# 

**Chapter 1**

# INTRODUCTION

# 1.1 INTRODUCTION

School Management System is a large database which can be used for managing your school day to day activities. It allows users to store almost all of their schools information either online or offline depending on the software built, including student information, employees, properties and study materials etc. most importantly, this information can be shared with authorized users, records can be searched and reports can be easily generated. School management system is configurable and can be configured to meet certain individual school’s needs. It could make the school staffs life easier than ever. Using this system, finding student’s information’s is just less than a minute which might have cost several minute even more than a day. At the end of academic term form masters can easily prints student’s statements.

School Management System helps headmasters to get the most accurate information to make more effective decisions. Teachers and headmasters gain time saving administrative tools, parents gain immediate access to their children’s grades and students can track their own progress.

School Management System equipped features makes it possible to generate schedules and reports in minutes and to retrieve attendance records, grade checks, report cards, transcripts, and form letters in just a few clicks.

School Management Systems helps Teachers to complete grade book, track students attendance, input class notes, create lesson plans and detailed reports, and communicate with other staff members, students, etc. It also helps Students to access assignments and tests, and view attendance records, grades, report cards. . School Management System (SMS) is a web enabled application developed in using (React, Html ,Css ) As Frontend and powerful Spring Boot as backend & MySQL as database.

To implement School application, schools do not need expensive hardware and software, they just need an internet connection and desktops. Our system works as a centralized database and application that schools can easily access the system from anywhere based on the login credentials.

This system saves the time of the student and of the administrator. It includes processes like registration of the student’s details, assigning the department based on their course, and maintenance of the record. This system reduces the cost and workforce required for this job. As the system is online the information is globally present to everyone.

This makes the system easy to handle and feasible for finding the omission with updating at the same time. As for the existing system, they use to maintain their record manually which makes it vulnerable to security. If filed a query to search or update in a manual system, it will take a lot of time to process the query and make a report which is a tedious job.

As the system used in the institute is outdated as it requires paper, files, and binders, which will require the human workforce to maintain them. To get registered in the institute, a student in this system should come to the university. Get the forms from the counter while standing in the queue which consumes a lot of the student’s time as well as of the management team.

As the number of the student increases in the institute manually managing the strength becomes a hectic job for the administrator. This computerized system stores all the data in the database which makes it easy to fetch and update whenever needed.

My School is a platform independent system that virtually any user can access from anywhere through a standard internet accessible system

We can also customize School Managment System for individual school needs.

**1.2** **OBJECTIVES** :-

– To build a responsive website to manage the different school activities.

- To track student’s grades from their parents.

- To facilitate distribution process of courses and classes for teachers.

- To facilitate grades entry process for students by teachers.

- To make a virtual community between the members of educational process.

**1.3** **SCOPE AND LIMITATION OF PROJECT** :-

School management system (SMS) has been designed to provide an easy way for students to get their grades, and for their parents to be familiar with that grades and the academic achievement for their kids .

However this project is facing some obstacles which is deny it from achievement it’s goals, like : lack of acceptance these idea from some teachers and headmasters, because of their poor knowledge about using that technology or they find that using computers in their works instead of the paper works so hard and difficult.

And also the same thing for some parents where they do not prefer that way for the same previous reasons.

**Chapter 2**

**METHODOLOGY**

# 3.1 INTRODUCTION

This chapter states the methodology used to reach the objectives of the project. The framework in which software is designed, developed, and maintained is known as the Software Development Life Cycle (SDLC).

It shows the steps, phases, milestones, and evolution of the software development process. There are many types of models used in software design and development. Among them are the spiral models, rapid development model, Evolutionary model, waterfall model, prototyping model, etc.

**3.2 METHODOLOGY**

Prototyping Model has been used to develop this application. The Prototyping model is a technique for quickly building a function but incomplete model of the information system. There are several kinds of prototypes but they all intend to reduce risk by building a quick and dirty replica or mockup of the intended system.

It can be used to demonstrate technical feasibility when the technical risk is high. It can also be used to better understand and elicit user requirements. In either case, the goal is to reduce risk and limit costs by increasing understanding of proposed solutions before committing more resources, as shown in figure 1.

**3.3 PROTOTYPE MODEL**

**Advantages of Prototyping**

- Reduces development time.

- Reduces development costs.

- Requires user involvement.

- Developers receive quantifiable user feedback.

- Facilitate system implementation since users know what to expect.

- Results in higher user satisfaction.

- Exposes developers to potential future system enhancements.

**Chapter 3**

**REQUIREMENTS**

**3.1 INTRODUCTION**

This chapter will provide a full description of the system and its users. Then it depicts the functional and non-functional requirements that have been collected using several methods like brainstorming, interview and e-surveys. After determining the most important requirements, requirement analysis was adopted using several tools such as use-case diagram, sequence diagram and activity diagram.

**3.2 PROJECT DESCRIPTION**

School Management System (SMS) is a web based system that serving Students, Teachers, Headmasters a the main idea of our system is to allow the students to be up to date with the school activities and with their grades, and the same thing for their parents in addition to be in touch permanently with the teachers and the headmaster for any complaint, recommendation or anything that related to their students. And for teachers it consider as an easy way to manage their job’s day, to be in touch with students and to enter their grades or anything they want to send it or view it to the students. And for the headmasters, they have a full control of the system, and like the teachers, it will be a great tool to manage their days and plans for the school, also they will be in a permanent connection with all the other users which will be a wonderful virtual educational social community that definitely will improve the student’s performance and education, and make the school for them really as they second’s home.

**3.3 USER DESCRIPTION**

|  |  |
| --- | --- |
| **Roles** | **Functions** |
| **Admin** | -**Log In/Out**  **-Register**  **-Approve user**  **-view feedbacks**  **- Add New Subject & delete**  **- Add New Class & Sections**  **- Add New Teacher & delete**  **- Edit Teacher**  **- Add New Student & delete**  **- Edit Student** |
| **Teacher** | **- Log In/Out**  **- Assign students Marks**  **- Post tasks and upload assignments**  **-View feedbacks**  **-Display result and add attendance** |
| **Student** | **-Edit profile**  **-See result**  **-Submit assignment**  **-See attendance**  **-Give feedbacks** |

There are three main users for the proposed system, these are Admin, Teaher and student

**3.4 SYSTEM REQUIREMENT**

Before creating any website or a mobile App, it is necessary to visualize the layout, design and all features intended to be incorporated.

In addition, how users will interact with each page and icon and how the website/App should perform (behavior, load time etc.).

Requirements are the necessary attributes in the system, a statement that identifies a capability, characteristic or quality factor of the system in order to have value and utility to the users. Once the requirements are set, developers can initiate the other technical work including system design, development, testing, implementation, and operation. For any system, there are functional and non-functional requirements to be considered while determining the requirements of the system.

The functional requirements are user “visible” features that are typically initiated by stakeholders of the system, such as generate report, login, and signup.

On the other hand, nonfunctional requirements are requirements that describe how the system will do what it is supposed to do, for example, security, reliability and maintainability

**3.5 CONSTRAINT**

School Managemet System shall be able to handle at least 1000 students data at a time.

**3.6 NON FUNCTIONAL REQUIREMENTS**

**3.6.1 Interface**

Go to Appendix B for user interfaces

**3.6.2 Performance**

* **Number of Concurrent Users:**

SMS shall be able to handle at least 1000 users.

* **Adding a User:**

The system is susceptible to any temporary server failure since it uses the strong feature of Struts 2 and Hibernate. Hence the examination will be continued even if the sever gets disconnected in between the examination.

**3.7 Other Requirements:**

* **Hardware Interfaces**

The SMS is expected to function on Intel PIII 900 MHz Processor equivalent or above, 2 GB RAM, 100MB HDD.

* **Software Interfaces**

The SMS shall work on MS Windows operating systems family (MS Windows 98, MS Windows NT Workstation, MS Windows 2000, MS Windows XP). It configures to work with Oracle database. This System works on Apache Tomcat server.

**3.8 FUNCTIONALITY**

**A) ADMIN (Headmaster)**

\* He can approve other user such as teachers and students

\* He can add another Teachers and Students and delete them

\* Can add subjects ,assign classes and sections

\* Can view the feedbacks

**B) TEACHER**

\* Enter Student's grades per Subject.

\* Contact with students and parents.

\* Add new assigments to students

\* Teacher can edit students profile

\* Teacher can display the results for students

**C) STUDENTS**

\* He can edit his own profile

\* He can see his result or grades

\* See attendance status

\* Give feedback

**3.9 FUNCTIONAL REQUIRNMENTS**

**Security**

Every user has his own account and only authorized users can access the system with username and password. The passwords and other validations like email have been made

to ensure the security of the users

**Performance**

Easy tracking of records and updating can be done.

**Availability**

The system are available to users anytime, anywhere, just need a PC and Internet Connection. Also the system work in multiple web browsers like (Chrome, Mozilla, Opera, and Internet Explorer).

**User Friendly**

The system have a friendly user interface and the system very interactive.

**Chapter 4**

**DESIGN**

**4.1 Database Design**

The following table structures depict the database design

# Table 1: User\_Info

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| \_id | int | NO | PRI | NULL | auto\_increment |
| first\_name | varchar(30) | YES |  | NULL |  |
| last\_name | varchar(30) | YES |  | NULL |  |
| dob | date | YES |  | NULL |  |
| mobile | varchar(15) | YES |  | NULL |  |
| gender | varchar(10) | YES |  | NULL |  |
| acc\_status | int | YES |  | NULL |  |
| username | varchar(30) | YES |  | NULL |  |
| email | varchar(30) | YES | UNI | NULL |  |
| password | varchar(100) | YES |  | NULL |  |
| type\_job | varchar(10) | YES |  | NULL |  |
| pincode | varchar(10) | YES |  | NULL |  |
| locality | varchar(100) | YES |  | NULL |  |

# Table 1: Student Details

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| student\_id | int | NO | PRI | NULL |  |
| roll\_no | int | NO | MUL | NULL | auto\_increment |
| class | varchar(2) | YES |  | NULL |  |
| section | varchar(2) | YES |  | NULL |  |
| fees | int | YES |  | 0 |  |

## **Table 3: Teacher Details**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| teacher\_id | int | NO | PRI | NULL | - |
| designation | varchar(25) | YES |  | NULL | - |
| salary | int | YES |  | 0 | - |

## **Table 4: Teacher Assignment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| teacher\_id | int | YES |  | NULL |  |
| assignment\_id | int | NO | PRI | NULL | auto\_increment |
| added\_assignment | varchar(100) | YES |  | NULL |  |
| submission\_time | timestamp | YES |  | CURRENT\_TIMESTAMP | DEFAULT\_GENERATED |
| subject\_code | varchar(10) | YES |  | NULL |  |
| assignment\_due | timestamp | YES |  | NULL |  |

## **Table 5: Teacher Attendance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| teacher\_id | int | YES |  | NULL |  |
| date\_of\_attendance | timestamp | NO | PRI | CURRENT\_TIMESTAMP | DEFAULT\_GENERATED |
| attendance\_status | varchar(10) | YES |  | NULL |  |

## **Table 6: Student Attendance**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| student\_id | int | YES |  | NULL |  |
| date\_of\_attendance | timestamp | NO | PRI | CURRENT\_TIMESTAMP | DEFAULT\_GENERATED |
| attendance\_status | varchar(10) | YES |  | NULL |  |

## **Table 7: Student Subjects**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| student\_id | int | YES | - | NULL | - |
| subject\_code | varchar(10) | YES | - | NULL | - |

## **Table 8: Subjects Details**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| subject\_name | varchar(25) | YES |  | NULL | - |
| class | varchar(2) | YES |  | NULL | - |
| subject\_code | varchar(10) | NO | PRI | NULL | - |
| max\_marks | int | YES |  | NULL | - |
| max\_assignment\_marks | int | YES |  | NULL | - |
| teacher\_id | int | YES |  | 0 | - |

## **Table 9: Student Feedback**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| feedback\_id | int | NO | PRI | NULL | auto\_increment |
| student\_id | int | YES |  | NULL |  |
| teacher\_id | int | YES |  | NULL |  |
| feedback\_remarks | varchar(100) | YES |  | NULL |  |
| feedback\_ratings | int | YES |  | NULL |  |

## **Table 10: Schedule**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| session\_id | int | NO | PRI | NULL | auto\_increment |
| teacher\_id | int | YES |  | NULL |  |
| class | varchar(2) | YES |  | NULL |  |
| subject\_code | varchar(10) | YES |  | NULL |  |
| start\_time | time | YES |  | NULL |  |
| end\_time | time | YES |  | NULL |  |

## **Table 11: Class**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| class | varchar(2) | NO | PRI | NULL | - |
| class\_rep | int | YES |  | NULL | - |
| teacher\_rep | int | YES |  | NULL | - |

## **Table 12: Marks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| marksheet\_id | int | NO | PRI | NULL | auto\_increment |
| student\_id | int | YES |  | NULL |  |
| subject\_code | varchar(10) | YES |  | NULL |  |
| marks | int | YES |  | NULL |  |
| assignment\_marks | int | YES |  | NULL |  |
| assignment\_solution | varchar(100) | YES |  | NULL |  |
| submission\_time | timestamp | YES |  | NULL |  |

## **Table 13: Address**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| city | varchar(25) | YES |  | NULL | - |
| district | varchar(25) | YES |  | NULL | - |
| state | varchar(25) | YES |  | NULL | - |
| country | varchar(25) | YES |  | NULL | - |
| pincode | varchar(10) | NO | PRI | NULL | - |

**4.2 CODING STANDARDS IMPLEMENTED**

### Naming and Capitalization

Below summarizes the naming recommendations for identifiers in Pascal casing is used mainly (i.e. capitalize first letter of each word) with camel casing (capitalize each word except for the first one) being used in certain circumstances.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | **Case** | **Examples** | **Additional Notes** |
| Class | Pascal | Teacher, Users, Subjects, Marks | Class names should be based on "objects" or "real things" and should generally be **nouns**. No ‘\_’ signs allowed. Do not use type prefixes like ‘C’ for class. |
| Method | Camel | getPincode, setSubjectList | Methods should use **verbs** or verb phrases. |
| Parameter | Camel | teacherId, firstName | Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios. |
| Interface | Pascal with "I" prefix | TeacherDao | Do not use the ‘\_’ sign |
| Property | Pascal | StudentAttendance, AssignmentSolution | Use a noun or noun phrase to name properties. |
| Associated private member variable | Camel | \_ findByMarksheetId, findByStd | Use underscore camel casing for the private member variables |

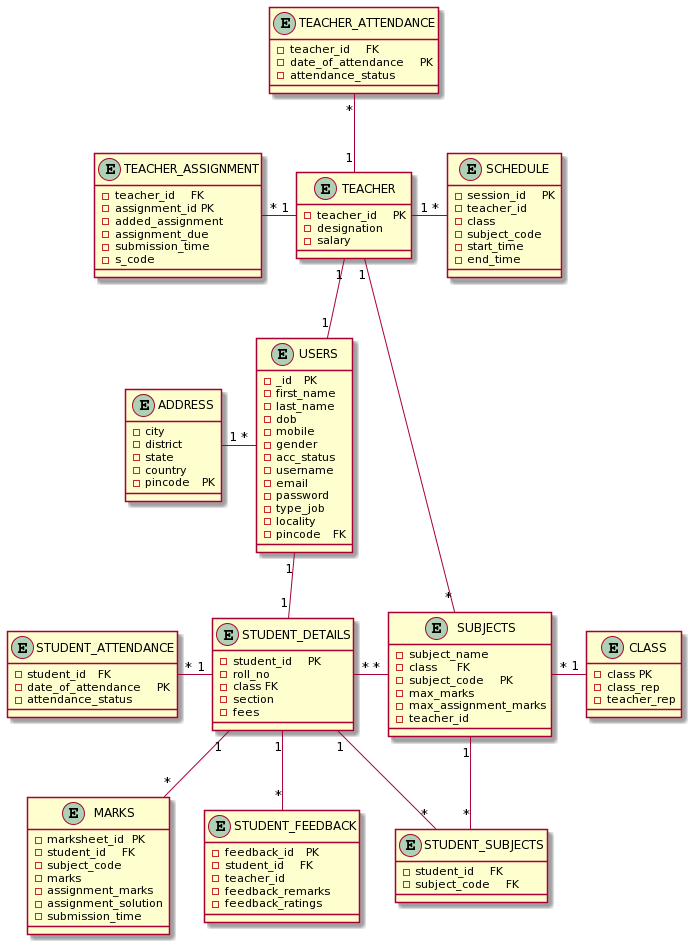
### Comments

* Comment each type, each non-public type member, and each region declaration.
* Use end-line comments only on variable declaration lines. End-line comments are comments that follow code on a single line.
* Separate comments from comment delimiters (apostrophe) or // with one space.
* Begin the comment text with an uppercase letter.
* End the comment with a period.
* Explain the code; do not repeat it.

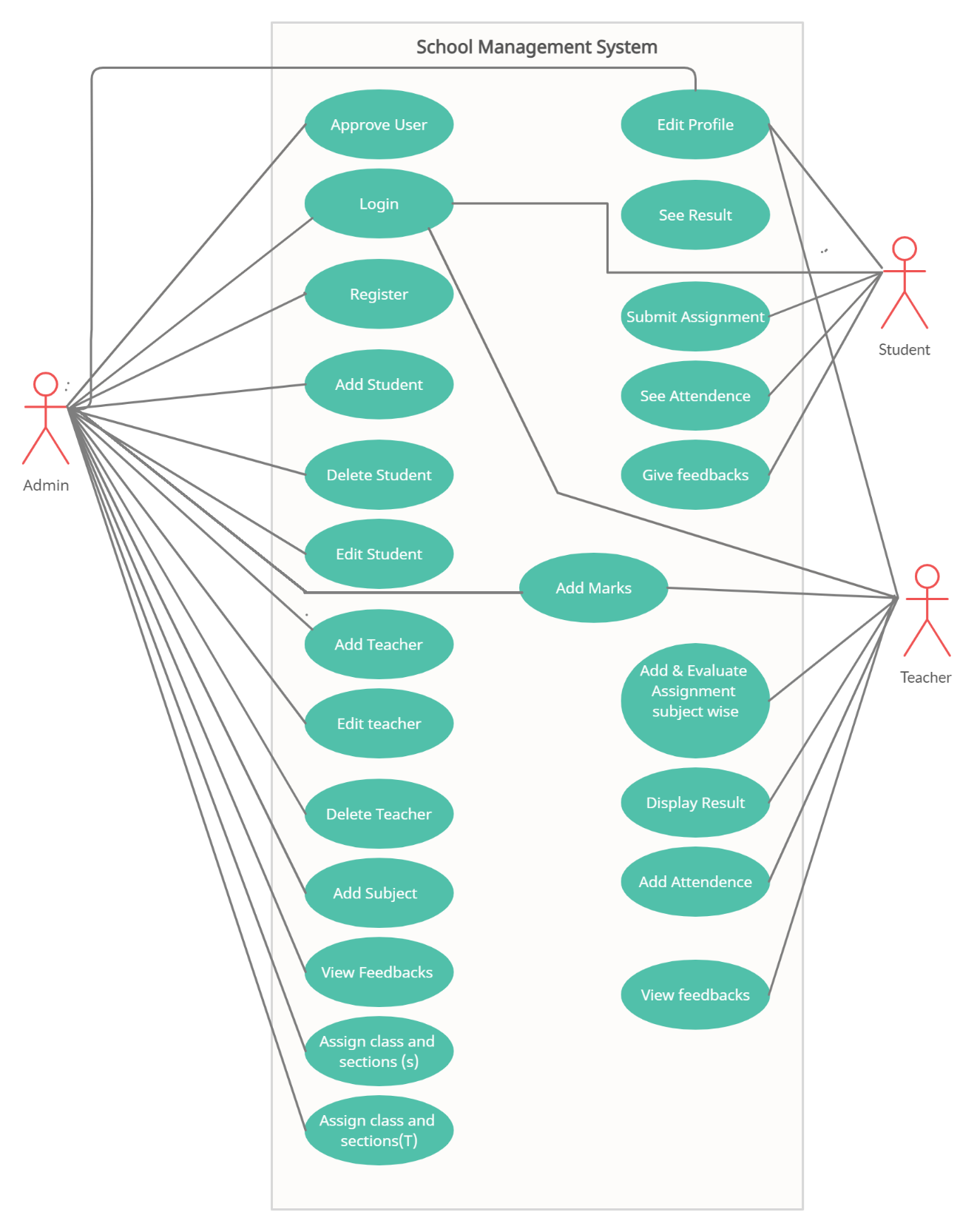
**Chapter 5**

**DIAGRAMS**

**5.1 ENTITY RELATIONSHIP DIAGRAM**



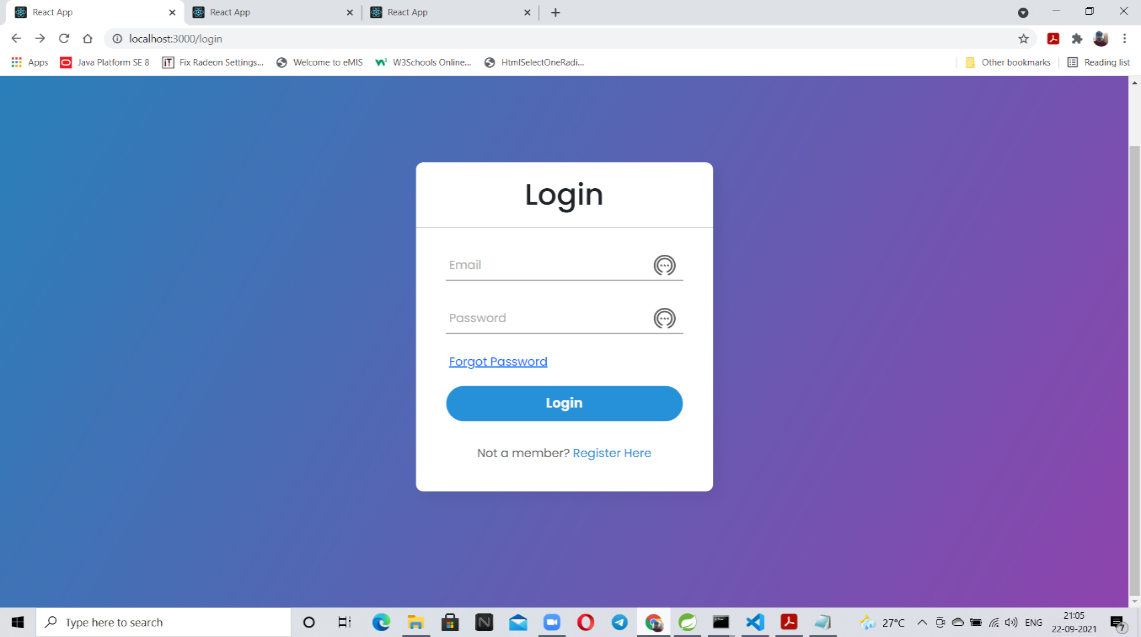
**5.2 USE CASE DIAGRAM**



**Chapter 5**

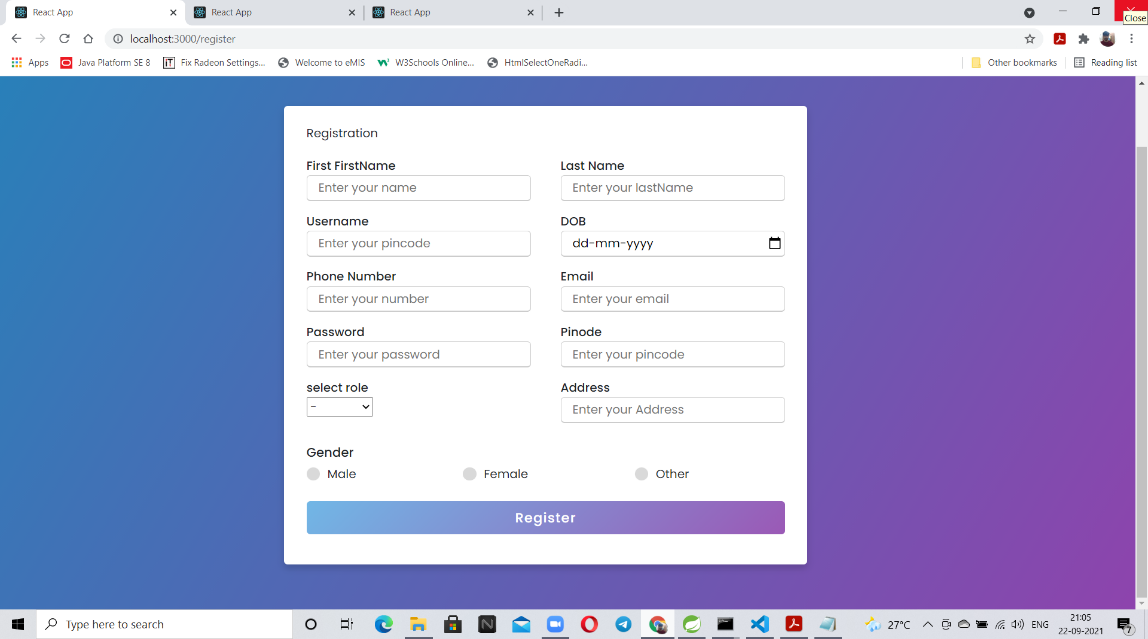
**APPENDIX**

**5.1 USER LOGIN PAGE**



**5.2 USER VERIFICATION PAGE**

**5.3 USER REGISTERATION PAGE**



**5.4 USER UPDATE PAGE**

