
Software Specification

for

Learning Management System

**Prepared by Andres
Abrego, Iliana Barrera,
Vanessa Cabrera, Jose
Gonzales and Lloyd
Wilson**

Created on April 15, 2018

Table of Contents

Introduction	3
Purpose	3
Team Organization	3
Overall Description	4
Product Perspective	4
Product Functions	4
User Classes and Characteristics	5
Operating Environment	6
Design and Implementation Constraints	6
Assumptions and Dependencies	6
External Interface Requirements	6
User Interfaces	6
Software Interfaces	10
Communications Interfaces	10
System Features	10
Login Request	10
Registration	11
Other Requirements	19

1. Introduction

1.1 Purpose

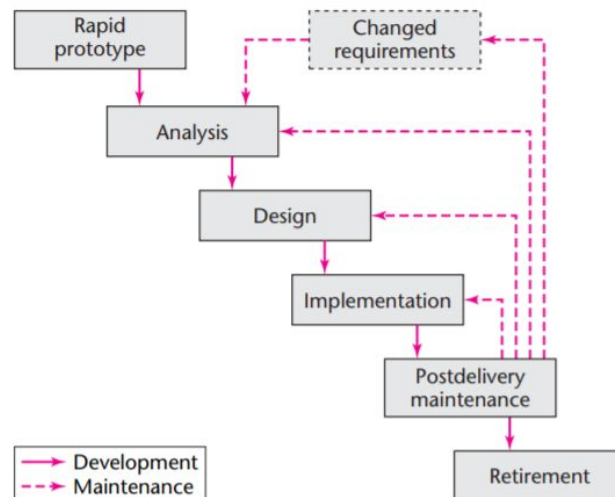
A proper description of our Learning Management System (LMS) and its functionalities is what will be shown throughout this SRS document. Explanations of the product environment, implemented use and data management are only some of the features being written about. Anyone with technical or non-technical knowledge on software engineering should be able to read this document and gain understanding of the LMS.

1.2 Team Organization

Our team is based on the Democratic Team approach where there is no designated leader or chief programmer. All tasks are divided according to skill and all decisions are voted on by majority. The team consists of:

Andres Abrego - Back End/Server
Iliana Barrera - Front End
Vanessa Cabrera - Front End
Jose Gonzalez - Back End/Framework
Lloyd Wilson - Back End/Database

In addition, our project was based on the Rapid Prototyping life cycle model. Our main goal was to build a functioning prototype to present to the class, who would interact with it and let us know of any issues or concerns. Afterwards, we would modify or finalize the prototype to create a fully functioning LMS product to submit.



2. Overall Description

2.1 Product Perspective

The goal is to create a Learning Management System: Software that allows organizations to manage education material and tools. An LMS involves data such as student information, academic grades, course curriculums and other academic resources. Using this data, it can calculate grades, track attendance, completed courses, predict GPA and much more.

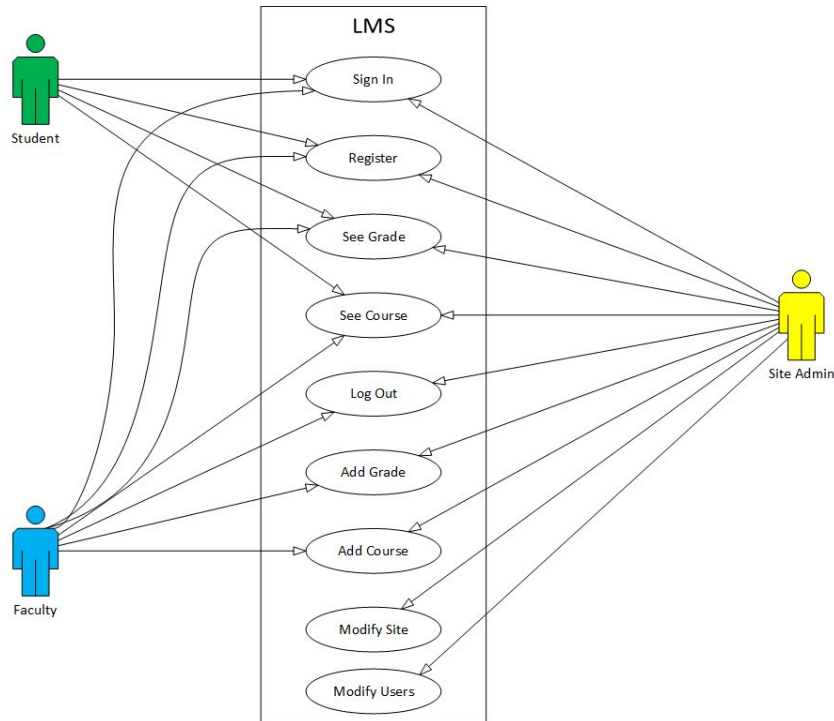
The product is a rendition of the already existing environment, Blackboard, and has almost identical characteristics and requirements. However, due to the fact that our task involves a group of 4 - 5 students, not all the functionalities of a fully implemented LMS will be duplicated. The requirements implemented in our smaller-scale LMS are as mentioned below.

2.2 Product Functions

The following are implemented core features and functionalities that will present in our LMS:

- 1) Registration
 - a) User will be able to create an account that will be added to the database
 - b) Information entered will be name, ID, e-mail (to log-in and distinguish between student and faculty) and a confirmed password
 - c) Available on the top-right navigation bar, so it is always accessible
- 2) User Log-In
 - a) If student: can log-in to view their current courses, grades and GPA
 - b) If faculty: can log-in to view list of courses they are teaching, student roster in each courses, ability to add students and add grades for specific students
- 3) User Log-Out
 - a) Can exit out of the current account sessions, regardless of whether it is a student, faculty or administrator
- 4) Add Grade
 - a) Only faculty or administrator will be able to input a numerical grade
 - b) Can only add a grade for one individual student at a time
 - c) Can only add a grade for an individual course at a time
- 5) Add Course
 - a) Only faculty or administrator can create a new course and its CRN number
 - b) Will then be able to add students or grades to each created course
 - c) Student in a course will be able to view this when they log-in
- 6) User Modification
 - a) Administrators can add or remove registered website users at will
 - b) Administrators can add or remove created courses or the grades added within
- 7) View Grades and Courses
 - a) Users will be able to see their enrolled courses and added grades at the time
 - b) Can view current GPA

2.3 User Classes and Characteristics



The system will consist of four types of users: new users, student, faculty, and admin. The system will assume that all users have basic web experience/expertise to use the software. With the only exception being the admin users as their privileges are superior access to the entire system.

The admin user will have full privileges over the whole system for which entitles them to be able to modify values in the system as well as the site as well. With this power the admin should be experienced with web development as well as database experience. This is designed to ensure the system can be maintained after deployment and accessible to modification on a per-client basis. Furthermore, with this ability each admin user must be trusted with the permission as this user can modify and change values for which may end irreversible once implemented. Therefore security over this user is crucial.

The faculty user will have limited privileges as their main purpose will be interaction with their students. This user can consist of teacher, professor, or lectures of a particular educational institution. The faculty user should be knowledgeable of general online sites as the system will be web-based. The faculty users will only be set by the admin users for which will give them access to all the privileges authorized to the faculty such as adding course, grades, and appending student to courses. Since the faculty users can only see the name of the students in their roster, there is little security concern about this user. This is due to as the only damage that a malicious user can do is change grades and add student to the classes as the ability to remove students is given to the admin.

The student users will be the base users once they have registered to the system. This user consists of any student of a particular educational institution. As the system needs web access, the system, users must have parental guidance if the user is of an elementary to secondary student. The

only privileges this user will have is see courses and grade posted from the faculty user. With this the security concerns on this users is low.

A new users will consist of any user that has not register to the system. The user must be affiliated to the educational institution either by being a student, faculty or appointed as site admin. Thus this user's privileges will be extremely limited until the users has register and log in to the system.

2.4 Operating Environment

Due to it being web-based, our LMS can be accessed on any device; whether mobile or desktop. All that is needed is access to a web browser with no specific hardware platform. Besides that, the web page itself is fairly independent as it is being hosted as an application on the Heroku platform.

However, the database used for the LMS does its limitations. It is created in PostgreSQL() with a specific amount of limited storage (which we do not expect to reach). The database is hosted on Heroku and connect to the front-end of the LMS using Python forms created in Django.

2.5 Design and Implementation Constraints

Most constraints were due to a lack of technical knowledge in specific areas. Due to being an online web-based LMS, time was taken out to learn about HTML, CSS and Javascript to produce multiple functioning pages that connect to each other.

2.6 Assumptions and Dependencies

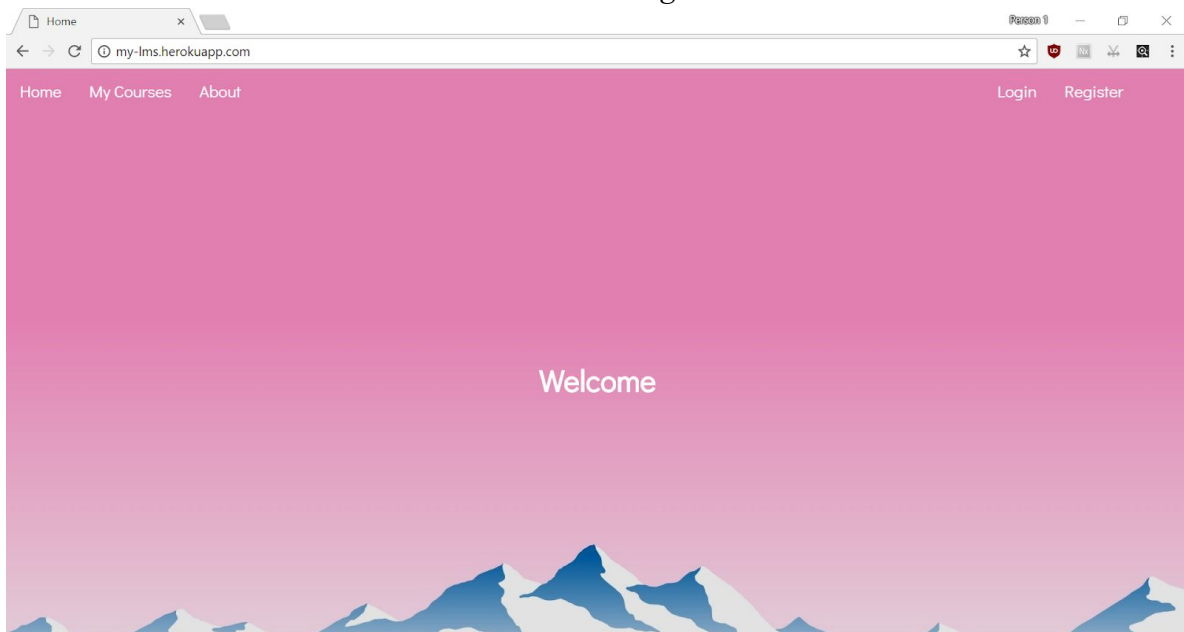
Some obvious assumptions were made when creating the LMS such as:

- A stable Internet connection needed to access the website application since the only way to access it is online
- 24/7 functioning Heroku servers that host the LMS application
- Having an effective time frame for completion
 - Project completion depended on the time frame given of a semester to implement as many functioning features
 - Time constraints were a major decision factor in which features to implement and which to not work on

3. External Interface Requirements

3.1 User Interfaces

Welcome Page



- Automatically the first page shown when the application is open
- Navigation Bar at the top offers options to be shown the welcome page, login-in, register or about
 - Every navigation bar option will turn a pale pink shade when the cursor hovers over it

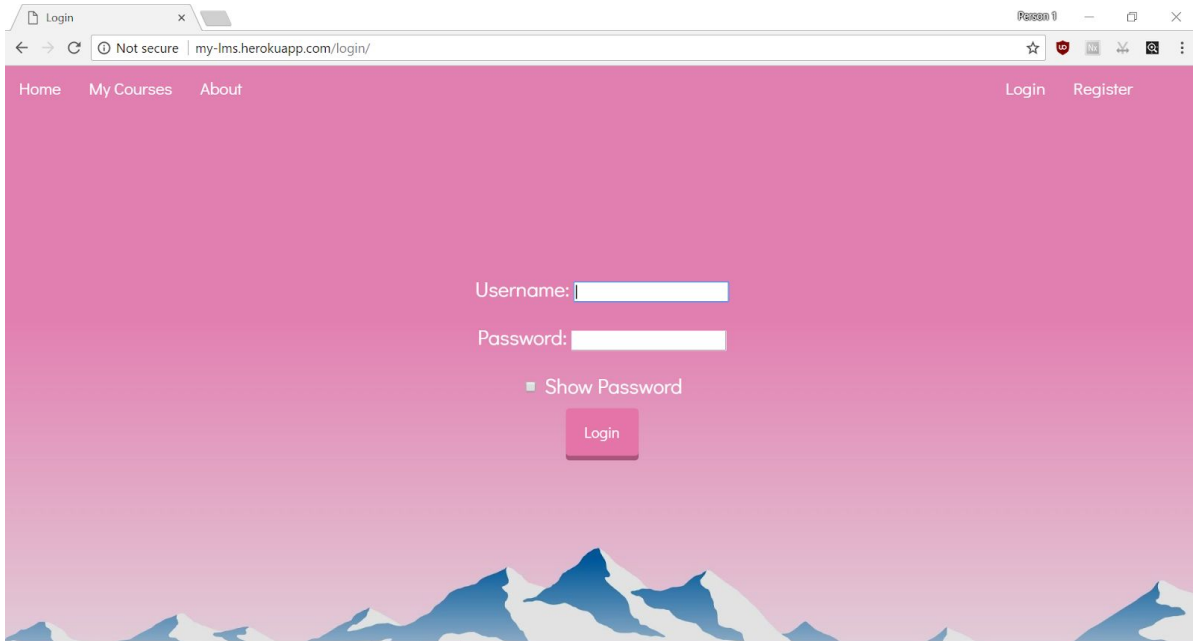
Registration Page

A screenshot of a web browser showing the 'Registration Page' of an application. The browser's address bar displays 'my-lms.herokuapp.com/register/'. The page has a pink header with navigation links: 'Home', 'My Courses', 'About', 'Login', and 'Register'. The main content area is pink with a registration form. The form fields are: 'Username:' (with a required field note: 'Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.'), 'First name:', 'Last name:', 'Student id:', 'Email:', and 'Password:'. At the bottom, there is a blue and white mountain range graphic.

- Prompts user to enter their information: username, full name, ID, password and confirmed password
 - Information is stored in database

- All boxes must be filled or else registration will not be fulfilled
 - Page will point out which boxes are not completed

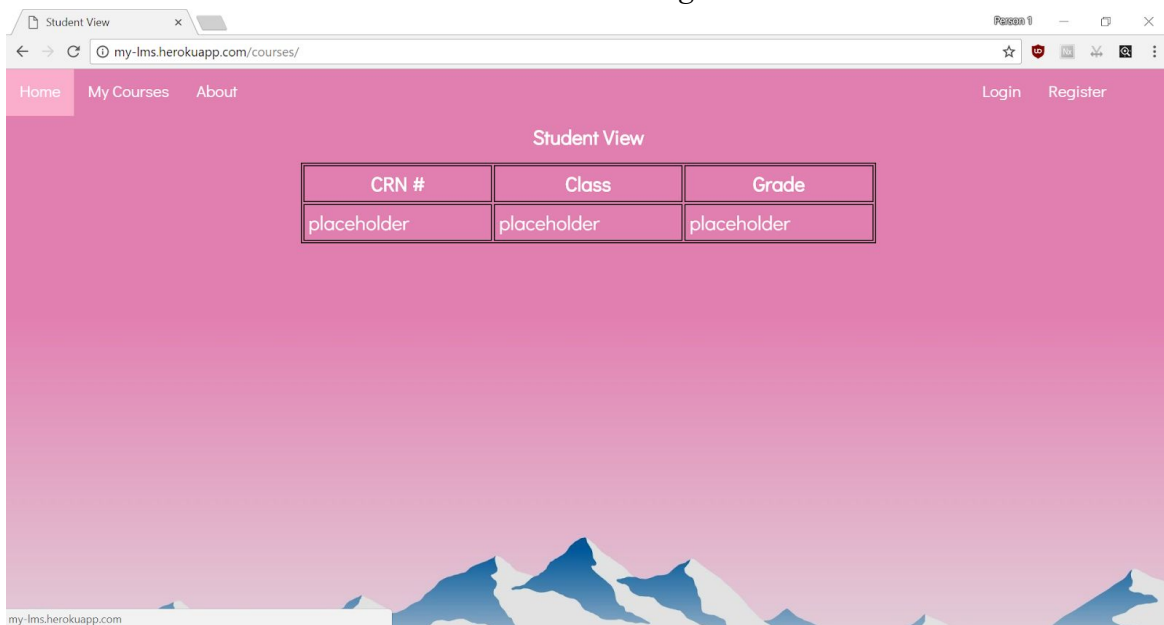
Log-In Page



A screenshot of a web browser showing the login page of 'my-lms.herokuapp.com'. The page has a pink background with a mountain range illustration at the bottom. The navigation bar includes 'Home', 'My Courses', 'About', 'Login', and 'Register'. The login form consists of two input fields: 'Username:' and 'Password:'. Below the password field is a checkbox labeled 'Show Password' and a pink 'Login' button.

- Prompts user to enter username and password to enter account whether a student or faculty
- Log-In button will only function is both username and password boxes are filled out
 - Otherwise, it will prompt user to fill the respective box

Student View Page



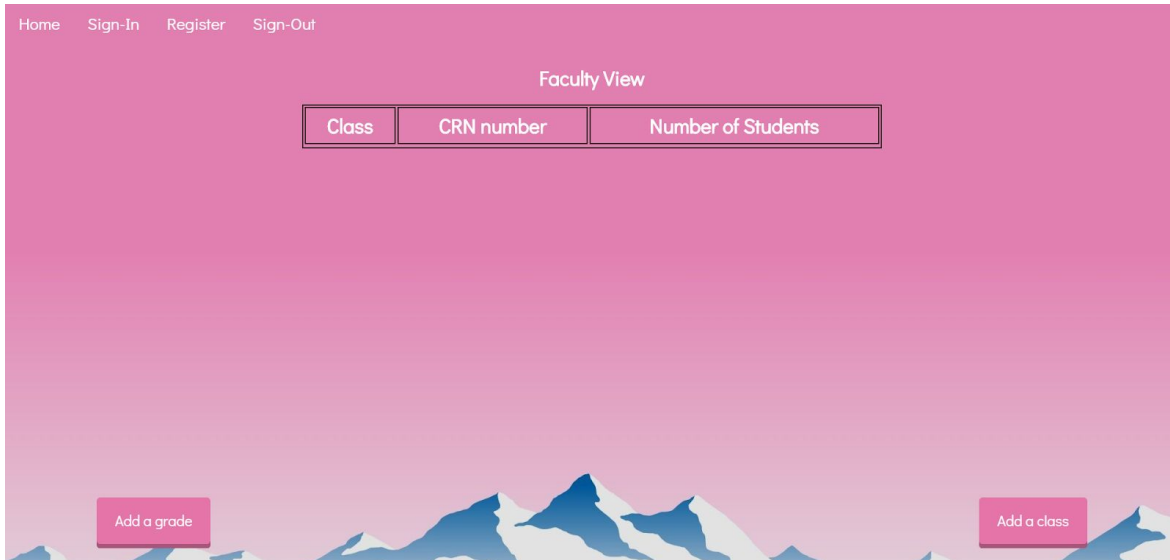
A screenshot of a web browser showing the 'Student View' page of 'my-lms.herokuapp.com'. The page has a pink background with a mountain range illustration at the bottom. The navigation bar includes 'Home', 'My Courses', 'About', 'Login', and 'Register'. The 'Student View' section contains a table with three columns: 'CRN #', 'Class', and 'Grade'. Each column has a placeholder value.

CRN #	Class	Grade
placeholder	placeholder	placeholder

- Student-created accounts will be able to view:

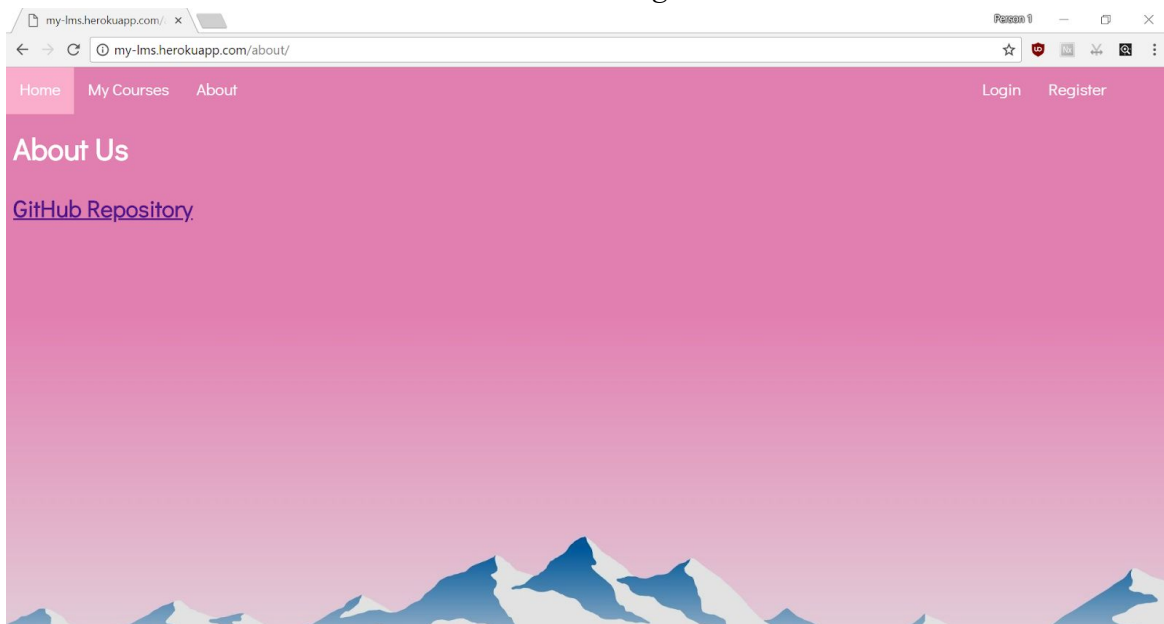
- Enrolled course CRN
- Enrolled course names
- Current grade in each course

Faculty View



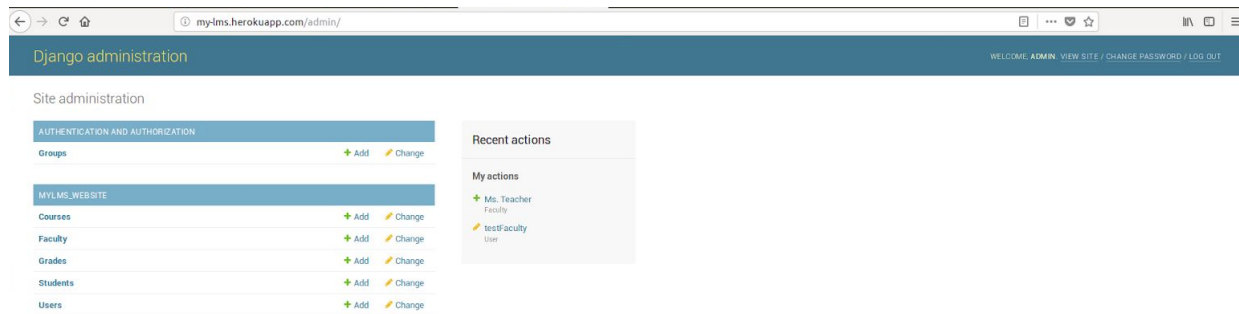
- Faculty-created accounts will be able to view:
 - Enrolled course CRN
 - Enrolled course names
 - Number of students in each course
- Add-A-Grade button allows faculty to enter a grade for a course
- Add-A-Class button allows faculty to create a new course

About-us Page



- Shows link to the GitHub Repository where all files needed to create the LMS are located

Administrator View



- Links to modify courses, faculty, students, grades and users overall

3.2 Software Interfaces

- Outgoing data from the LMS include created courses, added in grades and student information sent by the registered users to the Django server framework
- Incoming data from the server include any updates to software, total users, courses and grades

3.3 Communications Interfaces

The LMS application has a Python-coded server created in the Django platform that is web-based that is connected to the LMS application on Heroku. Also used to store the information is a PostgreSQL database that will store student information, faculty information, courses and grades.

All database hosted on Heroku are required to use a Secure Socket Layer (SSL). Furthermore, as far as security, all user passwords are hashed to prevent decryption.

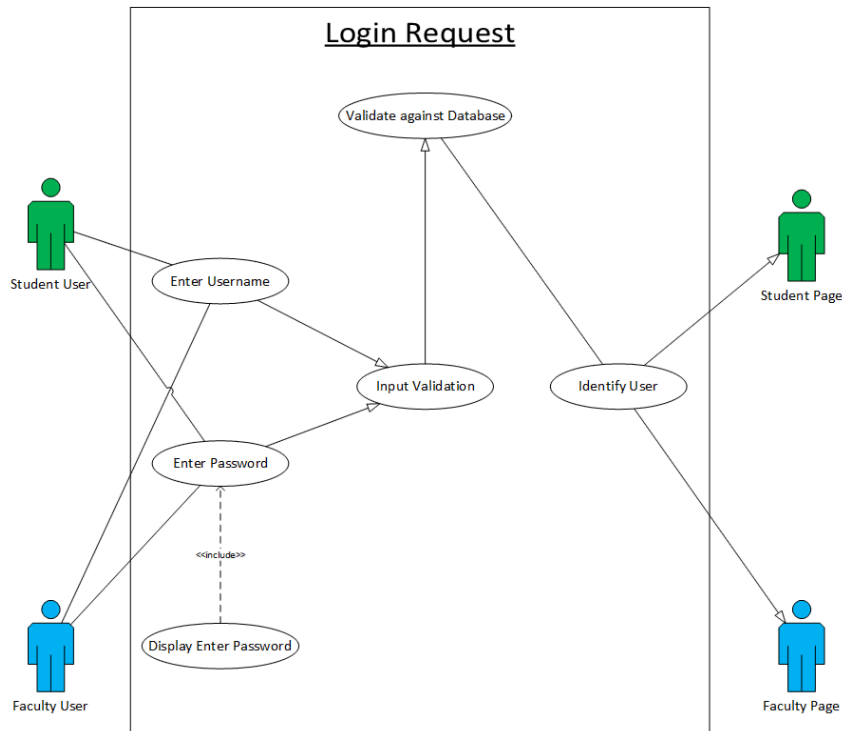
4. System Features

4.1 Login Request

4.1.1 Description and Priority

The system must let user to input there login information (username & password) to further access the LMS. For this to be completed the user must have already register in the system. Once login the user will be presented to the home page. The priority of the feature is high.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

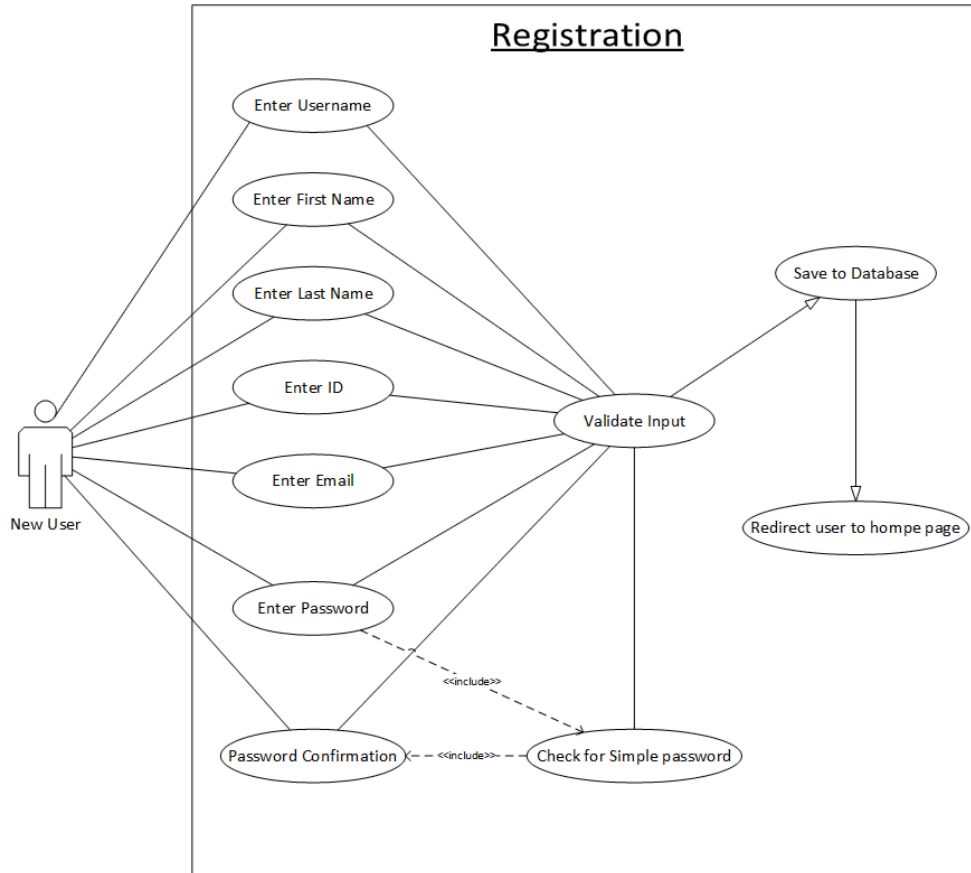
- REQ-1: User should be able to submit their login information
- REQ-2: Login Information must be validated against the login information store in the data based.
- REQ-3: Databased should identify user & determine if user is faculty or student
- REQ-4: If login information is incorrect the user will be prompted with error message
- REQ-5: User input password should not be visible by default
- REQ-6: User can toggle to see their inputted password between visible and invisible
- REQ-7: Once user is login information is validate transport user page to main page

4.2 Registration

4.1.1 Description and Priority

The system should allow new users to enter to the platform. This will be conducted by the integration of a registration page for which new users can input their information into the site to create an account. From this a user account will be created giving access to the site to the new user. Since the system needs users to fulfill other applications therefore leads this feature to be on high priority. As users' accounts must be added to the database as well as access the rest of the site.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

- REQ-1: User should be able to submit their information such as: Name, School ID email and password
- REQ-2: User should be able to create a username with special character and numbers
- REQ-3: Password should be validated to not be simple (commonly used passwords & numerical only passwords)
- REQ-4: Password should be validated to be longer than 8 characters
- REQ-5: Password should be validated to not have any personal information inputted from the other text boxes (no name, no id and email on password)
- REQ-6: Password must be confirmed from the user before submission.
- REQ-7: Upon submission system should identify empty input entry.
- REQ-8: If all entry are complete and satisfy other requirements the data should be stored

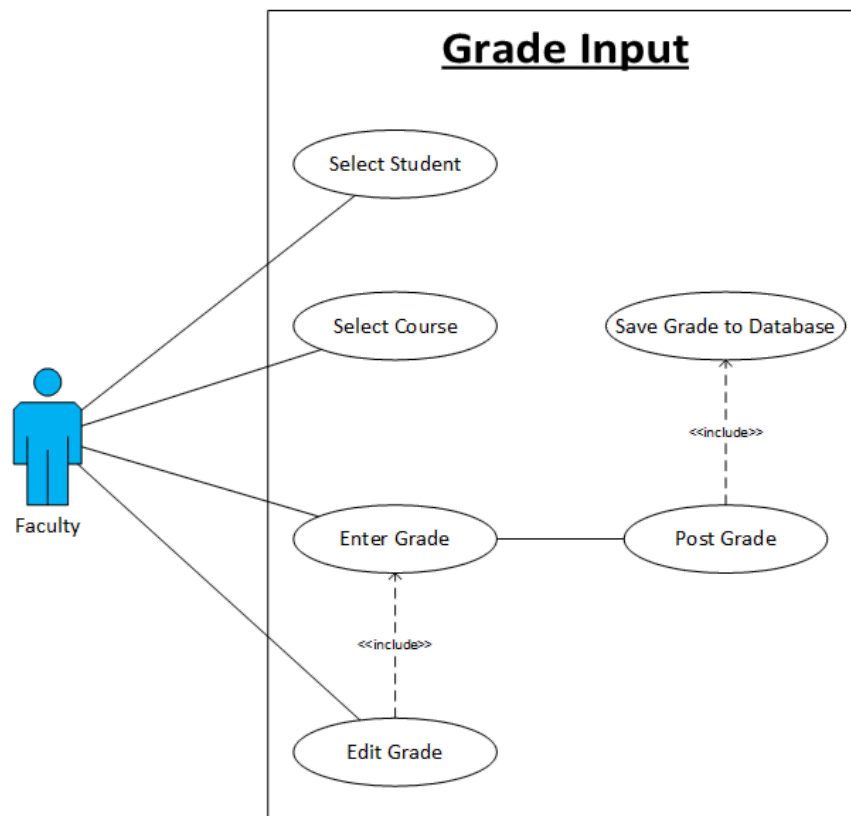
REQ-9: If submission is success redirect user to home page

4.3 Grade Input

4.1.1 Description and Priority

When a faculty users has log in to the system, they will have the ability to add grade to students users. The faculty user will select a class and a student for which they can input the grades. Once the grades have been “posted” the system will save them in the database for which they can be later modify or view by a student in there page. While this is an important part of the LMS core functionality, for the system to be functional it is not project crucial thus this feature will be denoted as medium or low priority. Once the base system is fully functional and live this will be change to a high priority.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

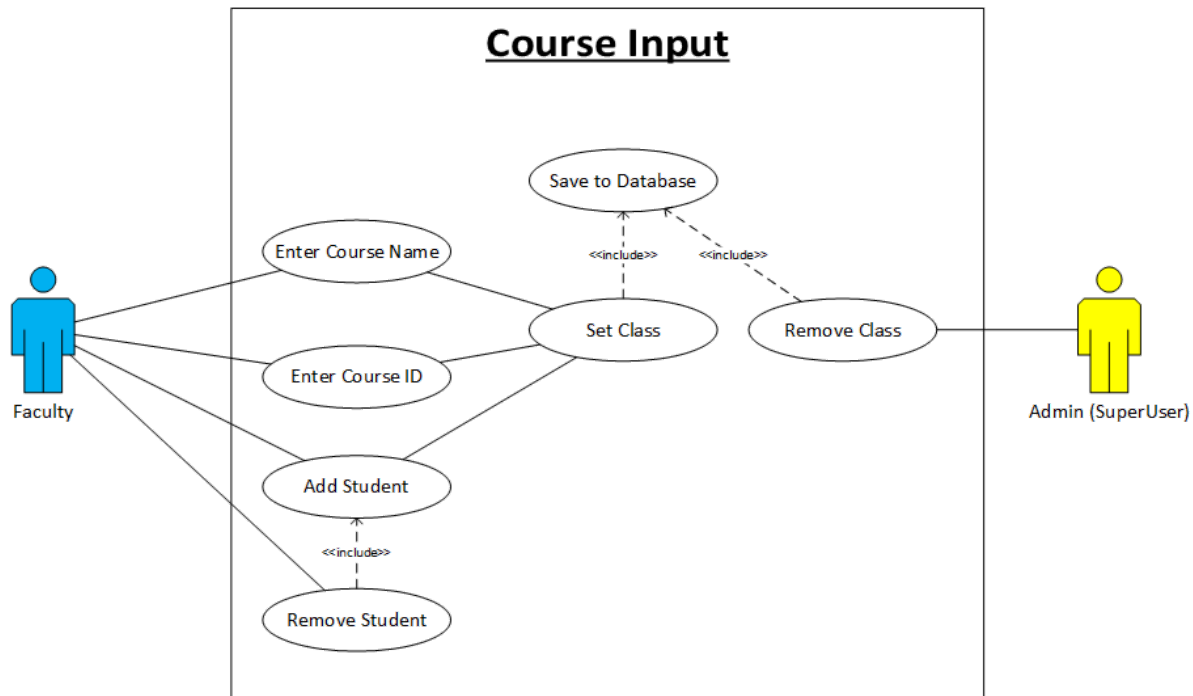
- REQ-1: Grade can only be inputted by faculty
- REQ-2: Grade will be specific to individual student in specific course
- REQ-3: Grade will be saved in the database
- REQ-4: Once grade is saved the faculty should be able to edit on a later date

4.3 Course Input

4.1.1 Description and Priority

When a faculty users is login in to the system, the user will have the ability to add course to themselves. Through which each course will have a course id as well as a name. After which the faculty can add/remove users to their course. Once the course is set, the database will be updated and student users that have been added will be be updated with there new course. The only time instance that a course can be remove is by the power of an site administrator/superuser. This super user can remove the course for which will delete the information from the database.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

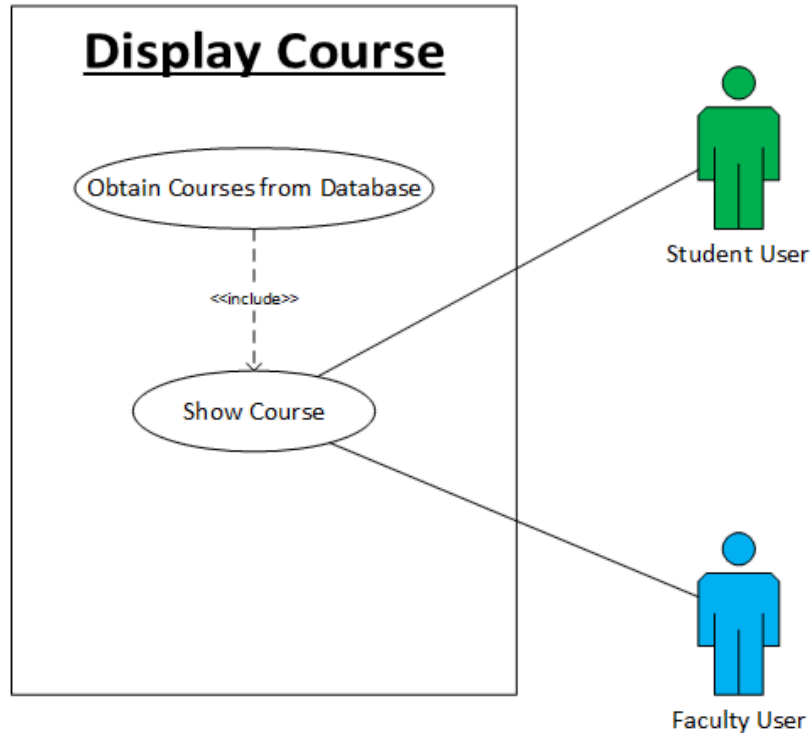
- REQ-1: Courses can only be inputted by the faculty
- REQ-2: Courses will have a specific course ID
- REQ-3: Course will have students enrolled to it
- REQ-4: Course can only be removed by a super user
- REQ-5: Faculty should be able to add student to the course

4.3 Display Courses

4.1.1 Description and Priority

Each user can see the courses they are attending/teaching in there respective page. All information will be obtain from the system database and will be display to each individual user. Since this a miniscule task to accomplish, the priority level on this feature is low as the implementation of this feature is not system critical to achieve a stable system.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

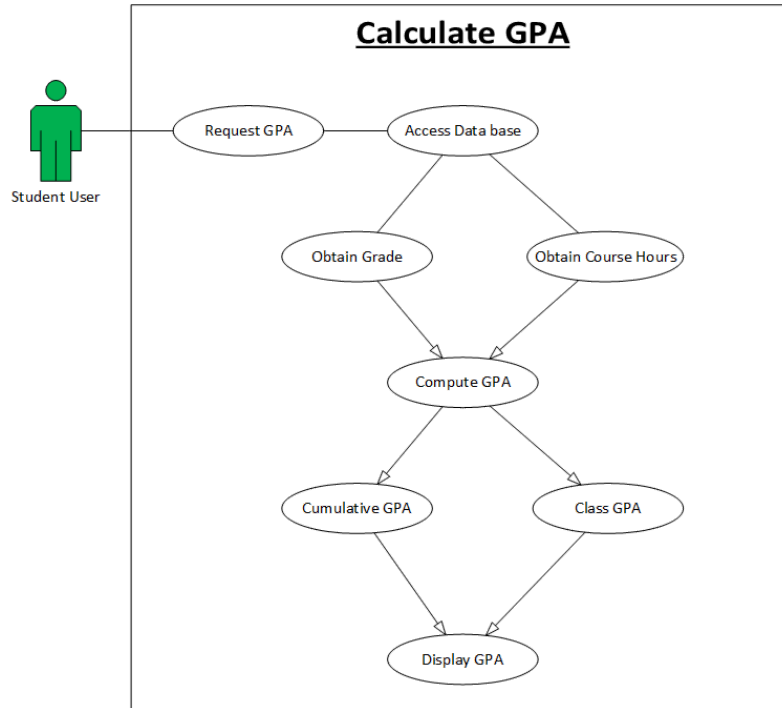
- REQ-1: Students will be able to see there courses enrolled
- REQ-2: Faculty should be able to see their courses they are teaching

4.3 Calculate GPA

4.1.1 Description and Priority

Once a student user is log in to the system they can see their courses they are attending as well as there grade (if submitted/posted by a faculty users). Once a grade are inputted the system will update the Grade Point Average (GPA). With all the grade inputted by faculty users as well as each course GPA has been calculated, the system will perform a cumulative GPA calculation to showcase the users. As this feature needs of input of grade to each student users feature, which has been denoted as a medium priority, then the priority of this feature will be low as will need the implementation of other system to conduct this feature.

4.1.2 Stimulus/Response Sequences



4.1.3 Functional Requirements

REQ-1: System should calculate student GPA depending on the grade and course taken

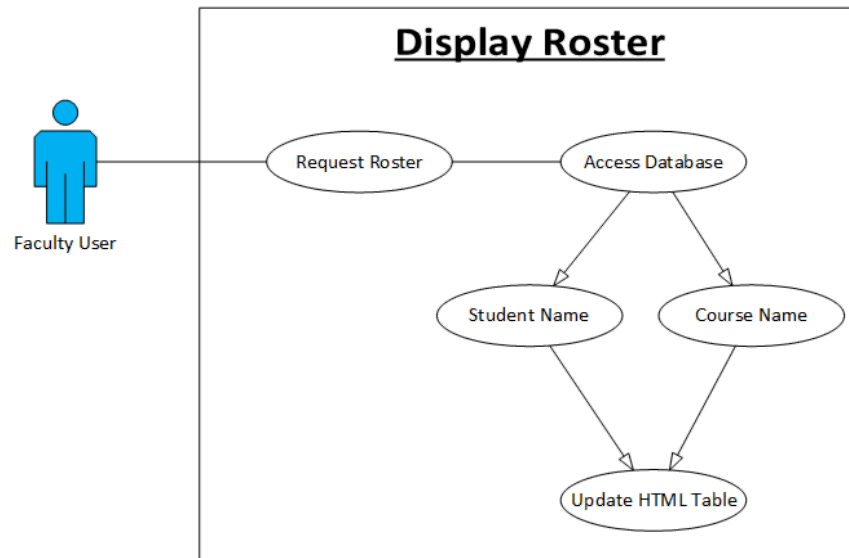
REQ-2: GPA will be evaluated per class basis as cumulative basis

4.3 Display Roster

4.1.1 Description and Priority

Once a faculty user is log in to the system as well as have added students to their course they will have the ability to see their students roster for each course. With this ability the faculty user can easily see their students in there class for which will be beneficial for other feature in the system. As this is a quality of life improvement for the faculty user the priority of this feature is low. Due to not being system crucial to have a stable build of the system

4.1.2 Stimulus/Response Sequences

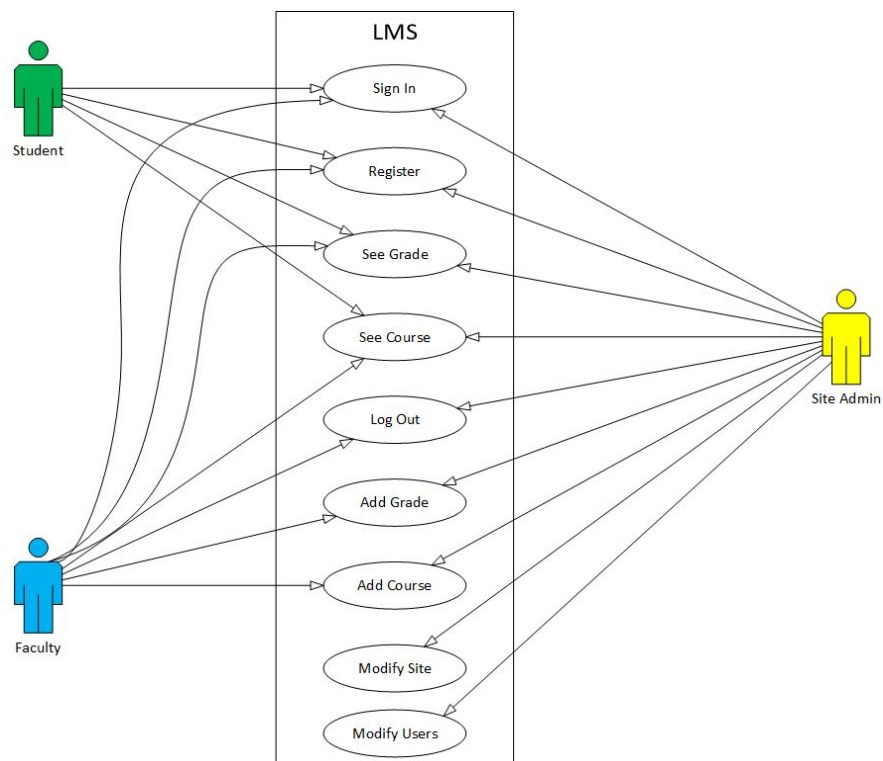


4.1.3 Functional Requirements

- REQ-1: Each faculty should be able see their roster per class
- REQ-2: System should be able to gather students from the database

4.3 Online Access

4.1.1 Description and Priority



The whole system will be accessible to all user via the internet access. From which all other subsystem features will be accessible. Any users can access the system through the uses of an

internet browser, therefore the system should be compatible with any web browser. Since the only point of access of the system is through online, this feature is system crucial as without it any subsequential system will be not accessible for which denote this feature as a critical high priority.

4.1.2 Stimulus/Response Sequences

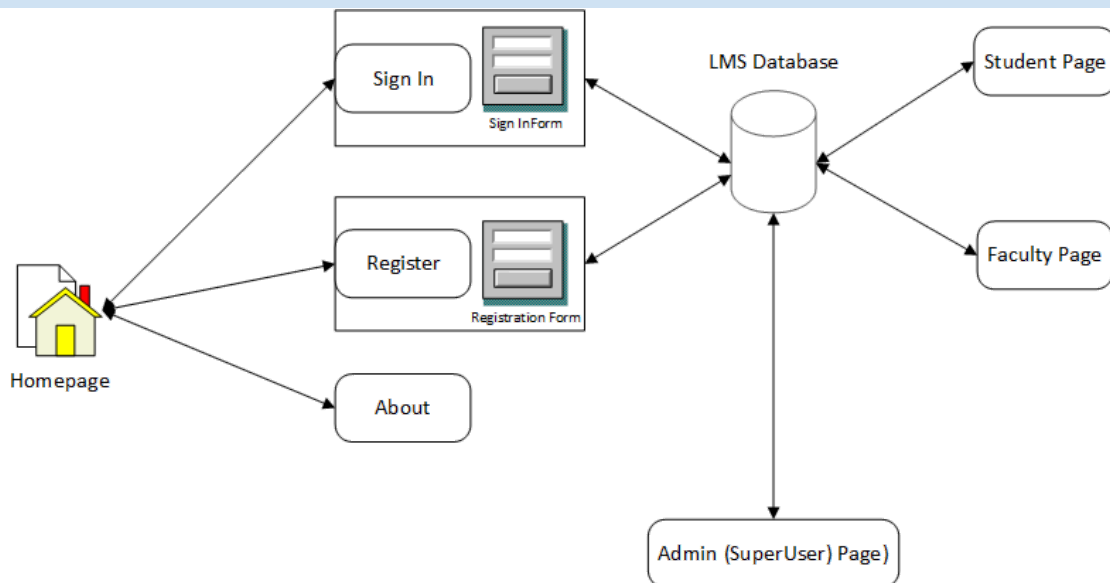
4.1.3 Functional Requirements

REQ-1: System should be easily accessible through online access

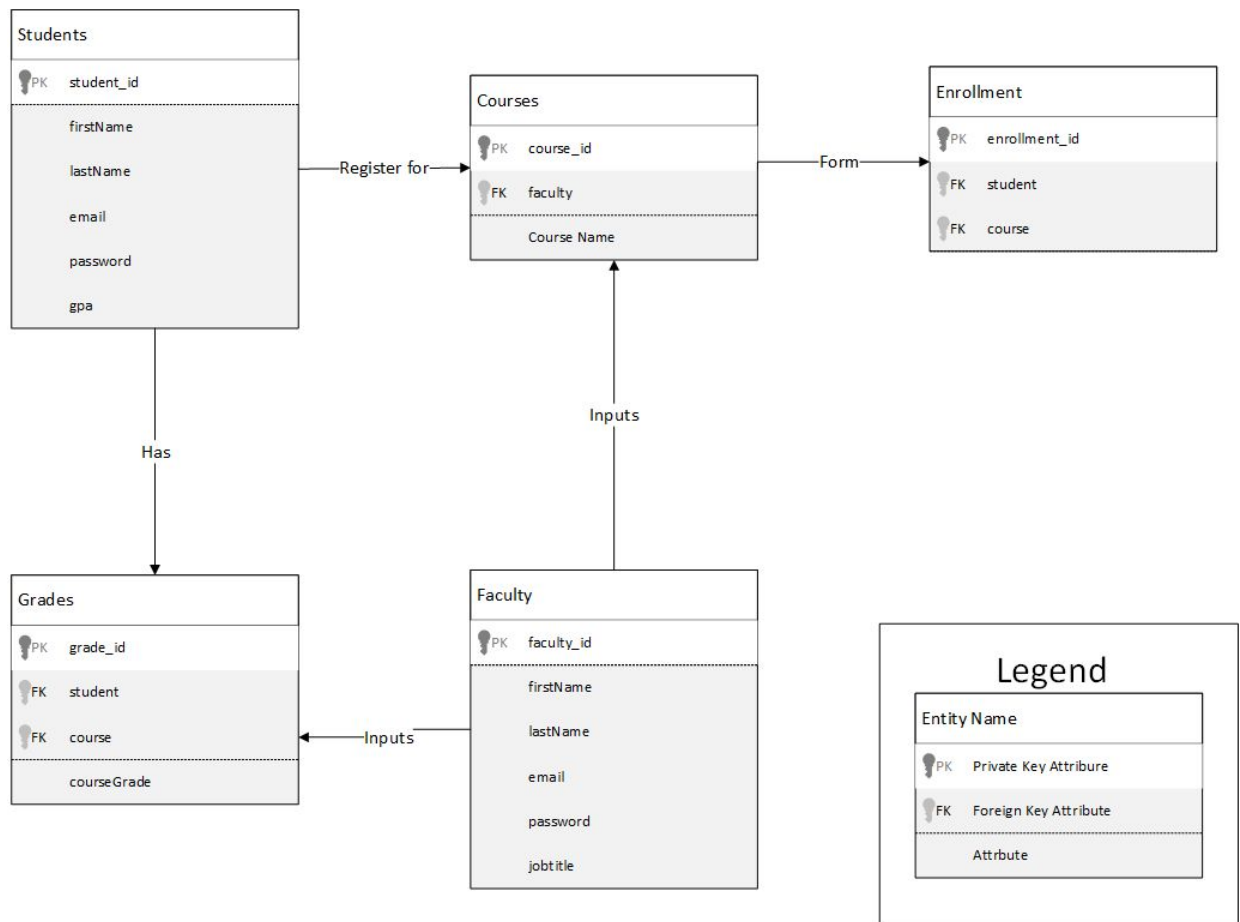
REQ-2: System should work on any internet browser

REQ-3: Users must access all subsequential web pages of the system

5. Other Requirements



The system has been created using the django web framework. This has been done as each webpage is denoted as a view that can be access by the overall system. While the overall system being denoted as an app leaves the possibility to integrate more feature at a later point in time. Using this framework lead us to an easier experience as the django will auto generate the SQL code by given the table information and thus ensuring that the database is up to date. The ability to query items to the database is by the use of forms for which can input and retrieve data from the database. While expendability can easy be done using this framework automatic testist can be created to ensure system is up to date and modification can be made live.



The database consists of four core tables that store the necessary information for the learning management system to function. The tables are named students, faculty, courses, and grades. The students table contains variables that holds the user name, student id and their gpa, with the student id being used to uniquely identify each student. The faculty table stores the faculty members name and login information and the faculty id which is used to uniquely identify the faculty member. The courses table holds the data on all available courses and has a foreign key reference to the faculty member that is teaching each specific course. The final table, the grades table associates the students and courses table through the use of foreign keys in order to track a students grade in each of their courses.

Appendix A: Glossary

Database:

All stored information entered into the learning management system.

Django:

Python-based framework

Learning Management System (LMS):

Software used by organizations to manage educational materials or tools

Heroku:

Cloud platform used to host applications of different languages

Software Requirements Specification:

Documentation used to specify a project's requirements and functionalities

User:

Any individual interacting with the LMS