**Software Design Specification Document**

**(CS360)**

**Smart School Management System**



Group Number: 6

|  |
| --- |
| Ahmed Anwar |
| Ammar Anwar |
| Usman Tariq |
| Muhammad Abdullah |
| Syed Arham Noman |

|  |  |
| --- | --- |
| Course: Software Engineering CS360  Instructor: Suleman Shahid  University: Lahore University of Management Sciences (LUMS) |  |

**Version: 1.0**

**Date: (29/03/2018)**

**Number of hours spent on this document:** 100

|  |  |
| --- | --- |
|  |  |

Contents

Contents ii

1 Change Log iii

1.1 Change log iii

2 Introduction 1

2.1 Document Purpose 1

2.2 Product Scope 1

2.3 Intended Audience and Document Overview 1

2.4 Definitions, Acronyms and Abbreviations 2

2.5 References and Acknowledgments 2

3 Overall Description 3

3.1 System overview 3

3.2 System constraints 3

3.3 Architectural strategies 5

4 System Architecture 7

4.1 System Architecture 7

4.2 Subsystem Architecture 10

4.3 Database Model 25

4.4 External Interface Requirements 27

5 User Interface Design 28

5.1 Description of the user interface 28

5.2 Information architecture 29

5.3 Screens 30

5.4 User interface design rules 60

6 Other Non-functional Requirements 61

6.1 Performance Requirements 61

6.2 Safety and Security Requirements 61

Legal Requirements 61

6.3 Software Quality Attributes 62

Appendix A - Group Log 63

Appendix B – Contribution Statement 64

# Change Log

## Change log

* We have added the option of admin to add and delete faculty
* Added faculty’s forget password functionality where faculty can change the password if he/she have forgotten the password by writing the email address and getting the password changing form.
* We are removing the functionality of adding graphs for showing students’ academic performances.

# Introduction

## Document Purpose

The Software Design Specification is made with the purpose of outlining the software architecture and design of the Smart School Management System in detail. The document will provide developers with insight in meeting client’s needs efficiently and effectively. It will cover all the systems and subsystems and components involved alongside their functionalities. Moreover, the document facilitates communication and better understanding of the system by providing visual representations of the system design and architecture.

## Product Scope

Smart School management system is web-based system that aims to assist school administration and faculty in efficiently managing the students enrolled. It will allow greater control of the administration which will improve accountability. Secondly, it will provide an interface to the instructors to efficiently manage the student academic records. Furthermore, the system will make the management more efficient by reducing the amount of paperwork and manual calculations. Lastly, it will allow faster and transparent transmission of student’s academic results to their respective parents.

## Intended Audience and Document Overview

This document is intended towards developers of Group 6 for assistance during the development stage. In addition, it is also oriented towards the teaching assistants and instructor for the Software Engineering Course CS 360 at Lahore University of Management Sciences.

The Software design document would demonstrate how the design will accomplish the functional and non- functional requirements captured in the Software Requirement specification (SRS). The document will provide a framework to the programmers through describing the high-level components and architecture, sub systems, interfaces, database design. This is achieved using architectural patterns, design patterns, sequence diagrams, class diagrams, relational models and user interfaces.

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
|  |  |
| Term | **Description** |
| Architectural Design | high level structures of a software system and the discipline of creating such structures and systems |
| Database | A collection of stored related data |
| Sequence Diagram | An interaction diagram that shows how process interact with one another and in what order |
| Activity Diagram | An activity diagram visually presents a series of actions or flow of control in a system |
| Database Scheme | The organization of data as a blueprint of how the database is constructed i.e. divided into database tables in the case of relational databases |
| TCP | Transmission Control Protocol |
| GUI | Graphical User Interface |
| LAN | Local Area Network |
| Load time | Time taken to deliver request |
| Coupling | Measure of degree of interdependencies between software modules |

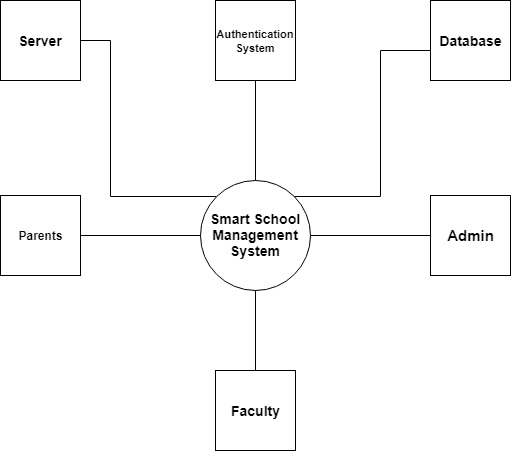
## References and Acknowledgments

* [GeeksforGeeks](https://www.geeksforgeeks.org/)
* [Draw.io](https://www.draw.io/)
* [StarUML](http://staruml.io/)
* [Balsamiq](https://balsamiq.com/)
* [IEEE Privacy Requirements: Present & Future](http://delivery.acm.org/10.1145/3110000/3103221/p13-anthonysamy.pdf?ip=221.120.220.11&id=3103221&acc=ACTIVE%20SERVICE&key=55CAB23EF39677E3%2EFE8F10057D3A2BA3%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1553537422_a9e00c7d83356bf612416a3e0e2af88c)
* [Prevention of Electronic Crimes Act 2016](https://moitt.gov.pk/userfiles1/file/PERSONAL-DATA-PROTECTION-BILL-July-18-Draft.pdf)
* [ISO 25010](https://iso25000.com/index.php/en/iso-25000-standards/iso-25010?limit=3&limitstart=0)
* [ISO 9241](https://en.wikipedia.org/wiki/ISO_9241)

# Overall Description

## System overview

The context diagram shows the external stakeholders that interact with the system.



## System constraints

### Hardware or software environment

The system will be only available to the user via a web-based interface in its initial stage. Therefore, it will not be optimized for using via a mobile phone browser. In addition, reliable internet connection is also a required for stable and smooth navigation throughout the system. However, frequent connectivity issues in Pakistan pertains which may impact the stability and accessibility of the system. Furthermore, the system should be able to run optimally on all popular web browsers like Chrome, Firefox, Opera etc.

### End-user environment

The end users are the school administration and the faculty. Both these user types are assumed to have enough but minimal knowledge of using a web-based interface. However, since, they will be shifting from traditional paper-based system to an online one, so they will need time to familiarize themselves with the new system. Therefore, the system design should be intuitive without posing additional challenges to the end user.

### Availability or volatility of resources

The database will be hosted on Heroku. Any maintenance/crash on the service provider’s end will severely impact the software, create additional recovery issues and will not enable to use the software.

### Network Communications

According to Cisco, the network requirements can be measured in terms of response time, throughput, and reliability. The software’s response time shall lie between 0.1-1 seconds. Since, the number of users is limited, thus not a lot of high-volume traffic is expected on the system which entails a lower throughput. However, reliability is very important i.e. the network should not experience frequent downtimes which can disrupt the result transmission process and create a bad user experience.

### Security requirements

The system should abide by the privacy standards set up by the Prevention of Electronic Crimes Act of 2016 of Pakistan. Thus, user’s data must be secured prevented from any malware attacks to ensure complete user privacy. Thus, abiding by the laws entailing this issue.

### Interoperability Requirements

The software shall be able to support expansion the expansion of the existing system in the future. This includes inclusion of additional stakeholders i.e. students, bank for fee collection and other system functionalities. Other users might be using different hardware or software interface which might create communication barriers between new and existing systems. Thus, to maintain coherency between different systems and keeping costs low, the system should be highly interoperable across different platforms without any restrictions.

### Standards Compliance

To ensure the delivery of highest quality of software, the system should meet the ISO standards for quality attributes like ISO 25010 which highlights the software quality attributes including efficiency, usability, portability etc. which have become the industry standards for gauging software quality.

### Data Repository and distribution

The data should only be accessible to user according to its privilege level. For instance, faculty can only see the academic information of the student and not his personal information. In addition, the database should provide range of query interfaces to accommodate various search behaviours and support multiple access points to find the data. For efficient information retrieval, it should identify and aggregate metadata records that describe the same object. Furthermore, it should also follow API search standards for interoperability.

## Architectural strategies

### Object Oriented methods

The software will be using Object Oriented software development methods as it is easier to identify relationships and communicate information between different objects. In addition, due to its modular structure, it is easier and quick to troubleshoot which will improve overall software maintainability. Furthermore, it allows reuse of code by inheritance which saves space and time.

### MVC Architectural Pattern

The system will be adopting the Model View Control Architectural Pattern as it allows efficient code reuse and parallel development by dividing the logic, user interface and controller into three interconnected parts, thus leading to low coupling. In addition, it leads to high cohesion as it allows logical grouping of related actions on a controller together.

### Layered Architecture

Since, the intended software is a web-application, so a layered architecture will be used alongside MVC. This allows, the software to be scalable and remotely accessible. In addition, it also enhances the security of the system by protecting the database tier. Furthermore, it also enables concurrency and synchronization.

### JavaScript

JavaScript will be used for the development of both front end and back end of the software because of its several benefits. Firstly, due to its client-side execution, it saves bandwidth and makes execution process faster. Secondly, it has a rich user interface interactivity which improves the GUI and improves user experience. In addition, it enhances interoperability as it is nicely integrated with other languages, therefore, can be used in variety of applications. Lastly, it is compatible with most of the modern web browsers

For the front-end development purpose, we will use ReactJS. Due to its component-based approach, it is very simple to learn. It supports the functionality of reusable components which will make the development process faster and flexible at the same time. React also uses virtual DOM which will make the user experience better and our development work faster. We will also be using Redux as it would help us in organizing our code and it would be easier for us to debug and test our code.

We will be using Express framework of node js as our server-side code as express would offer us high functionality along with various advance coding features. It has event-driven features and it also makes writing routing logic easy.

### Data

The system will be using an external database so that we do not have to administer the hardware and maintenance issues for the database. In addition, security and software updates also taken care of. However, the external database is a paid service.

Secondly, as there are not a lot of external systems involved, therefore distributed database will not be used in the system. In addition, DDMS is also very costly.

### Network Communication

The system will comply the network requirements issued by the ISO by using the TCP for network communications. In addition, it is interoperable as it allows cross-platform communication among heterogenous networks. Lastly, it is easily scalable as it enables addition of new networks without disrupting the current services.

### User Interface Paradigms

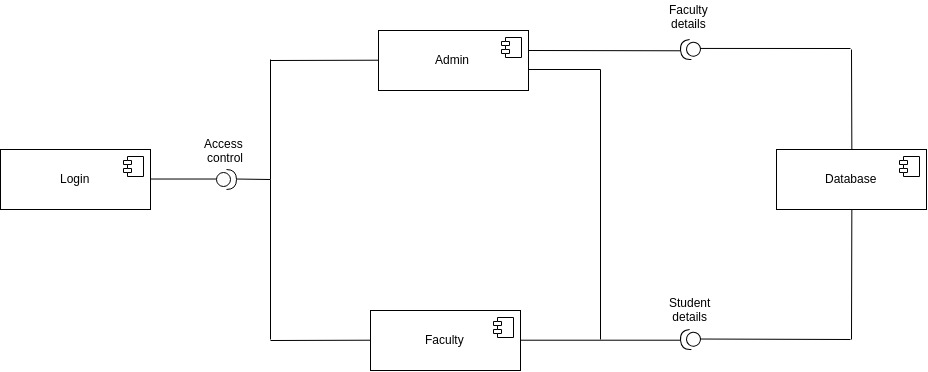
The user interface should have:

* Clarity: the content is displayed quickly and accurately.
* Detectability: user’s attention is directed towards the required information.
* Legibility: information is easy to read.
* Comprehensibility: the meaning is unambiguous and interpretable.
* Conciseness: users should not be overloaded with extraneous information.

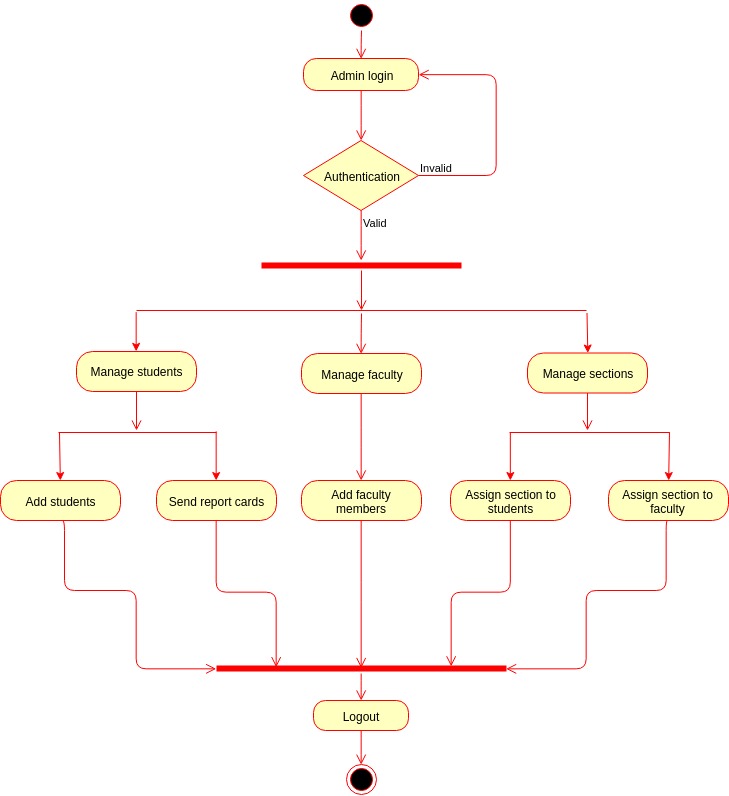
# System Architecture

## System Architecture

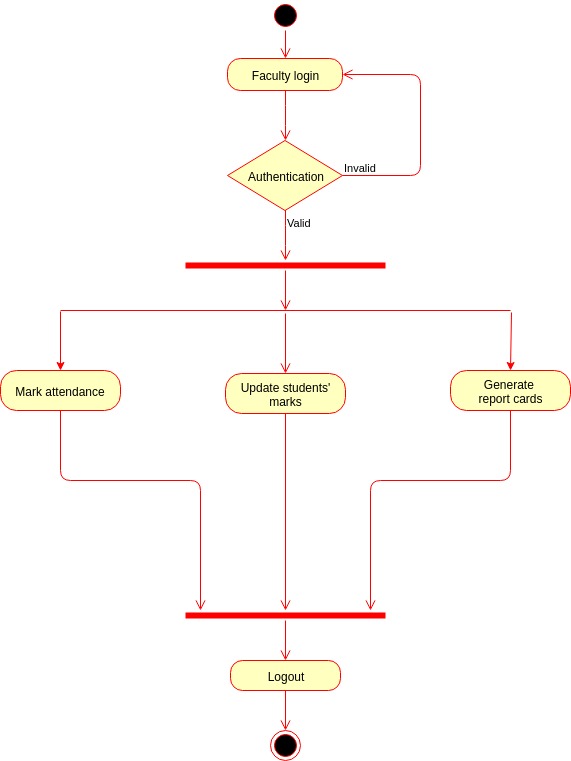
**Component Diagram**



**Activity Diagram Admin**



**Activity Diagram Faculty**

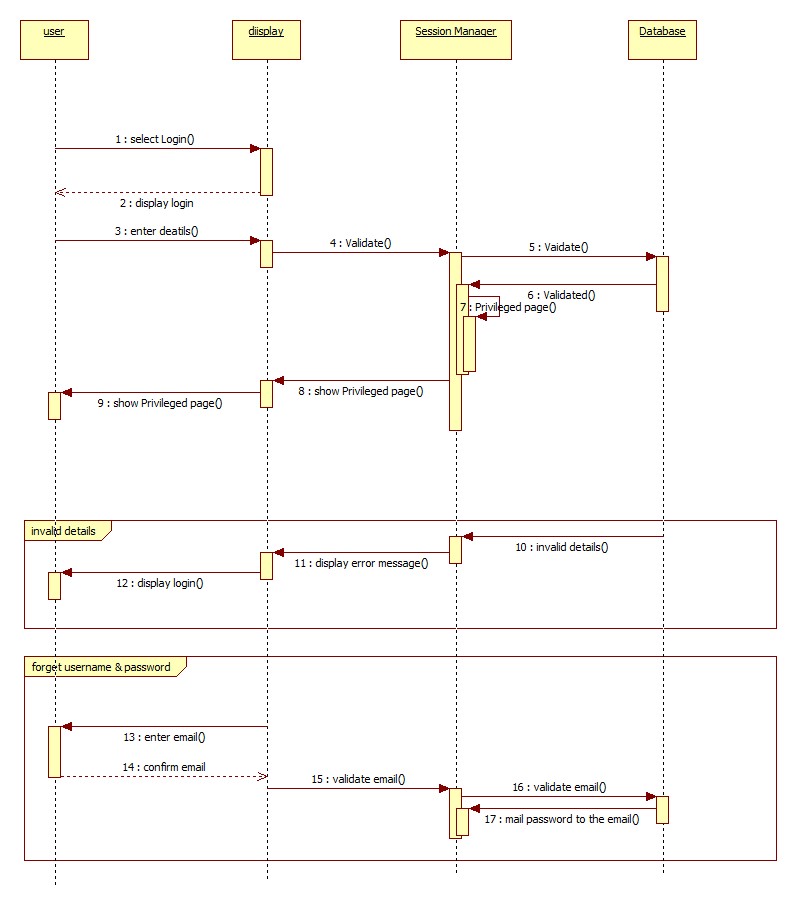


## Subsystem Architecture

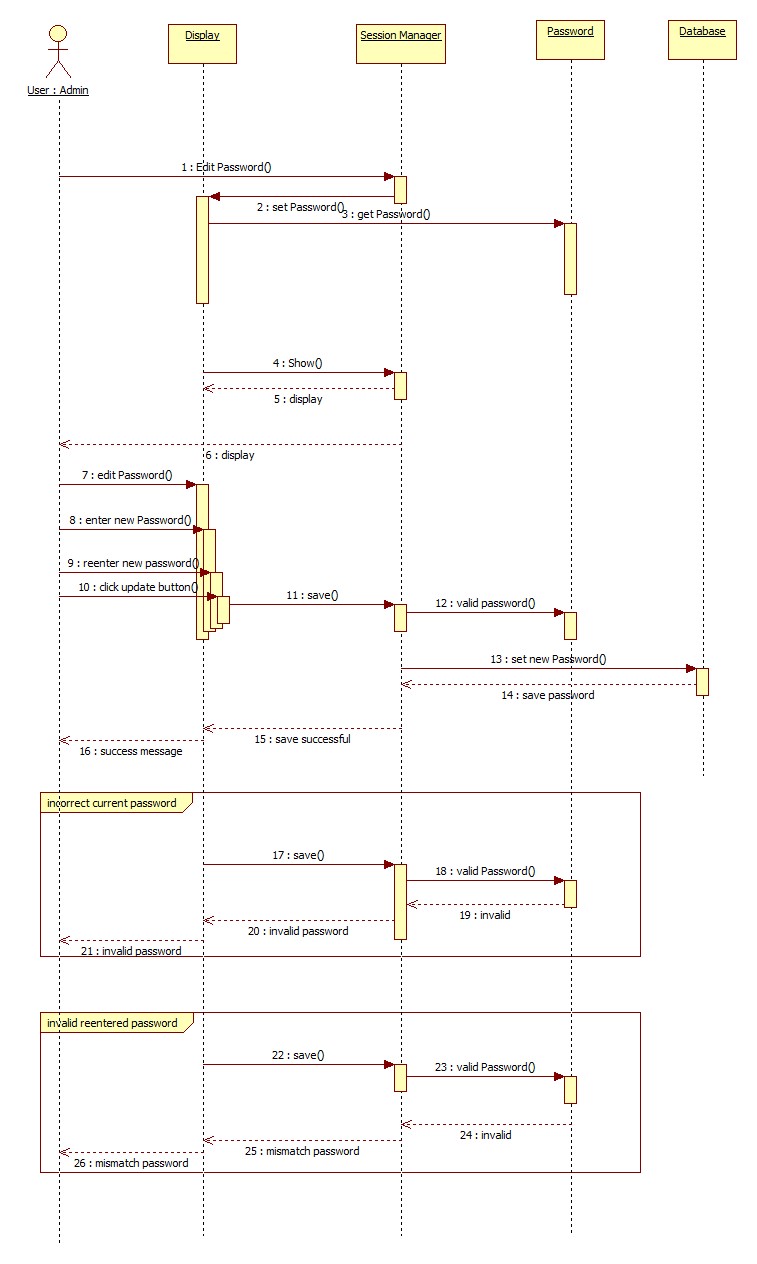
**Class Diagram**

****

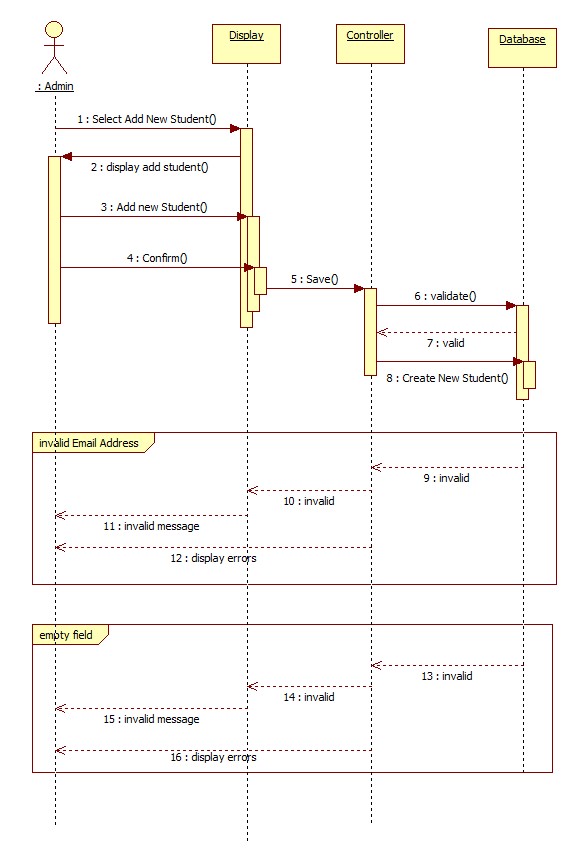
**Sequence Diagram 1: Login**



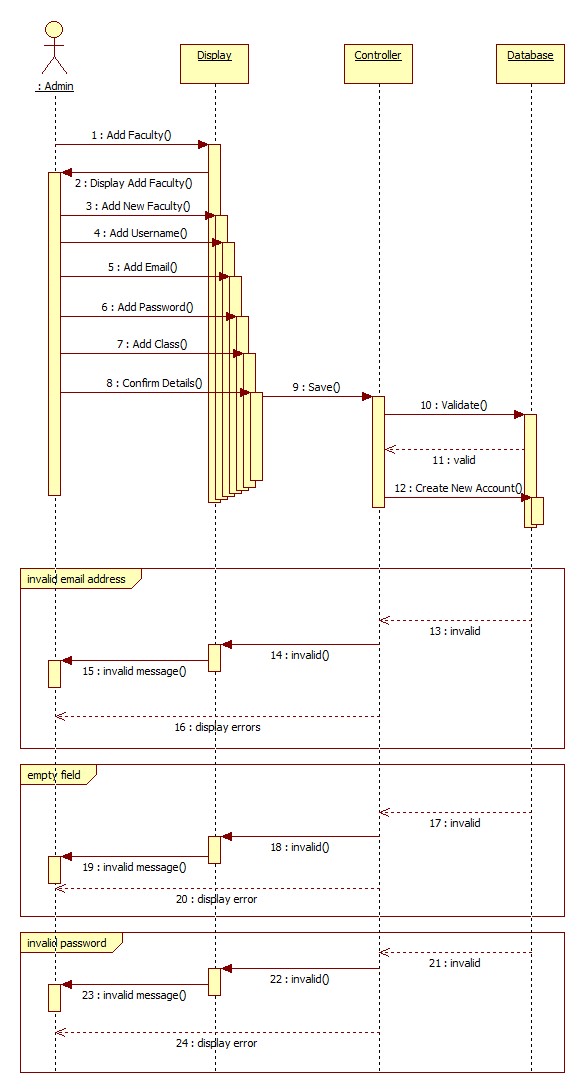
**Sequence Diagram 2: Change Password**

**

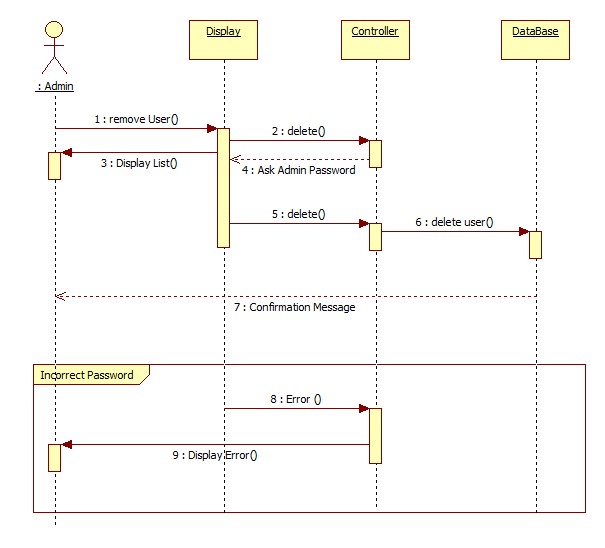
**Sequence Diagram 3: Add Student**



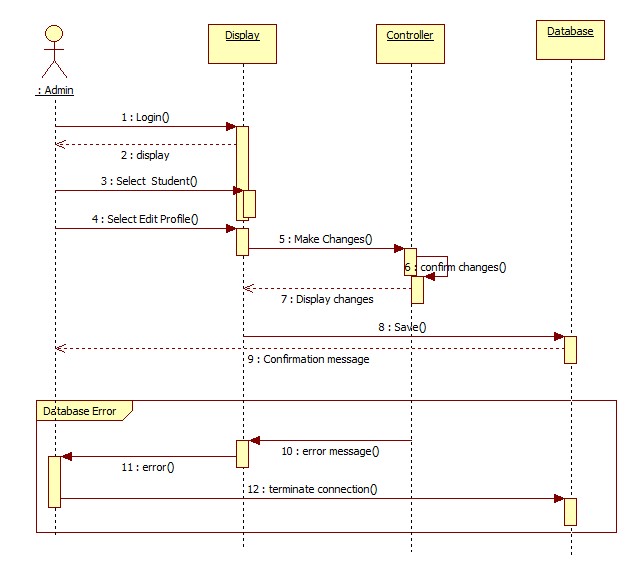
**Sequence Diagram 4: Add Faculty**



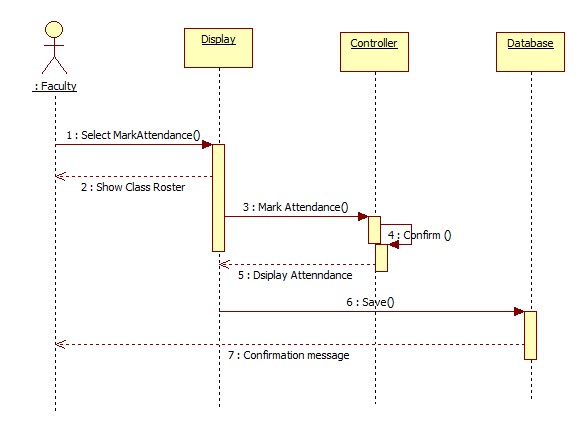
**Sequence Diagram 5: Remove User**

**

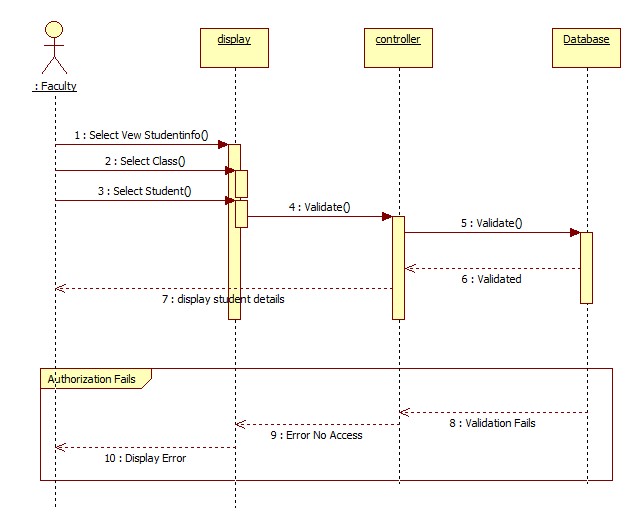
**Sequence Diagram 6: Edit Student Info**

****

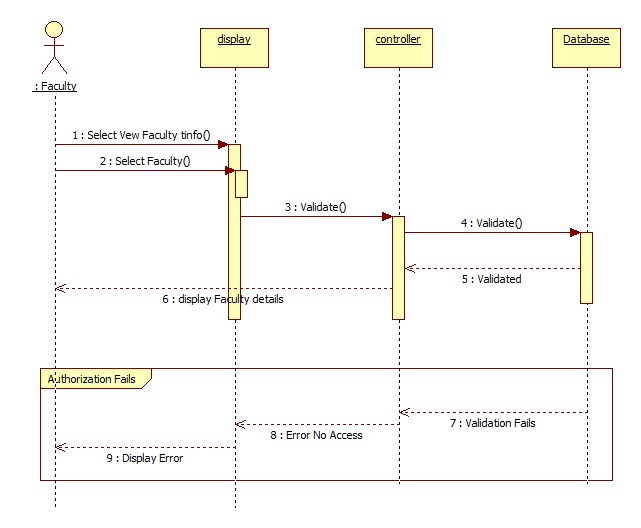
**Sequence Diagram 7: Mark Attendance**

****

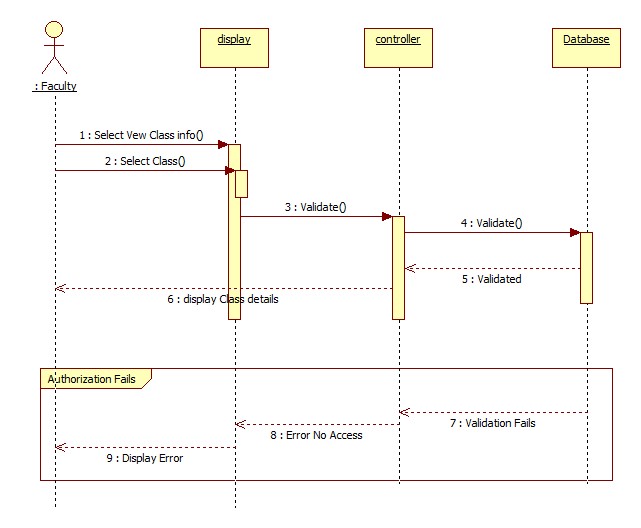
**Sequence Diagram 8: View Student Information**

****

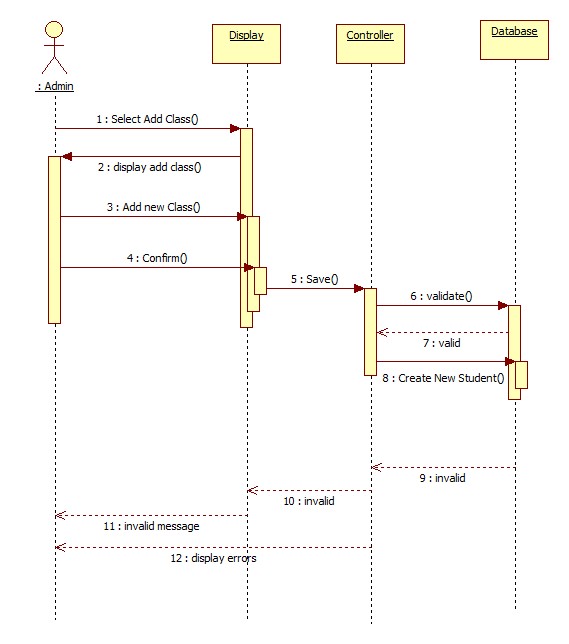
**Sequence Diagram 9: View Faculty**

****

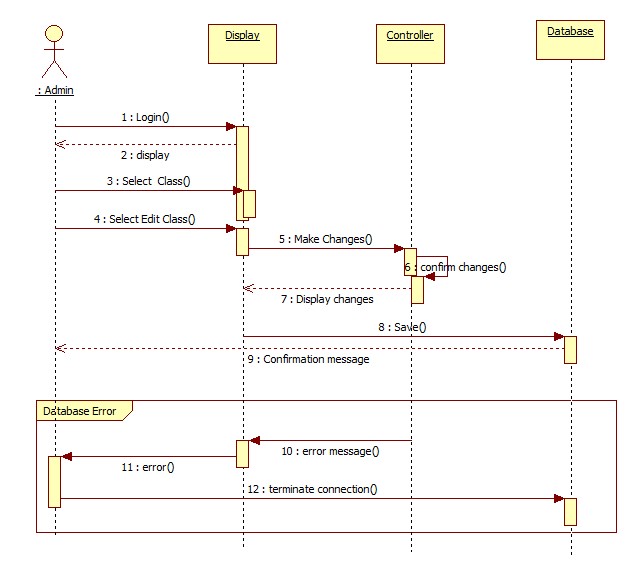
**Sequence Diagram 10: View Class**

****

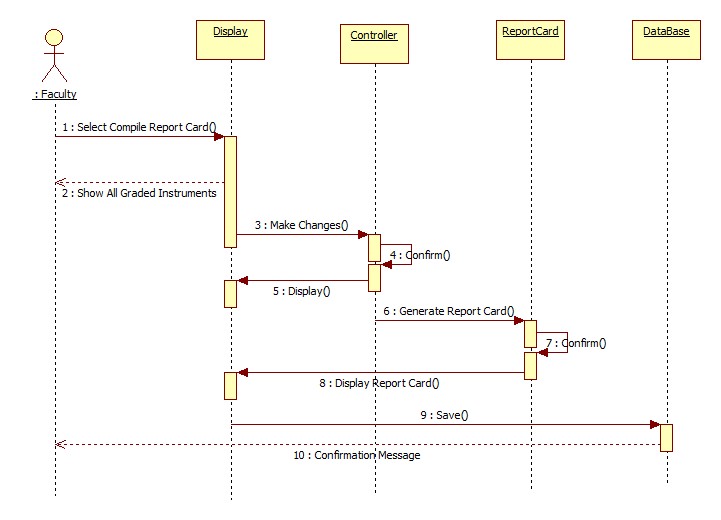
**Sequence Diagram 11: Add Class**

****

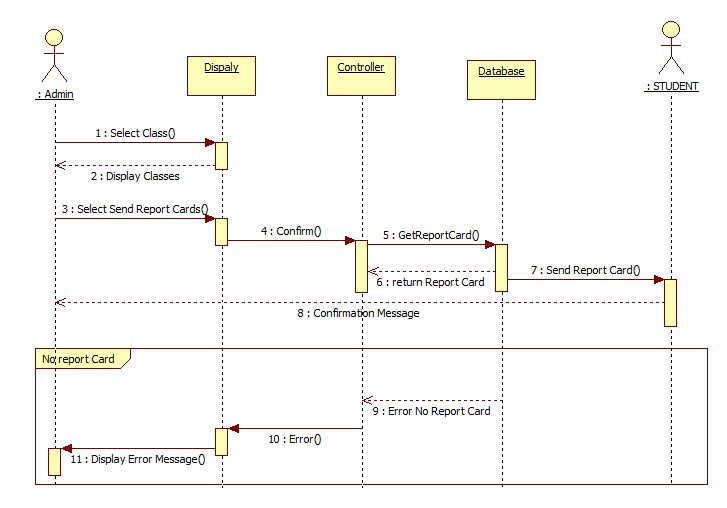
**Sequence Diagram 12: Edit Class Info**

****

**Sequence Diagram 13: Compile Report Card**

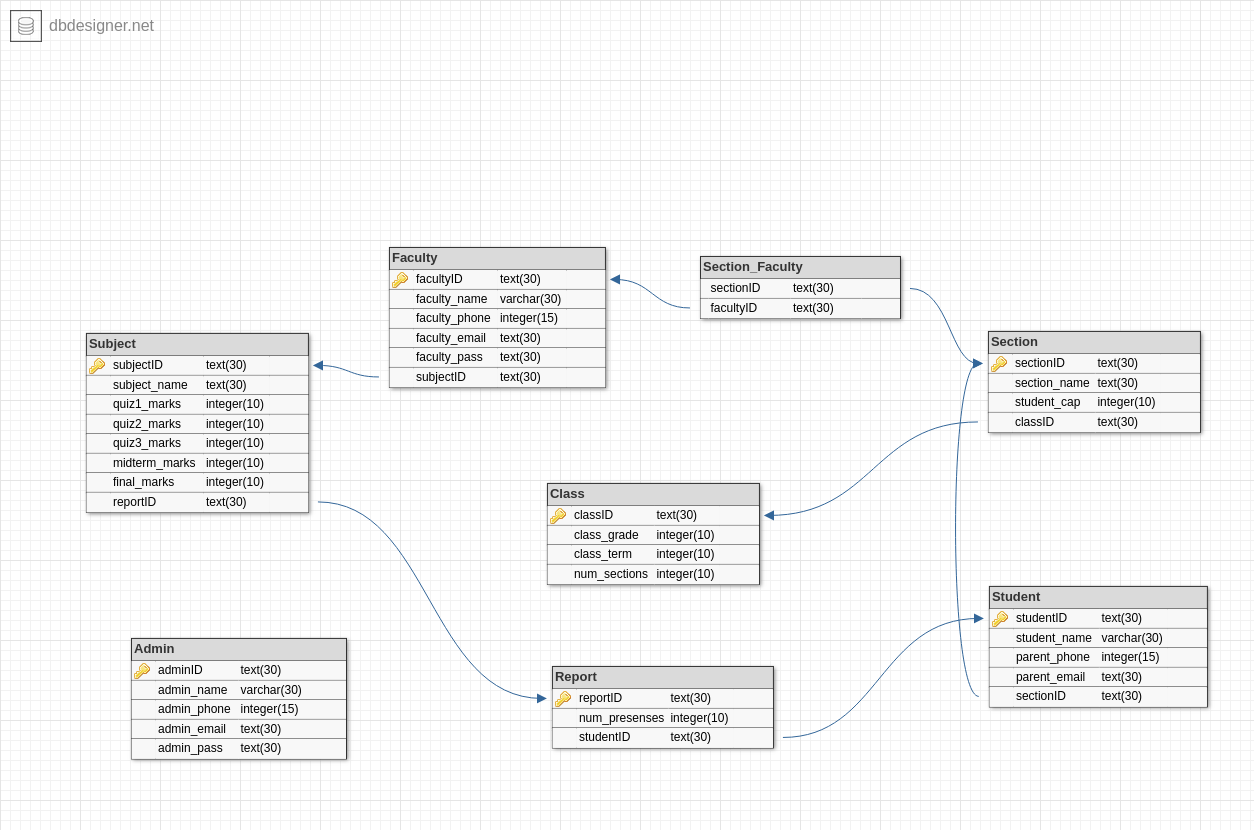
****

**Sequence Diagram 14: Send Report Card**

****

## Database Model

Database scheme and description of tables and relationships:



**Tables:**

**Admin**

This will contain all the details of the admins such as his ID, name, phone, email and password. This table will not be related to any other table.

**Faculty**

This will contain all the details of the faculty members such as his ID, name, phone, email and password and the subject he will be teaching. This will be related to ‘Subject’ through subjectID to show that this faculty member teaches which subjects.

**Section**

This will contain all the details about the sections such as sectionID, section name and the student capacity. This will be related to ‘Class’ through classID as a foreign key to store which class this section is related to.

**Section-Faculty**

This will only be a joining table to store which faculty member is assigned to which sections. This will have sectionID and facultyID as foreign keys to join ‘Section’ with ‘Faculty’.

**Student**

This will contain all the details of the students such as his ID, name, their parents’ phone and email. This will be related to ‘Section’ through sectionID as a foreign key to store which student is assigned to which section.

**Subject**

This will contain all the details about the subjects such as the subjectID, name, quiz 1-3 marks, mid-term marks and final exam marks. This will be related to ‘Report’ through reportID as a foreign key which will show that this subject appears on which report cards.

**Class**

This will contain all the details about the classes such as the classID, class grade, class term and the number of sections that class will have.

**Report**

This will contain all the details of the report cards such as reportID and the number of presences. This will be related to ‘Student’ studentID as a foreign key to show this report card is of which student.

**Relationships:**

Faculty – Section: n -> n

Section – Student: 1 -> n

Class – Section: 1 -> n

Report – Subject: 1 -> n

Student – Report: 1 -> 1

Faculty – Subject: 1 -> 1

### Database

MongoDB will be used for the Database because it:

* Can handle large unstructured data with ease.
* Has no complex joins.
* Supports dynamic queries on documents using document-based query language (NoSQL).
* Easily scalable
* Does not requires mapping of application objects to database objects.
* Uses internal memory for storing the working set, enabling faster access of data.

## External Interface Requirements

### User Interfaces

The users will be able to interact the system via a GUI. It enhances interactivity with the inclusion of buttons, tabs, scroll down options, dialogue boxes. The users i.e. school teachers and administration are expected to have some experience of using web-based interface. However, still there will be learning curve for them to familiarize themselves with the system interface. Therefore, the GUI will be optimized for users.

The interface will be providing less cluttered information, easy to navigate screens, using larger font size. In addition, symbols will be added that will be intuitive and will aid the user for navigating smoothly through the interface. The screen resolution will be kept limited to 1366 x 768 as the users are not expected to use high resolution monitors. In addition, message shorter message prompts will be hovered over the same screen instead of routing to an additional screen. Google’s material theme will be used to provide a minimalistic design to the interface so that it is visually appealing for the end user.

### Hardware Interfaces

The software is a web-based application which does not requires any additional hardware equipment or upgradation of existing computer systems. In addition, it does not require extensive computational power to run. However, reliable access to internet is required for using the software. Therefore, LAN, WLAN or ethernet devices are hardware interfaces that the system interacts with and which can impact the user experience of the system. In addition, the system also requires a web browser, that supports JavaScript.

### Communications Interfaces

The data should be transmitted securely over the internet without any loss to ensure consistent and accurate communication between different stakeholders. To ensure this issue, the system will be using HTTPS for sending data via browser to the website. This will enhance the security and trust of the system and prevent any malicious attack on the system.

# User Interface Design

## Description of the user interface

**Visual Studio Code**

Microsoft Visual Studio Code is a powerful text editor which supports the integration of all the popular extensions which will enhance the interoperability of our software. In addition, Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging. Lastly, it supports the integration of slack and VS live sharing extension which allows multiple users to work on the same program concurrently.

**ReactJS**

For the development purpose, we will use ReactJS. Due to its component-based approach, it is very simple to learn. It supports the functionality of reusable components which will make the development process faster and flexible at the same time. React also uses virtual DOM which will make the user experience better and our development work faster.

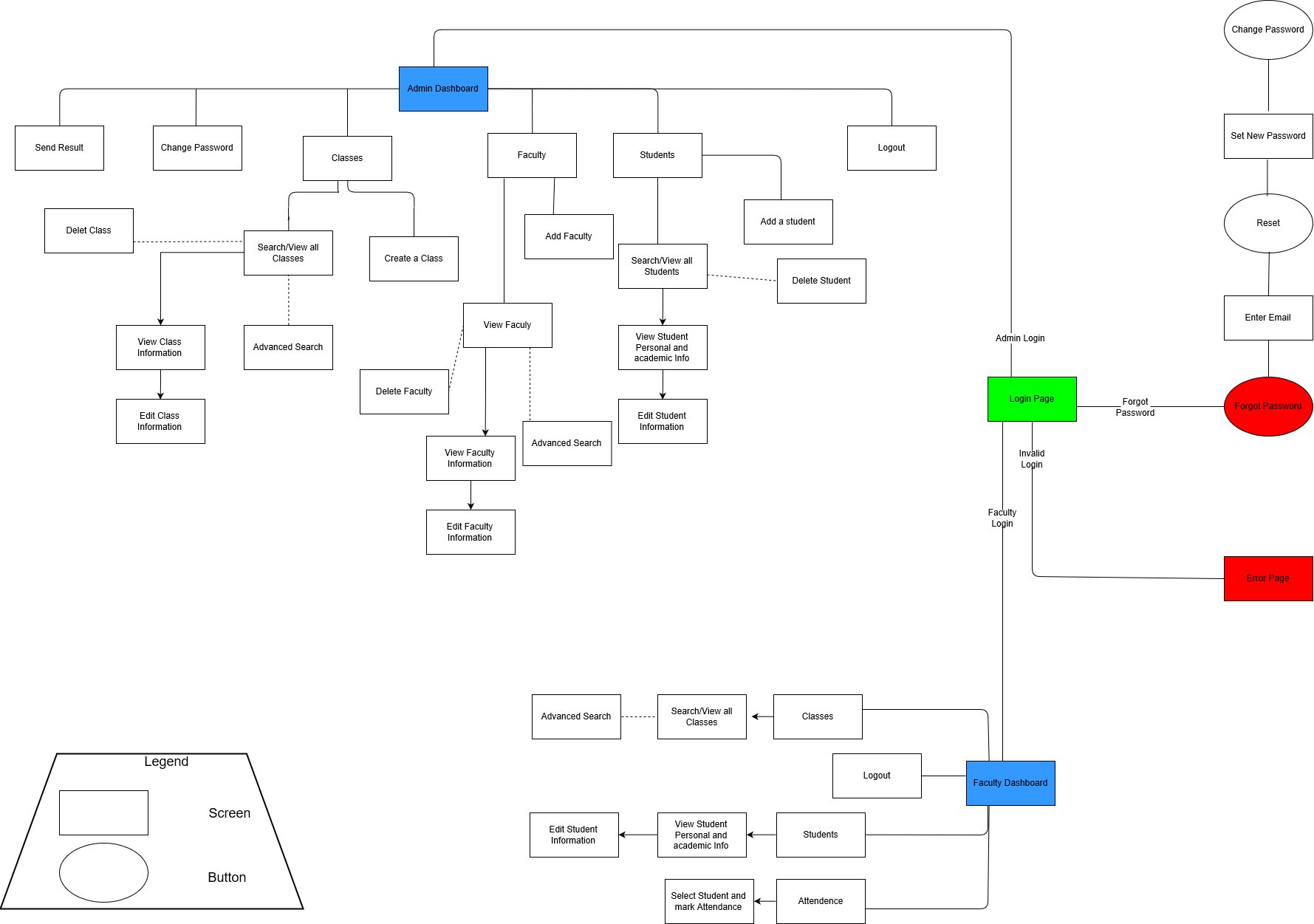
**React-Redux**

We will also be using Redux as it would help us in organizing our code and it would be easier for us to debug and test our code.

**MVC Presentation Layer**

Request comes to controller and controller decides whether this request needs any business logic to be implemented or not. If no it simply returns a view. If yes it interacts with Service Layer and Service Layer interacts with DAO Layer. Later the based on the response from Service Layer appropriate View will be shown by the Controller.

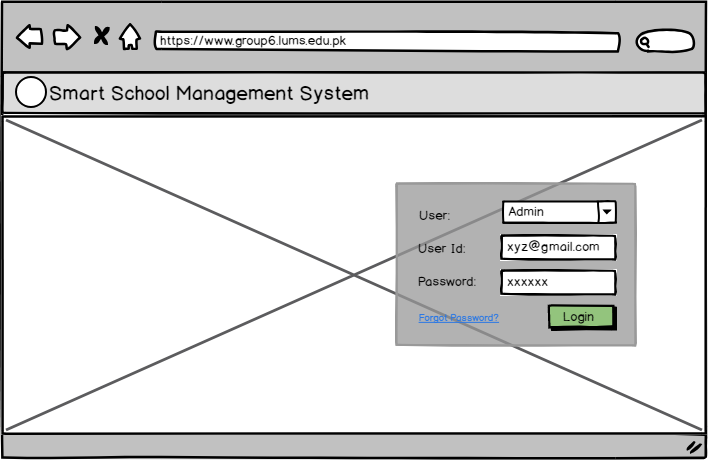
## Information architecture



## Screens

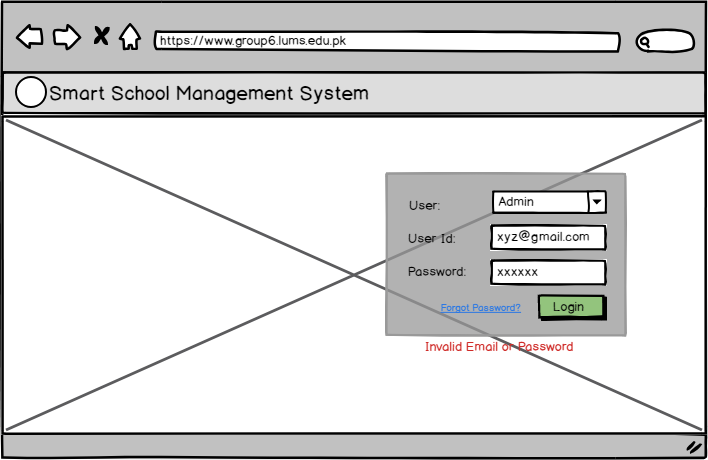
**Use Case 1: Admin and Faculty Login Functionality**

This screen shows the login page where both type of user land. Then user has the option of selecting his privilege level via a dropdown selection menu. Afterwards, User id and password is added which undergo input validation and then authenticated. This screen is satisfying the user requirement of admin and faculty login functionality.



**1.1**

Figure 1.2 shows the screen if the user input fails input validation. Thereafter, the user has the option to reenter his password. However, if he feels that he no longer remembers the correct password, then he has the option of selecting “Forgot Password” option.

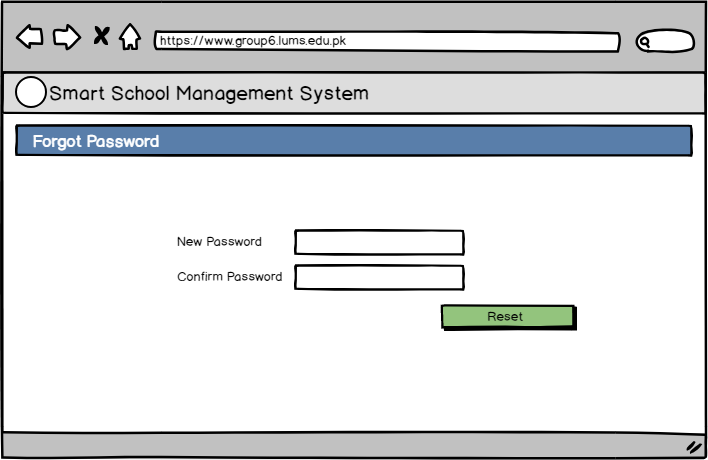


**1.2**

Afterwards, the user lands on the forgot password page where he is prompted to enter his email address.



**1.3**

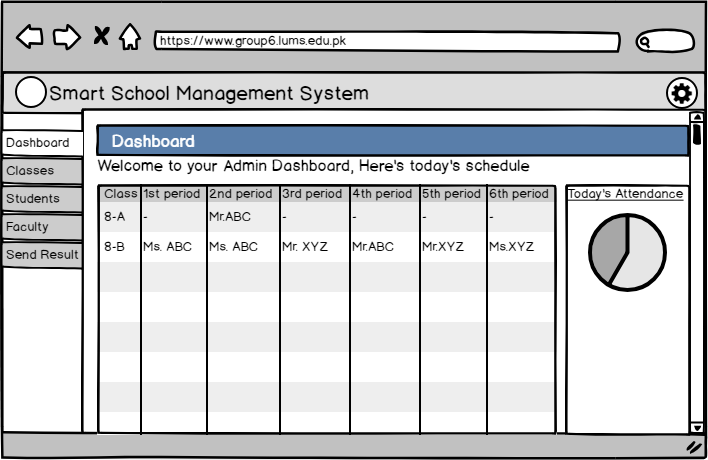


**1.4**

User enters his new password twice which is again validated and then he selects reset.

**Admin Dashboard**

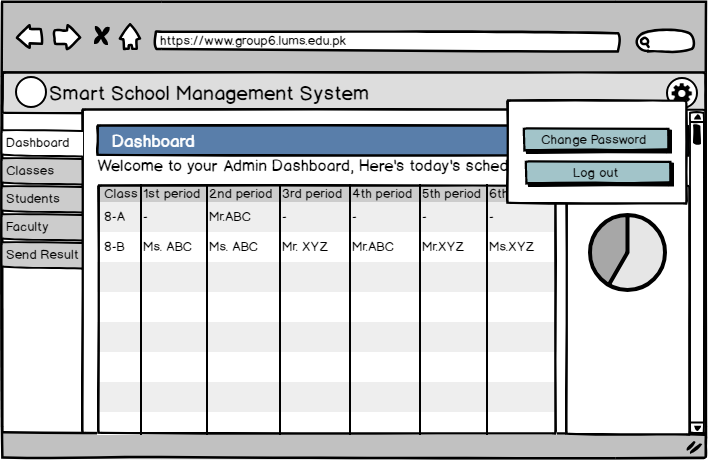
After successful login, the admin lands on a dashboard. The dashboard has a sidebar menu which each satisfy different use cases of the respective user. In addition, the user is shown infographics of school’s attendance and daily schedule.



**2**

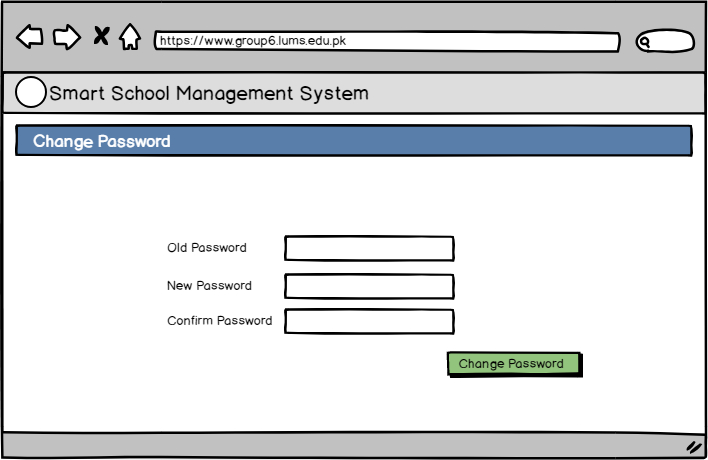
**Use Case 2: Change Password**

From the dashboard, the user has the option to either logout of change his password.



**2.1**

Selecting change Password prompts the user to add old and new password. Both undergo input validation. If the input validation fails, an error message shall be displayed on the same screen.

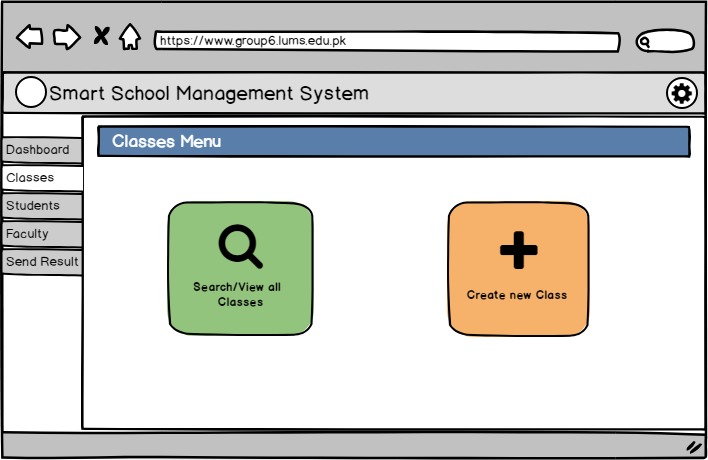


**2.2**

**Admin Use Cases**

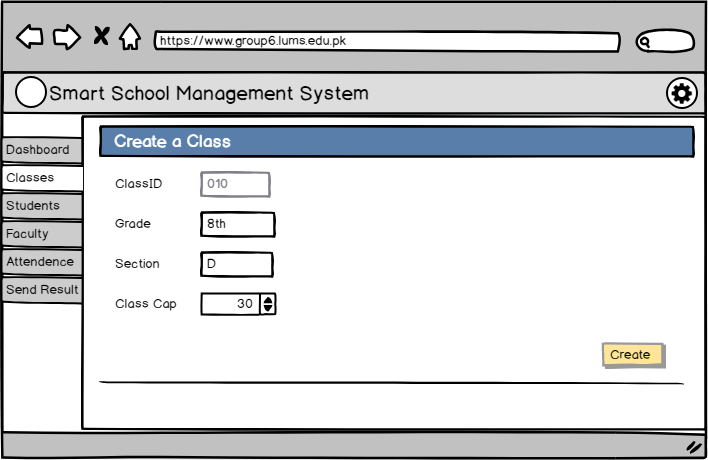
**Use Case 3: Add Class**

Admin has the option of adding a new class to the school. He selects classes option from the sidebar menu which takes him to the class menu i.e. figure 3.1. This action displays two buttons through which he can either add a new class or view that are already in the system.



**3.1**

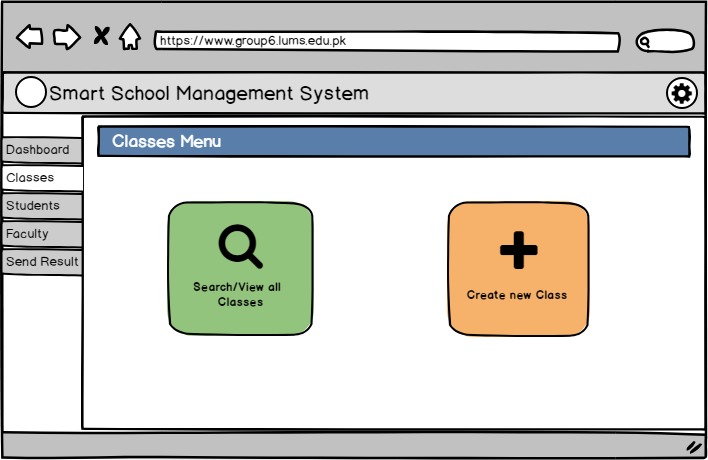
Thereafter, he selects the attributes in the box windows below in figure 3.2. The class cap can either get input via typing or by selecting from the dropdown menu placed adjacent to the respective box. The class ID is automatically generated by the server.

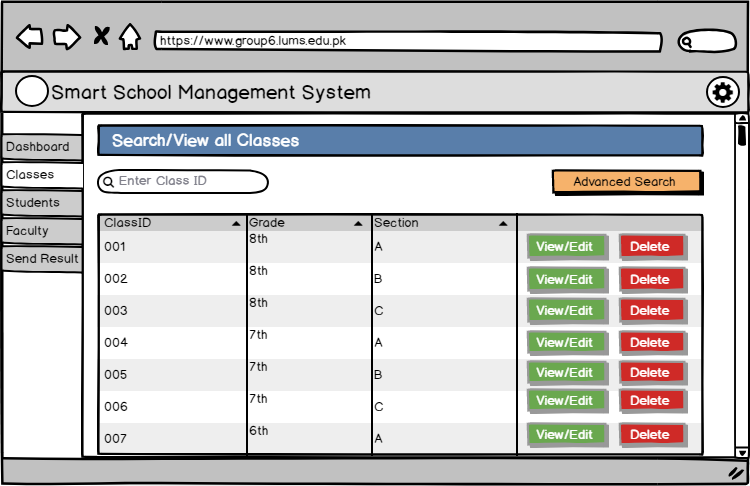


**3.2**

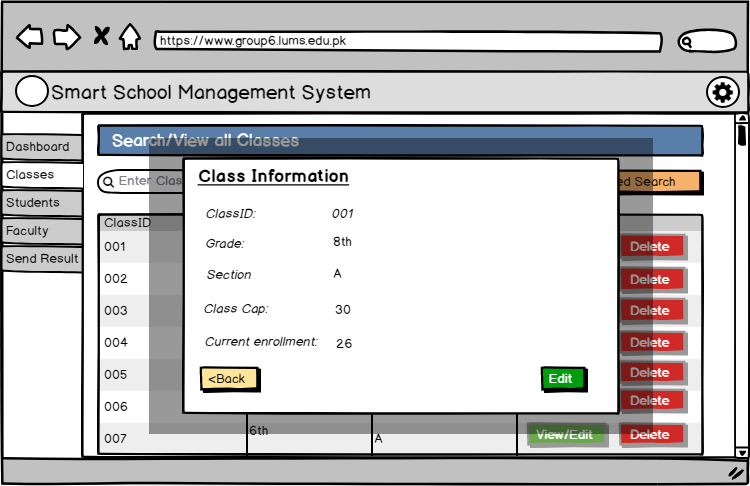
**Use Case 4: View/Edit Classes**

The user has the option of viewing and editing the class information by clicking the view all classes button on the classes screen.

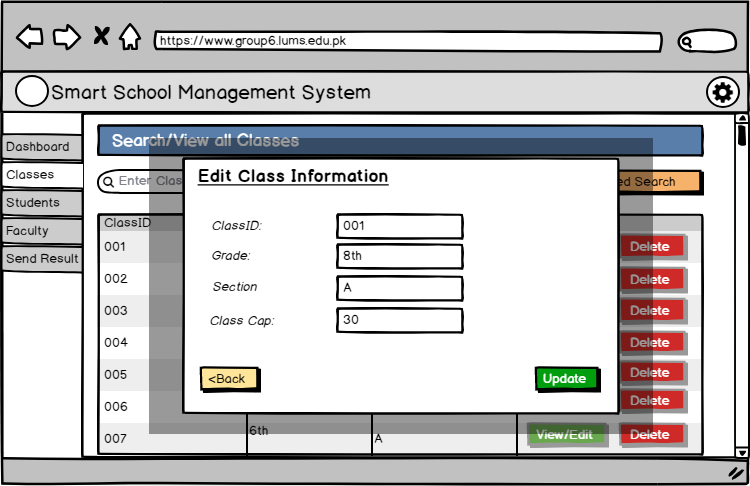
** 4.1**



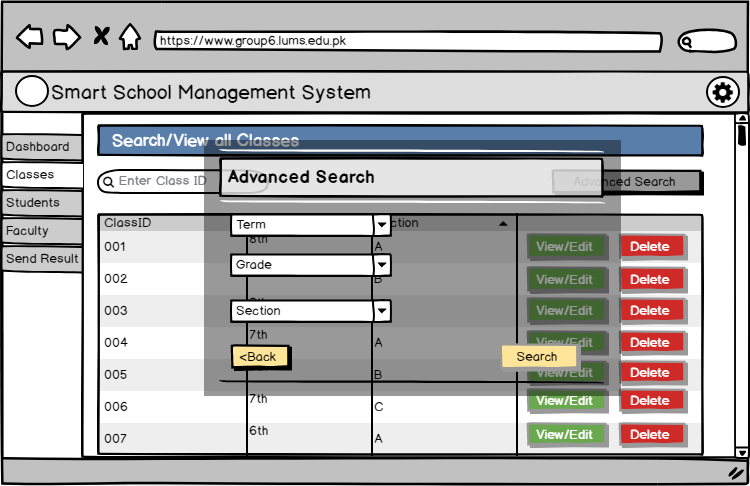
**4.2**



**4.3**



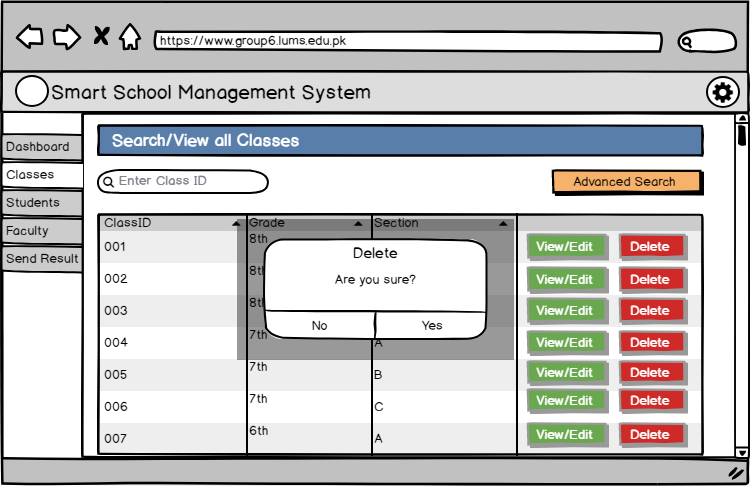
**4.4**



**4.5**

**Use Case 5: Delete Class**

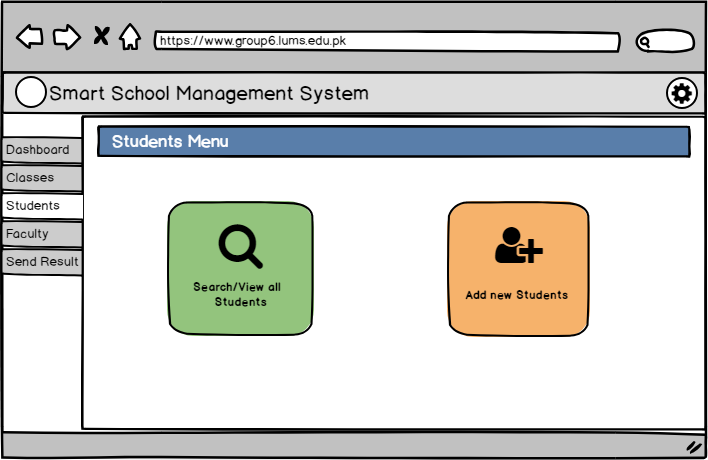
A particular class can also be deleted after selection.



**5.1**

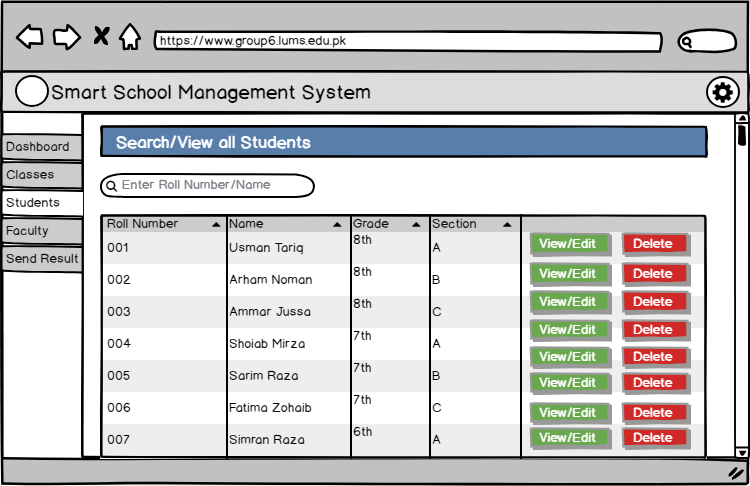
**Use Case 6: View Student information**

Students tab can be selected from the sidebar menu which displays buttons to either add new student or view existing student.



**6.1**

After clicking on view student information, user has the option of either manually finding the desired student via a scroll bar present on the right corner or he can enter the student’s roll number or name in the search box.

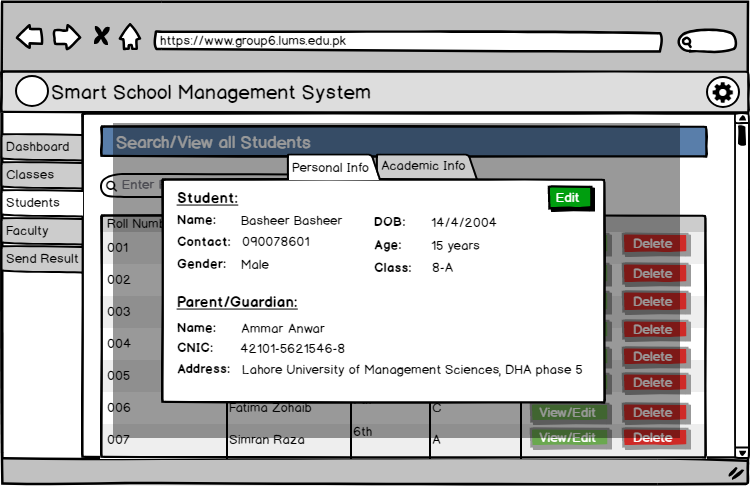


**6.2**

After successful search the user is displayed the student’s information.

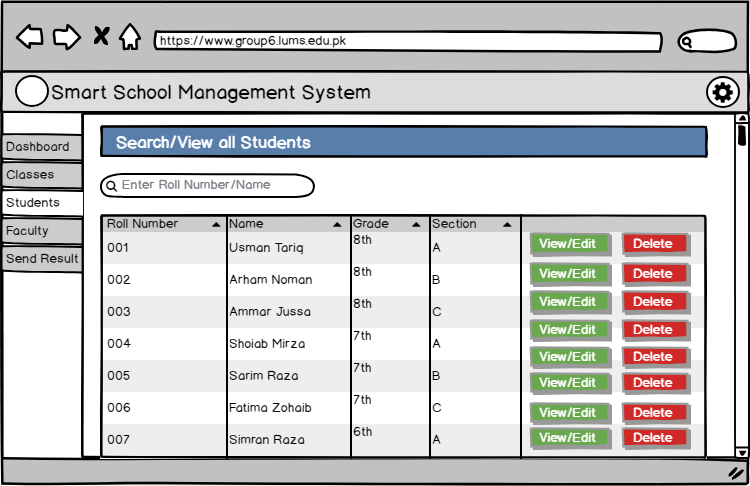


**6.3**

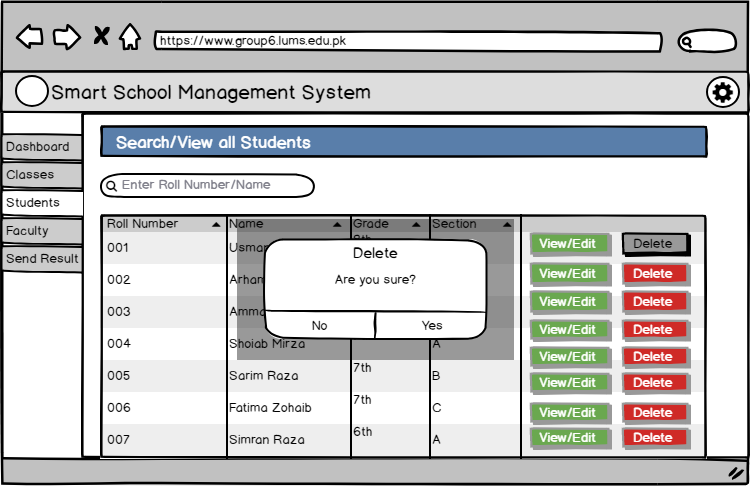


**6.4**

**Use Case 7: Delete Student**



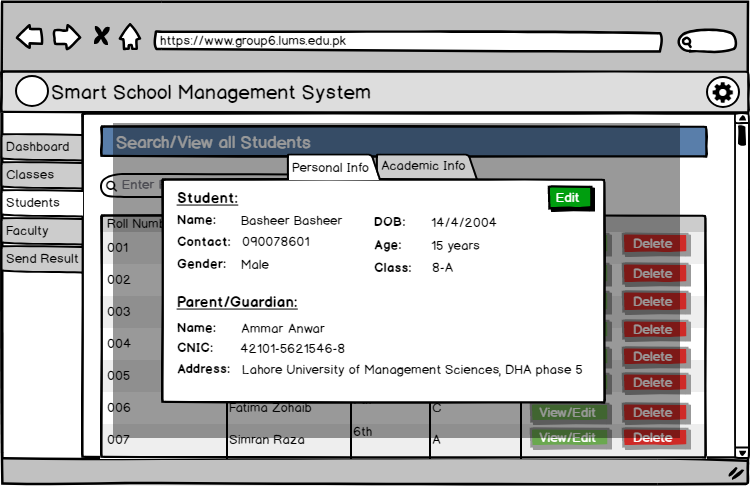
**7.1**



**7.2**

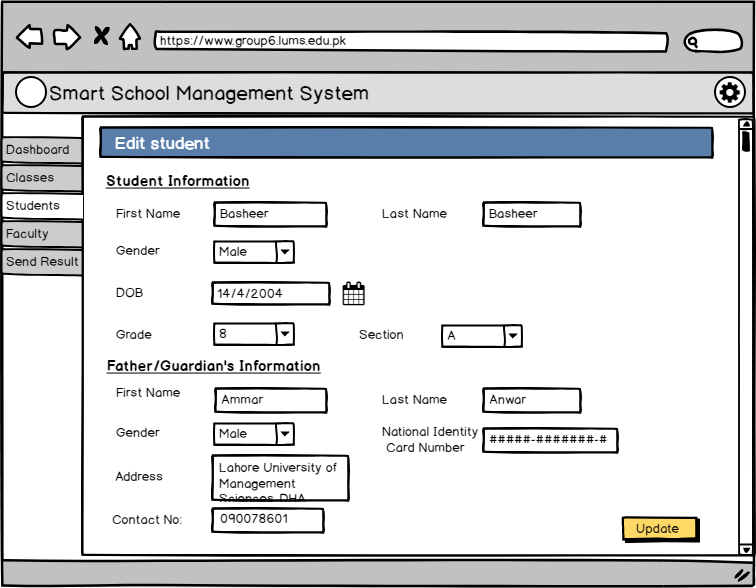
**Use Case 8: Edit Student Personal Information**

The user can opt to edit the student information by selecting the edit option when the personal information screen is shown.



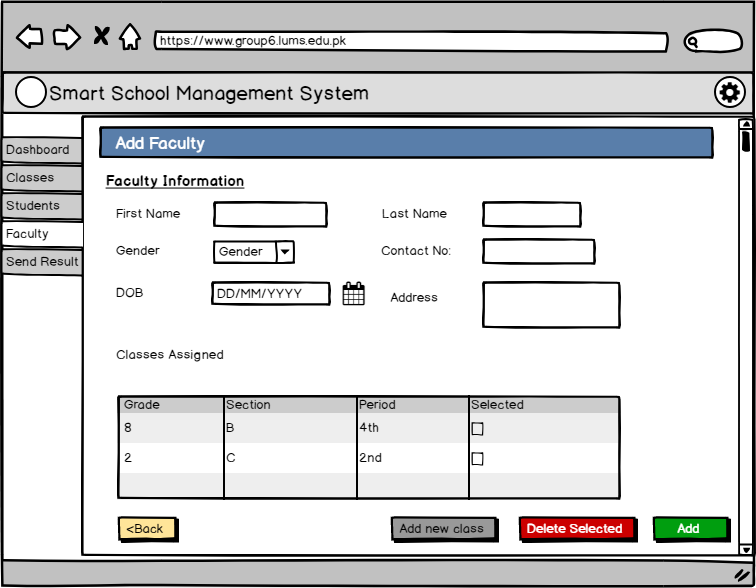
**8.1**

Thereafter, a form will appear displaying the user’s previous attributes and the option to modify them. After filling in the desired parameters, the user can click save to update the student information.

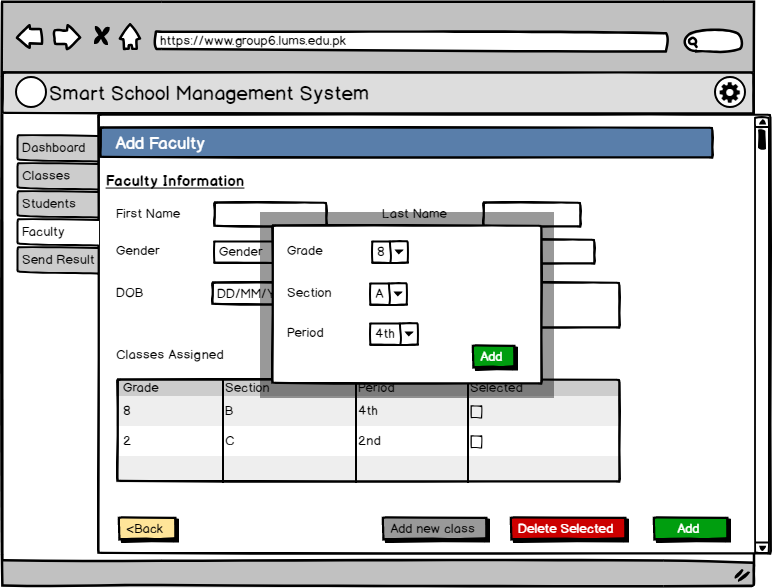


**8.2**

**Use Case 9: Add Faculty**



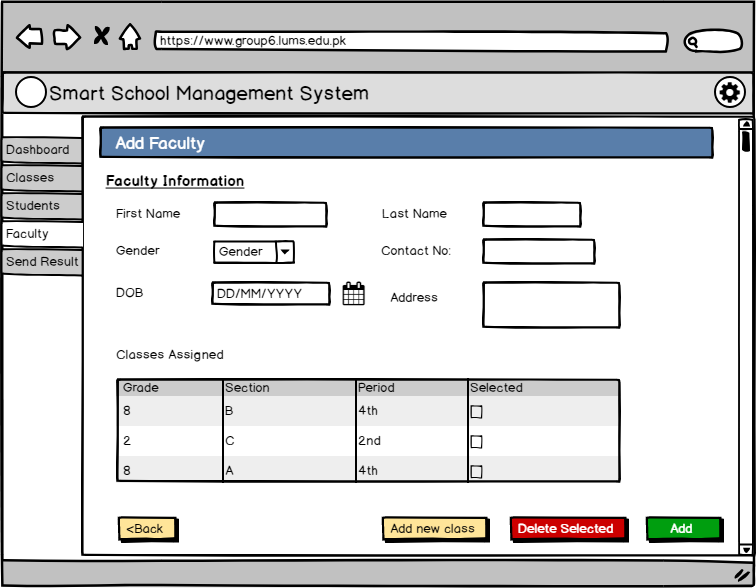
**9.1**



**9.2**

User has also the option of assigning new classes to the faculty.

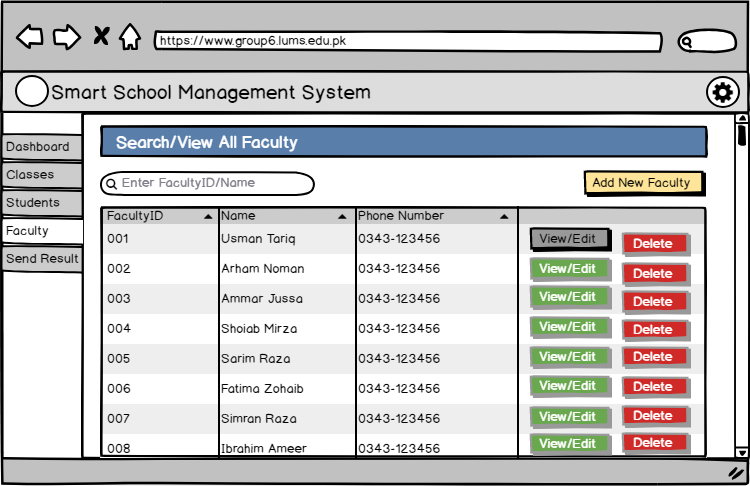
Faculty will be added to the system after all attributes are filled and validated.



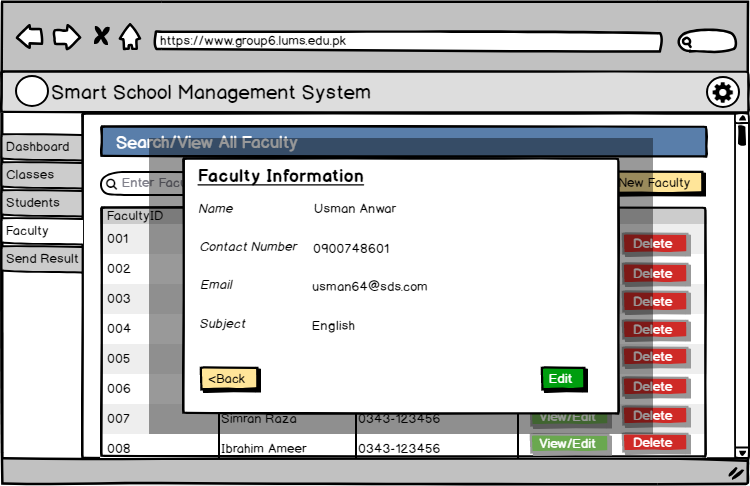
**9.3**

**Use Case 10: View/Edit Faculty**

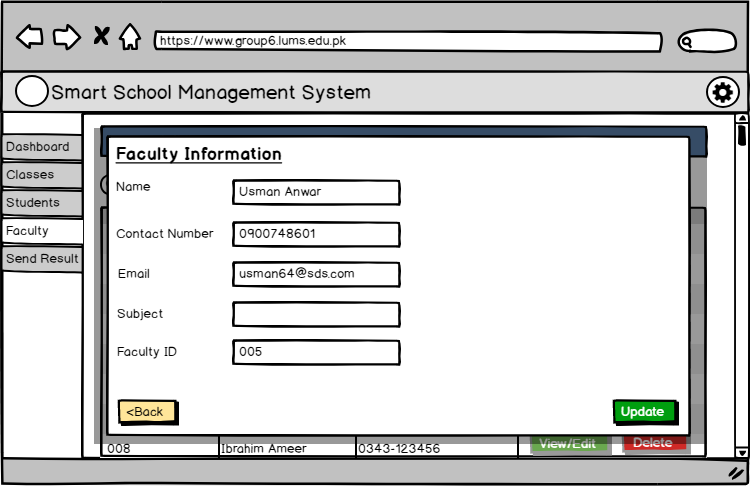
Admin can also update the fields of the faculty in the system.



**10.1**



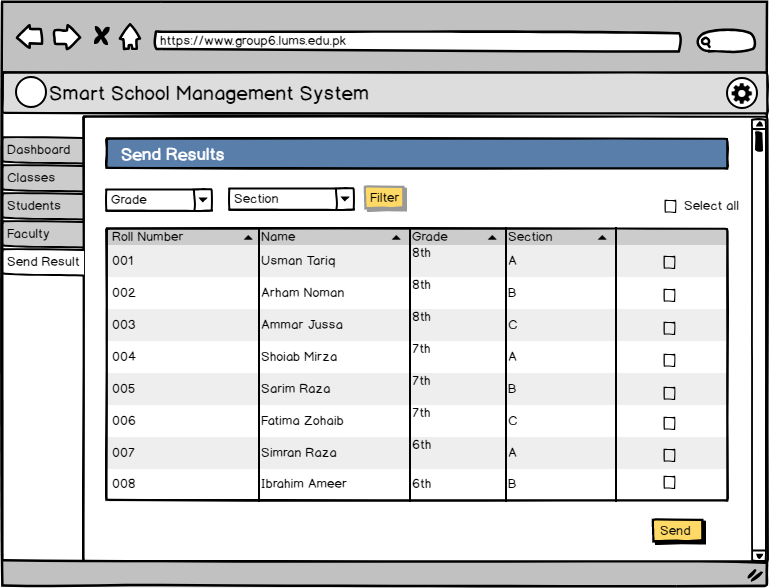
**10.2**



**10.3**

**Use Case 11: Send Report Cards**

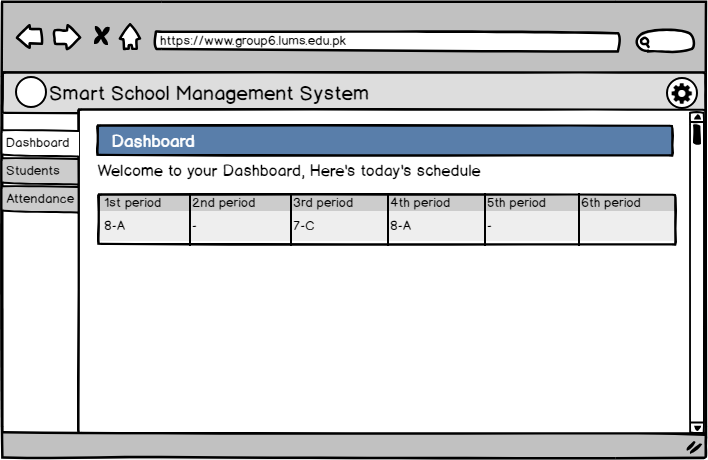
From the sidebar menu, the user can navigate to send result option which displays roster of students, and the check box option. The user can opt to send results to specific users or choose to select all.



**11.1**

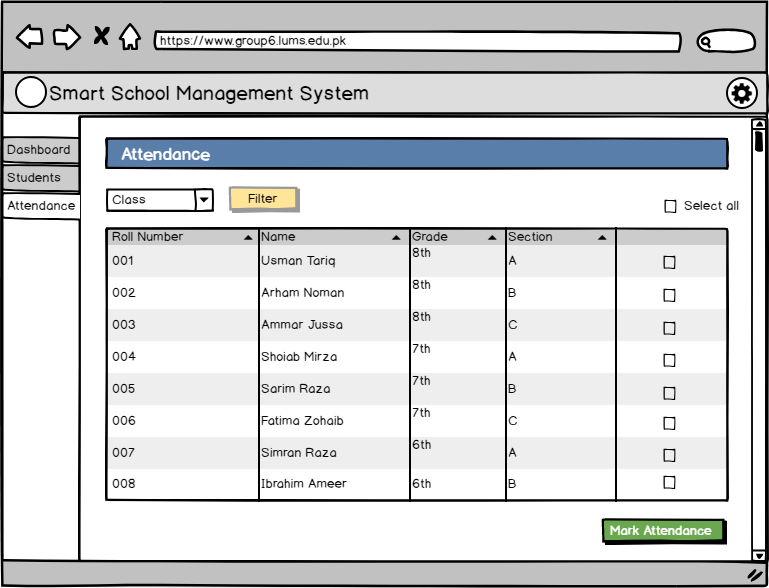
**Faculty Use Cases**

**Faculty Dashboard**



**Use Case 12: Mark Attendance**

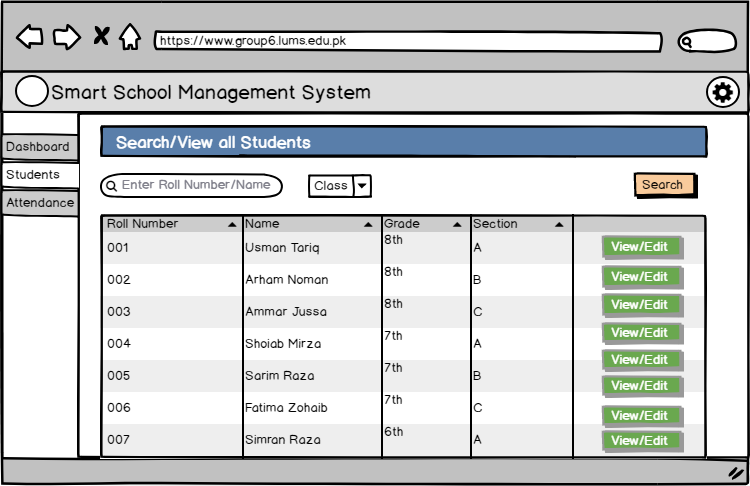
Faculty navigates to the mark attendance tab from the sidebar menu. The screen shows the option of selecting the desired class. Check boxes allow faculty to mark attendance.



**12.1**

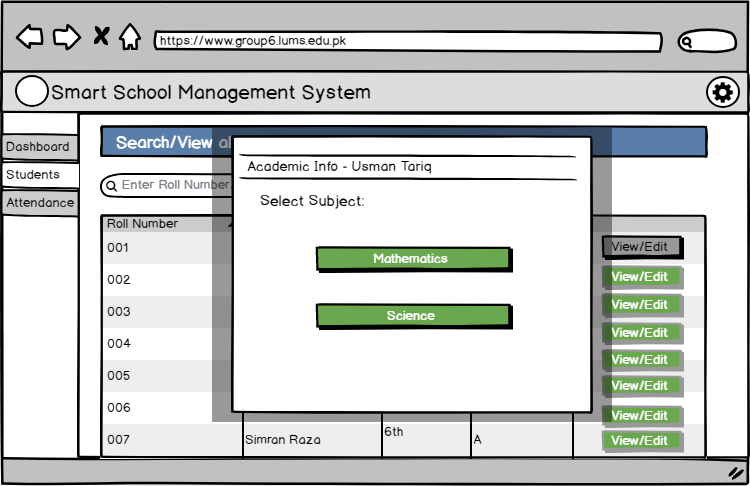
**Use Case 13: View/Edit Student Marks**

User navigates to the student’s tab in the sidebar menu. User can search the desired student via scroll down option or search box option.

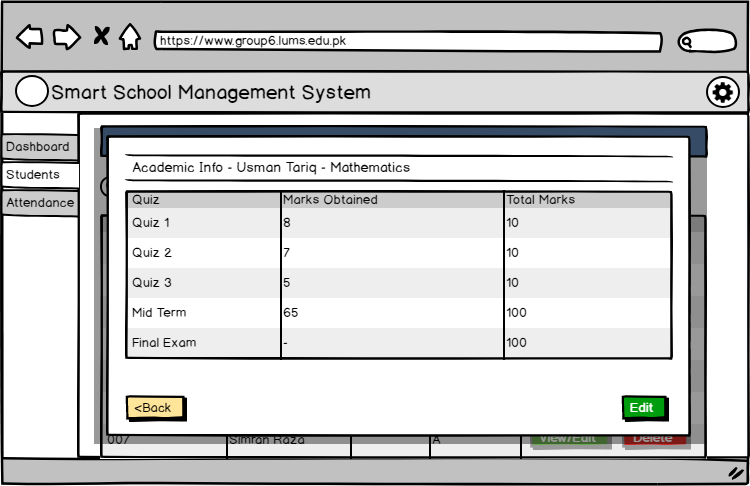


**13.1**

User will select the subject he wants to see the marks of.

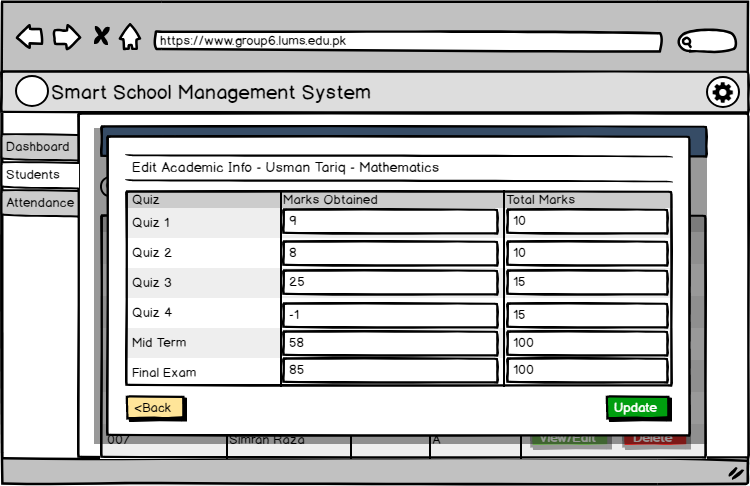


**13.2**



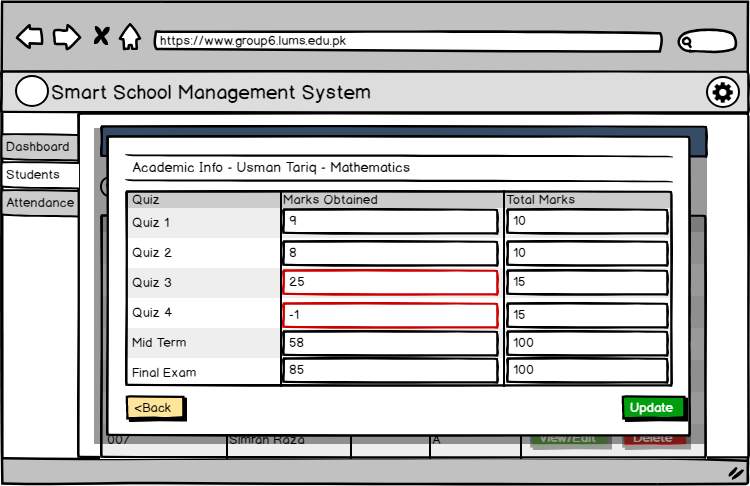
**13.3**

User has the option of editing the student marks.



**13.4**

The user enters the desired marks in the marks filed. The values undergo validation. If the values are invalid, the display highlights the invalid fields and prompts the user to enter the values again.



**13.5**

## User interface design rules

1. Allow users to use either the keyboard or mouse (**flexible**)

Users will be able enter data and navigate using any method, either keyboard or mouse. (Scrolling, Selecting)

1. Allow users to change focus (**interruptible**) Users will open several windows once but focus working on only one interface. This will open different interfaces in different web browser tabs, making it easier to change among tasks.
2. Provide immediate and reversible actions, and feedback. Most of the time users will be given help to revise their selections in the same interface where they make changes. E.g. there will be options to reenter marks on the interface where user enters negative marks, making it easier to do changes quickly.
3. Provide meaningful paths and exits (**navigable**) Finding a relevant interface and exiting from it should be easy and sensible. Operations should be able to understand without much technical knowledge for all users. Users should be able to guess where to find certain interfaces without complex procedures.
4. Accommodate users with different skill levels (**accessible**). System will be used by users with many skill levels. And all of them should be able get their job done without specific technical support. This system will be designed with simple interfaces, easy to understand operations.

# Other Non-functional Requirements

## Performance Requirements

1. Average response cycle shall lie between 1-10 seconds.
2. Peak response time must be no more than 20 seconds.
3. The system must support at least 50 concurrent users as multiple users are expected to work concurrently.
4. The database must be able to store data of at least 500 student records without any duplicates and shall have the ability to expand at later stage.
5. The database capacity must not be bounded by a certain value to support expansion subsequently.
6. Any operation done must instantaneously update the database.
7. The error rate shall be 5% i.e. requests resulting in errors
8. The system shall be able to handle at least 1000 requests per second.
9. Average load time must be no more than 0.5 seconds.
10. The software shall be optimized to use no more than 512 MB of Ram so that it is able to run on low-end systems.

## Safety and Security Requirements

## Legal Requirements

The system should abide by the legal guidelines provided by the, “Prevention of Electronic Crimes Act 2016”, which provides legal constraints regarding the protection and misuse of user’s data. Thus, user’s data must be highly secured and not misused at any stage. To ensure this the data will be protected using secured database Heroku.

### Confidentiality

The network communication must also be secured to prevent misuse of user’s data. Furthermore, export of user personal information like contact information should be restricted.

### Authentication

The user shall be able to access the data according to his respective privileged level. At each instance of login, the user type is authenticated. In addition, critical operations must re-authenticate the user.

### Safety

If there is extensive damage to a wide portion of the database due to catastrophic failure, the recovery method should restore a past copy of the database that was backed up to archival storage and reconstruct a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure.

### Security

The system shall prevent itself from malware attacks and viruses. In addition, it shall address following security requirements.

* Integrity: prevention of unauthorized modification of information.
* Availability: prevention of unauthorized withhold of information
* Restrictions on unauthorized inter-module communication
* Auditability: Easier traceback
* Store admin and faculty passwords in the form of hash (SHA-1 hashing algorithm)

## Software Quality Attributes

### Scalability

The system should be able to expand and allow integration of multiple school branches without any providing any major restrictions or conflicts. The database will support inclusion of additional data. However, a paid version shall be required for this expansion. Furthermore, the use of client server architecture makes it easier to scale without interrupting the existing framework.

### Availability

The system shall be using Heroku as the database platform for our application. This always enables the application to be available to users. However, the availability will be dependent on a stable internet connection.

### 6.3.2 Interoperability

The system should be able to communicate, send, receive data over various platforms like google drive, and export/import data in different tools like Microsoft excel. This shall be achieved via using technologies that complement each other like… and provide no integration restrictions.

### 6.3.3 Modifiability

The system shall be easy to modify so that expansion of the system at a later stage cost less.

This will be achieved by minimizing coupling and improving cohesion. In addition, the use of MVC architecture will reduce maintenance costs at subsequent stages.

### 6.3.4 Reliability

The system shall have an uptime rate of 95%. In addition, it will have a failure rate of 10 in 1000 runs. This will be done by rigorous testing after development.

### 6.3.5 Usability

The user interface should be optimized to according to the user profiles as they would need some time to familiarize themselves with the new user interface. To ensure this, intuitive and minimalistic interface will be provided to the user.

Appendix A - Group Log

**26 Feb:** Gathered to discuss about the document (1 hour)

**5 Mar:** Assigned roles. Went through all the diagrams from the slides (1 hour)

**12 Mar:** Discussed all the sections of the document. (2 hours)

**17 Mar:** Assigned work for spring break (1 hour)

**25 Mar:** Reviewed all the work done. Checked each other’s diagrams and reviewed all the text (2 hours)

**26 Mar:** Finalized all the screens and information architecture (2 hours)

**27 Mar:** Finalized class diagrams and database schemas (3 hours)

**28 Mar:** Finalized all the description of the sections (3 hours)

**29 Mar:** Compiled all the work done. (8 hours)

Appendix B – Contribution Statement

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Contributions in this phase | Approx. Number of hours | Remarks |
| Muhammad Usman Tariq | Did all the screen images and user interface design | 22 |  |
| Ahmed Anwar | Did all the description part + Sequence diagrams + context diagram | 22 |  |
| Syed Arham Noman | Did some of the screen images + information architecture diagram | 17 |  |
| Muhammad Abdullah | Did activity + component diagram + some description | 16 |  |
| Ammar Anwar Jussa | Assigned Roles + database schema + description | 16 |  |