SOFTWARE ENGINEERING (IT300)

School Management System

Submitted by:

Akiri Sahithi - 16IT203

Faculty: Dr. Biju R Mohan and Raksha N
Department of Information Technology,
National Institute of Technology-Karnataka
Mangalore, India.

Session 2018-2019

INDEX

1. Introduction 2

2. Software Requirements Specification 3

3. Design 14

4. Work Done

    4.1 Development Environment 30

    4.2 Testing 30

    4.3 Performance of the Application 48

    4.4 Cost Estimation 57

5. Conclusion and Future Work 58

1. INTRODUCTION

1.1 Scope of Work

1.1.1 Project Objectives:

This project is used in schools coordinating among admin, teachers, and parents of the students. This can replace the tedious work of teachers taking attendance on paper and calculating the percentage for each student. This helps increase parent-teacher interaction. It can be used by educational institutes or colleges to maintain the records of students easily. It can solve 3 major issues of Computerization, Automation, and Easy Interaction.

Computerization: All the details regarding the school, whether it is small or big, will be computerized.
No Redundant Data: As this management system will be centralized, the chances of duplicate data in the system are close to nil.
Automation: The automation feature of this management system will mitigate the task of writing papers. E.g. there is no need to write the students' report cards on paper with a pen.It can simply be done online on the system and can be forwarded to the students and their parents.
Easy Interaction: In today's rush hour of life, it is difficult for a parent to go to the school of his / her child every time a teacher calls. With this management system, it will be easier for a parent and a teacher to be in touch every day. As a matter of fact, it will be easier for each individual person who is associated with the system to be in touch as needed.

1.1.2 Project Scope:

The project will consist of three parts, admin, parent, and teacher modules where admin can add all information about students and teachers, parents can view their child's academic details and can interact with teachers, and teachers can upload attendance and grades of students and can also interact with parents.

1.2 Usage Scenarios

Education is the backbone of human society. So, it is important to provide an elegant, strong, and quality education to every generation's youth to make sure that their personality is developed in a better way.

In the offline system, it is an overhead to keep the records related to faculty, student, parents and other school staff on the papers. Everything related to their progress in the system is marked manually. E.g. A report of a student's attendance is generated monthly is shown to his / her parents. Now, a regular student, going to school every day, is marked absent for a day by mistake. It is a burden to take out the register and view the records.
Key Features are

- Multi-User Account System
- Responsive User Interface
- Parent Monitoring Feature
- Profile System
- Exam Marks Management
- Daily Attendance
- Internal Messaging

2. SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

2.1.1 Product Perspective

The School Management System is self-contained. This is used in schools coordinating among admin, teachers, and parents of the students. This can replace the tedious work of teachers taking attendance on paper and calculating the percentage for each student. This helps increase parent-teacher interaction. This Project's student information system provides us with a simple interface for the maintenance of student information. It can be used by educational institutes or colleges to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant, and collecting relevant information may be very time-consuming. It also greatly reduces the number of paper resources needed in attendance and grade data management.

2.1.2 Product Functions

Functionalities of an Admin:

S.No	Function	Description
1	Add Student	The admin can add student information(details) by editing or updating the database
2	Add Teacher	The admin can add teacher information(details) by editing or updating the database

3	Add Course	The admin can add course information by editing or updating database
4	View Details	The admin can view all student, course, teacher details by collecting information from database
5	View Feedback	The admin can view all teacher feedback given by parents by collecting information from database
6	View Suggestions	The admin can view all suggestions given by parents and teachers details by collecting information from database
7	Upload notices/events	The admin can upload events or notices regarding school which all parents and teachers details by collecting information from database
8	Login/Logout	Admin can log in and logout of the application.

Functionalities of a Teacher:

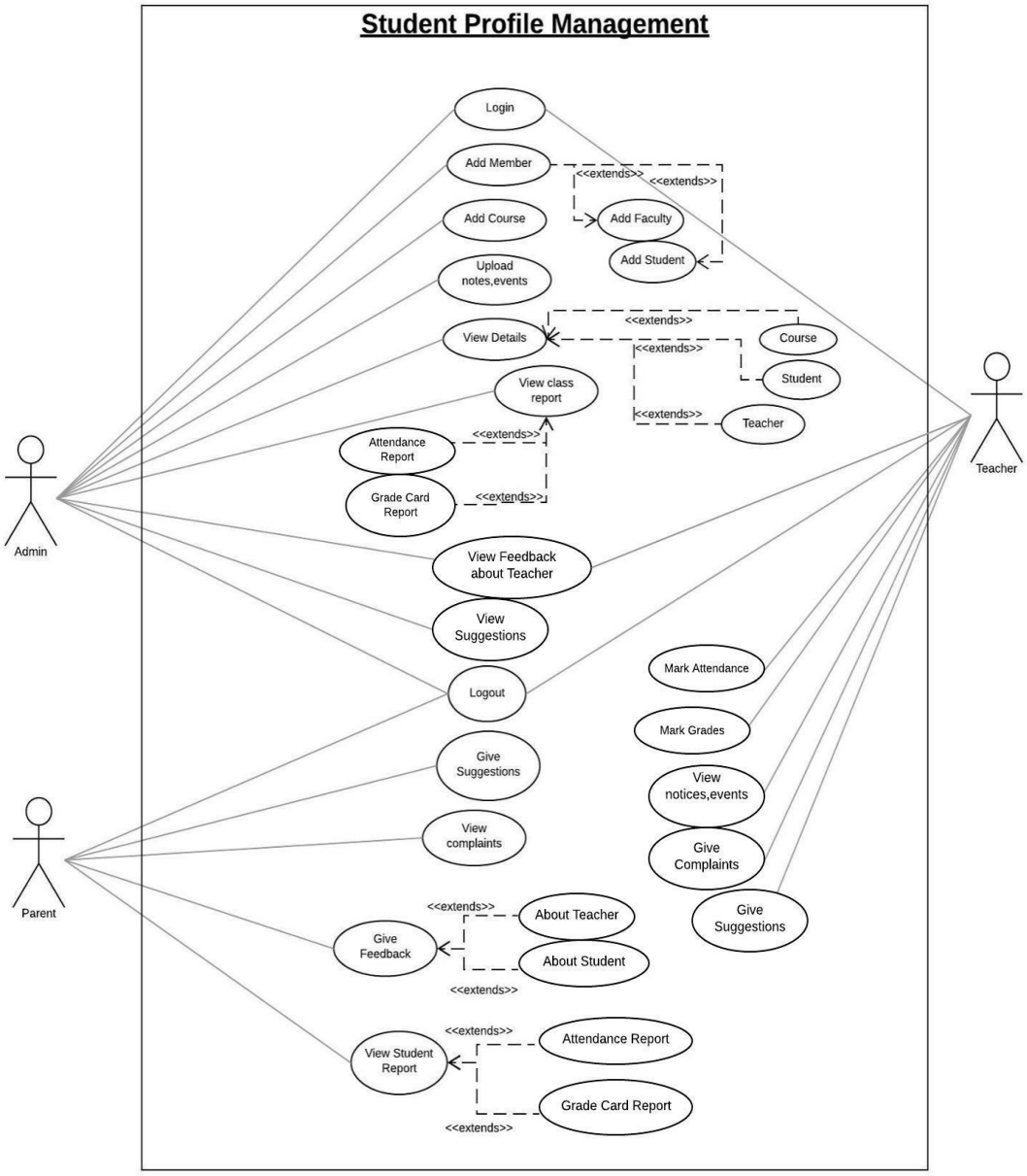
S.No	Function	Description
1	Mark attendance	The teacher can upload students attendance % which will be uploaded in database
2	Mark grades	The teacher can upload students grades which will be uploaded in database
3	View Notices	The teacher should be able view notices/events uploaded by admin by getting information from database
4	Give complaints	The teacher can give complaints about their students which will be uploaded in database
5	Give Suggestions	The teacher can give suggestions about school which will be uploaded in database
6	Login/Logout	The teacher can login and logout from his session

Functionalities of a Parent:

S.No	Function	Description
1	View attendance	The parent should be able to view his child’s attendance % by collecting information from database
2	View grades	The parent should be able to view his child’s grade card by collecting information from database

3	Give feedback	The parent should be able to give feedback about his child's teacher which will be uploaded in database
4	View complaints	The parent should be able to view complaints about his child given by teacher by collecting information from database
5	View notices/events	The parent should be able view notices/events uploaded by admin by getting information from database
6	Give suggestions	The parent should be able to give suggestions about school which will be uploaded in database
7	Login/Logout	The parent can login and logout from their session

2.1.3 User Classes and Characteristics



Student profile management system is a powerful, yet easy program to use so, all users with some basic experience with computers are able to use it efficiently.

- **Parent:**
  They are the guardians of the student and inspect their progress of them and also provide feedback regarding Faculty and teaching methodology.
  A parent should be able to do the following functions:

  1. View student's attendance.

2. View students' grades.
3. Give feedback regarding the faculty and the teaching methodology.
4. View complaints about the student.
5. View notices/events happening in school.
6. Give suggestions about the application or school rules.

- **Teacher:**
  They are the persons who are assigned to teach different courses created by the admin. Teachers evaluate students and keep parents notified about their progress.
  They should be able to:

  1. Mark attendance to the students
  2. Mark grades to the students
  3. View Notices issued by the Admin/Principal
  4. Give complaints if any un-intended behavior is observed among students.
  5. Give Suggestions about the application or school rules.

- **Admin/Principal:**
  The person who handles all over management of the school. - Refers to the superuser who is the Central Authority and has been vested with the privilege to manage the entire system. It can be any higher official in the respected school management or a dean or principal of the school. He can manage all details of the student, teacher, and course.
  They should be able to:

  1. Register a new student to the school
  2. Add a new teacher
  3. Add a course
  4. View details of both existing students and teachers
  5. View feedback given by parents about teachers.
  6. View suggestions about the application or school rules.

2.1.4 Operating Environment

Operating environment for this school management system can be like this:

- Operating system: Windows, Ubuntu
- Database: MySQL
- Distributed database
- Platform: XAMPP

2.1.5 Design and Implementation Constraints

Technology Constraints: Proposed web application can be implemented with PHP/JAVA for back-end design purpose & For the database purpose, we can opt for Mysql/Oracle.

Interface Constraints: Since, this is a Web-based application so it should work on major browsers like Internet explorer, Mozilla Firefox, Google Chrome, Opera, etc.

Safety and Security Constraints: Since the application is intended for authenticated users only, so anonymous persons should not be able to access and operate the user data.

2.1.6 Assumptions and Dependencies

• The users and Admin must have basic knowledge of computers and the English Language.
• Should be connected to the XAMPP server always for functioning.

2.2 External Interface Requirements

2.2.1 User Interfaces

As present-day users are more involved in web interfaces for their daily needs User-friendly interface is definitely necessary. It should be easy to navigate without any learning curve involved. A decent and pleasant appearance with ease of navigation should help users.

Some standard interface designing rules like Schneiderman's golden rules of interface design, and Donald Norman’s design principles are to be followed for an efficient interface.

2.2.2 Hardware Interfaces

The server is directly connected to the client systems. The client systems have access to the database on the server.
Recommended Operating Systems:

- Windows: 7 or newer
- MAC: OS X v10.7 or higher
- Linux: Ubuntu

We strongly recommend a computer fewer than 5 years old.

- Processor: Minimum 1 GHz; Recommended 2 GHz or more
- Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
- Hard Drive: Minimum 32 GB; Recommended 64 GB or more
- Memory (RAM): Minimum 1 GB; Recommended 4 GB or above

2.2.3 Software Interfaces

2.2.3.1 Front End

Client:

The Admin, Parent, Teacher online interfaces are built using HTML (Hyper Text Markup Language) and CSS (Cascading style sheets), PHP, Javascript.

- **HTML - HTML or Hypertext Markup Language** is the main markup language for creating web pages and other information that can be displayed in a web browser.HTML is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start tag, and the second tag is the end tag (they are also called opening tags and closing tags).
- **CSS- Cascading Style Sheets (CSS)** is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to style web pages and interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG, and XUL. CSS is a cornerstone specification of the web and almost all web pages use CSS style sheets to describe their presentation.
- **PHP- PHP is a server-side scripting language** designed for web development but also used as a general-purpose programming language. While PHP originally stood for Personal Home Page, it now stands for PHP: Hypertext Preprocessor, a recursive backronym.PHP code is interpreted by a web server with a PHP processor module, which generates the resulting web page: PHP commands can be embedded directly into an HTML source document rather than calling an external file to process data.
- **JAVASCRIPT**- JavaScript (JS) is a dynamic computer programming language. It is most commonly used as part of web browsers, whose implementations allow client-side scripts to interact with the user, control the browser, communicate asynchronously, and alter the document content that is displayed. It is also being used in server-side programming, game development, and the creation of desktop and mobile applications. JavaScript is a prototype-based scripting language with dynamic typing and has first-class functions. Its syntax was influenced by C.

2.2.3.2. Back End:

The back end is designed using MySQL which is used to design the databases.

- **MYSQL- MySQL** ("My S-Q-L", officially, but also called "My Sequel") is (as of July 2013) the world's second most widely used open-source relational database management system

(RDBMS). It is named after co-founder Michael Widenius's daughter, My. The SQL phrase stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements.

2.2.3 Communications Interfaces

This software package should be securely accessible through internet communication channels(wired or wireless). This project supports all types of web browsers. This contains simple electronic forms for the student and teacher details collection during the process of registration.

HTTP a classic "client-server" protocol is required to reach out to the majority of the audience. Users click a link on their web browser (the client), and the browser sends a request over the internet to a web server that houses the site the user requested. The server sends back the content of the site, such as text and images, which display in users' web browsers.

2.3 System Features

The features of the student management system are:

- Student Attendance List Creation: It allows the admin to create a student attendance sheet consisting of name, roll number, date, Absent/Present mark, and subject. Admin has to fill in student names along with associated roll numbers.

- Profile Marking: The faculty has the list of students. He may see the list of call roll numbers and select absent if the student is absent or select present if the student is present. He can also upload the grades.

- Academic data Storage: This data is now stored in the database. Faculty and admin may also view it anytime.

- Attendance sheet transfer: The faculty can transfer the file to a server (normal computer) where this data can be stored and maintained by the school or college.

- Adding teachers and classes: Admin can add teachers and assign them to subjects.

Database schema structure used in this project is

- Admin table

	#	Name	Type	Collation	Attributes	Null	Default
☐	1	username	varchar(20)	latin1_swedish_ci		No	*None*
☐	2	password	varchar(20)	latin1_swedish_ci		No	*None*

- Teacher table

	#	Name	Type	Collation	Attributes	Null	Default
☐	1	id	int(11)			No	*None*
☐	2	tname	varchar(20)	latin1_swedish_ci		No	*None*
☐	3	tmobile	bigint(11)			No	*None*
☐	4	temail	varchar(20)	latin1_swedish_ci		No	*None*
☐	5	tqualification	varchar(20)	latin1_swedish_ci		No	*None*
☐	6	tpassword	varchar(20)	latin1_swedish_ci		No	*None*
☐	7	taddress	varchar(30)	latin1_swedish_ci		No	*None*
☐	8	Gender	varchar(10)	latin1_swedish_ci		No	*None*

- Student table

#	Name	Type	Collation	Attributes	Null	Default
1	**sid**	varchar(11)	latin1_swedish_ci		No	*None*
2	**sname**	varchar(20)	latin1_swedish_ci		No	*None*
3	**sclass**	int(3)			No	*None*
4	**sphone**	bigint(11)			No	*None*
5	**sparent**	varchar(20)	latin1_swedish_ci		No	*None*
6	**sparent_email**	varchar(20)	latin1_swedish_ci		No	*None*
7	**spassword**	varchar(20)	latin1_swedish_ci		No	*None*
8	**parent_address**	varchar(40)	latin1_swedish_ci		No	*None*

- Grades table

#	Name	Type	Collation	Attributes	Null	Default
1	**sid**	varchar(11)	latin1_swedish_ci		No	*None*
2	**hgrade**	varchar(2)	latin1_swedish_ci		No	*None*
3	**agrade**	varchar(2)	latin1_swedish_ci		No	*None*
4	**course_id**	int(11)			No	*None*
5	**sattendance**	float(5,2)			No	*None*

2.4 Other Nonfunctional Requirements

2.4.1 Performance Requirements

- Product Requirements:
    1. Efficiency Requirements: When a student profile management system will be implemented admin and users should be able to easily access the library as uploading and viewing information will be very faster. Less latency, lower CPU utilization time, Speed index, etc.
    2. Accessibility Requirements: The system should accurately perform member login, member validation, report uploading, viewing information, and search. Each component of the software should be well-defined and visually

compatible with the user.

3. Usability Requirements: The system is designed for a user-friendly environment so that admin, parent, and staff of the school can perform the various tasks easily and in an effective way

- Organisational Requirements:
    1. Implementation Requirements: In implementing the whole system it uses HTML on the front end with PHP as server-side scripting language which will be used for database connectivity and the backend i.e the database part is developed using MySQL

2.4.2 Safety Requirements

There will be unique login ID and passwords for each user. So accessing the personal data of an individual can be prevented.

- Attendance and Scores of a student can be viewed only by his parent after logging in.
- A teacher should log into one's account to update the performance of respective students.
- Admin has to log in to make announcements so that no other user can make false announcements.

2.4.3 Security Requirements

Information transmission should be securely transmitted to the server without any changes in information. As the system provides the right tools for problem-solving it must be made sure that the system is reliable in its operations and for securing sensitive details.

The system should provide a secure login to the users by using advanced secure login algorithms and provide access only to authorized users as security is the key requirement of this system. This system uses a database application, hence the 'Weak Authentication' threat should be taken care of.

2.4.4 Software Quality Attributes

System should be:-

- Consistent in performance with respect to interface or actions required to perform the tasks.
- Safe and Secure functioning of the application over various domains.
- Robust in nature to cope with errors that can occur while using it.
- Scalable: As the server has to support three modules of Admin, Parent and Teacher
- Flexible: Modules designed should support any future possible re-use.
- User friendly: To make users easier and intuitive usage of the system.

- Efficient: In the implementation of a system to use resources in an optimal way.
- Inter-operable: Across various platforms and technologies.
- Upgradable: To any changes that are wished to be inculcated in the future.
- Available all the time.
- Maintainability: The administrators should be able to maintain the system

2.4.5 Other Requirements

- Organisational Requirements: In implementing the whole system it uses HTML on the front end with PHP as server-side scripting language which will be used for database connectivity and the backend i.e the database part is developed using MySQL

3. SYSTEM DESIGN

3.1 Design Goals

The application comprises many features and hence the system is divided into various components. The main objective of this section is to elaborate on the system design and to give an overview of the various components of the application including their interfaces. It also provides information about the relationship between the various components and the different data elements used by each of the components. It also explains the overall system design. The application has a client-server architecture with the application running on the client side and the files residing on the server side with which the user interacts through the application. The following sections contain class diagrams, sequence diagrams, and activity diagrams representing the various components and their interactions as also a detailed description of each of the components.

3.2 System Architecture

3.2.1 Class Diagram

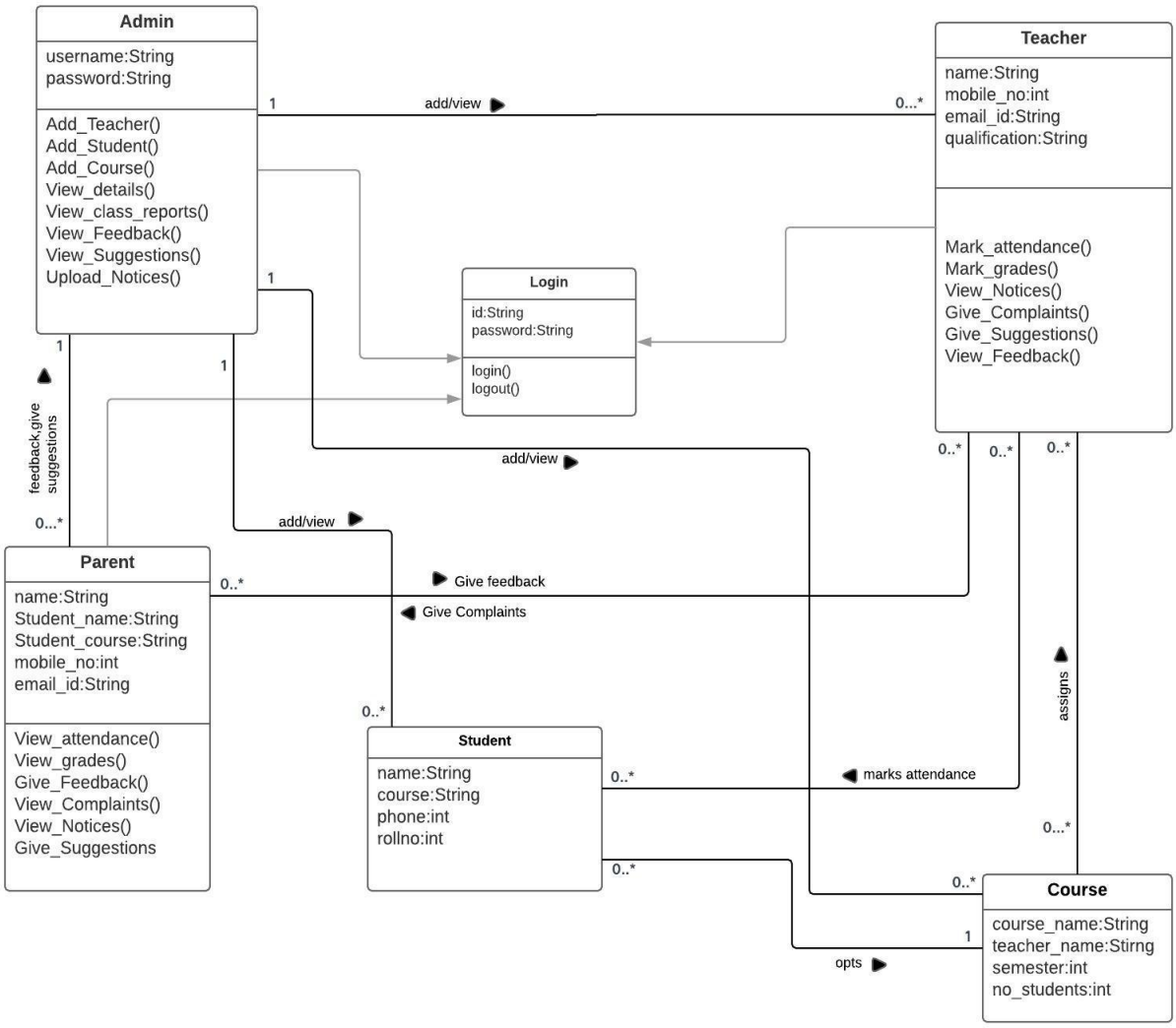


Fig Class Diagram of School Management System

3.2.2 Data Flow Diagrams

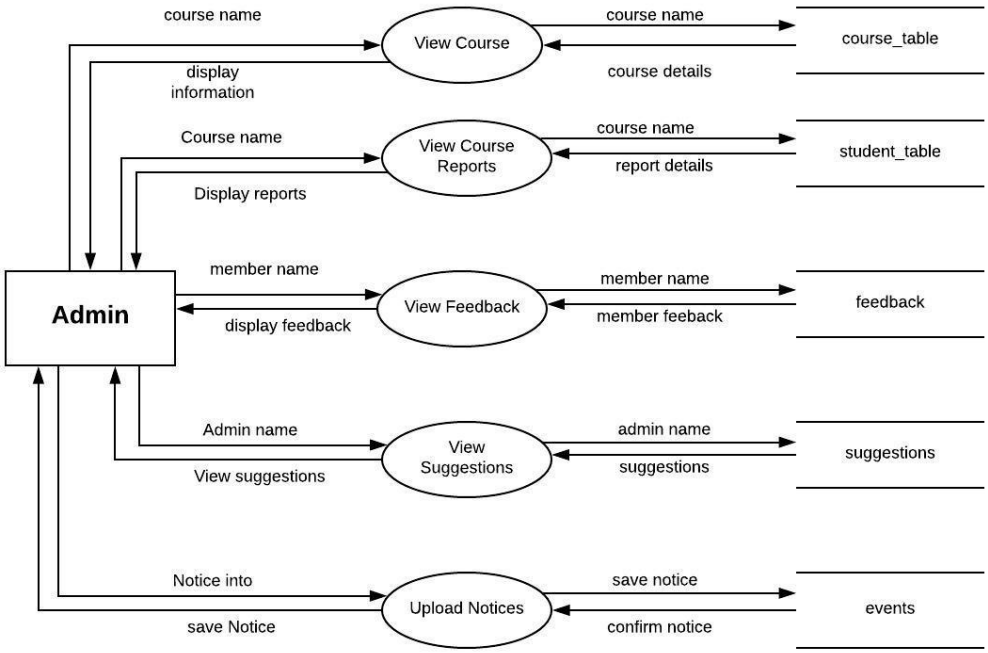


Fig Data-flow Diagram of Admin in School Management System

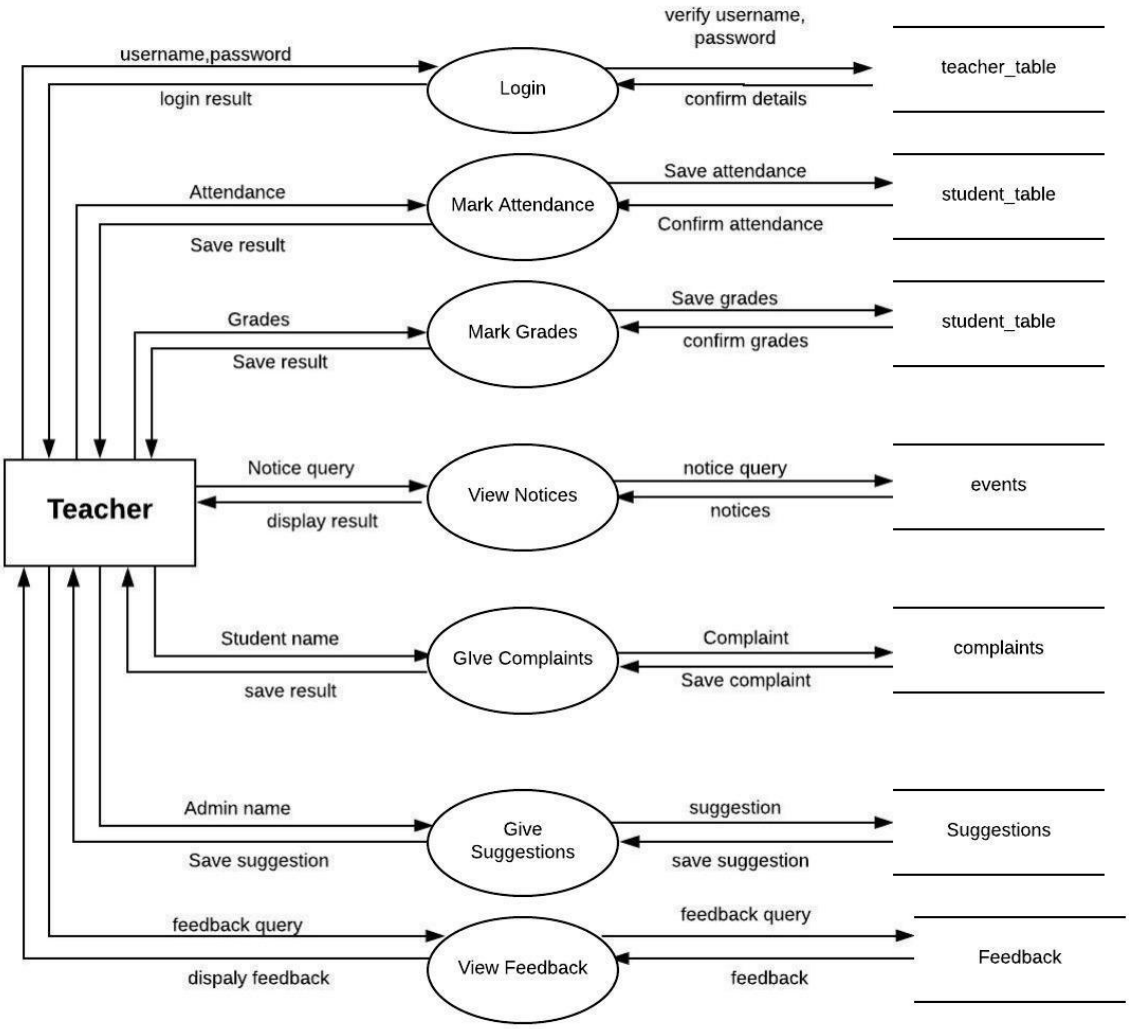


Fig Data-flow Diagram of Teacher in School Management System

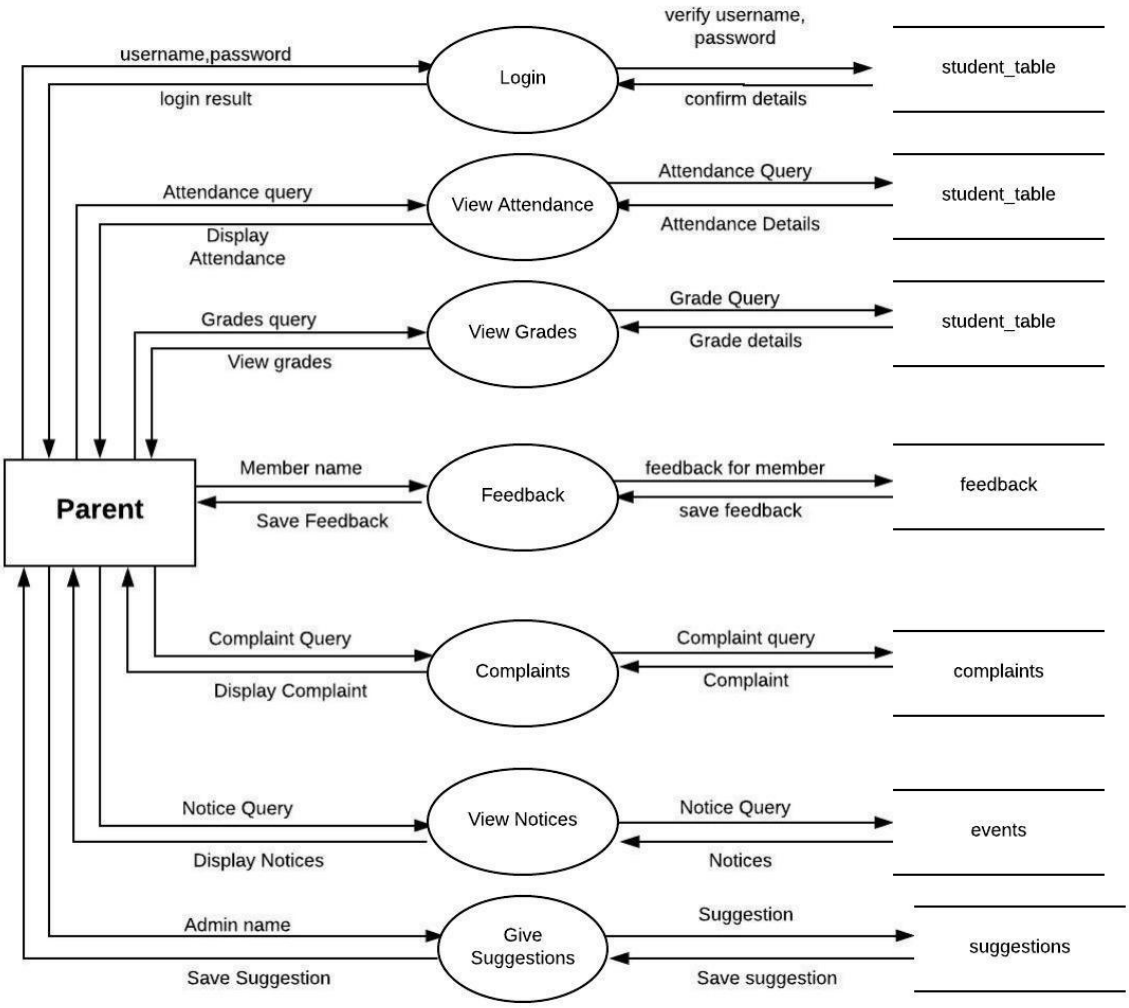


Fig Data-flow Diagram of Parent in School Management System

3.2.3 Navigation Diagram

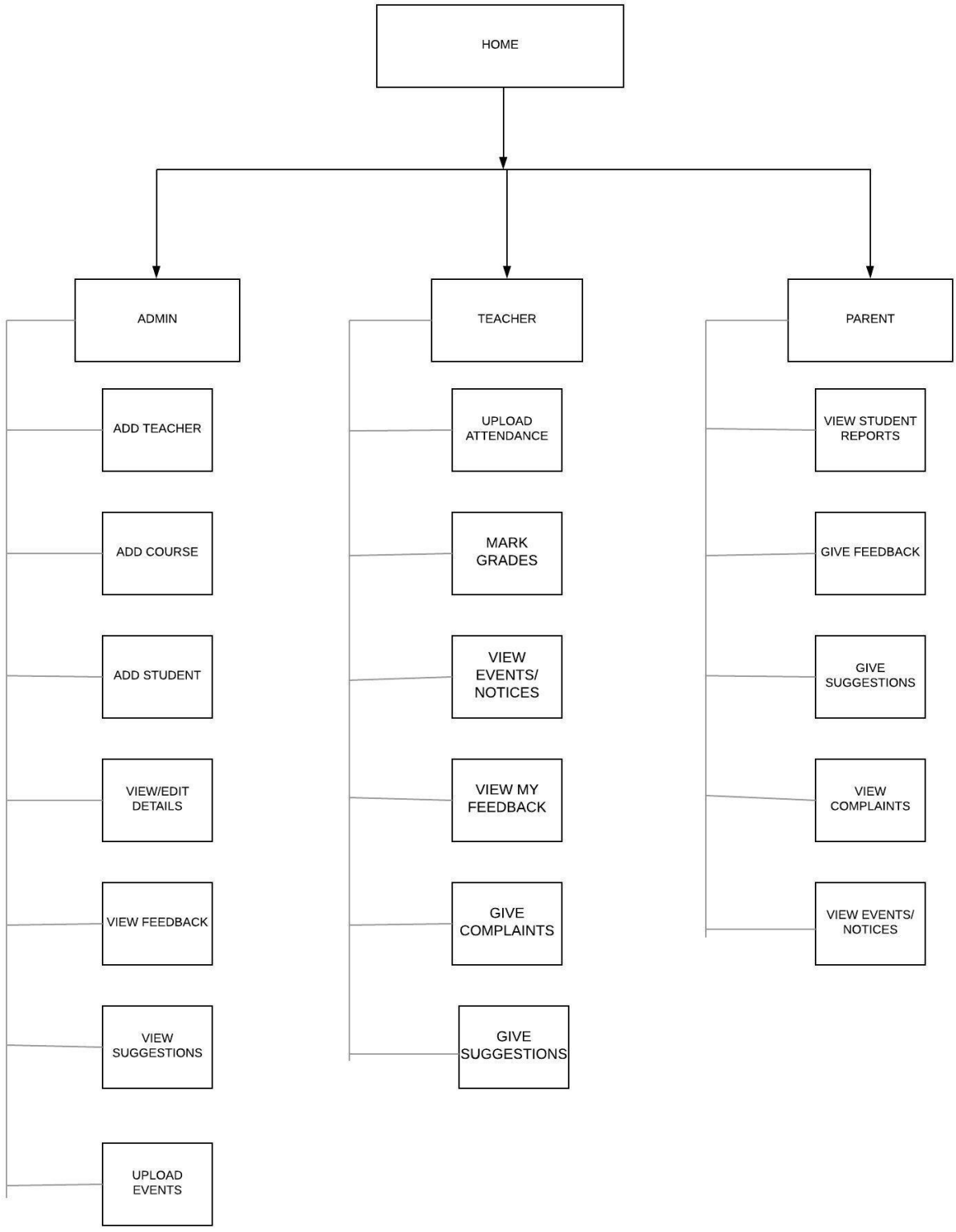


Fig Navigation Diagram of School Management System

3.2.4 Component Diagram

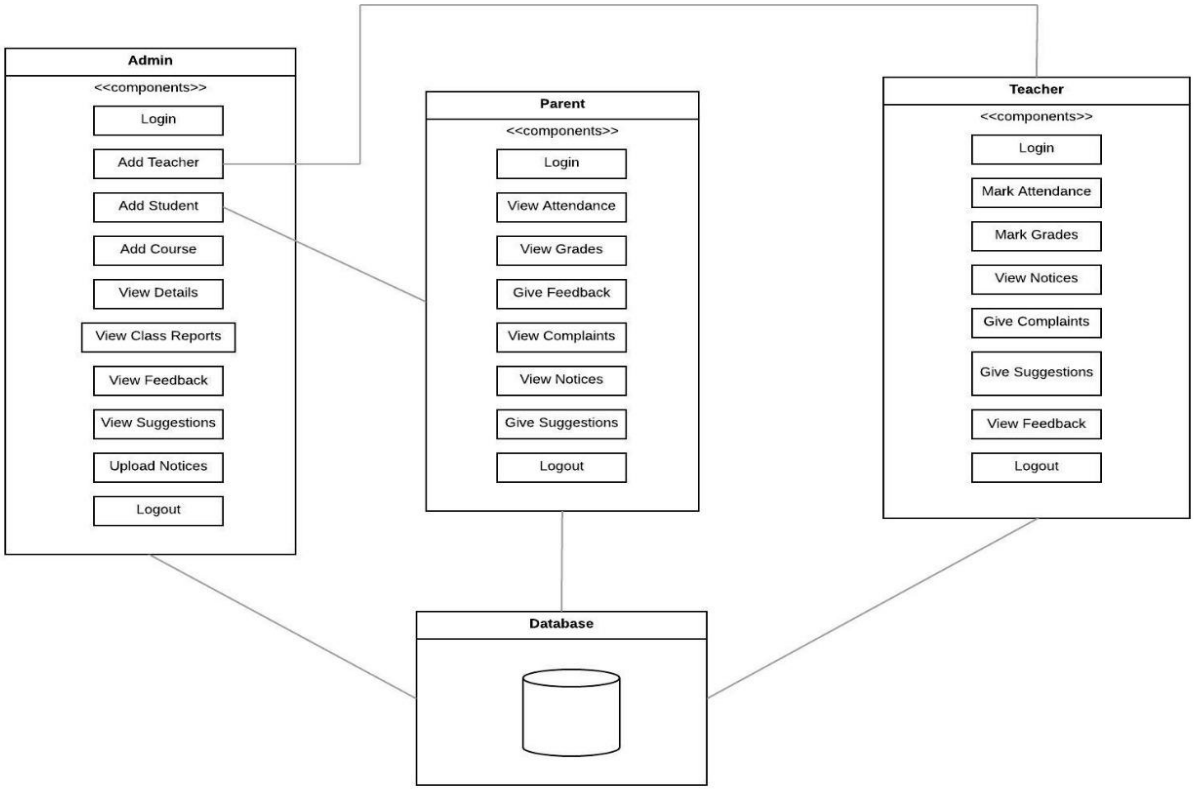


Fig Component Diagram of School Management System

3.2.5 ER Diagram

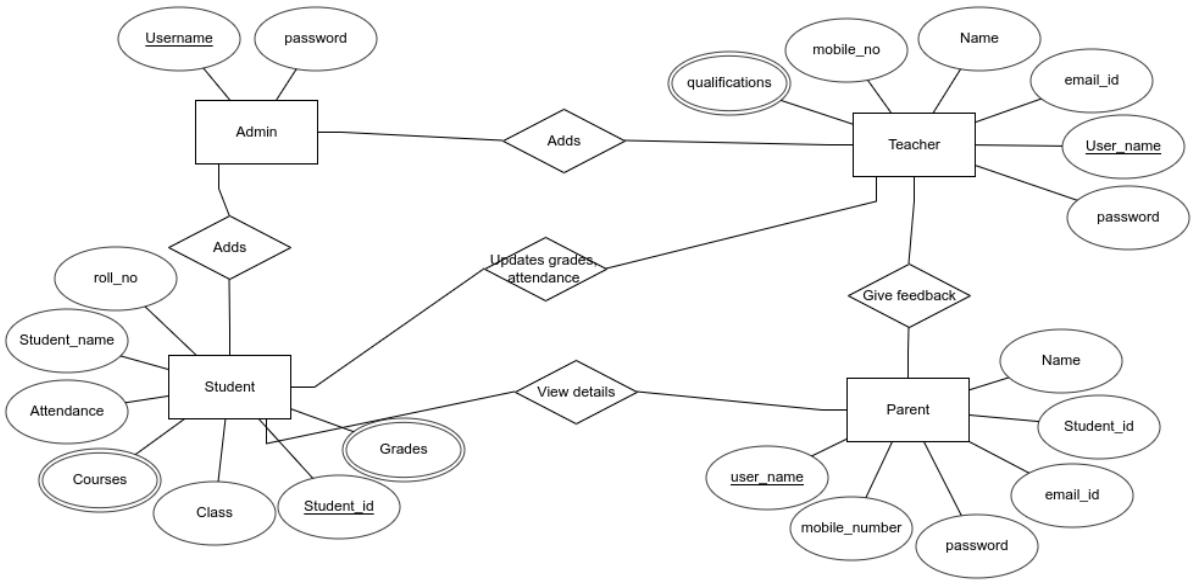


Fig ER Diagram of School Management System

3.3 Interaction Diagrams

3.2.1 Sequence Diagram

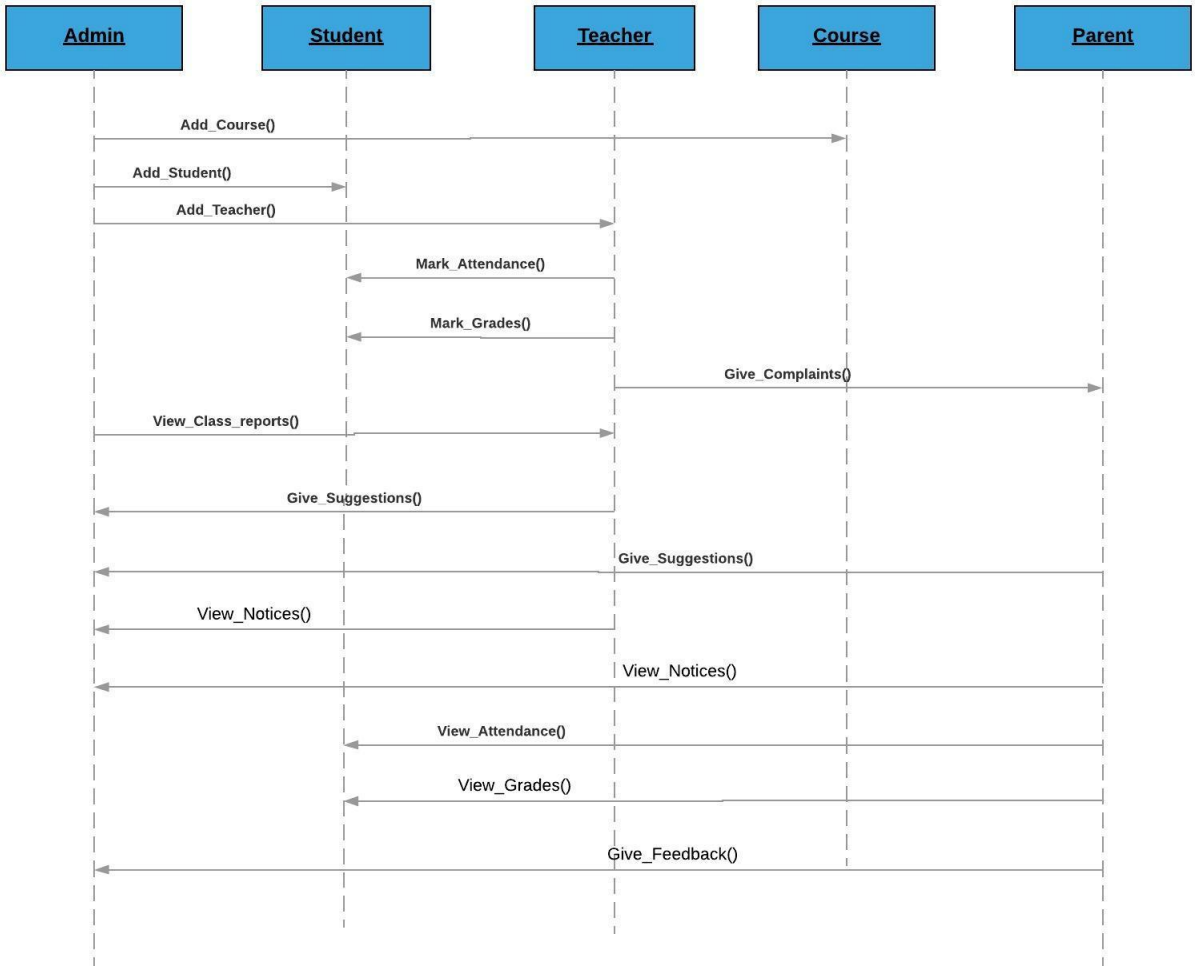
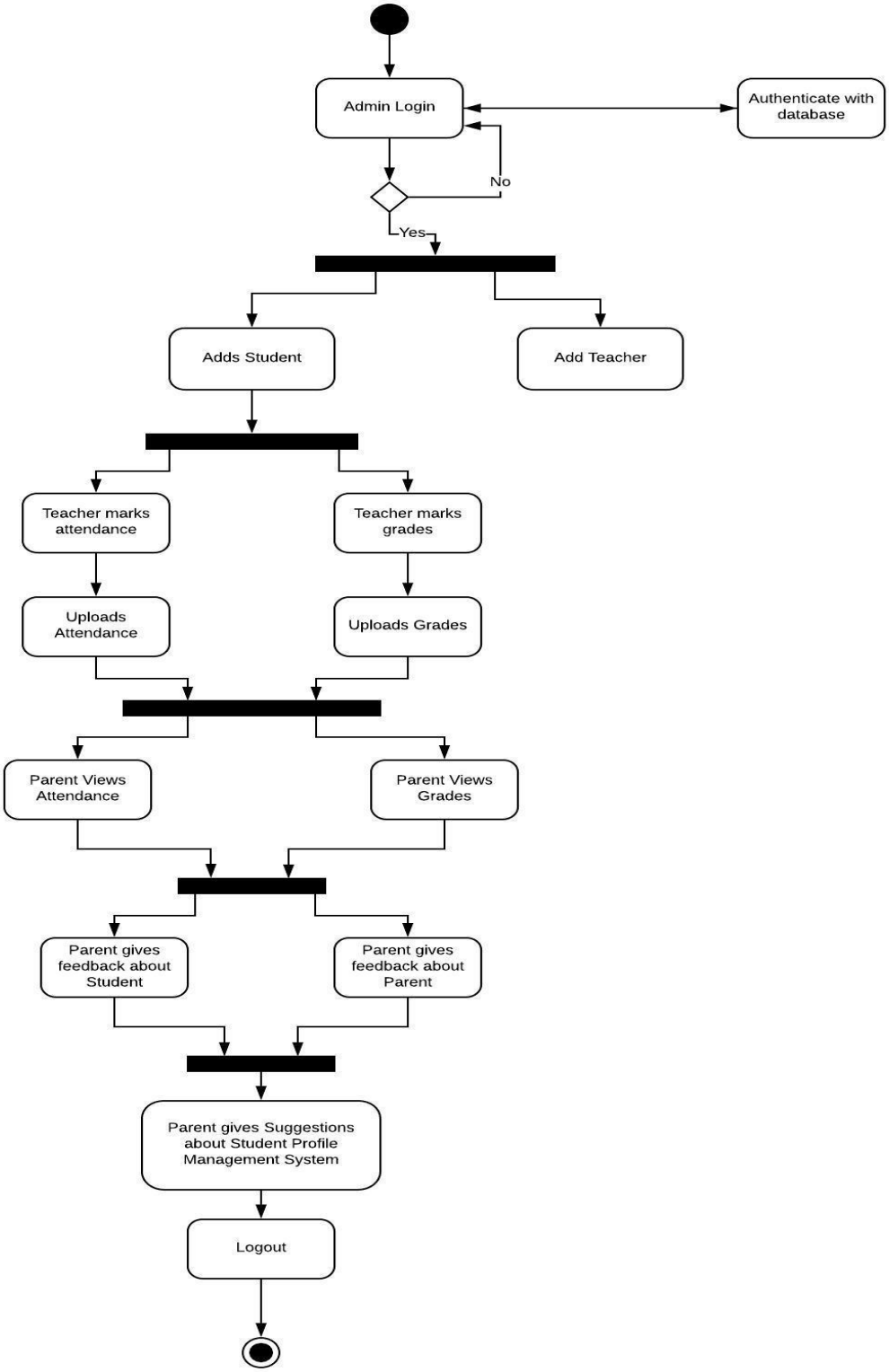


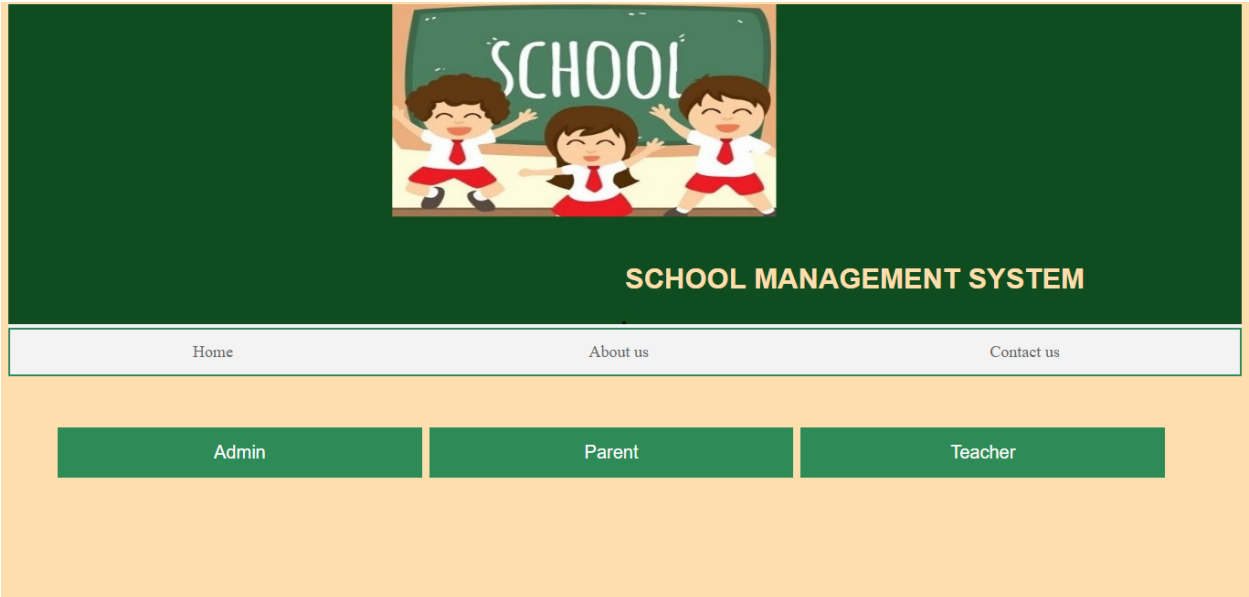
Fig. Sequence Diagram for desired functionalities

3.2.2 Activity Diagram

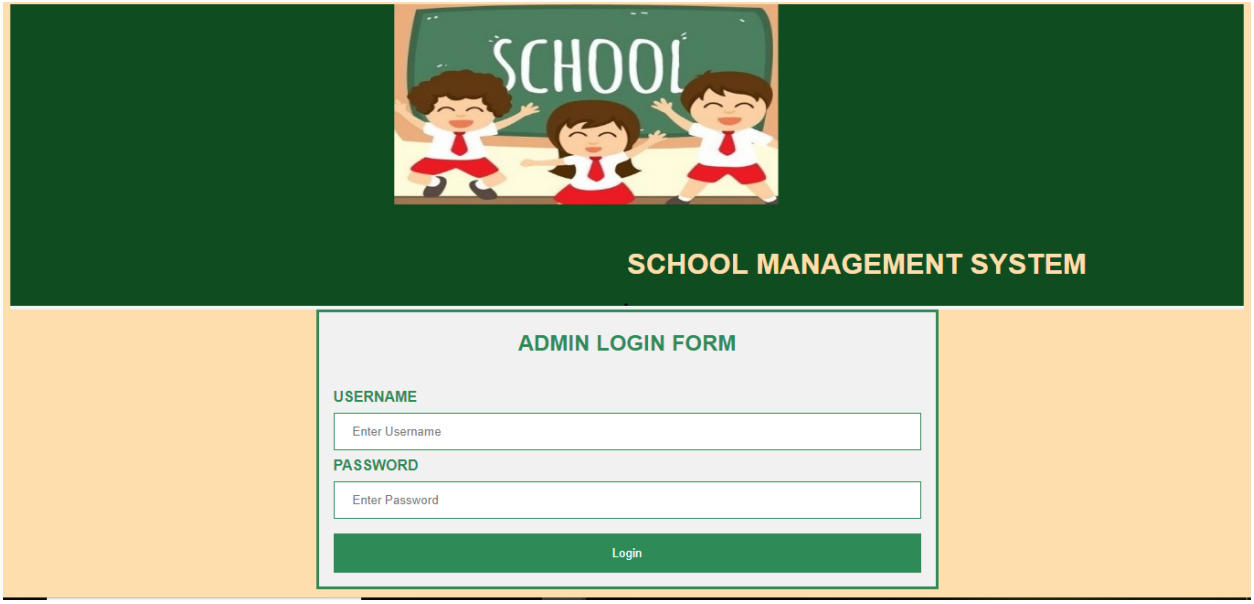


3.4 Detailed Design Methodologies

3.4.1 Detailed Description of the User Interface



Admin Interface:



SCHOOL MANAGEMENT SYSTEM

ADMIN PROFILE

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

ADD TEACHER

Teacher-Name: Enter Teacher Name

Mobile-no: Enter Mobile-no

Email: Enter Email

Qualification: Enter Qualification

Gender: Male Female

Address: Enter Address

ADD TEACHER

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

ADD COURSE

Course-name: Enter course name

Teacher: Select Teacher

Class: Enter Class

ADD

ADD STUDENT

Student-Name	Enter Student Name
Class	Enter Class
Mobile-No	Enter Mobile-no
Parent-Name	Enter Parent-name
Parent-Emailid	Enter Parent's Email-id
Gender	Male Female
Address	Enter Address

ADD STUDENT

Student details

S.No	Student Name	Course	Parent Name	Parent-Mobile	Parent-Email	Edit Details
1	Prameela	10	Ramesh	8889996661	ramesh@gmail.com	Edit details
1	Rishi	1	Gowtham	7654321890	gowtham.v@gmail.com	Edit details
1	Jahnavi	2	Ganesh	7901234568	ganesh@gmail.com	Edit details
1	ram	3	rajesh	9493141091	rajesh.rk@gmail.com	Edit details
1	Si Kavya	4	Sanjeev	8901234567	sanjeev@gmail.com	Edit details
1	Anusha	5	Mahesh	9012345678	mahesh@gmail.com	Edit details
1	Shreya	6	Koteswara Rao	9959905837	koti@gmail.com	Edit details
1	Ajay	7	Anand	77890123456	anand@gmail.com	Edit details
1	Sowmya	8	Vinod	8890123456	vinod@gmail.com	Edit details
1	Arnav Singh	9	Ajith Singh	9998887776	ajith@gmail.com	Edit details

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

Teacher details

S.No	Teacher Name	Qualification	Phone-no	Email	Edit details
1	Radha Rani	MSc	9999999999	radha.rani@gmail.com	Edit details
2	Karthik	BSc., B.Ed	9876543210	karthik@gmail.com	Edit details
3	Gowtham Kumar	B.Tech	9876567890	gowtham@gmail.com	Edit details
4	Ravi Varma	BBA	9123456780	ravi@gmail.com	Edit details
5	Padma	MBA	8123456790	padma@gmail.com	Edit details
6	Srinivas	MSc., B.Ed	9876123450	srinu@gmail.com	Edit details
7	Hema Latha	MSc., M.Phil	7654321890	latha@gmail.com	Edit details
8	Indira Devi	BSc., B.Ed	7890654321	indira@gmail.com	Edit details

Course details

S.No	Course	Teacher Name	Class	Edit
1	social	Radha Rani	3	Edit details
2	social	Radha Rani	1	Edit details
3	social	Radha Rani	2	Edit details
4	science	Karthik	3	Edit details
5	science	Karthik	1	Edit details
6	social	Radha Rani	4	Edit details
7	social	Radha Rani	5	Edit details
8	social	Gowtham Kumar	6	Edit details
9	social	Gowtham Kumar	7	Edit details
10	social	Gowtham Kumar	8	Edit details
		Gowtham		

SCHOOL MANAGEMENT SYSTEM

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

SUGGESTIONS

S.No	Suggestions
1	Kindly leave the students early in the evening
2	Ensure teachers do not beat students hardly.

Add Teacher | Add Course | Add Student | View/Edit Details | View Feedback | View suggestions | Upload Events | Logout

Upload Events/Notices

Event Date: 01 / dd / yyyy

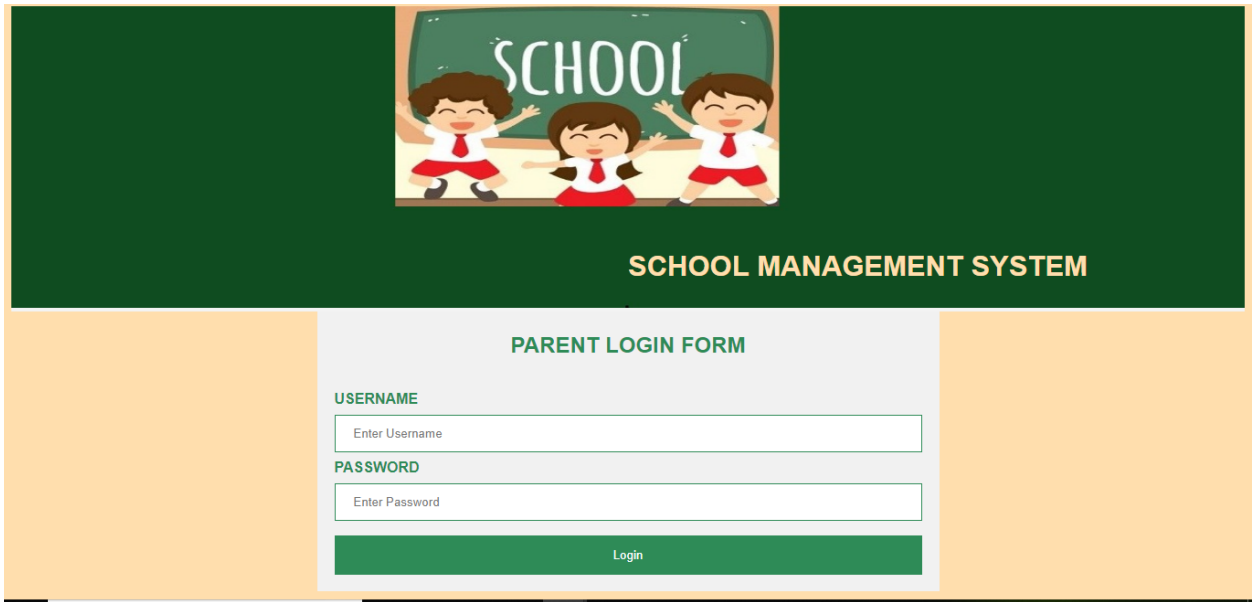
April 2019

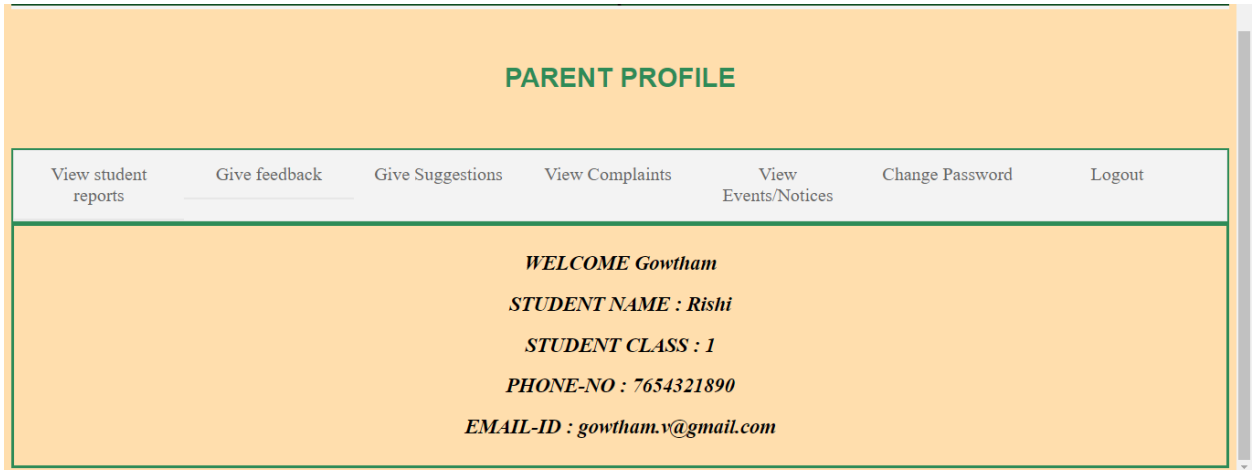
Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4

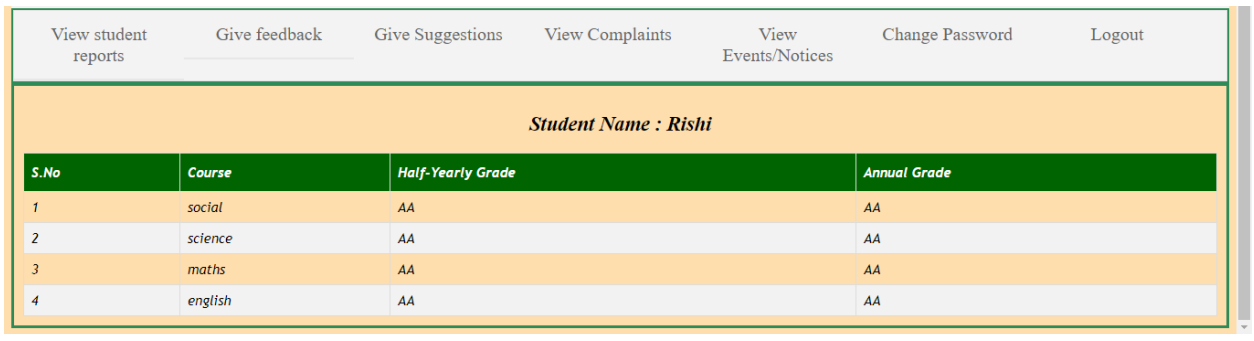
Event/Notice :

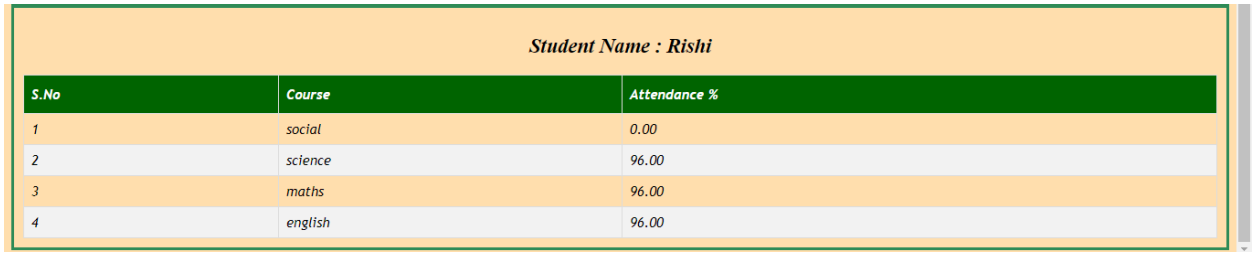
SUBMIT

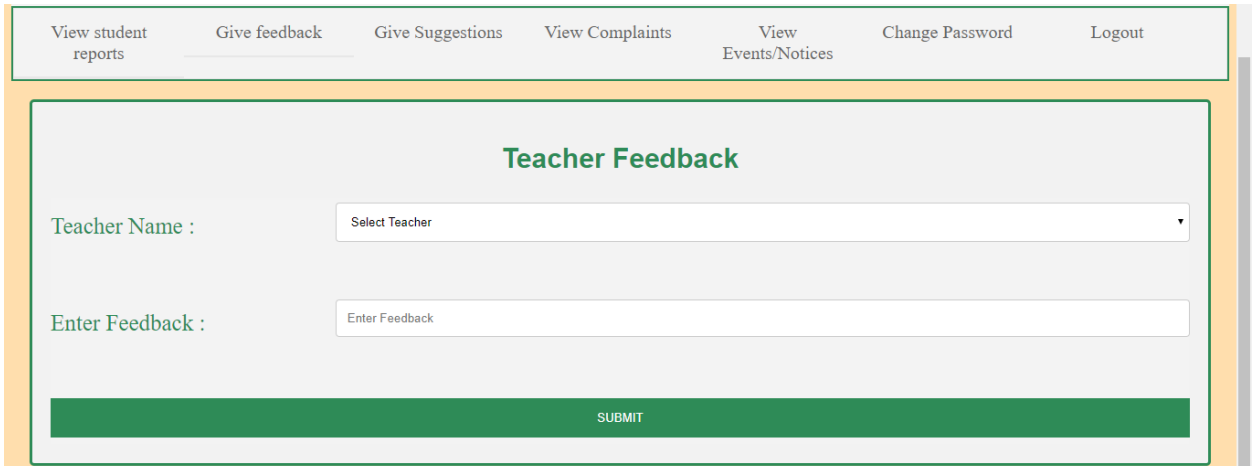
Parent Interface:

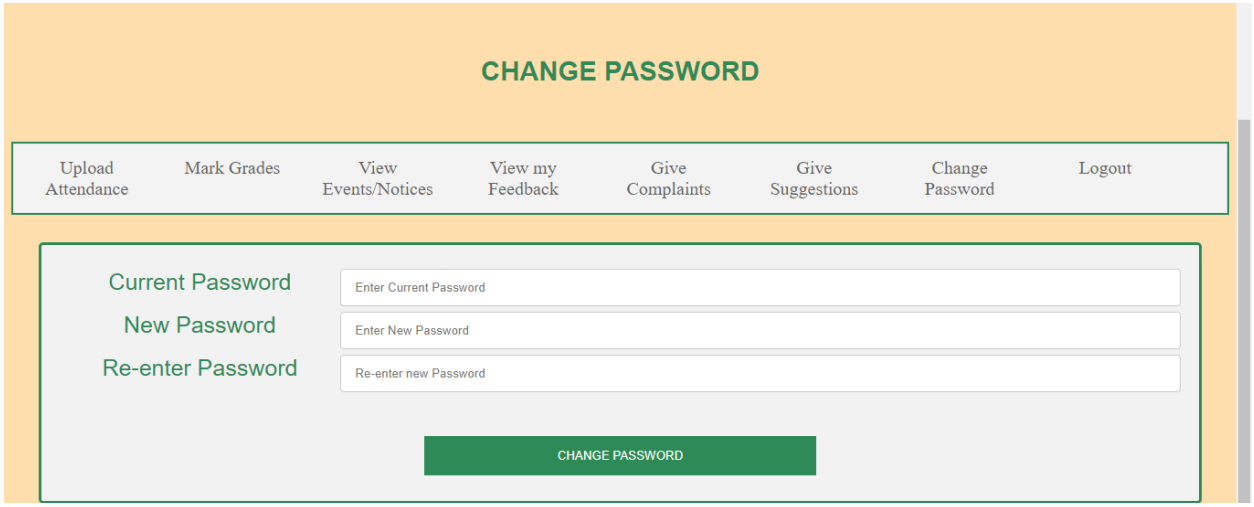




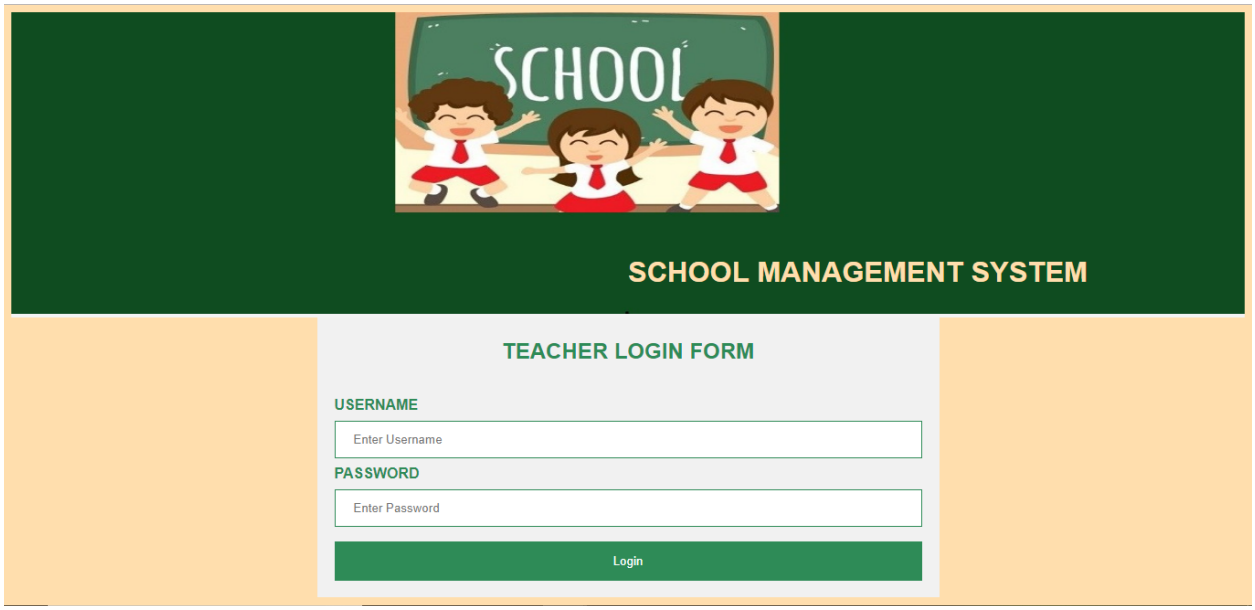








Teacher Interface:



Upload Attendance%

S.No	Student Name	Upload Attendance%
3_1	ram	Mark Attendance
1_1	Rishi	Mark Attendance
2_1	Jahnavi	Mark Attendance
4_1	Si Kavya	Mark Attendance
5_1	Anusha	Mark Attendance

Upload Grades

S.No	Student Name	Upload Grade
3_1	ram	Mark Grades
1_1	Rishi	Mark Grades
2_1	Jahnavi	Mark Grades
4_1	Si Kavya	Mark Grades
5_1	Anusha	Mark Grades

Upload Attendance | Mark Grades | View Events/Notices | View my Feedback | Give Complaints | Give Suggestions | Change Password | Logout

Complaint Box

Student Name : Enter Student Name

Roll No : Enter Roll No

Complaint : Enter Complaint

SUBMIT

4. WORK DONE

4.1 Development Environment

To implement the project, the operating system used was windows. The server XAMPP was used to run the php files and Jenkins is used as development tool. PhpMYAdmin(SQL) database was used.

4.2 Testing

What is testing?

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

- A Test case is a sequence of steps to test if a functionality/feature of an application is working correctly.

4.2.1 Manual Test-case Design

Characteristics of a test case:

1. A good test case is one that has a high probability of detecting an as−yet undiscovered error.
2. Test cases must be written for :
    - Invalid and Unexpected input conditions
    - Valid and Expected input conditions
3. No redundancy
4. Not too simple or too complex

The test case should be accurate, economical, traceable, repeatable, reusable.

Why is testing done? To know

- Which modules have been tested?
- How many modules or features are stable?
- Which modules need more work according to the number of defects found in respective modules?
- Are sufficient input combinations exercised and covered in test cases?
- Does the app give out correct error messages if the user does not use it the way it was intended to be used?
- Is the app tested on a different browser or different platforms for mobile to test compatibility?
- Does the UI conform to the specifications?
- Are the features traceable to the requirement spec? Have all of them been covered?

- Are the user scenarios traceable to the use case document? Have all of them been covered?
- Are the test cases good enough? Are they finding defects?
- Is software ready to move to production? Is testing enough?
- And finally, what is the quality of the application?

Test case Template Description:

1. **Test Focus**: To identify what type of feature of the test case is testing
   Ex: Functionality, Validation, etc.
2. **Test Description**: The purpose of the test case and what feature of the unit it is testing.
3. **Test Type:** Indicates whether the test case checks the functional requirement or field validation or UI design or non-functional requirement.
4. **Pre and Postconditions**: What is required from the tester before executing this test case
5. **Test Steps**: Steps describing exactly how the test should be carried out
6. **Test Data**: Input data which is required to execute the test cases
7. **Expected Results**: The expected behavior of the Unit after the execution of the test case
8. **Actual Results**: The actual behavior observed after the test case is Executed
9. **Test Results**: PASS if the expected and actual results match. Fail if they don't match.
10. **Remark:** Any comments on the test case or test execution.
11. **Test Environment**: The environment (Hardware/Software/Network) in which the test was executed.

1. Test case: Login Testing

S.No.	Test element	Observation
1.	Test Focus	User login functionality checking.
2.	Test Description	To check whether the web application allows a user to login with a wrong password.
3.	Test Type	Functionality check
4.	Pre and Post conditions	Pre: ● A user is expected to be allotted with a username and password beforehand. ● If a user is admin, it should be tested on the admin login page. Same is the case with parent and teacher. Post: ● Direct back to the login page again.
5.	Test steps	● Go to School Management System home page. ● Choose an actor and it directs to the login page of the actor. ● Enter the username which is the email provided to the user. ● Enter a wrong password which doesn't match with the password stored. ● Now, click the login button provided at bottom of the page.
6.	Test data	1. Username: sekhar@gmail.com 2. Password: sekhar
7.	Expected Result	Should not log in the user.
8.	Actual result	Did not log in the user into the home page and also informed that the credentials are wrong.
9.	Test Status	PASS
10.	Remarks	The application not only denied the invalid login but also informed the user about the wrong credentials entered. This increases the usability of the system.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

2. Test case: Form validation of name

S.No.	Test element	Observation
1.	Test Focus	Validation of input name
2.	Test Description	To verify whether the name column takes any invalid characters other than alphabets and spaces in Add Teacher functionality in the Admin portal.
3.	Test Type	Field Validation check
4.	Pre and Post conditions	● Admin should be given login credentials to perform this action. ● Admin has to login to the system.
5.	Test steps	● Go to School Management System home page. ● Choose Admin as an actor and it directs to the login page of the actor. ● Then log in and direct to the admin page. ● Select 'Add Teacher' functionality in the navigation bar. ● Enter the details needed to add a teacher. ● Click 'Add Teacher' button at the bottom of the page.
6.	Test data	1. Teacher name: Seetha Sri5678 2. Mobile No: 9886296865 3. Email: seetha.sri@gmai.com 4. Qualification: B.Ed 5. Gender: Female 6. Address: Srinivas Nagar, Surathkal, NITK.
7.	Expected Result	The application should not accept and sustain such invalid entries.
8.	Actual result	Did not add the teacher with given details and gave notification that name should contain only letters.
9.	Test Status	PASS
10.	Remarks	Specific reason and format required are stated.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

3. Test case: Form validation of email

S.No.	Test element	Observation
1.	Test Focus	Validation of input email
2.	Test Description	To verify whether the email column takes any format in Add Teacher functionality in the Admin portal.
3.	Test Type	Field Validation check
4.	Pre and Post conditions	● Admin should be given login credentials to perform this action. ● Admin has to login to the system.
5.	Test steps	● Go to School Management System home page. ● Choose Admin as an actor and it directs to the login page of the actor. ● Then log in and direct to the admin page. ● Select 'Add Teacher' functionality in the navigation bar. ● Enter the details needed to add teacher. ● Click 'Add Teacher' button at the bottom of the page.
6.	Test data	1. Teacher name: Seetha Sri 2. Mobile No: 9886296865 3. Email: seetha.sri@gmail 4. Qualification: B.Ed 5. Gender: Female 6. Address: Srinivas Nagar, Surathkal, NITK.
7.	Expected Result	The application should not accept and sustain such invalid entries.
8.	Actual result	Did not add the teacher with given details and gave notification that email is invalid and requested to enter a valid email
9.	Test Status	PASS
10.	Remarks	Specific reason and format required are stated.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

4. Test case: Validation for change in password

S.No.	Test element	Observation
1.	Test Focus	Validation of input password in change password module.
2.	Test Description	To verify whether password column in change password takes length less than 8 characters or not in both teacher and parent modules.
3.	Test type	Field validation check and length check
4.	Pre and Post conditions	Pre: ● An authorized teacher and parent is logged into the system. ● He/She has navigated to 'Change Password' page
5.	Test steps	● Go to School Management System home page. ● Choose Teacher/Parent as an actor and it directs to the login page of the actor. ● Then log in and direct to their home page. ● Select 'Change Password' functionality in the navigation bar. ● Enter a new password. ● Click 'Change Password' button at the bottom of the page.
6.	Test data	1. New Password: Jessie 2. Re-enter Password: Jessie
7.	Expected Result	The application should not accept and sustain such invalid entries and ask the user to enter a valid password of length >= 8.
8.	Actual result	The application didn't accept given a password which is of length < 8.
9.	Test Status	PASS
10.	Remarks	The change password is working properly.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

5. Test case: For uploading events by admin

S.No.	Test element	Observation
1.	Test Focus	Upload events checking
2.	Test Description	To check if the interface takes in events and date of events inputted by admin and show it at teacher and parent Interface.
3.	Test type	Functionality check
4.	Pre and Post conditions	Pre: ● User/admin is authorised. ● He has navigated to upload events page. Post: ● An authorized teacher or parent should be able to view the uploaded events at their interface.
5.	Test steps	1. Goto school management home page and click on admin module. 2. Login into admin module using a valid username and password. 3. Navigate to the Upload events page. 4. Upload date and description of event.
6.	Test data	● Date: 20/4/2019 ● Event: School Annual Day
7.	Expect Result	An authorized parent and teacher should be able to view the uploaded event with event description and date at their interface.
8.	Actual result	The authorised parent and teacher are able to view the inputted at their respective interface.
9.	Test Status	PASS
10.	Remarks	The upload events function in the application is working well.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

6. Test case: To check upload attendance page

S.No.	Test element	Observation
1.	Test Focus	Upload attendance checking
2.	Test Description	An authorized teacher uploads the attendance of students of her class and the respective parents of students should be able to view grades and attendance of their children.
3.	Test type	Functionality check
4.	Pre and Post conditions	Pre: ● The respective teacher is authorized. ● He/She has navigated to upload grades/attendance page. Post: ● An authorized parent after logging in into their portal should be able to view their child's attendance which implies teacher has entered grades correctly.
5.	Test steps	1. Goto school management page and login click on teacher login. 2. Log in into teacher module with a valid username and password. 3. Navigate to upload attendance page 4. Click on respective student's details and upload grades/attendance.
6.	Test data	● Attendance = 70%
7.	Expected Result	An authorized parent of the student should view the attendance.
8.	Actual result	An authorized parent of a student is able to view attendance.
9.	Test Status	PASS
10.	Remarks	The upload attendance module in the application is working properly.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

7. Test case: To upload suggestions by parents

S.No.	Test element	Observation
1.	Test Focus	To test if suggestions box takes an empty string as input
2.	Test Description	To test if the suggestions box in parents module can take an empty string as input or not
3.	Test Type	Null Check
4.	Pre and Post conditions	Pre: ● An authorized parent is signed in. ● He/She has navigated to 'Give Suggestions' page. Post: ● Enters an empty string as input to the suggestion box.
5.	Test steps	1. Goto school management system home page. 2. Log in into parent module using a valid username and password. 3. Go to 'Upload suggestions' page. 4. Enter empty string as suggestion. 5. Click upload suggestion.
6.	Test data	● Suggestion = ""(Empty string)
7.	Expected Result	The website shouldn't accept those invalid details and give messages as invalid input.
8.	Actual result	The website didn't accept an empty string as input.
9.	Test Status	PASS
10.	Remarks	The module won't accept empty details for required fields.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

8. Test case: To test if empty password is valid

S.No.	Test element	Observation
1.	Test Focus	To test if empty password is valid or not
2.	Test Description	To test if an empty password entered by user is taken by system
3.	Test Type	Form Validation Check
4.	Pre and Post conditions	Pre: ● A user who has a valid username Post: ● He should not be able to login into the system
5.	Test steps	1. Go to school management system page and click on login as admin or parent or student. 2. Enter a username and empty password
6.	Test data	● Username = “Valid Username” ● Password = “”(Empty String)
7.	Expected Result	The website shouldn’t accept those empty strings and give messages as invalid input.
8.	Actual result	The website didn’t accept an empty string as input.
9.	Test Status	PASS
10.	Remarks	The module won't accept empty details for required fields.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

9. Test case: To check upload grades page

S.No.	Test element	Observation
1.	Test Focus	Upload attendance checking
2.	Test Description	An authorized teacher uploads the grades of students of her class and the respective parents of students should be able to view grades and attendance of their children.
3.	Test type	Functionality check
4.	Pre and Post conditions	Pre: ● The respective teacher is authorized. ● He/She has navigated to upload grades page. Post: ● An authorized parent after logging in into their portal should be able to view their child's grades and attendance which implies teacher has entered grades correctly.
5.	Test steps	5. Goto school management page and login click on teacher login. 6. Log in into teacher module with a valid username and password. 7. Navigate to upload grades page 8. Click on respective student's details and upload grades/attendance.
6.	Test data	● Half Yearly grades = AB ● Annual grades = AA
7.	Expected Result	An authorized parent of the student should view the grades.
8.	Actual result	An authorized parent of a student is able to view both annual and half-yearly grades.
9.	Test Status	PASS
10.	Remarks	The upload grades module in the application is working properly.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

10. Test case: To check View Complaints module

S.No.	Test element	Observation
1.	Test Focus	View complaints uploaded by teachers
2.	Test Description	An authorized teacher uploads the complaints of students of her class and the respective parents of students should be able to view complaints about their children if any.
3.	Test type	Functionality check
4.	Pre and Post conditions	Pre: ● The respective teacher is authorized. ● He/She has navigated to view complaints page and has uploaded grades. Post: ● An authorized parent after logging in into their portal should be able to view complaints about their child's which implies teacher has entered complaints correctly.
5.	Test steps	1. Goto school management page and login click on parent login. 2. Log in into parent module with a valid username and password. 3. Navigate to view complaints page 4. View Complaints.
6.	Test data	● Student RollNo : 3_1 ● Complaint : "Sleeping in class".
7.	Expected Result	An authorized parent of the student should view the complaints.
8.	Actual result	An authorized parent of a student is able to view complaints entered.
9.	Test Status	PASS
10.	Remarks	The uploaded complaints and view complaints module in the application is working properly.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

11. Test case: Check Updated password.

S.No.	Test element	Observation
1.	Test Focus	To check if the updated password works.
2.	Test Description	When a user changes their password after logging in, whether the change is reflected or not for the next log in.
3.	Test Type	Functionality Check
4.	Pre and Post conditions	Pre: ● User should be given registered by the Admin . ● User must login to the system.
5.	Test steps	1. Goto school management page and login click on teacher login. 2. Log in into parent module with a valid username and password. 3. Click on Change Password button in the navigation pane. 4. Enter your current password, your new password and re-enter the new password.
6.	Test data	● Current password: ramesh1970 ● New password: ramesh45 ● Re-enter new password: ramesh45
7.	Expected Result	The password should be updated and allow the user to login with the new password.
8.	Actual result	The web app allowed the new password to login the user.
9.	Test Status	PASS
10.	Remarks	The change of password is reflected .
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

12. Test case: Verify Add Teacher .

S.No.	Test element	Observation
1.	Test Focus	To check if the Teacher is added by Admin .
2.	Test Description	When Admin adds a Teacher profile, whether a profile is created with the given details or not.
3.	Test Type	Functionality Check
4.	Pre and Post conditions	Pre: ● Admin must gather the necessary details of the Teacher.
5.	Test steps	1. Goto school management page and select Admin profile, login to application. 2. Select ‘Add Teacher’ functionality. 3. Enter the details of Teacher. 4. Click Add Teacher button.
6.	Test data	● Teacher name: Seetha Sri ● Mobile No: 9886296865 ● Email: seetha.sri@gmail.com ● Qualification: B.Ed ● Gender: Female ● Address: Srinivas Nagar, Surathkal, NITK.
7.	Expected Result	The Teacher details should be added in the database of ‘School Management System’, ‘teacher_table’
8.	Actual result	Teacher details are added.
9.	Test Status	PASS
10.	Remarks	The Teacher details are reflected in database indicates success of functionality .
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

13. Test case: Verify Add Student .

S.No.	Test element	Observation
1.	Test Focus	To check if the Student is added by Admin .
2.	Test Description	When Admin adds a Student profile, whether a profile is created with the given details or not.
3.	Test Type	Functionality Check
4.	Pre and Post conditions	Pre: ● Admin must gather the necessary details of the Student.
5.	Test steps	1. Goto school management page and select Admin profile, login to application. 2. Select ‘Add Student’ functionality. 3. Enter the details of Student and some details of Parent. 4. Click Add Student button.
6.	Test data	● Student name: Dinesh Kumar ● Student Class: 3 ● Gender: Male ● Mobile No: 9886296865 ● Parent Name: Ramesh Kumar ● Email: ramesh.1970@gmail.com ● Address: Srinivas Nagar, Surathkal, NITK.
7.	Expected Result	The Student details should be added in the database of ‘School Management System’, ‘student_table’
8.	Actual result	Student details are added.
9.	Test Status	PASS
10.	Remarks	The Student details are reflected in database indicates success of functionality .
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

14. Test case: Verify Add Course.

S.No.	Test element	Observation
1.	Test Focus	To check if the Course is added by Admin .
2.	Test Description	When Admin adds a new Course, whether a new course with given details is generated or not.
3.	Test Type	Functionality Check
4.	Pre and Post conditions	Pre: ● Admin must decide upon the Course and Teacher to who it has to be assigned.
5.	Test steps	1. Goto school management page and select Admin profile, login to application. 2. Select ‘Add Course’ functionality. 3. Enter the details of Course and select Teacher. 4. Click Add Course button.
6.	Test data	● Course Name: English ● Course Class: 6 ● Teacher Name: M. Divya Sree
7.	Expected Result	The Student details should be added in the database of ‘School Management System’, ‘course_table’
8.	Actual result	Course details are added.
9.	Test Status	PASS
10.	Remarks	The Course details are reflected in database indicates success of functionality .
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

15. Test case: To test if an unauthorised user can login or not

S.No.	Test element	Observation
1.	Test Focus	To test if an unauthorized user can log in or not.
2.	Test Description	To test if an empty password entered by the user is taken by system.
3.	Test Type	Functionality check
4.	Pre and Post conditions	Pre: ● A user has access to the website
5.	Test steps	1. Go to school management system page and click on login as admin or parent or student. 2. Enter a random username and password.
6.	Test data	● Username = “random username” ● Password = “Random input”
7.	Expected Result	The website shouldn’t accept those strings and give messages as invalid input and user should not be able to log in.
8.	Actual result	The website didn’t accept any random username and password.
9.	Test Status	PASS
10.	Remarks	The module won't allow any invalid user to login into the system.
11.	Test Environment	● OS: Windows 10 ● Browser: Chrome ● Local Server: Xampp

4.2.2 Automated Testing

Automation testing is a technique uses an application to implement entire life cycle of the software in less time and provides efficiency and effectiveness to the testing software.

Selenium automation tools is used for this application.Selenium is a browser automation tool, commonly used for writing end-to-end tests of web applications. A browser automation tool does exactly what you would expect: automate the control of a browser so that repetitive tasks can be automated.

Following test cases are implemented on the Selenium tool and run on Eclipse IDE.

Test Cases:

- **Login with Wrong password:**
  When a user tries to Log In using a wrong password, the application should not direct to a login page, instead it has to be informed that password is invalid.

  Inputs:
  1. Username: radha.rani@gmail.com
  2. Password: radharani7

  Output:
  Invalid Login credentials.
  Test passed

- **Field Validation of Name:**
  In the functionality of Add Teacher, Teacher name has to be only in Alphabets and Spaces. Any characters other than that should be detected and informed.

  Inputs:
  1. Name: Keerthi Sri5
  2. Phone Number: 9886296865
  3. Email: keerthi.k@gmail.com
  4. Qualification: BEd.
  5. Gender: Female
  6. Address: Srinivas Nagar, Surathkal.

  Outputs:
  Invalid Name
  Test passed

- **Field Validation of Email-ID:**
  In the functionality of Add Student, Parent email has to be in the standard format of email. Any incomplete or invalid formats other than that should be detected and informed.

  Inputs:
  1. Name: Vasanth Kumar
  2. Gender: Male
  3. Class: 5
  4. Phone Number: 9886296865
  5. Father's Name: Kiran Kumar
  6. Email: kumar.vasanth
  7. Address: Srinivas Nagar, Surathkal.

  Output:
  Invalid Email
  Test passed

- **Field Validation of Phone Number:**
  In the functionality of Add Student, Parent phone number has to be in the standard format of phone number. Using of alphabets, special characters, or in incomplete form should be detected and informed.
  Inputs:
  1. Name: Vasanth Kumar
  2. Gender: Male
  3. Class: 5
  4. Phone Number: 12345
  5. Father's Name: Kiran Kumar
  6. Email: kumar.vasanth@gmail.com
  7. Address: Srinivas Nagar, Surathkal.

  Output:
  Invalid phone number
  Test passed

- **Confidentiality Check:**
  In case a user tries to open a page that corresponds to an action which is to be allowed only to a specific user, system should direct to the login page directly and thus protecting the confidentiality of the users.
  Output:
  Test passed

- **Navigation Check:**
  This test is done in order to check the connectivity between a series of pages. From one page six other pages are opened one after the other in a row. Then the final page is compared to the expected page. This confirms the navigation between the pages.
  Output:
  Test passed

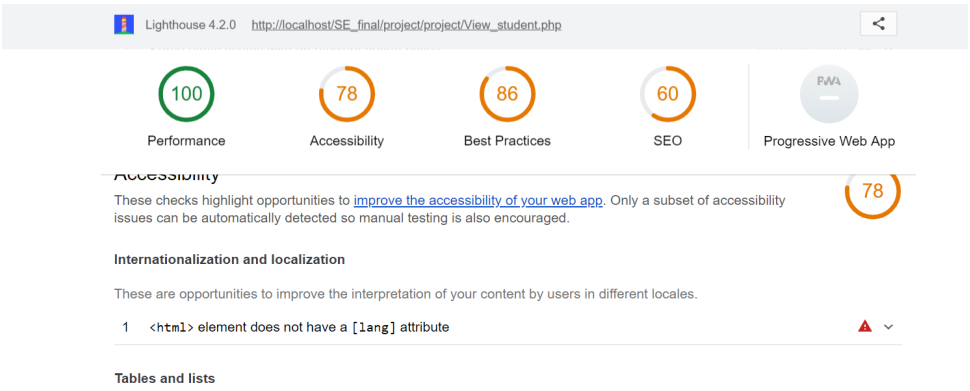
4.3 Performance of the Application

Performance of the application is tested using "**Google Lighthouse**". The improvements made after testing are as following:

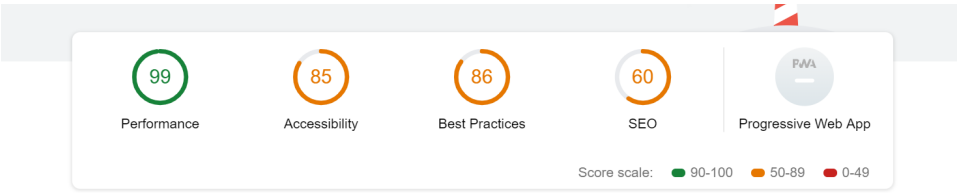
1. Adding lang attribute:

The value of "lang" attribute represents the primary language of document. This is important because, if the language of a webpage is not specified, the screen reader assumes the default language set by the user resulting in a strange accent. It is essential to specify a language and ensure that it is valid so website text is pronounced correctly.

The accessibility increased from 78% to 85% after adding 'lang' attribute to html tag.



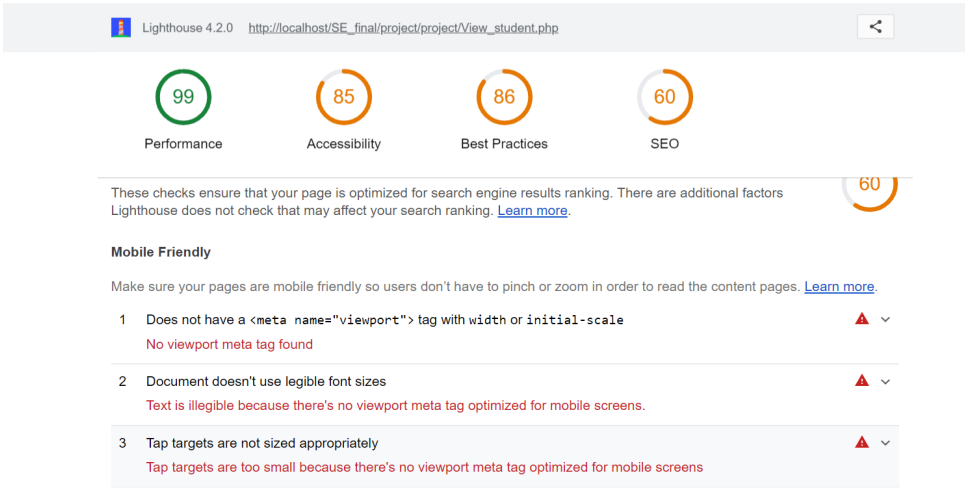
After adding:



2. Adding viewport meta tag:

The viewport is the user's visible area of a web page. The viewport varies with the device, and will be smaller on a mobile phone than on a computer screen. So viewport meta tag is added to set it dynamically according to device size.

The SEO is increased from 60% to 90% after adding the meta viewport tag.

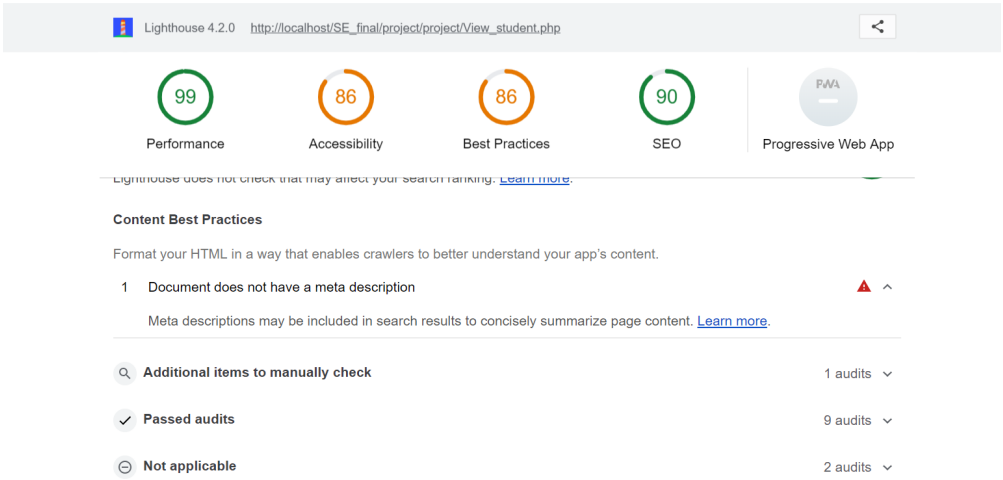


After adding:

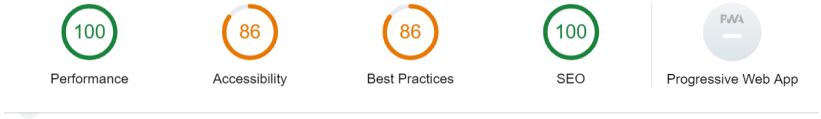
3. Adding description meta tag:

The “description” meta tag includes the details of the website.

SEO increased from 90% to 100% after adding the tag.

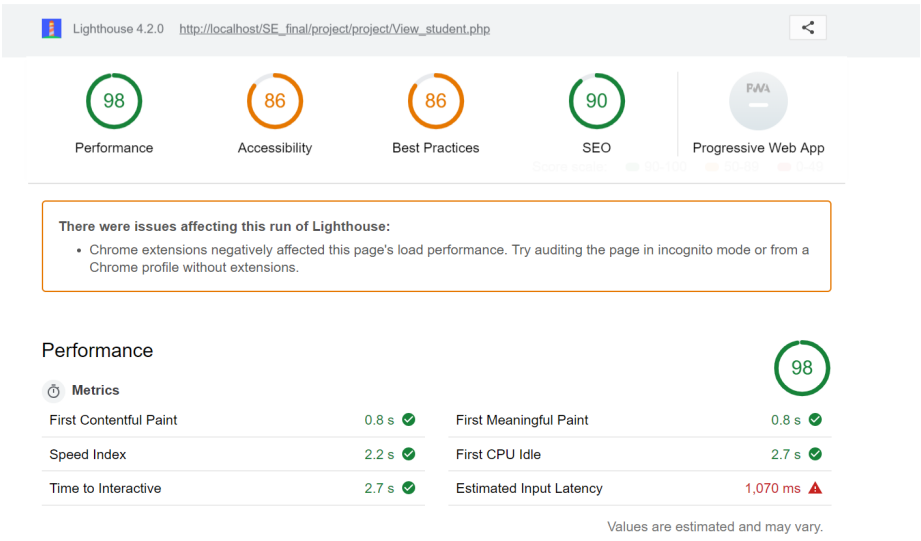


After adding:

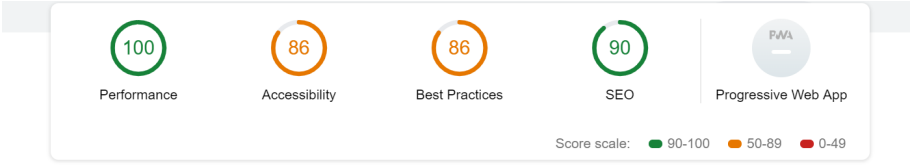


4. Removing commented code:

Performance increased from 98% to 100% after deleting the extra commented lines of code.



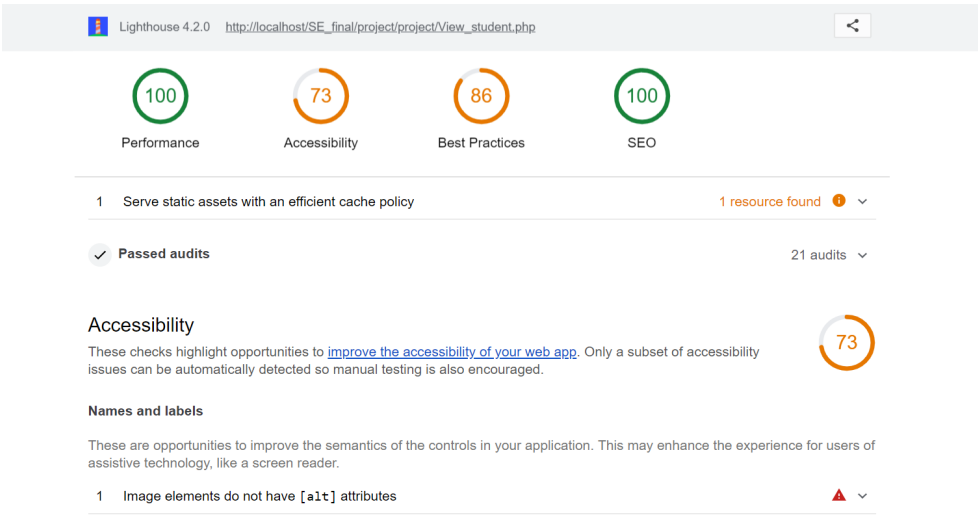
After removing:



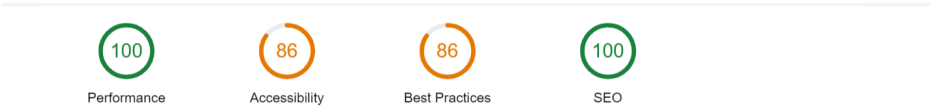
5.Adding alt attribute in img tag:

The alt attribute specifies an alternate text for an image, if the image cannot be displayed. The alt attribute provides alternative information for an image if a user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).

The accessibility increased from 73% to 86% after adding the “alt” attribute to img “tag”.



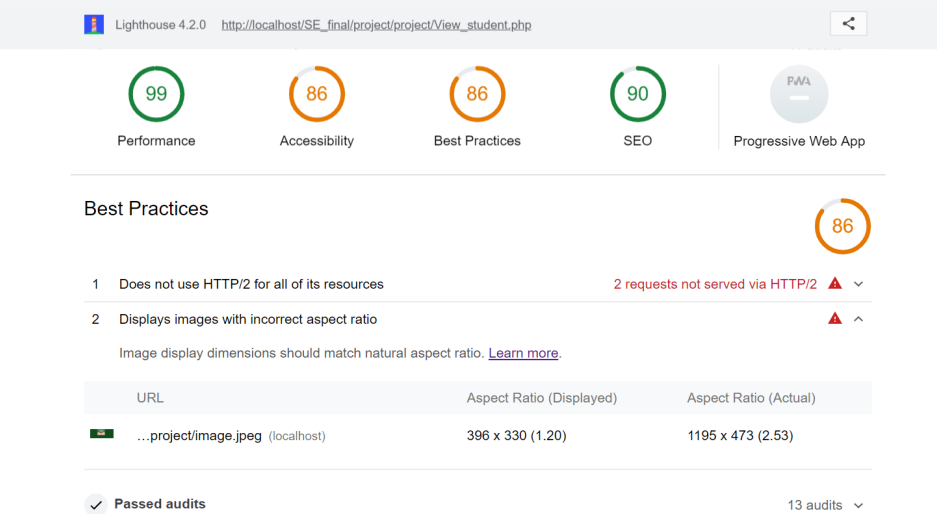
After adding:



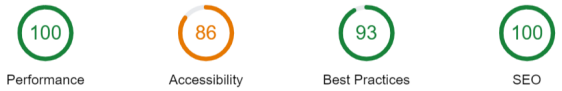
6. Changing image’s aspect ratio:

If a rendered image has a significantly different aspect ratio from the aspect ratio in its source file (the "natural" aspect ratio), then the rendered image may look distorted, possibly creating an unpleasant user experience.So it is better to check aspect ratio.

The Best Practices scale raised from 86% to 93% after displaying image with the correct aspect ratio.



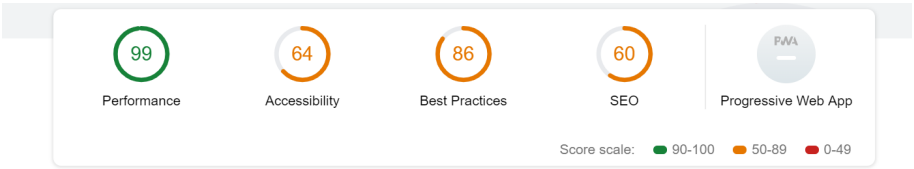
After changing:



7. Adding lang attribute:

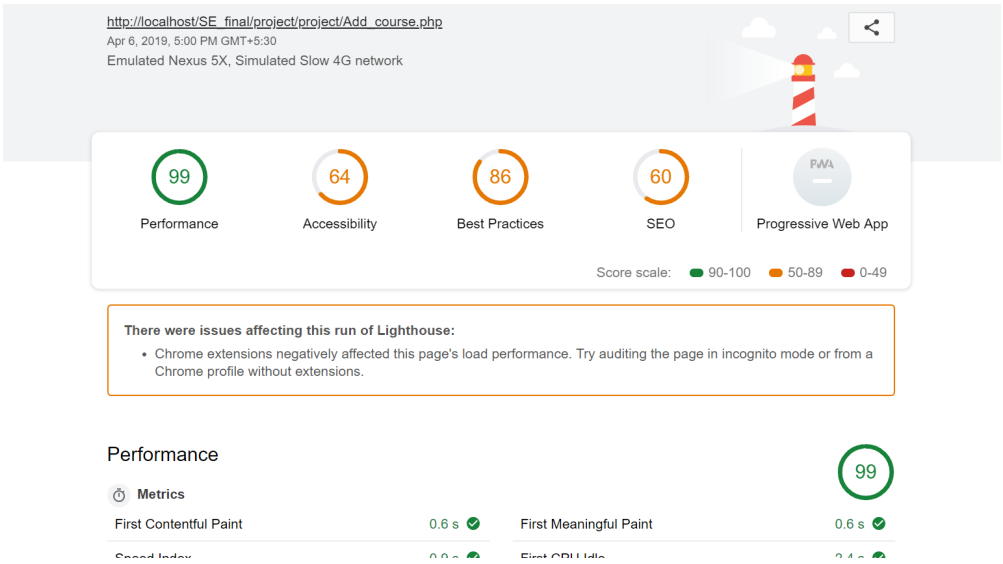
Accessibility of Add_course page increased from 57% to 64% after adding “lang” attribute to the html tag.

After adding:

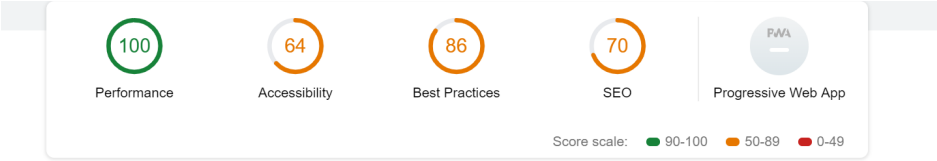


8. Adding description meta tag:

SEO increased from 60% to 70% after adding description meta tag.

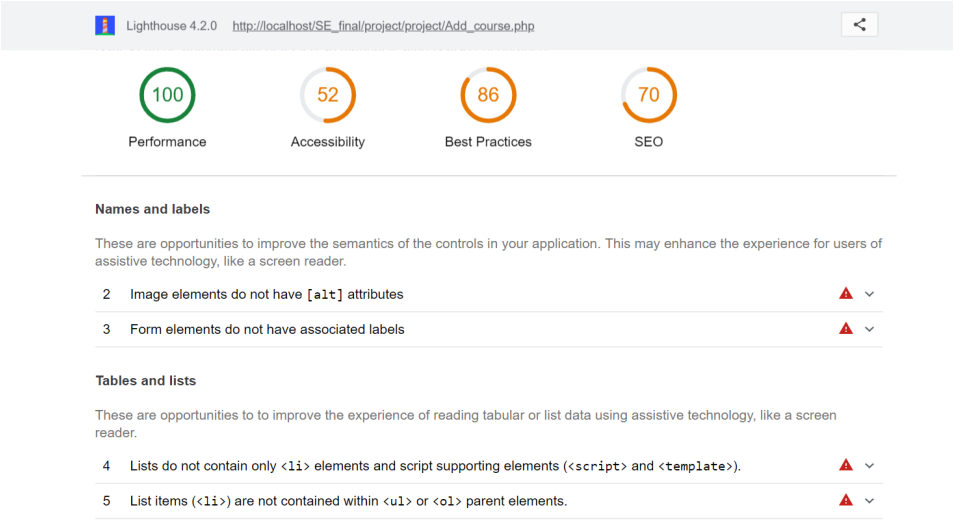


After adding:

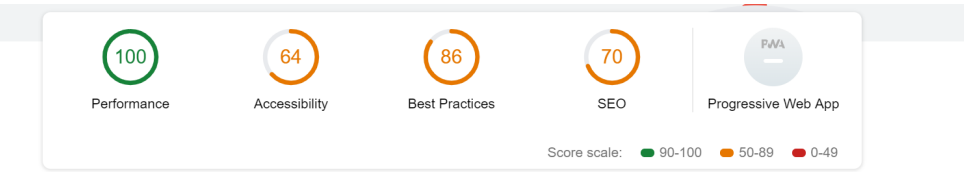


9. Adding alt to img tag:

The accessibility increased from 52% to 64% after adding “alt” attribute to “img“ tag.

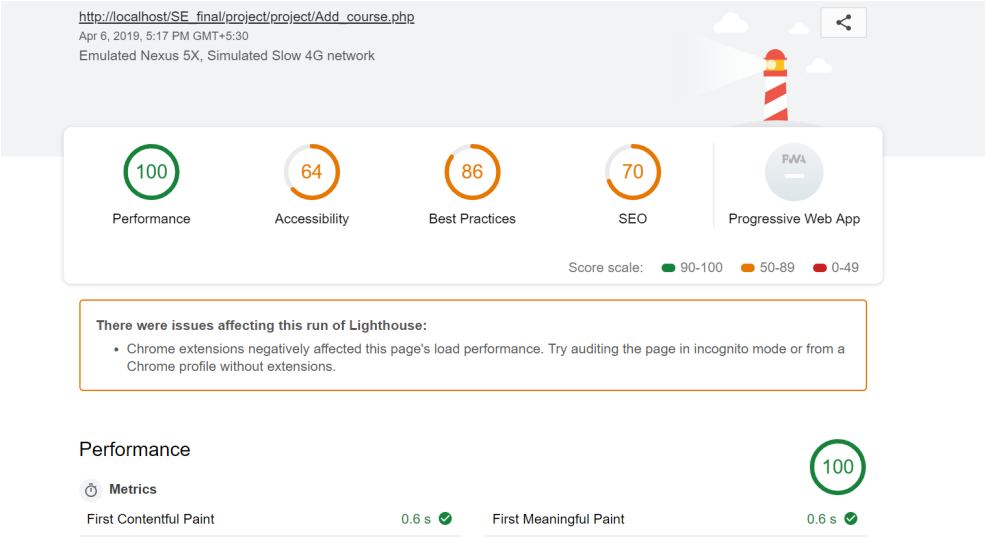


After adding:

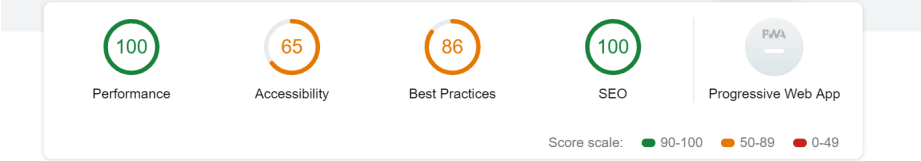


10. Adding viewport meta tag:

SEO increased from 70% to 100% after adding the viewport meta tag.

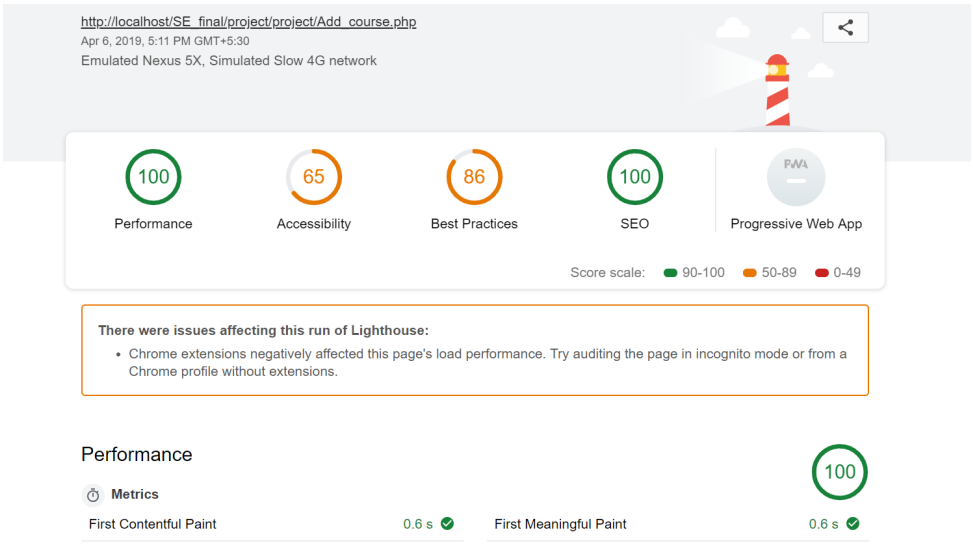


After adding:

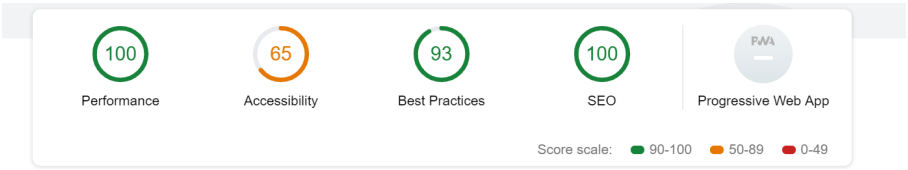


11. Changing aspect ratio of the image:

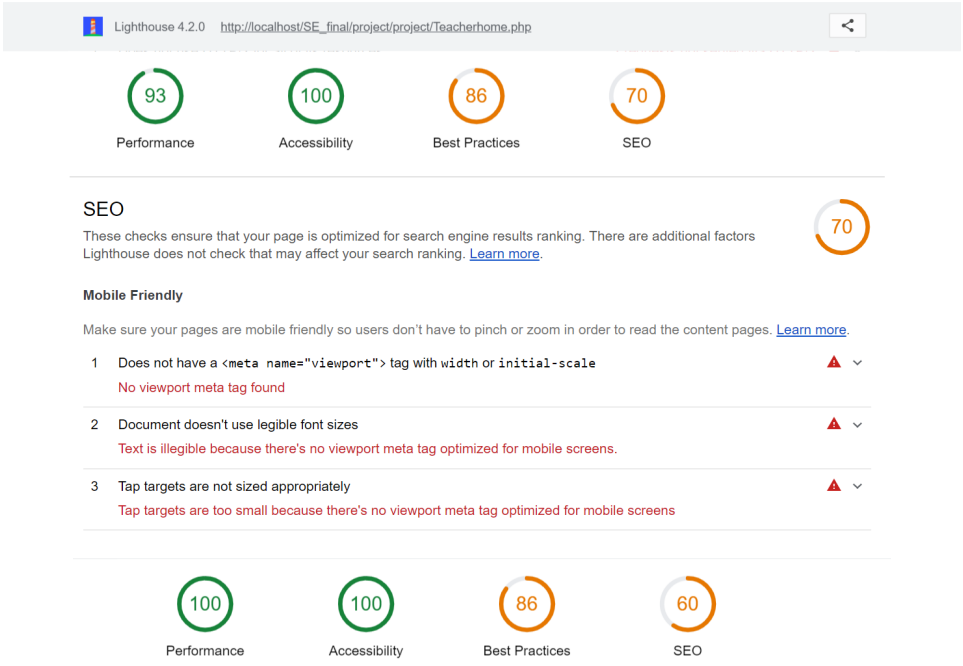
The scale of Best Practices increased from 86% to 93% after changing the aspect ratio of the image displayed.



After changing:

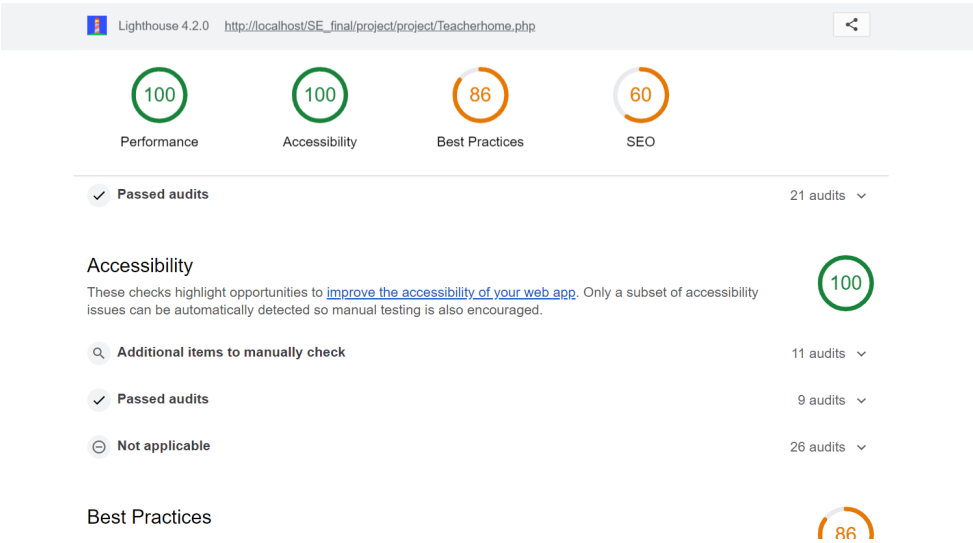


12. For Home page:

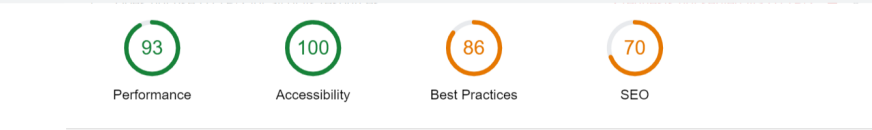


13. Adding description meta tag:

SEO increased from 60% to 70% after adding description meta tag.

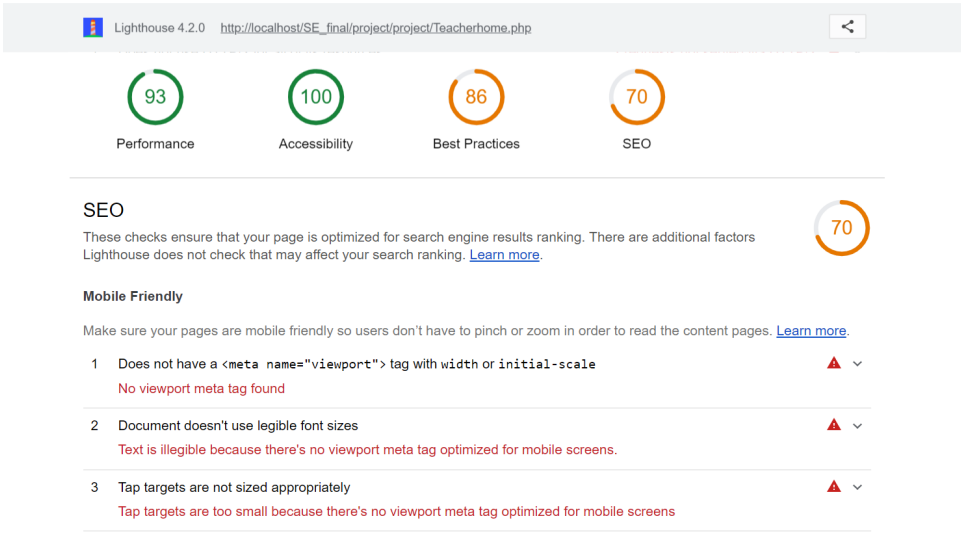


After adding:

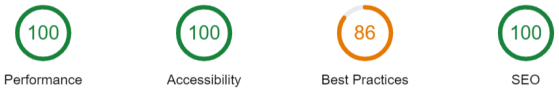


14. Adding viewport meta tag:

SEO increased from 70% to 100% after adding viewport meta tag.

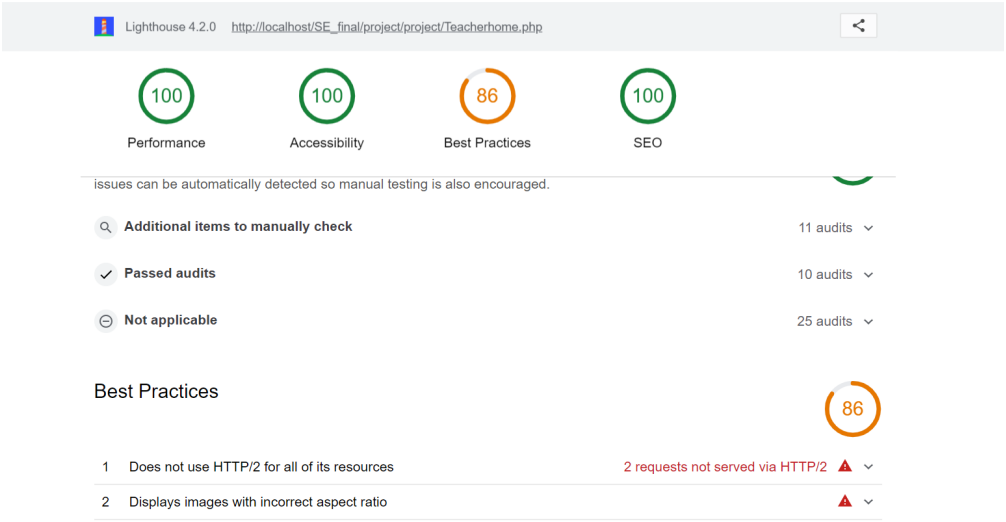


After adding:

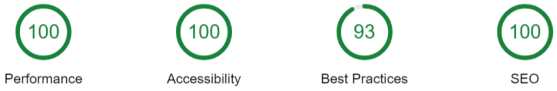


15. Changing aspect ratio:

The scale of Best Practices raised from 86% to 93% after changing the aspect ratio of the image displayed.



After changing:



4.4 Cost Estimation of the Application

Cost Estimation of the Application is done through COCOMO Model. The model uses a basic regression formula with parameters that are derived from historical project data and current as well as future project characteristics.

Organic: A development project can be considered of organic type, if the project deals with developing a well understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar types of projects.

Semi-detached: A development project can be considered of semi-detached type, if the development consists of a mixture of experienced and inexperienced staff. Team members may have limited experience on related systems but may be unfamiliar with some aspects of the system being developed.

Embedded: A development project is considered to be of embedded type, if the software being developed is strongly coupled to complex hardware, or if the stringent regulations on the operational procedures exist.

COCOMO estimation model is given by the following expressions:

- Effort = a1x (KLOC)^a2 pm
- Tdev = b1x (Effort)^b2 Months
- P= Effort/ Tdev

where

- KLOC is the estimated size of the software product expressed in Kilo Lines of Code.
- P is the no of persons required to complete the work .
- a1, a2 , b1, b2 are constants for each category of software products.
- Tdev is the estimated time to develop the software, expressed in months.
- Effort is the total effort required to develop the software product, expressed in person months (PMs).

This application comes under Organic category.
So the parameters a1, a2, b1 and b2 values are :
a1=2.4
a2=1.05
b1=2.5
b2=0.35
KLOC=8776/1000=8.776

Effort=2.4x (8.776)^1.05 pm = 23.47 person month
Tdev= 2.5x (Effort)^0.35months = 7.49 months
People= 3.13 people

5. CONCLUSION AND FUTURE WORK

5.1 Future Work:

As the system includes the basic things like grades, attendance, notices, complaints, feedbacks and so on, it fits with every different type of school scenarios.

To meet the future requirements and the drastically increasing digitalization, we can make another module for the student who can access the teaching material provided by the course instructor. Teacher can include videos too for better understanding and all these features can increase the performance of the Web Application created.

Also we can make some features self-tuned, like a student can fill in the application form along with all his details to get in to the school and all admin has to do is just accept the request if certain constraints are satisfied. Whereas before it was like Admin himself has to fill out everything and make the student enrolled. If this is done, the workload on Admin is going to be reduced and also there exists accuracy in data provided by student as there's no any intermediate person anymore.

These things are thought of for the improvement of the project performance and no activity causes degradation of the functionalities. There is a future scope of this facility that many more features such as online lectures video tutorials can be added by teachers as well as online assignments submission facility , a feature of group chat where students can discuss various issues of engineering can be added to this project thus making it more interactive more user friendly and project which fulfills each users need in the best way possible.

5.2 Conclusion

The School Management System which capable of storing school resources such as students and staff of the school and their relationship was implemented. It is easily to track the relations of students and courses they have taken, courses and teacher they are given by using the friendly interface of the system. The system can work in local or distributed manner. It means that the system can be used on local machines for management of one school or can be located on one server and clients from different schools can connect to the server and obtain requested information.