This package's classes are accessed by web methods defined in User Management Service.

#### 4.3.2.7 Database

This is the database of the whole system. All the user's and academic data is maintained in this database. It has been designed by keeping data integrity and confidentiality principles in mind. Also, database normalization principles are applied during database design.

### 4.3.3 Use Cases

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

The various user classes identified the following use cases and primary actors for the AAS:

| **Actors** | **Use Cases** |
|---|---|
| Staff | • Login<br>• Manage Accounts<br>• Manage Subjects and Grades<br>• Logout |
| Students | • Login<br>• View Academic Records<br>• View Academic Penalties<br>• View Ineligible Subjects<br>• View Subjects for Improvement<br>• View Projected GPA<br>• View Lists for Graduation<br>• Logout |
| Faculty | • Login<br>• Approve Academic Penalties<br>• View Academic Penalties<br>• View Subjects for Improvement |

| | |
|---|---|
| | • View Lists for Graduation<br>• Logout |

*Table 4-1 - Use Cases*

The aforementioned use cases can be represented in the form of the following Use Case Diagram.

### 4.3.3.1 Use Case Diagram

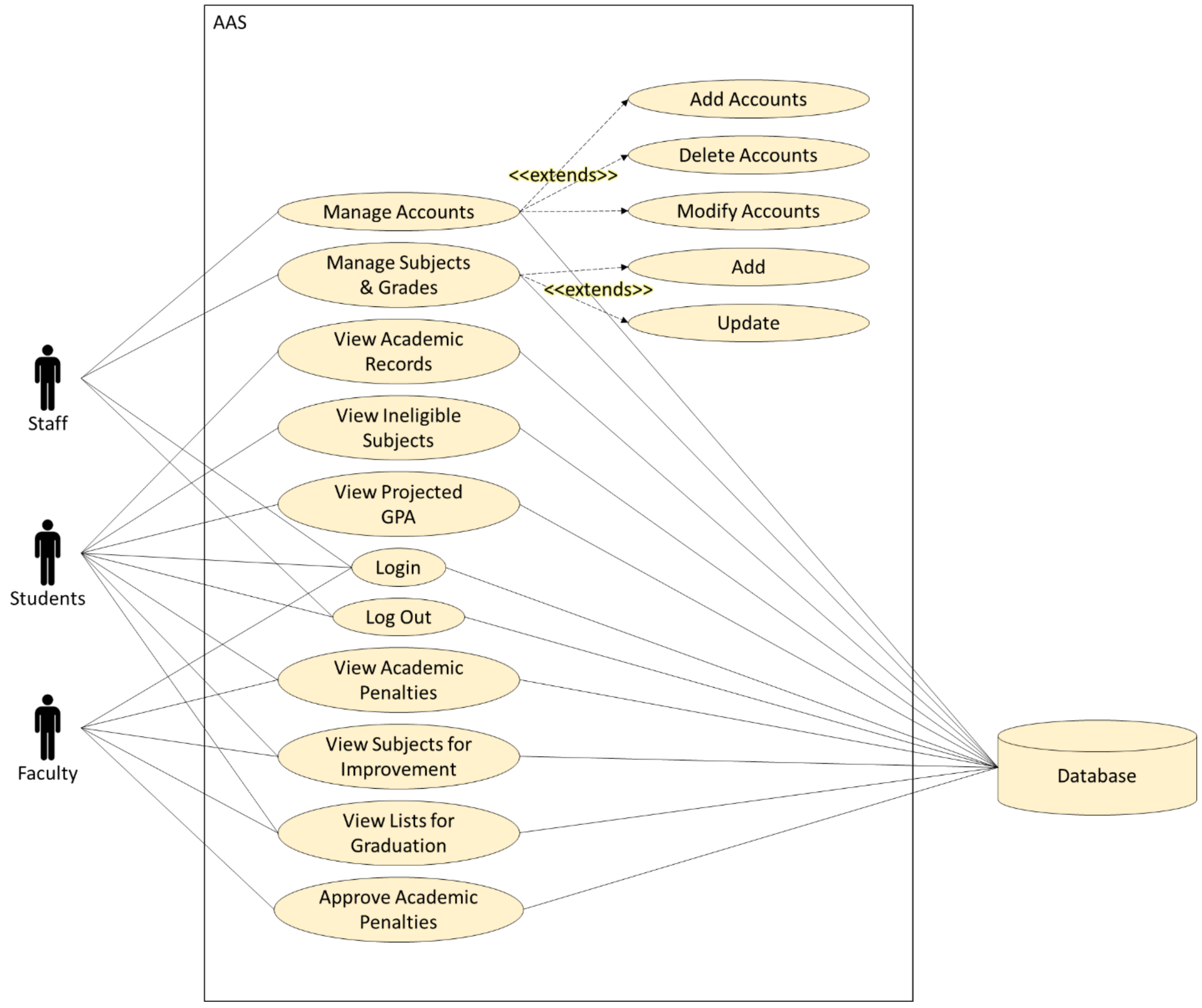


*Figure 4-3 - Use Case Diagram*

#### 4.3.3.2 Use Cases Description

| | | | |
|---|---|---|---|
| Use Case ID: | 1 | | |
| Use Case Name: | Manage Accounts | | |
| Actors: | Staff | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | Staff has to log in to the system and create accounts to provide access to all the students. This will include creating, modifying and disabling accounts. | | |
| Preconditions: | The staff has to log in. | | |
| Post-conditions: | Changes in the account must be updated in the database. | | |
| Normal Flow (Primary Scenario): | 1. The staff will enter, modify or delete the accounts of the students.<br>2. Changes will be updated in the database. | | |
| Alternative Flows: | 1. An error is encountered during the modification of database.<br>2. Proper functionality of the database will be checked. | | |

*Table 4-2 - Manage Accounts Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 2 | | |
| Use Case Name: | Login | | |
| Actors: | Staff, Students, Faculty | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |

| | |
|---|---|
| Description: | A user tries to log in to the system. |
| Preconditions: | The user has to open the login page first. |
| Post-conditions: | If the use case was successful, the actor is now logged into the system. If not, the system state remains unchanged. |
| Normal Flow (Primary Scenario): | 1. The system requests that the actor enter his/her name and password.<br>2. The actor enters his/her name and password.<br>3. The system validates the entered name and password and logs the actor into the system. |
| Alternative Flows: | If in the Basic Flow, the actor enters an invalid name and/or password, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the login, at which point the use case ends. |

*Table 4-3 - Login Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 3 | | |
| Use Case Name: | Manage Subjects and Grades | | |
| Actors: | Staff | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | Staff has to enter or update the subjects and grades for students at the start and end of every semester. | | |

| | |
|---|---|
| Preconditions: | Staff has to log in, and be in the provision of the data regarding subjects and grades. |
| Post-conditions: | Changes in data must be updated in the database. |
| Normal Flow (Primary Scenario): | 1. The staff will enter, modify or delete the subjects and grades of the students at start and end of every semester.<br>2. Changes will be updated in the database. |
| Alternative Flows: | 1. An error is encountered during the modification of database.<br>2. Proper functionality of the database will be checked. |

*Table 4-4 - Manage Subjects and Grades Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 4 | | |
| Use Case Name: | View Academic Records | | |
| Actors: | Students | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | When students log in to their profile, they will be provided access to their academic records. This will include name, course, semester, year of study, CGPA, credit hours studied, and GPA Progression. | | |
| Preconditions: | Student has to log in. | | |
| Post-conditions: | Data will be displayed on student's profile. | | |
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile.<br>2. Data will be displayed to students. | | |

| | |
|---|---|
| Alternative Flows: | - |

*Table 4-5 - View Academic Records Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 5 | | |
| Use Case Name: | View Ineligible Subjects | | |
| Actors: | Students | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | If a student has F Grade in a prerequisite subject, a list of subjects will be displayed which the individual cannot study. | | |
| Preconditions: | Student has to log in. | | |
| Post-conditions: | List of ineligible subjects will be displayed on student's profile. | | |
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile.<br>2. Data will be displayed to students. | | |
| Alternative Flows: | If there are no F Grades, the list will be shown as vacant. | | |

*Table 4-6 - View Ineligible Subjects*

| | | | |
|---|---|---|---|
| Use Case ID: | 6 | | |
| Use Case Name: | View Projected GPA | | |
| Actors: | Students | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |

| | |
|---|---|
| Description: | Students can choose any of the five subjects (in which they have scored worst grades) and their projected GPA will be displayed if they improve them to A Grade. |
| Preconditions: | 1. A student has to log in.<br>2. A student must select subjects for improvement to view projected GPA after improvement. |
| Post-conditions: | Projected GPA will be displayed. |
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile and select the subjects he/she wishes to improve.<br>2. Projected GPA will be displayed to students. |
| Alternative Flows: | If a student has A Grade in all subjects, a vacant list of improvement subjects will be displayed, and this option will be disabled. |

*Table 4-7 - View Projected GPA*

| | | | |
|---|---|---|---|
| Use Case ID: | 7 | | |
| Use Case Name: | View Academic Penalties | | |
| Actors: | Students, Faculty | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | Students can view a list of academic penalties awarded to them. Faculty can view whole lists of academic penalties that have been awarded. | | |

| | |
|---|---|
| Preconditions: | 1. An actor has to log in.<br>2. Faculty must request which list of academic penalties they want to view. |
| Post-conditions: | Individual data for students and combined data for faculty will be displayed. |
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile and his academic penalties will be displayed on the profile.<br>2. Faculty will choose which type of academic penalty's list they want to view. It will be displayed on a screen. |
| Alternative Flows: | A vacant list will be generated in case of no penalties. |

*Table 4-8 - View Academic Penalties Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 8 | | |
| Use Case Name: | View Subjects for Improvement | | |
| Actors: | Students, Faculty | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | Students can view the list of top 5 subjects in which they have scored the worst grades. These will be the subjects that will benefit them the most in improvement. A list of subjects will be available for faculty which displays the subjects from which most students can get the benefit. | | |
| Preconditions: | 1. An actor has to log in. | | |

| | |
|---|---|
| | 2. Faculty should specify the option for viewing the list. |
| Post-conditions: | Subjects for improvement will be displayed, along with previous grade and Improvement Margin. |
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile and the subjects will be displayed on their profile.<br>2. The faculty can choose to view the list. It would be displayed on their screen. |
| Alternative Flows: | If a student has A Grade in all subjects, a vacant list of improvement subjects will be displayed. |

*Table 4-9 - View Subjects for Improvement Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 9 | | |
| Use Case Name: | View Lists for Graduation | | |
| Actors: | Students, Faculty | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | Students can view if they have fulfilled the graduation criteria. A list will also be generated for the faculty displaying data for an entire course. | | |
| Preconditions: | 1. An actor has to log in.<br>2. Faculty should specify the option for viewing the list. | | |
| Post-conditions: | Individual's graduation criteria for students and list of entire course for faculty will be displayed. | | |

| | |
|---|---|
| Normal Flow (Primary Scenario): | 1. The student will log in to the profile and the graduation criteria status will be displayed on his/her profile.<br>2. Faculty can click the option for viewing the list. |
| Alternative Flows: | - |

*Table 4-10 - View Lists for Graduation Use Case*

| | | | |
|---|---|---|---|
| Use Case ID: | 10 | | |
| Use Case Name: | Logout | | |
| Actors: | Staff, Students, Faculty | | |
| Created by: | Hamza | Last Updated by: | Hamza |
| Date Created: | 24/01/2018 | Date Last Updated: | 24/01/2018 |
| Description: | A user attempts to log out of the system. | | |
| Preconditions: | User has to be logged in first. | | |
| Post-conditions: | The user will be logged out and sent to the log in page. | | |
| Normal Flow (Primary Scenario): | 1. User clicks the Sign Out button.<br>2. The user is signed out and sent to the Login Screen. | | |
| Alternative Flows: | - | | |

*Table 4-11 - Logout Use Case*

## 4.3.4 Class Diagram with Description

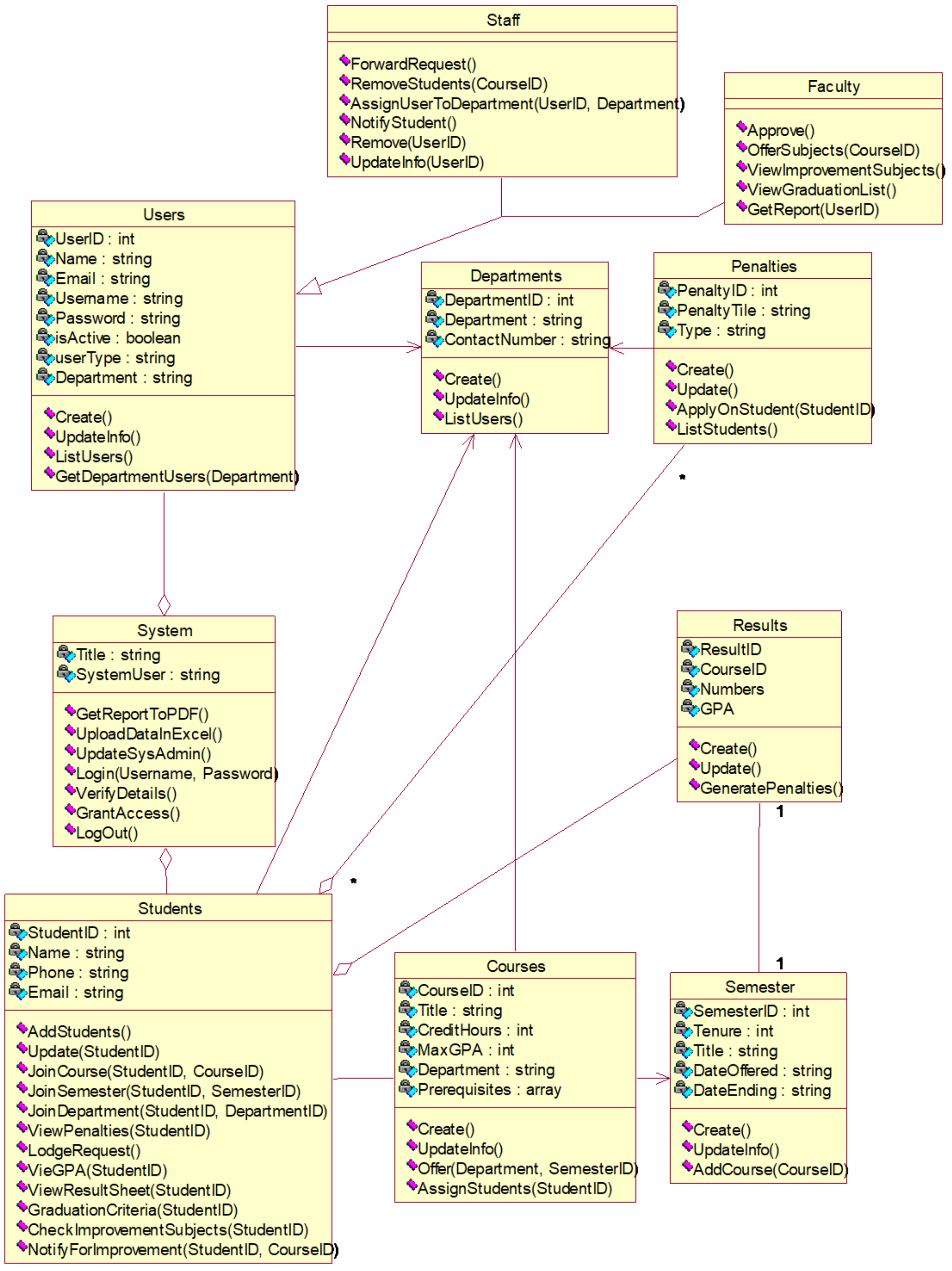


*Figure 4-4 - Class Diagram*

| Class Name | Description |
|---|---|
| System | System class contains the origin for the function AAS to perform. It is the main class acting as a gateway to all the other classes. |
| User | User class contains all the information related to user management. It has aggregation with other classes of user categorization and the functions that perform all the user management functions. Users may include faculty and staff. |
| Student | Student class contains all the information related to students and their results. It has aggregation with other classes of user categorization and the functions that perform all the user management functions. |
| Department | This contains the data for the departments so that users can be affiliated with their specific department |
| Penalties | Penalties Class involves data for the penalties that are applied to the students. It can include relegations, withdrawals, probations etc. |
| Results | Results Class keeps a record of the results of all the students, and help in achieving the overall goal of academic analysis. |
| Semester | Semester Class stores the data about the semesters that are being offered and the students enrolled in them. |
| Courses | Courses Class stores further information about the students. This can enable the staff to deal with the students in the form of groups as courses. |

*Table 4-12 - Class Diagram Description*

## 4.4 User Interface Issues

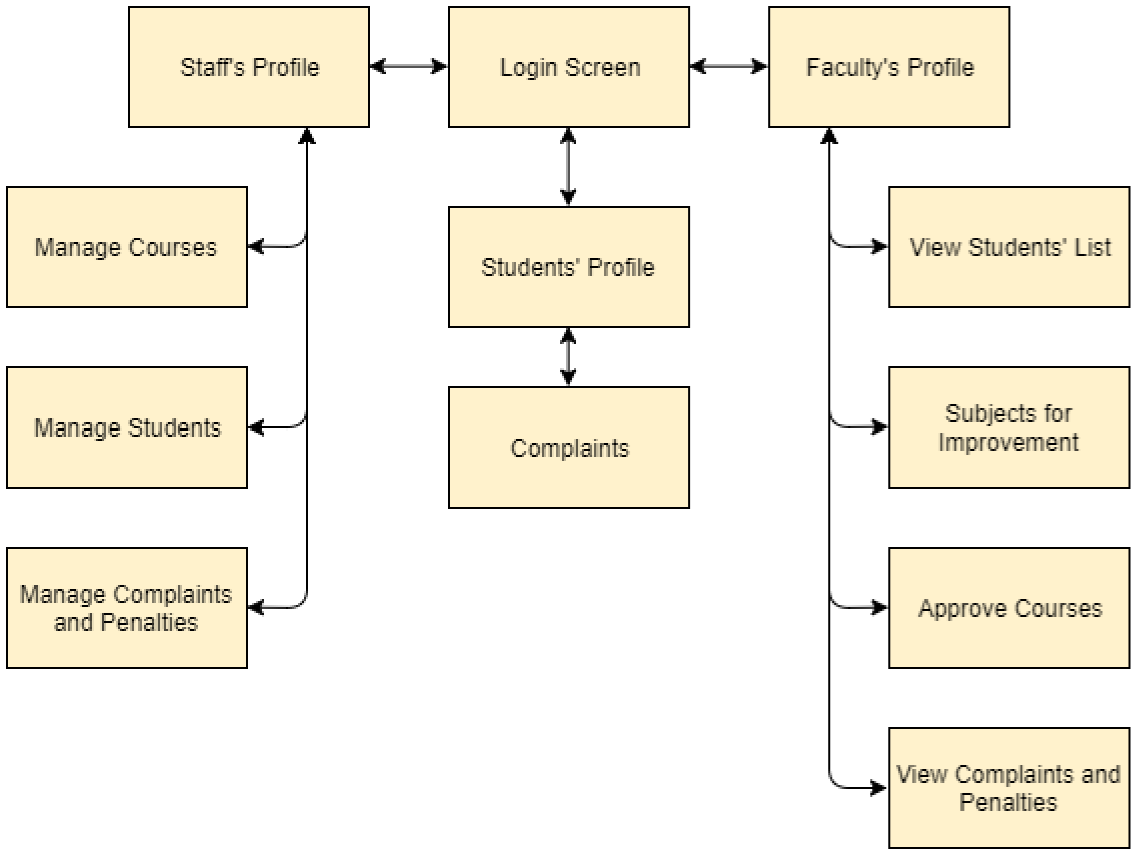


*Figure 4-5 - Interface Diagram*

### 4.4.1 Description of the Diagram

#### 4.4.1.1 Login Screen

This is the initial login screen, which takes username and password as input. After a successful login, the user is sent to the interface specified for his or her category.

#### 4.4.1.2 Staff's Profile

If the logged in user is a staff, the options for managing data about the students, courses, semesters and results are displayed.

#### 4.4.1.3 Manage Courses

It allows the staff to add data for new courses, modify for existing ones, and remove for graduates.

#### 4.4.1.4 Manage Students

It allows the staff to enter data about students, and modify their records in the database.

#### 4.4.1.5 Manage Complaints & Penalties

This interface provides the staff with the opportunity to manage the complaints launched by the students, as well as to initiate the academic penalties that can be further approved by the faculty.

#### 4.4.1.6 Students' Profile

If the logged in user is a student, the students' profile is displayed. This takes data from the database and displays it to students. It includes name, course, semester, year of study, CGPA, graduation criteria, GPA progression, academic penalties, subjects for improvement and credit hours.

#### 4.4.1.7 Complaints

If the student has any complaint, he can initiate it from this interface. This complaint is sent to the staff, who forwards it to the faculty.

#### 4.4.1.8 View Students' List

It allows the faculty to view the list of specific students and their academic data.

#### 4.4.1.9 Subjects for Improvement

Using specific mathematical formulae, the list of subjects that can be offered to students, such that most students can get benefit from them, are displayed.

#### 4.4.1.10 Approve Courses

The interface enables the faculty to approve the courses that are offered to the students.

#### 4.4.1.11 View Complaints & Penalties

The complaints of students and the penalties that are awarded to them are displayed to the faculty, for approval and evaluation purposes.

## 4.4.2 Activity Diagrams

### 4.4.2.1 User Management

The diagram below displays how the users are logged into the system and displayed specific interfaces according to their category.

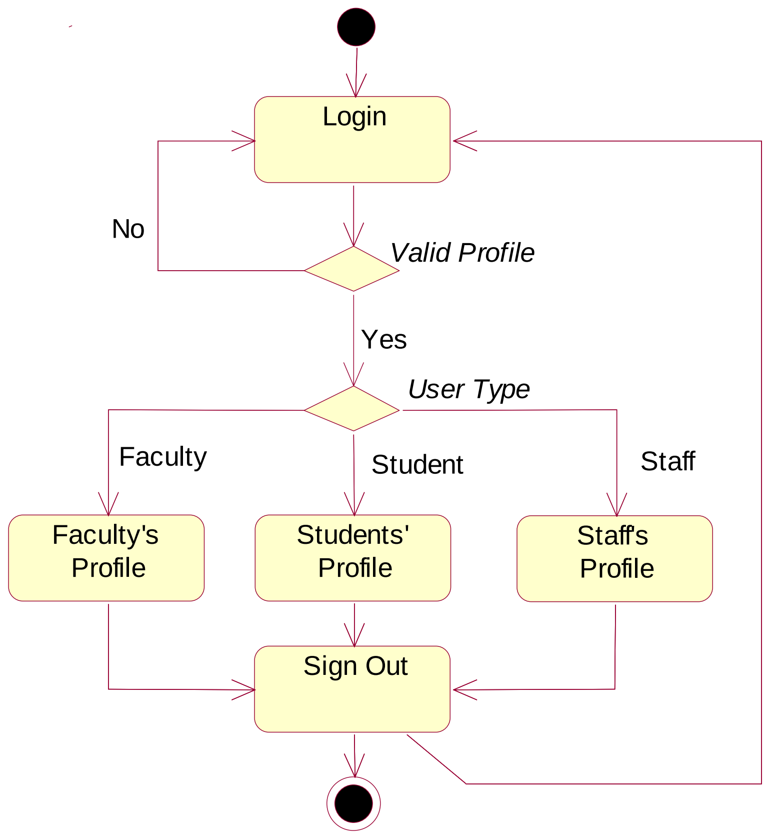


*Figure 4-6 - User Management – Activity Diagram*

#### 4.4.2.2 Faculty's Usage

Figure 5-7 displays the basic features that are available after a faculty member successfully logs into his account. These include management of penalties, improvement subjects, and graduation criteria lists.

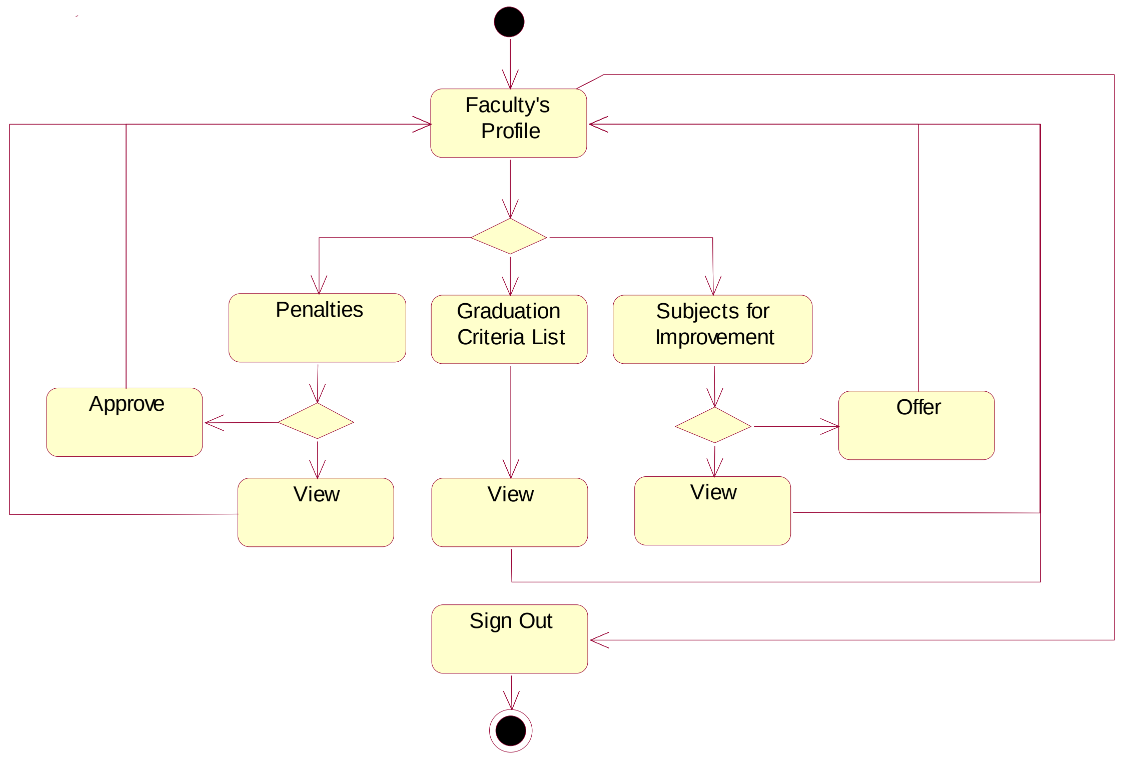


*Figure 4-7 – Faculty's Usage – Activity Diagram*

#### 4.4.2.3 Students' Usage

The diagram below describes the features that are displayed after login to a student. All the basic data is displayed on the main profile page, with a further feature enabling the students to lodge complaints.

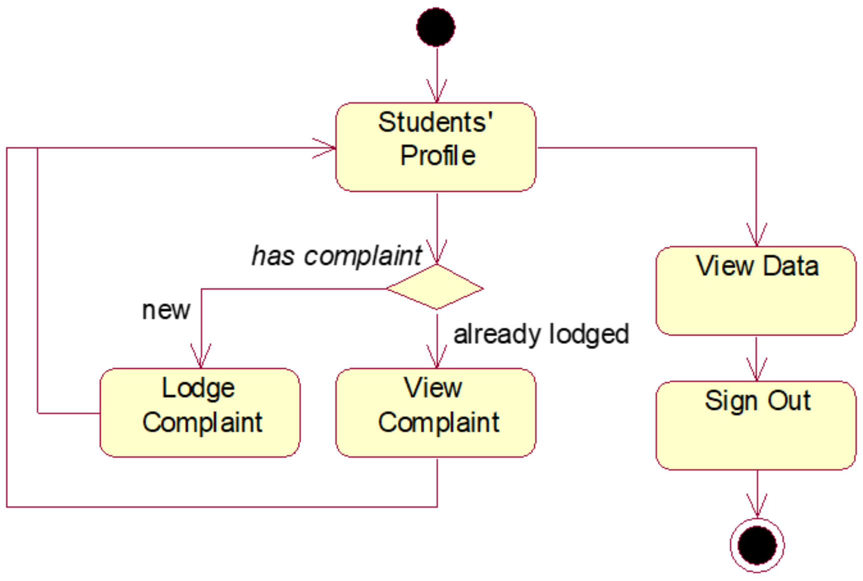


*Figure 4-8 - Students' Usage – Activity Diagram*

#### 4.4.2.4 Staff's Usage

If in case the logged in user belongs to the category of management staff, the following features can be available, which are portrayed in the form of an activity diagram.

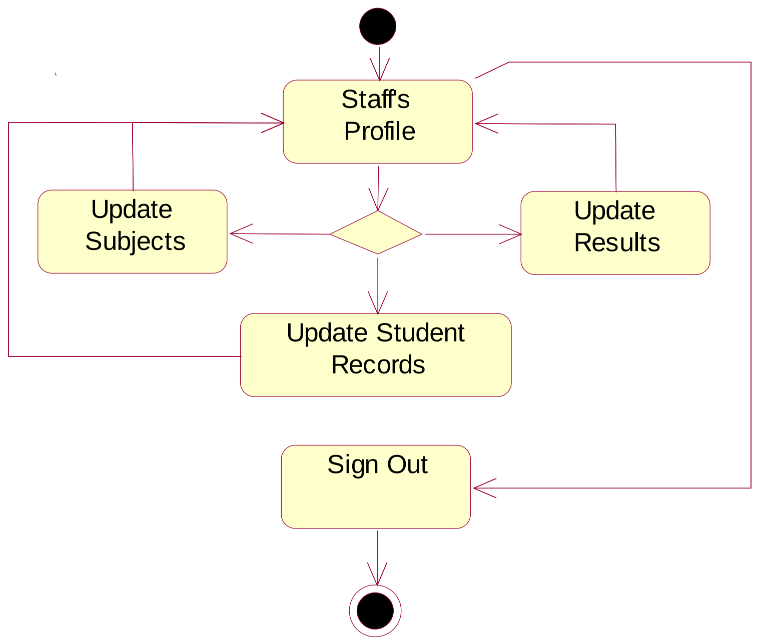


*Figure 4-9 - Staff's Usage – Activity Diagram*

### 4.4.3 Sequence Diagrams

#### 4.4.3.1 View Results

The following diagram shows the sequence of events as a student logs into his or her profile to view the academic performance.

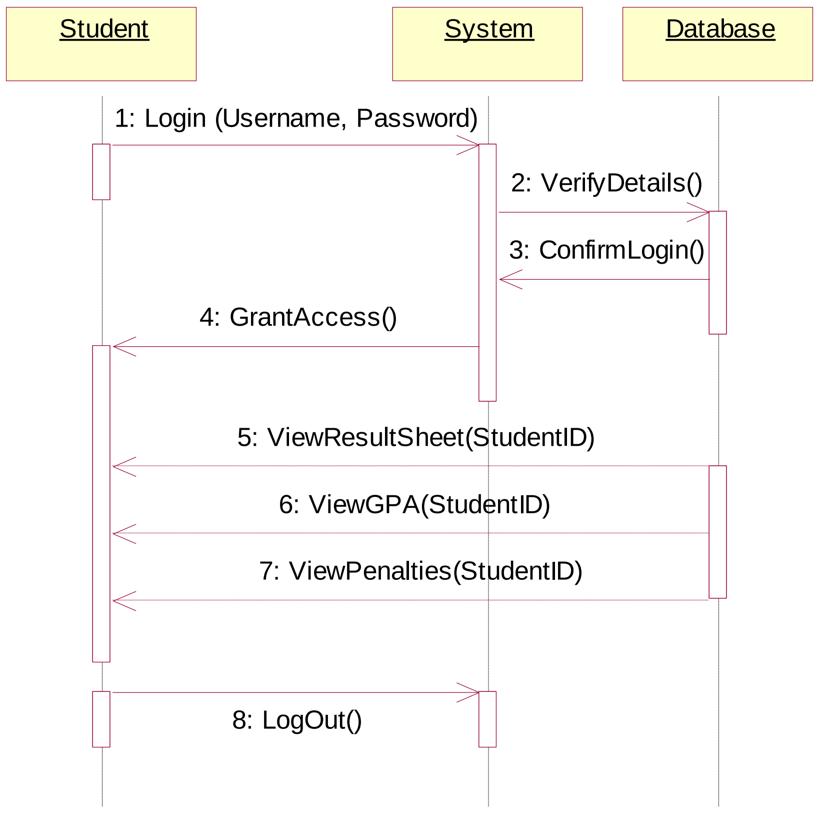


*Figure 4-10 - View Results - Sequence Diagram*

#### 4.4.3.2 Lodge Complaint

This sequence diagram displays the procedure of lodging of complaints and viewing their status, by the students.

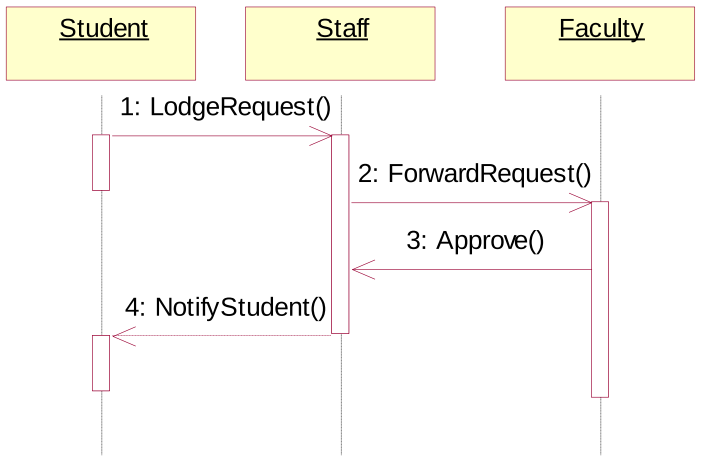


*Figure 4-11 - Lodge Complaint - Sequence Diagram*

### 4.4.3.3 Approve & View Penalties

All the academic penalties are forwarded to the Faculty for approval. These lists can be viewed by faculty before and after approval. After approval, the students are notified.

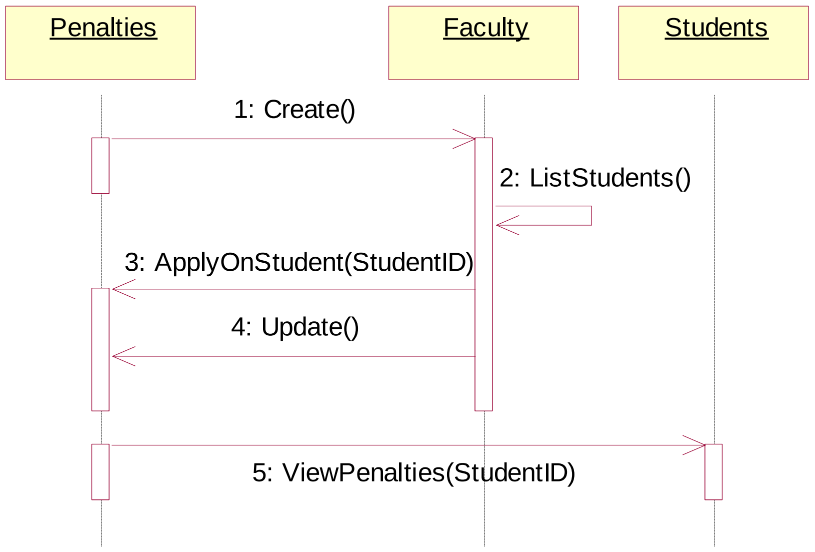


*Figure 4-12 - Approve & View Penalties - Sequence Diagram*

### 4.4.3.4 Graduation Lists

When a course reaches the end of their academic tenure, it needs to be checked if all students have fulfilled the graduation criteria. This is portrayed in the following diagram.

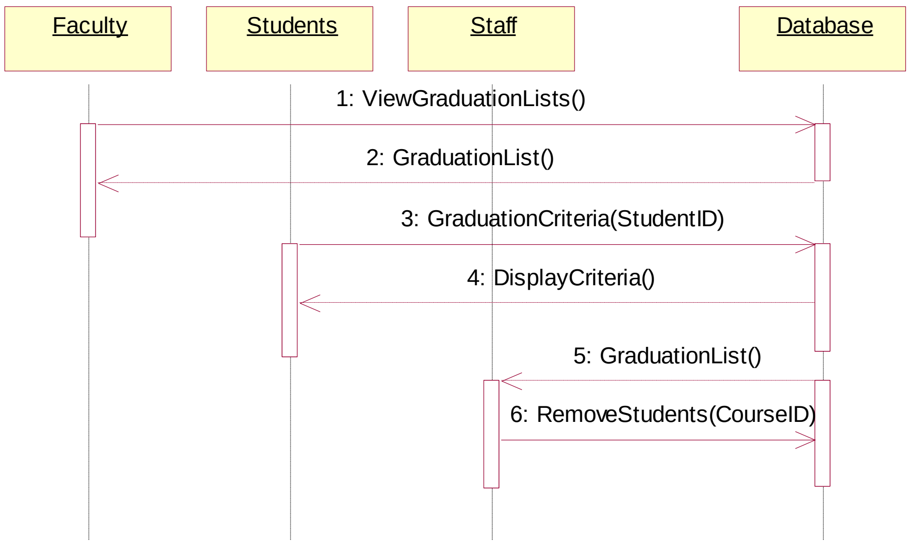


*Figure 4-13 - Graduation Criteria - Sequence Diagram*

#### 4.4.3.5 Improvement Subjects

The following diagram presents the sequence in which a subject offered for improvement is interacted with, by various users.

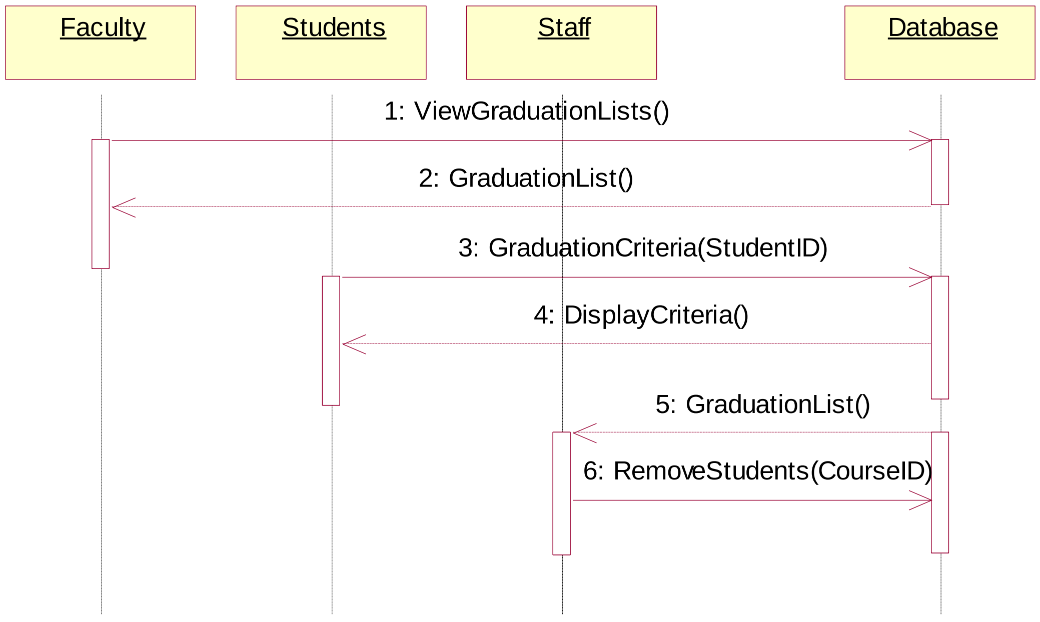
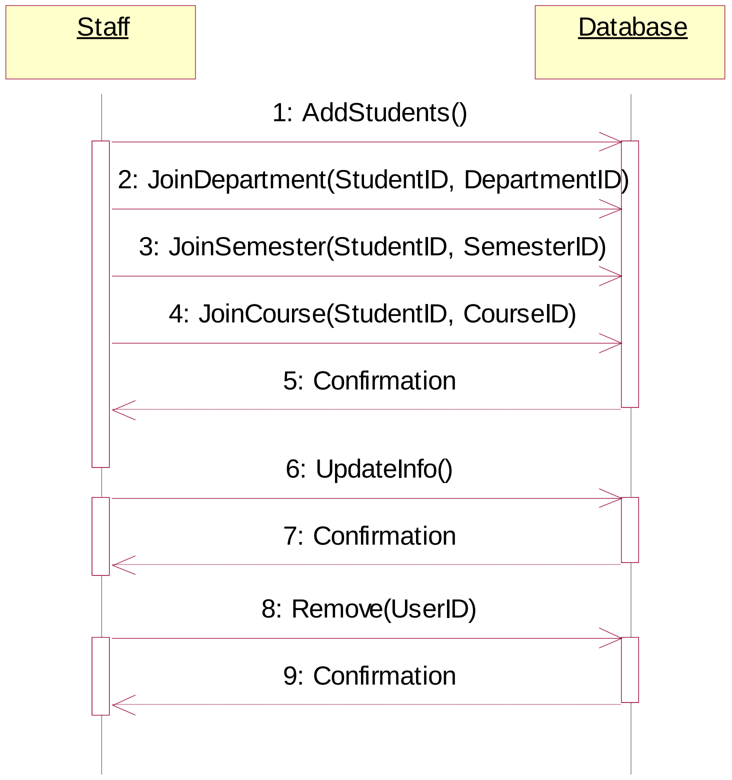


*Figure 4-14 - Improvement Subjects - Sequence Diagram*

### 4.4.3.6 Manage Users

AAS puts the responsibility of managing the students and other users on the head of Management Staff. This includes adding data for a new course, including their departments, courses, semesters etc. This information requires update over the passage of time. Moreover, when a course has complete their tenure of education, their access can be removed. All these tasks are represented in the form of following sequence diagram.



Staff
Database
1: AddStudents()
2: JoinDepartment(StudentID, DepartmentID)
3: JoinSemester(StudentID, SemesterID)
4: JoinCourse(StudentID, CourseID)
5: Confirmation
6: UpdateInfo()
7: Confirmation
8: Remove(UserID)
9: Confirmation

*Figure 4-15 - Manage Users - Sequence Diagram*

### 4.4.3.7 Manage Subjects

The following sequence diagram represents the management of subjects by the management staff.

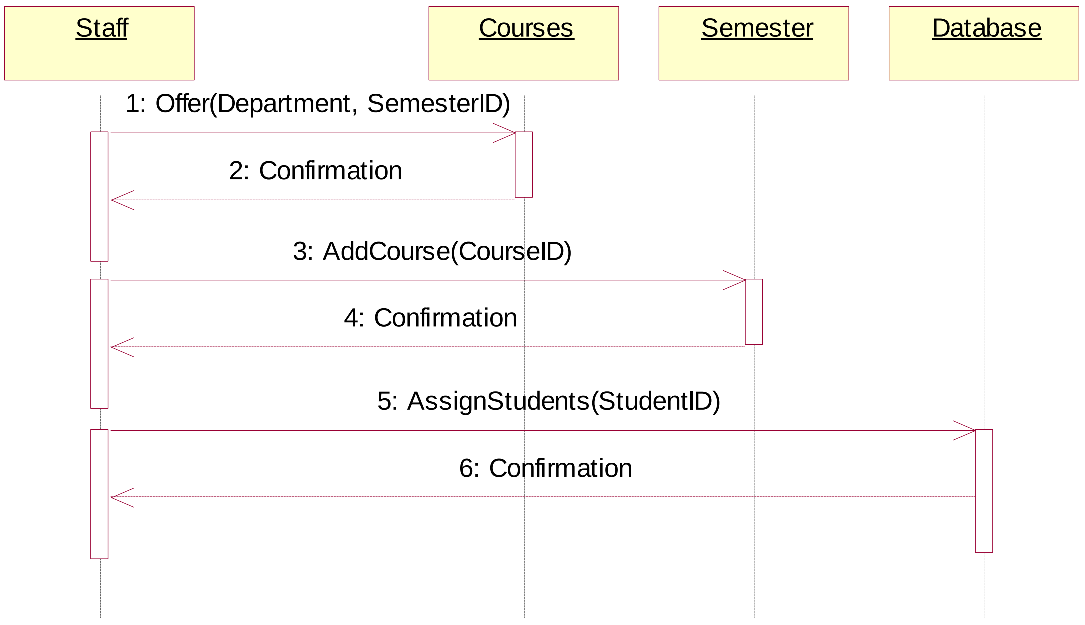


*Figure 4-16 - Manage Subjects - Sequence Diagram*

#### 4.4.3.8 Manage Results

For the AAS to function properly, it needs to keep the results updated in following way:

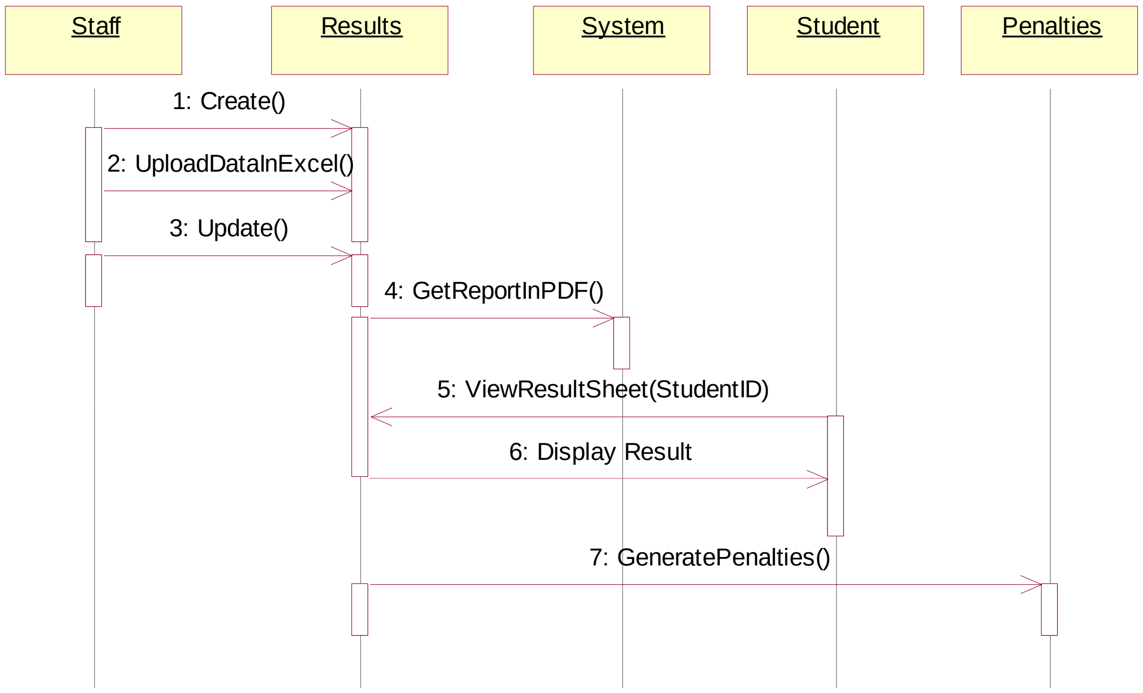


*Figure 4-17 - Manage Results - Sequence Diagram*

## 4.5 Detailed Description of Components

### 4.5.1 User Management

| | |
|---|---|
| Identification | **User Management** |
| Type | Component |
| Purpose | • To manage a set of users (Student, Faculty, Staff). It will manage the user credential database and validation of user credentials in case of login activity.<br>• All of this functionality is hidden from the user - the nitty-gritty details of managing the users are the job of the System class. |
| Function | • Login(Username, Password)<br>• Logout()<br>• VerifyDetails()<br>• GrantsAccess() |
| Subordinates | User database will be used to hold the user records. |
| Dependencies | All user management functions depends on it. |
| Interfaces | Represents User management options to the user<br>• Login(button)<br>• Logout(button) |
| Resources | User Database |
| Processing | • Login(username, password) logs in the user<br>• Logout(user) logs out the user |

|  |  |
|---|---|
|  | • VerifyDetails() verify user credentials from user database<br>• GrantsAccess() grants user access in the user database |
| Data | Graphics for interface support.<br>A database repository for maintaining the user record. |

*Table 4-13 - User Management Component*

### 4.5.2 Results

|  |  |
|---|---|
| Identification | **Results** |
| Type | Class |
| Purpose | • To search the records of the students.<br>• To generate the penalties based on the results.<br>• To allow results to be exported. |
| Function | • Create()<br>• Update()<br>• GeneratePenalties() |
| Subordinates | User database will be used to hold the user records. |
| Dependencies | All academic analysis functions depend on it. |
| Interfaces | • View results individually<br>• View combined results<br>• Export to PDF |
| Resources | Results Database |

| | |
|---|---|
| Processing | • Create() creates the result<br>• Update() is used to update the already stored results<br>• GeneratePenalties() generates penalties for students falling below certain criteria. |
| Data | Services and user repository |

*Table 4-14 - Results Component*

### 4.5.3 User Access Rights

| | |
|---|---|
| Identification | **User Access Rights** |
| Type | Component |
| Purpose | • To classify the users into categories on the basis of privileges assigned to each user category.<br>• The privileges contain user rights of interacting the app. |
| Function | • Categorize(User)<br>• GrantAccess() |
| Subordinates | User database will be used to hold the user records. |
| Dependencies | All user management functions depend on it. |
| Interfaces | • Sends user to the profile specified for its category |
| Resources | User Database |
| Processing | • Categorize (User) is used to categorize the user into one of the defined categories.<br>• GrantAccess () grants access according to that category. |

| | |
|---|---|
| Data | Database repository for maintaining the user record, based on their individual access rights. |

*Table 4-15 - User Access Rights Component*

### 4.5.4 Button

| | |
|---|---|
| Identification | **Button** |
| Type | Component |
| Purpose | • Class to allow a user to choose various Menu options<br>• Graphic user interface representation |
| Function | • Represents various options available to the user<br>• ButtonCallback (button) – a Callback function, used when the button is clicked in order to call the requisite function related to that button. |
| Subordinates | A graphical picture will be used for the button's graphic. |
| Dependencies | Depends on the menu class |
| Interfaces | • button->text() [text overlay on the button]<br>• button->bitmap() [graphic for the button]<br>• button->screenX(), button->screenY() [screen coordinates for the button] |
| Resources | Nil |
| Processing | • ButtonCallback(button()<br>• Callback functions for the button. |
| Data | A graphical picture to hold the visual graphic of the button. |

*Table 4-16 - Button Component*

### 4.5.5 Menu Component

| | |
|---|---|
| Identification | **Menu Component** |
| Type | Component |
| Purpose | • User interface to access different features of the Application<br>• Separate menu class instances for each menu screen |
| Function | • Display a window of buttons representing different menu choices.<br>• For each menu choice, the menu class will call the function display (button) to display button onscreen. |
| Subordinates | An array will hold different buttons, one for each option of the menu. |
| Dependencies | Calls instances of the button class. |
| Interfaces | Represented in UIs as menus |
| Resources | Nil |
| Processing | display(button) displays the button onscreen |
| Data | Array of button classes |

*Table 4-17 - Menu Component*

### 4.5.6 System

| | |
|---|---|
| Identification | **System Class** |
| Type | Class |
| Purpose | • To manage user's login and logout |

| | |
|---|---|
| | • To provide imports from Excel files and exports to PDF files |
| Function | • Works along with the login screen<br>• Imports and Exports can be used for data entry or reports |
| Subordinates | Main System User |
| Dependencies | Aggregation of Students and Users; requires data from the database for authentication |
| Interfaces | Login Screen, as well as UIs for import and export of data |
| Resources | Database, Online Server |
| Processing | GetReportToPDF() generates PDF files of lists and results<br>UploadDataInExcel() takes excel files as input and stores in DB<br>UpdateSysAdmin() used to manage the Administrator<br>Login(Username, Password), VerifyDetails(), GrantAccess() and Logout() used for User Management |
| Data | Data about System Administrator |

*Table 4-18 - System Class Component*

### 4.5.7 Student

| | |
|---|---|
| Identification | **Student Class** |
| Type | Class |
| Purpose | Provides UI for students and store data before sending it to database |

| | |
|---|---|
| Function | • Gets data of students<br>• Assigns them dept., course, semester etc.<br>• Includes all functions for displaying data on the profile |
| Subordinates | All functions that deal directly with students. It includes those for input of data in the database, and those for output, both. |
| Dependencies | Stores data in the database. Has an aggregation relationship with System Class. |
| Interfaces | Student's Profile |
| Resources | Constant connection with database |
| Processing | • AddStudents(), Update(StudentID), JoinCourse(StudentID, CourseID), JoinSemester(StudentID, Semester), Join Department(StudentID, Department):<br>All these functions edit or update data of students in database.<br>• LodgeRequest():<br>for launching complaints<br>• ViewPenalties(StudentID), ViewGPA(StudentID), ViewResultSheet(StudentID), GraduationCriteria(StudentID), CheckImprovementSubjects(StudentID), NotifyForImprovement(StudentID, CourseID):<br>Accesses data from database and displays it to student |

| | |
|---|---|
| Data | Students' Data from Database |

Table 4-19 - Student Class Component

### 4.5.8 Users

| | |
|---|---|
| Identification | **Users Class** |
| Type | Class |
| Purpose | • To provide interface for faculty & staff and store their data before sending it to database |
| Function | • Gets data of faculty and staff<br>• Performs the managerial functions that the faculty and staff has to perform |
| Subordinates | All functions that deal directly with administrative tasks of students. Works in relation with Penalties, Semester, Courses, and Departments classes. |
| Dependencies | Stores data in the database. Has an aggregation relationship with System Class. |
| Interfaces | Faculty's and Staff's profile |
| Resources | Constant connection with database |

| | |
|---|---|
| Processing | • Create(), Remove(UserID), UpdateInfo(), ListUsers(), RemoveStudents(CourseID): Deals with user management<br>• GetDeptUsers(DeptID), AssignUsersToDept(UserID, DeptID): Deals with departmental data<br>• ForwardRequest(), Approve(), NotifyStudent(): Involves notifications and complaints<br>• ViewGraduationList(), ViewImprovementSubjects(), OfferSubjects(CourseID): Managerial roles for Faculty |
| Data | Students' Data from Database |

*Table 4-20 - Users Class Component*

### 4.5.9 Departments

| | |
|---|---|
| Identification | **Department Class** |
| Type | Class |
| Purpose | • To manage data of Depts. at backhand |
| Function | • Tracks members of a department |
| Subordinates | Students, Faculty and Staff are related to at least one of the instances of Dept |
| Dependencies | On students, faculty and staff |
| Interfaces | - |

| | |
|---|---|
| Resources | Database |
| Processing | • Create(), UpdateInfo(): tracks Dept.'s data<br>• ListUsers(): tracks members of a dept. |
| Data | Users' Data from Database |

*Table 4-21 - Department Class Component*

### 4.5.10 Penalties

| | |
|---|---|
| Identification | **Penalties Class** |
| Type | Class |
| Purpose | • To manage data regarding academic penalties<br>• To manage data regarding complaints |
| Function | • Awarding penalties on students, considering their results |
| Subordinates | Academic Penalties and Complaints |
| Dependencies | On Students and Results |
| Interfaces | Display Penalties to Users in their respective profiles |
| Resources | Result Database |
| Processing | • Create(), Update (): generates and edits penalties<br>• ApplyOnStudent(StudentID): Awarding penalties to individuals<br>• ListStudents(): to see a list of all awardees of penalties |
| Data | Results from Database |

*Table 4-22 - Penalties Class Component*

## 4.5.11 Courses

| | |
|---|---|
| Identification | **Courses Class** |
| Type | Class |
| Purpose | • To store data regarding subjects being offered |
| Function | • Keeps track of all subjects that are being, or can be offered. This data can come in handy for calculation of subjects for improvement.<br>• Manages a list of prerequisites, to avoid students studying a subject they haven't qualified for. |
| Subordinates | It only comprises of data of courses offered in MCS CSE Dept, their Credit Hours, Maximum Securable Points etc. |
| Dependencies | On department and semester |
| Interfaces | Displayed as Subjects for Improvement to Students and Faculty |
| Resources | Database, use of mathematical formulae to calculate the margin of improvement etc. |
| Processing | • Create(), UpdateInfo(): tracks course's data<br>• Offer(DeptID, SemesterID): offers subjects to study in specific semesters<br>• AssignStudents (StudentID): to assign students to a subject that is being offered. |
| Data | Courses' Data, List of Prerequisites |

*Table 4-23 - Courses Class Component*

### 4.5.12 Semester

| | |
|---|---|
| Identification | **Semester Class** |
| Type | Class |
| Purpose | • To manage data of Semesters that are being held |
| Function | • Keeps track that which course or individuals are studying in which semester<br>• Keeps track on which subjects are being offered in which semester |
| Subordinates | Includes data regarding Semester's starting and ending date, subjects offered etc. |
| Dependencies | On Department and Courses |
| Interfaces | Semester Data being displayed at students' profile. Data management from staff's profile |
| Resources | Database |
| Processing | • Create(), UpdateInfo(): tracks Semester's data<br>• AddCourse(CourseID): adds courses that are making up the semester |
| Data | Courses' Data |

*Table 4-24 - Semester Class Component*

## 4.6 Summary

The purpose of this chapter was to deliver a portrayal of the design of the system suitable enough to allow for software development, with an understanding of what is built and how it is developed. It provides information essential to getting a description of the details for the software and the system to be built. It presents a design view and detailed description of AAS. It explains the purpose, features, interfaces, what the system do, its entire processes in detail, the constraints under which it must operate and how the system reacts to inputs and what will be its outputs.

CHAPTER 5

# Implementation

# 5. Implementation

## 5.1 Introduction

The preceding chapter discoursed comprehensive design of the AAS. This design is converted into an application by utilizing numerous technologies and tools. The implementation details are conferred in the subsequent divisions providing minutiae of the system's inner functioning.

## 5.2 Tools & Technologies

### 5.2.1 PHP

Hypertext Preprocessor (PHP) is a server-side scripting language designed for Web development but also used as a general-purpose programming language.

### 5.2.2 Atom

Atom is a free and open-source text and source code editor for macOS, Linux, and Microsoft Windows with support for plug-ins written in Node.js, and embedded Git Control, developed by GitHub.

### 5.2.3 HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current major version of the HTML standard.

### 5.2.4 SQL

SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.

### 5.2.5 MySQL

MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

### 5.2.6 JS

JavaScript, often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm.

### 5.2.7 jQuery

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It is free, open-source software using the permissive MIT License.

### 5.2.8 Apache Server

The Apache HTTP Server, colloquially called Apache, is a free and open-source cross-platform web server, released under the terms of Apache License 2.0.

## 5.3 UI Design

The system supports an intuitive and easy to use UI that has an extremely shallow learning curve and require minimum training to be operated at maximum efficiency. In this way, users of the web application won't feel reluctant to use the application to keep an eye on the results, while the faculty and staff won't feel reluctant to adopt the new system as means of academic analysis either.

### 5.3.1 Login Screen

Following are the sketches of UI implementation for AAS. This will be the first screen that the user sees upon opening the AAS. The interface is simple and self-explanatory.

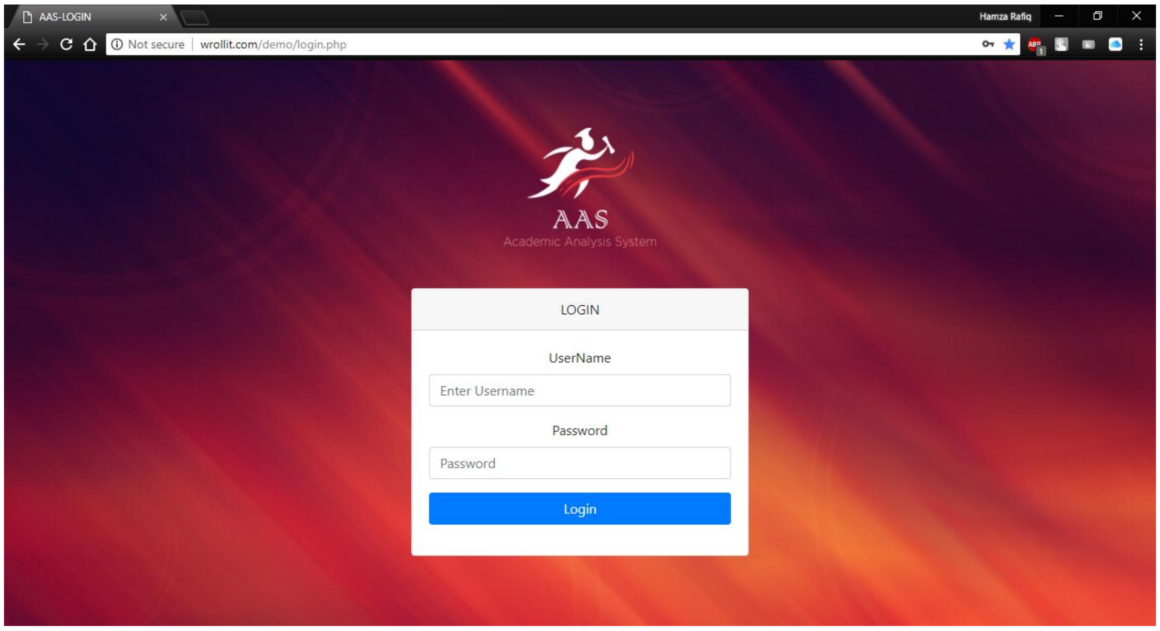
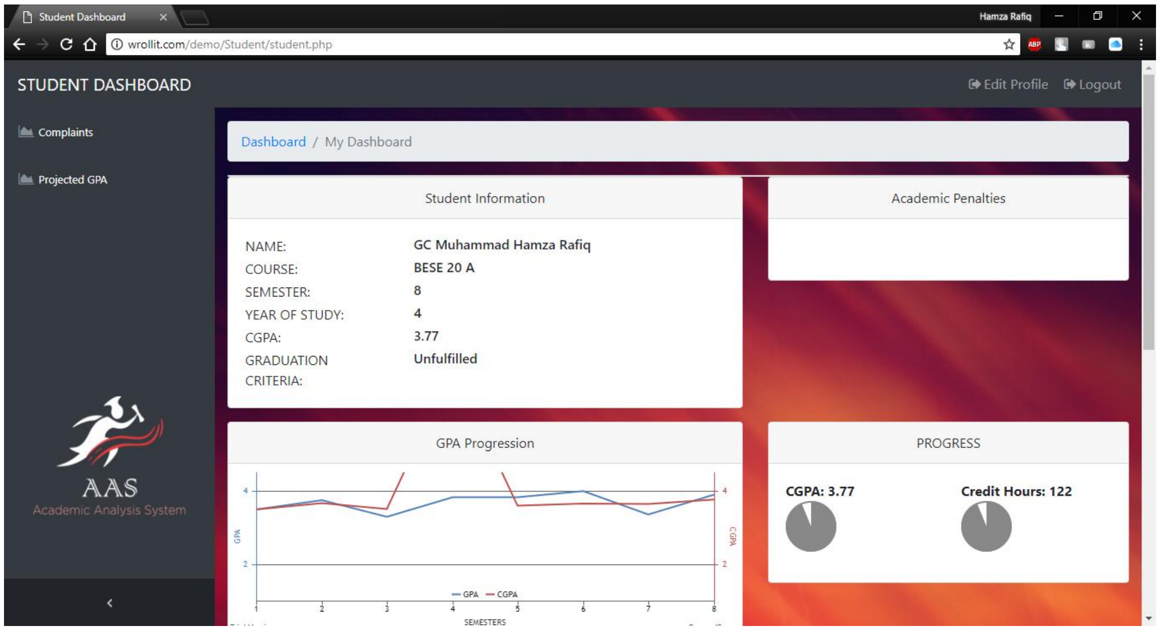


*Figure 5-1 - Login Screen UI*

### 5.3.2 Students' Profile

If a student logs in to the system, this will be the screen that is presented to him or her. It provides a minimalist, yet explanatory UI which covers all required aspects on one screen. The graphical methods for representation of data make the data insights more visible.


Student Dashboard

wrollit.com/demo/Student/student.php

STUDENT DASHBOARD

Edit Profile Logout

Complaints

Projected GPA

Dashboard / My Dashboard

Student Information

|  |  |
|---|---|
| NAME: | GC Muhammad Hamza Rafiq |
| COURSE: | BESE 20 A |
| SEMESTER: | 8 |
| YEAR OF STUDY: | 4 |
| CGPA: | 3.77 |
| GRADUATION CRITERIA: | Unfulfilled |

Academic Penalties

GPA Progression

PROGRESS

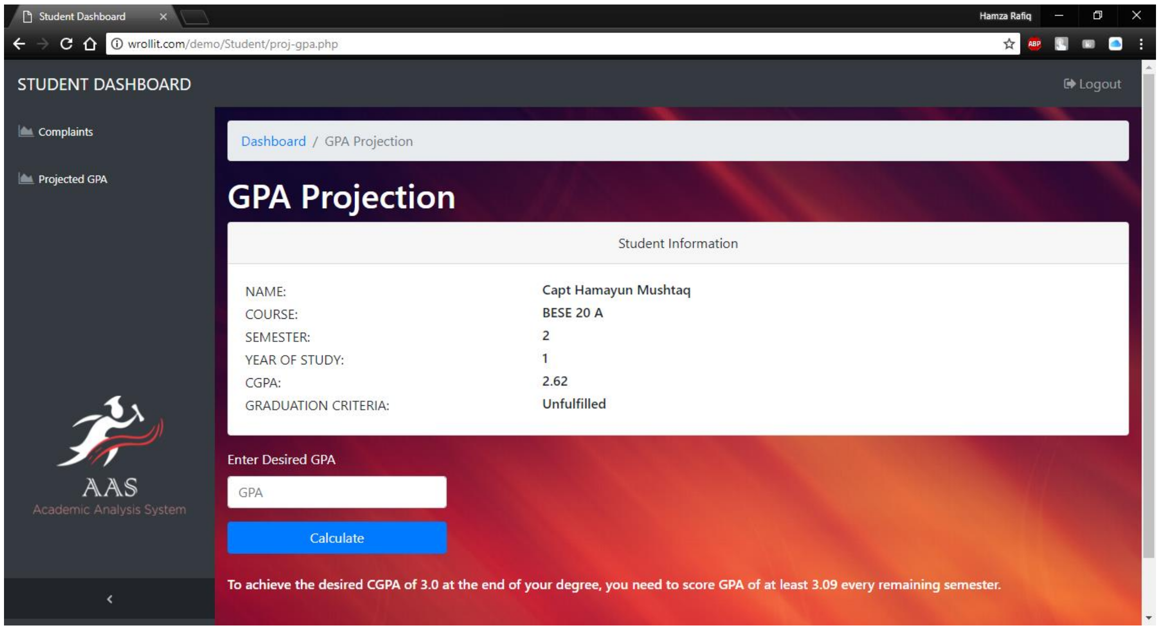
CGPA: 3.77

Credit Hours: 122

AAS Academic Analysis System

*Figure 5-2 - Student's Profile UI*

The system provides the students with an opportunity to view a projection of their GPAs.


Student Dashboard

wrollit.com/demo/Student/proj-gpa.php

STUDENT DASHBOARD

Logout

Complaints

Projected GPA

Dashboard / GPA Projection

GPA Projection

Student Information

|  |  |
|---|---|
| NAME: | Capt Hamayun Mushtaq |
| COURSE: | BESE 20 A |
| SEMESTER: | 2 |
| YEAR OF STUDY: | 1 |
| CGPA: | 2.62 |
| GRADUATION CRITERIA: | Unfulfilled |

Enter Desired GPA

GPA

Calculate

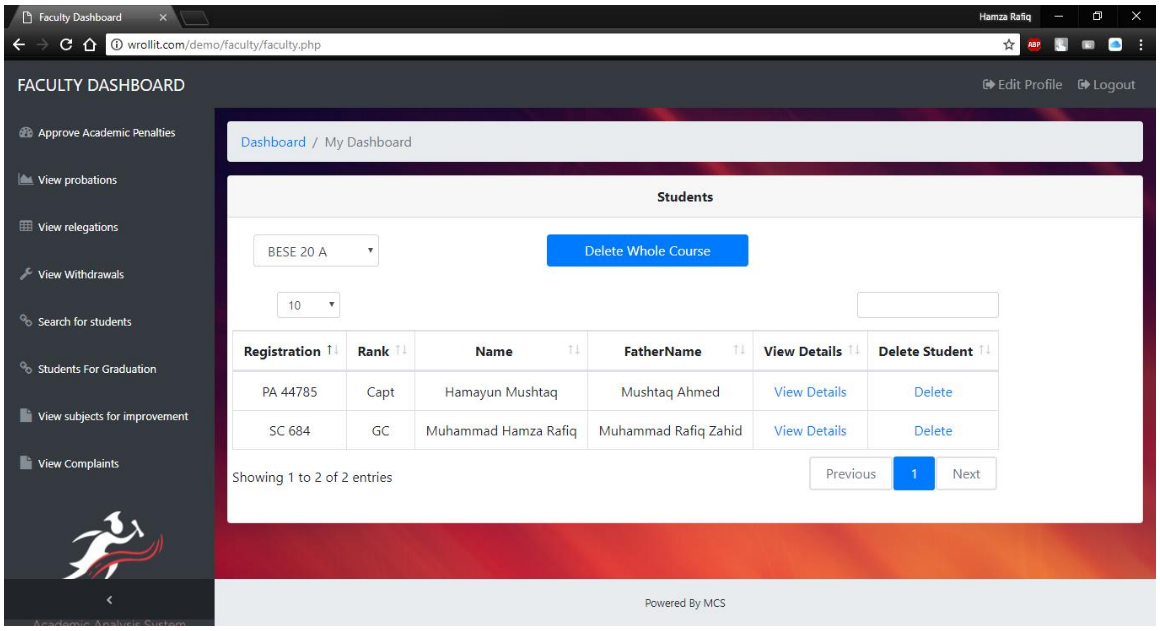
To achieve the desired CGPA of 3.0 at the end of your degree, you need to score GPA of at least 3.09 every remaining semester.

AAS Academic Analysis System

*Figure 5-3 - GPA Projection UI*

### 5.3.3 Faculty's Profile

The easy-to-understand basic screen for the faculty will look like the following image.



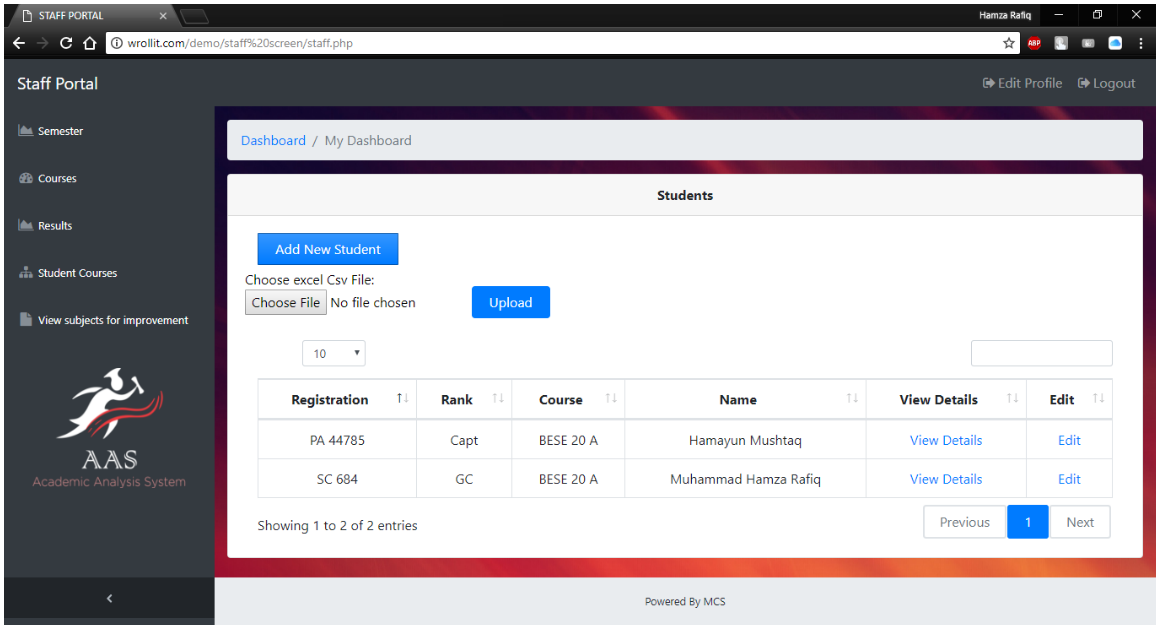
| Registration | Rank | Name | FatherName | View Details | Delete Student |
|---|---|---|---|---|---|
| PA 44785 | Capt | Hamayun Mushtaq | Mushtaq Ahmed | View Details | Delete |
| SC 684 | GC | Muhammad Hamza Rafiq | Muhammad Rafiq Zahid | View Details | Delete |

Showing 1 to 2 of 2 entries

*Figure 5-4 - Faculty's Profile*

### 5.3.4 Staff's Profile

The following interface is available for usage by the management staff.



| Registration | Rank | Course | Name | View Details | Edit |
|---|---|---|---|---|---|
| PA 44785 | Capt | BESE 20 A | Hamayun Mushtaq | View Details | Edit |
| SC 684 | GC | BESE 20 A | Muhammad Hamza Rafiq | View Details | Edit |

Showing 1 to 2 of 2 entries

*Figure 5-5 - Staff's Profile*

## 5.4 Summary

Implementation details of AAS are discussed in this chapter. Different functionalities and strategies to develop the system have also been pondered upon. A brief introduction to different tools and technologies employed is also given.

# CHAPTER 6

# Testing

# 6. Testing

## 6.1 Introduction

The purpose of this chapter is to elicit all material that is essential to plan and control the test efforts for the development of this project. It specifies the test plan for AAS application during the development phase and provides a rationale behind the necessity of these tests. This document provides an overview of the tests that were implemented, the items that were targeted by the tests, along with the testing approach that was deployed. This testing is being done according to the elicited requirements in Software Requirements Specification Document for AAS.

## 6.2 Test Items

- Account Management
- Academic Records
- Academic Penalties
- Graduation Criteria Lists
- Roadmap for Students
- Improvement Subjects
- Semester Management
- Course Management

## 6.3 Tested Features

- Login

- Create accounts
- Manage accounts
- Remove users
- Changing password
- Adding individual subjects
- Adding subjects for the whole course
- Improvement subjects
- Update subjects
- Remove subjects
- Display academic records
- Approval of penalties
- View subjects for improvement
- View graduation criteria list
- Import files from Excel
- Export files to .pdf

## 6.4 Features not to be tested

Following tests are not done on the system since they are out of the scope of this project.

- Security Testing

## 6.5 Approach

The system is working in modules so the testing phase was initiated by testing each module separately i.e. unit testing, and then step by step integrating modules to test them with each other i.e. integration testing, followed by the testing of the complete application as a whole.

## 6.6 Item Pass/Fail Criteria

- Items will pass the test if the actual output of each of the test case is same as the desired output of the system.
- Any transfer of data between any modules is updated in the database.

## 6.7 Suspension Criteria and Resumption Requirements

### 6.7.1 Suspension Criteria

- The build contains many serious defects which seriously limit testing progress.
- Software/hardware problem
- Assigned resources are not available when needed to be tested.

### 6.7.2 Resumption Requirements

- Resumption will only occur when the problems that caused the suspension have been resolved.

## 6.8 Testing Tasks

- Development of test cases
- Execution of tests based on the developed test cases
- Report defects from the executed test cases, if any
- Provision of complete test report
- Incorporate changes later in the stage of the project development

## 6.9 Staffing and Training Needs

- A 1-hour training session can be arranged to guide the staff and faculty that will be managing this application.
- The portfolio for students is self-explanatory and doesn't require any special training.

## 6.10 Test Cases

| | |
|---|---|
| Test Case Name | Application Startup Testing |
| Test Case ID | 1 |
| Description | This feature sends the user to the login screen of the web application when he/she enters the website https://goo.gl/ceHBkQ in the web browser. |
| Testing Technique Used | Black Box Testing |
| Preconditions | The computer is on and is connected to the internet, and a web browser is running. |
| Input Values | URL: https://goo.gl/ceHBkQ |
| Valid Inputs | The specified URL address |
| Steps | 1. Enter URL in the browser. 2. Press Enter. |
| Expected Output | The user will be sent to the login screen of AAS. |
| Actual Output | Successful opening of the application |
| Status | PASS |

*Table 6-1 - Application Startup Testing*

| | |
|---|---|
| Test Case Name | Login Feature Testing |
| Test Case ID | 2 |
| Description | This feature asks the user to enter his/her credentials for login. This test case is aimed to check that feature works according to user requirement. |
| Testing Technique Used | Black Box Testing |
| Preconditions | System is running and connected to a database. The user has opened the login webpage. |
| Input Values | 1. Username<br>2. Password |
| Valid Inputs | 1. Valid and authorized username<br>2. Valid and authorized password |
| Steps | 1. Enter username.<br>2. Enter password.<br>3. Click "Login". |
| Expected Output | The user credentials will be passed to the server for verification. The valid users will be directed to their dashboard after login. |
| Actual Output | Successful login. The user is directed to the profile dashboard. |
| Status | PASS |

*Table 6-2 - Login Feature Testing*

| | |
|---|---|
| Test Case Name | Student Account Creation Testing |
| Test Case ID | 3 |
| Description | This test case checks the procedure of creation of user accounts for students by staff. This feature will be provided to staff on the dashboard screen. |
| Testing Technique Used | Black Box Testing |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | Name, Father's Name, CNIC, Rank, Unit, Weight, Department, Medical Category, Service, Registration Number, Blood Group, Height, Date of Birth, Date of Commission, Father's Address, Father's Occupation |
| Valid Inputs | Alphanumeric values for the fields stated above |
| Steps | 1. On Staff's Dashboard, click "Add New Student". 2. Enter the data for the student. 3. Click "Register". |
| Expected Output | The new student is registered and added into the database and can be viewed now in the students' list. |
| Actual Output | The account is created in database and is visible on the student's list. |
| Status | PASS |

*Table 6-3 - Student Account Creation Testing*

| | |
|---|---|
| Test Case Name | Adding New Semester Testing |
| Test Case ID | 4 |
| Description | This test case checks that the staff can create new semesters as per the requirement, so new courses can be added for a particular student. |
| Testing Technique Used | Black Box Testing |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | 1. Title (e.g. Spring)<br>2. Year<br>3. Semester No. |
| Valid Inputs | Alphanumeric characters |
| Steps | 1. Click on "Semester" from the side panel.<br>2. Click on "Add new semester".<br>3. Enter the data.<br>4. Click "Register". |
| Expected Output | New semester will be added in database and semester's list. |
| Actual Output | The database and list are updated with the new semester. |
| Status | PASS |

*Table 6-4 - Adding New Semester Testing*

| | |
|---|---|
| Test Case Name | Semester Deletion Testing |
| Test Case ID | 5 |

| | |
|---|---|
| Description | This test case checks that the staff can delete already created semesters as per the requirement. |
| Testing Technique Used | Black Box Testing |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | Mouse Click |
| Valid Inputs | Mouse Click |
| Steps | 1. Click on “Semester” from the side panel. 2. Click on “Delete” in front of the semester that is to be deleted. |
| Expected Output | Semester will be deleted in database and semester’s list. |
| Actual Output | The database and list are updated with the semester deleted. |
| Status | PASS |

*Table 6-5 - Semester Deletion Testing*

| | |
|---|---|
| Test Case Name | Subject Generation Testing |
| Test Case ID | 6 |
| Description | This test case checks that the staff can create new subjects as per the requirement, so new courses can be added for students. |
| Testing Technique Used | Black Box |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |

|  |  |
|---|---|
| Input Values | Course Name, Credit Hours, Semester, Instructor |
| Valid Inputs | Alphanumeric Characters |
| Steps | 1. Click on “Courses” from the side panel.<br>2. Click on “Add new course”.<br>3. Enter the data.<br>4. Click “Register”. |
| Expected Output | New subject will be added in database and semester’s list. |
| Actual Output | The database and list are updated with the new subject. |
| Status | PASS |

*Table 6-6 - Subject Generation Testing*

|  |  |
|---|---|
| Test Case Name | Subject Deletion Testing |
| Test Case ID | 7 |
| Description | This test case checks that the staff can delete already created subjects as per the requirement. |
| Testing Technique Used | Black Box |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | Mouse Click |
| Valid Inputs | Mouse Click |
| Steps | 1. Click on “Courses” from the side panel.<br>2. Click on “Delete” in front of the subject that is to be deleted. |

| | |
|---|---|
| Expected Output | Subject will be deleted in database and semester's list. |
| Actual Output | The database and list are updated with the subject deleted. |
| Status | PASS |

*Table 6-7 - Subject Deletion Testing*

| | |
|---|---|
| Test Case Name | Students' Course Registration Testing |
| Test Case ID | 8 |
| Description | This test case checks the process of registration of new courses for a specified student by staff. |
| Testing Technique Used | Black Box |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | Student, Semester ID, Subject |
| Valid Inputs | Items selected from the dropdown lists of already registered students, subjects and semesters. |
| Steps | 1. Click on "Student Courses" from the side panel. 2. Select the student who needs the subject to be added to. 3. Select Semester and Course from the dropdown list. 4. Click "Select". |
| Expected Output | New subject will be registered for the student. |
| Actual Output | New subject is registered for the student. |
| Status | PASS |

*Table 6-8 - Students' Course Registration Testing*

| | |
|---|---|
| Test Case Name | Students' Result Addition Testing |
| Test Case ID | 9 |
| Description | This test case checks the process of adding results to the already registered subjects of a student. |
| Testing Technique Used | Black Box |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. |
| Input Values | Subject, Student, Points, Grade |
| Valid Inputs | Either from dropdown lists or alphanumeric characters |
| Steps | 1. Click on "Results" from the side panel. 2. Select Subject. 3. Click "Submit" against the student required. 4. Enter Marks and Grade, and click "Register". |
| Expected Output | The result will be updated in the database. It will now be shown in completed courses for the student, and CGPA is updated. |
| Actual Output | Same as expected |
| Status | PASS |

*Table 6-9 - Students' Result Addition Testing*

| | |
|---|---|
| Test Case Name | Subjects to Offer for Improvement Testing |
| Test Case ID | 10 |

| | |
|---|---|
| Description | This test case checks the list of the subjects that are most suited to be offered for improvement. Average grade points are calculated for all the subjects and all the students. The ones with the lowest average are preferred. |
| Testing Technique Used | Black Box |
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. Some results must be already fed. |
| Input Values | None |
| Valid Inputs | None |
| Steps | Click on "View Subjects for Improvement" in the side panel. |
| Expected Output | Average grade points are calculated for all the subjects and all the students. The ones with the lowest average are preferred. |
| Actual Output | List is generated just as required. |
| Status | PASS |

*Table 6-10 - Subjects to Offer for Improvement Testing*

| | |
|---|---|
| Test Case Name | Penalty Initiation Testing |
| Test Case ID | 11 |
| Description | This test case checks if the staff can successfully generate penalties for students, and these penalties are forwarded to Faculty for approval. |
| Testing Technique Used | Black Box |

| | |
|---|---|
| Preconditions | Staff is logged into the account and is currently viewing the dashboard. Some students must be already registered. |
| Input Values | Student, Type of penalty |
| Valid Inputs | From dropdown lists |
| Steps | 1. Select student for a penalty on the dashboard, and click "View Details".<br>2. Click "Add Penalty".<br>3. Select the type of penalty and click "Register". |
| Expected Output | Penalty will be forwarded to the Faculty. |
| Actual Output | Penalty is forwarded to Faculty. |
| Status | PASS |

*Table 6-11 - Penalty Initiation Testing*

| | |
|---|---|
| Test Case Name | Penalty Approval Testing |
| Test Case ID | 12 |
| Description | This test case checks if the faculty can successfully approve or deny penalties for students. These penalties were forwarded by Staff. |
| Testing Technique Used | Black Box |
| Preconditions | Faculty is logged into the account and is currently viewing the dashboard. Some penalties must be already initiated. |
| Input Values | Penalty Status |
| Valid Inputs | Approve / Deny |

| | |
|---|---|
| Steps | 1. Select "Approve Academic Penalties" from the side panel.<br>2. Click "Manage" against the penalty which needs to be confirmed.<br>3. Select "Approve" or "Dismiss" and click "Register". |
| Expected Output | The penalty is sent to the penalty's list and if the status was approved, it would be notified to the student as well. |
| Actual Output | Same as expected. |
| Status | PASS |

*Table 6-12 - Penalty Approval Testing*

| | |
|---|---|
| Test Case Name | View Academic Penalties Testing |
| Test Case ID | 13 |
| Description | This test case checks if the approved or dismissed penalties are displayed to the faculty for the sake of the record. |
| Testing Technique Used | Black Box |
| Preconditions | Faculty is logged into the account and is currently viewing the dashboard. Some penalties must be already approved. |
| Input Values | Mouse Click |
| Valid Inputs | Mouse Click |
| Steps | From the side panel, select either of the following:<br>1. View Probations<br>2. View Relegations |

| | |
|---|---|
| | 3. View Withdrawals |
| Expected Output | The approved and dismissed penalties will be displayed. |
| Actual Output | The approved and dismissed penalties are displayed. |
| Status | PASS |

*Table 6-13 - View Academic Penalties Testing*

| | |
|---|---|
| Test Case Name | Student Records Test Case |
| Test Case ID | 14 |
| Description | This test case checks the data displayed on the dashboard when a student logs in. It includes student information, academic penalties, current courses, pie charts and courses for improvement. |
| Testing Technique Used | Black Box |
| Preconditions | Student logs in the system. |
| Input Values | None |
| Valid Inputs | None |
| Steps | Student logs into his profile with the provided credentials. |
| Expected Output | The academic records will be displayed. |
| Actual Output | The academic records are displayed. |
| Status | PASS |

*Table 6-14 - Student Records Test Case*

| | |
|---|---|
| Test Case Name | GPA Projector Testing |
| Test Case ID | 15 |

|  |  |
|---|---|
| Description | This test case checks the feature in which the user enters the GPA he wants to end his degree on. The system calculates a minimum GPA which he must acquire to achieve his desired GPA. |
| Testing Technique Used | Black Box |
| Preconditions | Student is logged into the system. |
| Input Values | Desired GPA |
| Valid Inputs | Float Value |
| Steps | 1. Select "Projected GPA" from the side panel.<br>2. Enter the desired GPA.<br>3. Click "Calculate". |
| Expected Output | A minimum required GPA will be displayed. |
| Actual Output | As per expected. |
| Status | PASS |

*Table 6-15 - GPA Projector Testing*

|  |  |
|---|---|
| Test Case Name | Logout Testing |
| Test Case ID | 16 |
| Description | This test case checks for the logout feature for the users. |
| Testing Technique Used | Black Box |
| Preconditions | User must be logged into the system. |
| Input Values | Mouse Click |
| Valid Inputs | Mouse Click |

|  |  |
|---|---|
| Steps | 1. Click on “Logout” button. <br> 2. Confirm the logout. |
| Expected Output | Logs out of the system. |
| Actual Output | Same as expected |
| Status | PASS |

*Table 6-16 - Logout Testing*

## 6.11 Summary

From the extensive black box testing, it can be concluded that the system is working fine and in accordance with the functional requirements stated in the SRS document. The system can be further improved and enhanced during maintenance and upgrade phase. A detailed white box testing can also highlight the errors and bugs, if present, in the code of the application.

CHAPTER 7

# Conclusion and Future Work

# 7. Conclusion & Future Work

## 7.1 Conclusion

Various applications exist that perform the functions of an academic management system, but there will be very few applications that analyze the data and find meaningful reports out of it, that can not only tell us more about the data but also share the burden which was being performed manually.

In MCS CSE Dept, software that have been in use of similar nature included MIMS, LMS and CMS. These software, although enriched by lots of features, still do not address the issue of being easy-to-learn and easy-to-use. AAS looks forward to being an application that not only addresses these issues but also offer modifiability for future implementations.

## 7.2 Design Decisions & Tradeoffs

The user interface has been kept simple and friendly so even a user with only basic knowledge of web applications can use it effectively.

The need for the details of the students being entered by them has been eliminated. This is done to ensure that the sign in only requires a username and password, which is provided to the users by an administrative authority. This, however, can result in slight rigidity in this regard.

The current results uploaded on CMS are a great source for data entry, but the option of affiliating AAS with CMS, so it accesses its results from CMS, is a tricky one. This not only requires AAS to be very adaptable but would also require compromise in CMS' privacy policy.

The system considers the removal of data of students who have graduated after a certain amount of period, to make the database non-complex, and unburdened. As the implementation atmosphere only comprises of one department at the moment, it would be easy to keep an eye on its working, progress, strengths, weaknesses and feedbacks.

Since this application is a new development effort for implementation in MCS, the focus was to develop an application that is simple enough for an average user to use, while still providing the required features along with.

## 7.3 Future Work

As specified in the external scope of the product, in the coming years, this software can be modified and adjusted for implementation for Post Graduate and PhD students, as well as implementation in other departments like MCS EE Dept with slight changes.

The project can be further enhanced by incorporating confidentiality and shifting it over to HTTPS server. A survey can be conducted among the users of the application, and considering their reviews, changes can be made in the application on annual basis.

# Bibliography

[1] NUST, "National University of Sciences and Technology (Academic Programs) Regulations", 2016.

[2] CMS, "CMS", 2015. [Online]. Available: https://cms.must.edu.pk/. [Accessed: 20- Jun- 2018].

[3] LMS, "NUST LMS Portal", *LMS*, 2018. [Online]. Available: https://lms.nust.edu.pk/. [Accessed: 20- Jun- 2018].

[4] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*, 1st ed. Boston: Addison-Wesley Professional, 2015.

[5] B. Boehm, "Software Risk Management: Principles and Practices", *IEEE Software*, vol. 8, no. 1, pp. 32-41, 1991.

[6] KillerPHP, "PHP Video Tutorials for Web Designers - KillerPHP.com", *KillerPHP.com*, 2018. [Online]. Available: https://www.killerphp.com/. [Accessed: 20- Jun- 2018].

[7] "W3Schools Online Web Tutorials", *W3schools.com*, 2018. [Online]. Available: https://www.w3schools.com/. [Accessed: 20- Jun- 2018].