**“INDEX”**

**1. INTRODUCTION**

**1.1 OBJECTIVES**

**1.2 LIMITATIONS**

**2. DATABASE DESIGN**

**2.1 SOFTWARE REQUIREMENT SPECIFICATION**

**2.1.1 DATA REQUIREMENTS**

**2.1.2 FUNCTIONAL REQUIREMENTS**

**2.1.3 SOFTWARE AND HARDWARE REQUIREMENTS**

**2.2 CONCEPTUAL DESIGN**

**2.3 LOGICAL DESIGN**

**2.4 IMPLEMENTATION**

**2.4.1 PRESENTAION LAYER (FRONT END)**

**2.4.2 MIDDLE LAYER**

**2.4.3 DATA BASE SERVICES LAYER (BACK END)**

**3. SNAPSHOTS**

**3.1 HOME PAGE**

**3.2 LOGIN PAGE**

**3.3 REGISTRATION PAGE**

**4. SOURCE CODE**

**5. CONCLUSION**

**REFERENCES**

**Chapter 1**

**INTRODUCTION**

* 1. **OBJECTIVES:**
* The main objective of the project is to design and develop user friendly school database management system.
* To study then functioning of school database management system.
* To provide a good user interface to both teacher and admin.
* To provide accurate and synchronized database to both staff and admin.
* To develop an accurate and flexible system, it will eliminate the data redundancy and maintain a data consistency in database system.
* Provide a security to the data by using login id and unique password method.
* Improved well arrange attendance list and mark list in database system.
* Designing a user friendly and understandable interface option to admin and teacher.
  1. **LIMITATIONS:**
* Time consumption in data entry as the records are to be manually maintained in database.
* Storage increases as files and products increase it requires a lot of secondary storage device.

**Chapter 2**

**DATABASE DESIGN**

The database-planning phase begins when a customer requests to develop a database project. It is set of tasks or activities, which decide the resources required in the database development and time limits of different activities.

**2.1 SOFTWARE REQUIREMENT SPECIFICATION (SRS)**

**2.1.1 REQUIREMENT COLLECTION AND ANALYSIS**

During this step, the database designers interview prospective database users to understand and document their **data requirements**. The result of this step is a concisely written set of users’ requirements. These requirements should be specified in as detailed and complete a form as possible. In parallel with specifying the data requirements, it is useful to specify the known **functional requirements** of the application. These consist of the user defined **operations** (or **transactions**) that will be applied to the database, including both retrievals and updates.

**DATA REQUIREMENTS:**

* A user consists an unique user-name, email id and password.
* A admin consists an admin id and password.
* A teacher with student name ,attendance, marks sheet and courses.
* Admin with teacher details, student mark list and student attendance.

**2.1.2 FUNCTIONAL REQUIREMENTS:**

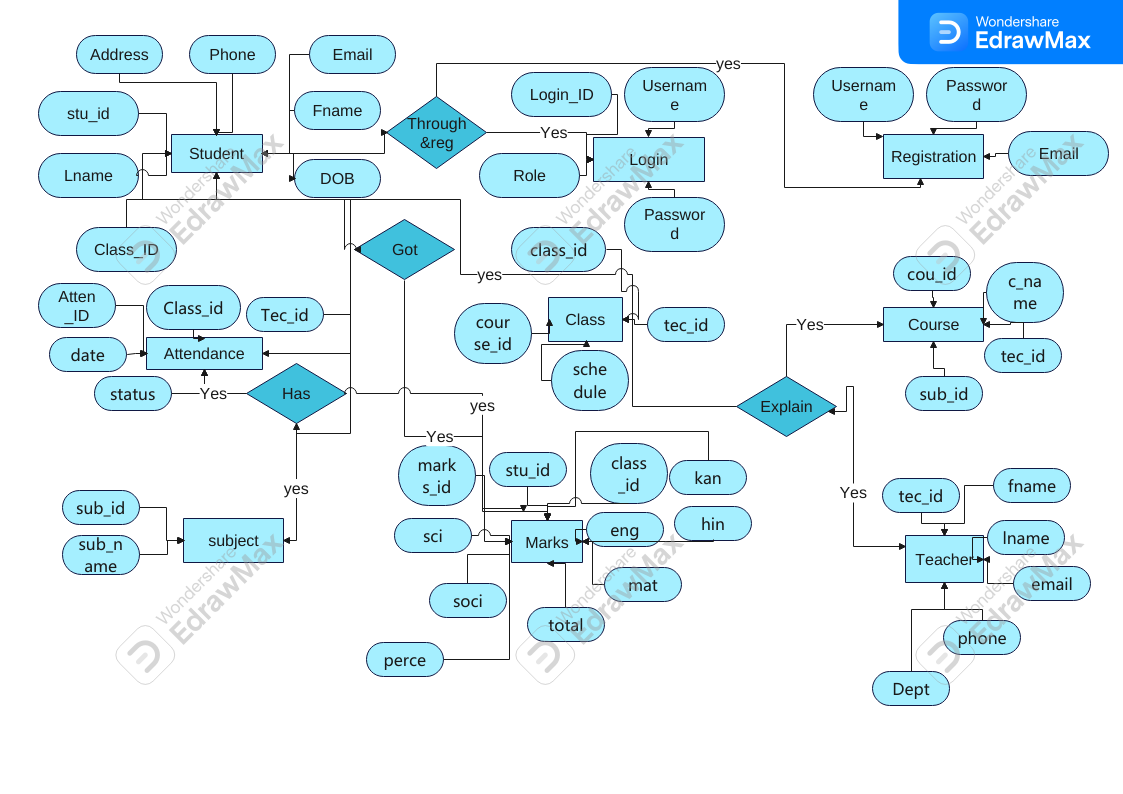
* **Admin management:** The admin should allow, Tracking of staff assignments, roles, and responsibilities within the school management.
* **User management:** The system should allow user to tracking of student assignments, roles, and attendance within the school.
* **Reporting and analytics:** The system should maintain teacher details and their login and contact details.

**2.1.3 SOFTWARE AND HARDWARE REQUIREMENTS**

* Intel core i3 2nd generation is used as processor because it is faster & provide reliable and stable working environment.
* A RAM size of 1GB is used as it will provide fast reading & writing capabilities.
* An operating system with Windows 10 OR higher versions
* Visual studio code or python Ide 9.0
* Django 4.9 version.
* Oracle Sql software.

**2.2 CONCEPTUAL DESIGN**

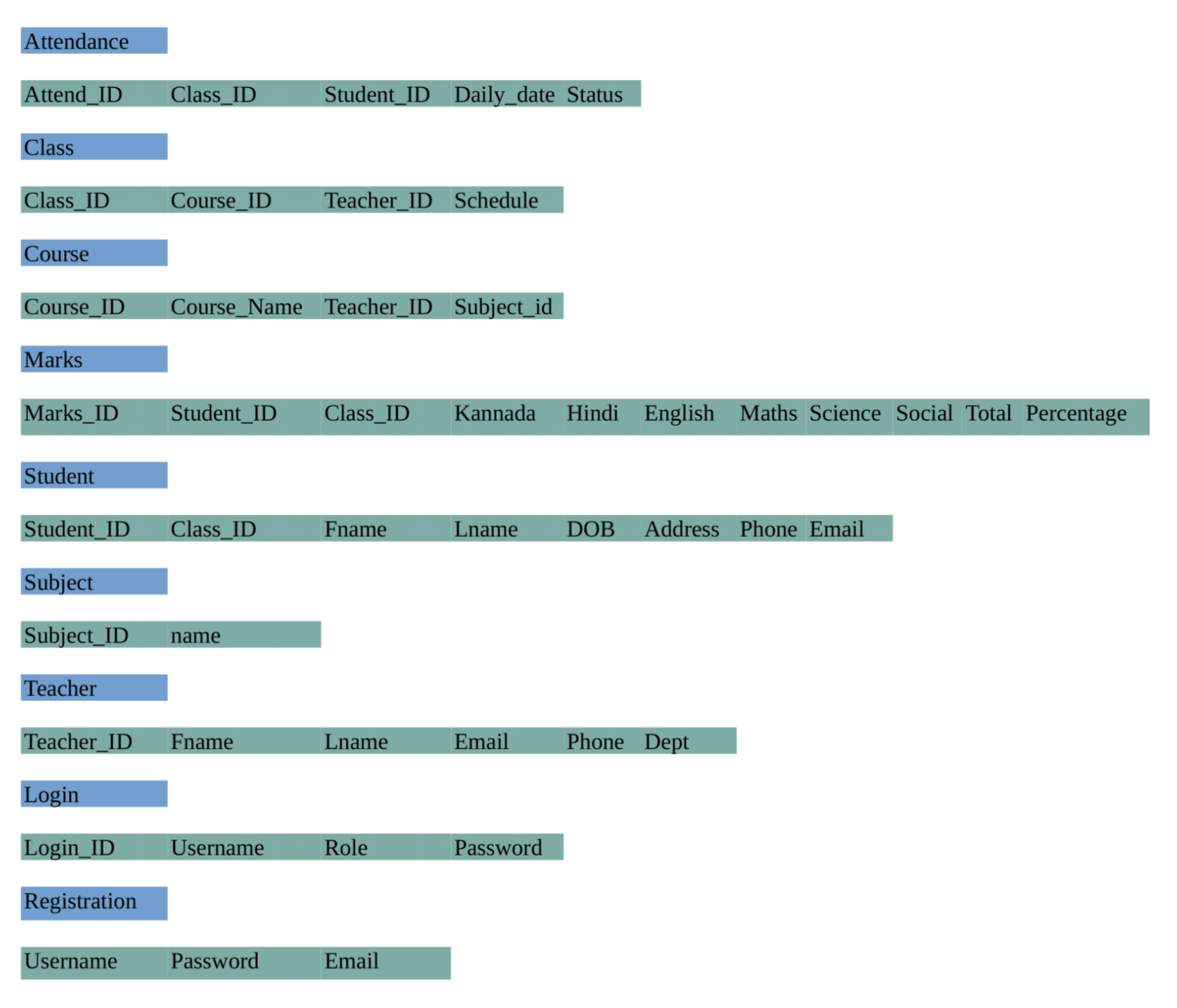
Once all the requirements have been collected and analyzed, the next step is to create a conceptual schema for the database, using a high-level conceptual data model. This step is called the conceptual design.. The conceptual schema is a concise description of the data requirements of the users and includes detailed descriptions of the entity types, relationships, and constraints; these are expressed using the concepts provided by the high-level data model. Because these concepts do not include implementation details, they are usually easier to understand and can be used to communicate with nontechnical users. The high-level conceptual schema can also be used as a reference to ensure that all users’ data requirements are met and that the requirements do not conflict. This approach enables database designers to concentrate on specifying the properties of the data, without being concerned with storage and implementation details, which makes it is easier to create a good conceptual database design. The result of this step is an Entity- Relationship (ER) diagram.



Fig***: E-R diagram.***

**2.3 Logical Design**

The next step in database design is the actual implementation of the database, using a commercial DBMS. Most current commercial DBMSs use an implementation data model—such as the relational (SQL) model—so the conceptual schema is transformed from the high-level data model into the implementation data model. This step is called **logical design** or **data model mapping**; its result is a database schema in the implementation data model of the DBMS. The ER to relation mapping algorithm is used to convert conceptual design into logical design. In this phase, the primary keys and foreign keys are defined.

ER RELATIONAL MODEL

**2.4 IMPLEMENTATION**

The various system tools that have been used in developing both the presentation layer, middle layer and database layer of the project are being discussed in this chapter.

**2.4.1 PRESENTAION LAYER (FRONT END):**

HTML, CSS, JAVA SCRIPTS are utilized to implement the front end**.**

**HTML (Hyper Text Markup Language):** HTML is a syntax used to format a text document on the web. Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

**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. Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language.CSS is used to set the visual style of web pages and user interfaces written in HTML and XHTML. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.

PHP: PHP which stands for Hypertext Preprocessor, is a widely-used server-side scripting language primarily designed for web development but also used as a general-purpose programming language.

**2.4.2 MIDDLE LAYER:**

Middle layer is implemented by using python django module , it’s a web application framework written in python it runs the web-server , It was created by adrian holovaty and simon willson .django is based on WSGI –complaint web servers. It supports Oracle sql, ,Mysql , sqlite as an external backend server.

Web application framework is a collection of libraries and modules that enables a web application developer to write application without bother about low level protocols.

It’s a core framework which includes standalone web server for development and testing, It forms a serialization, validation system that can translate between HTML forms and values into suitable for storage in backend database.

It has its own nomenclature such as callable objects generating the http response views, It consists of an object relational mapper.

**2.4.3 DATABASE SERVICES LAYER (BACK END)**

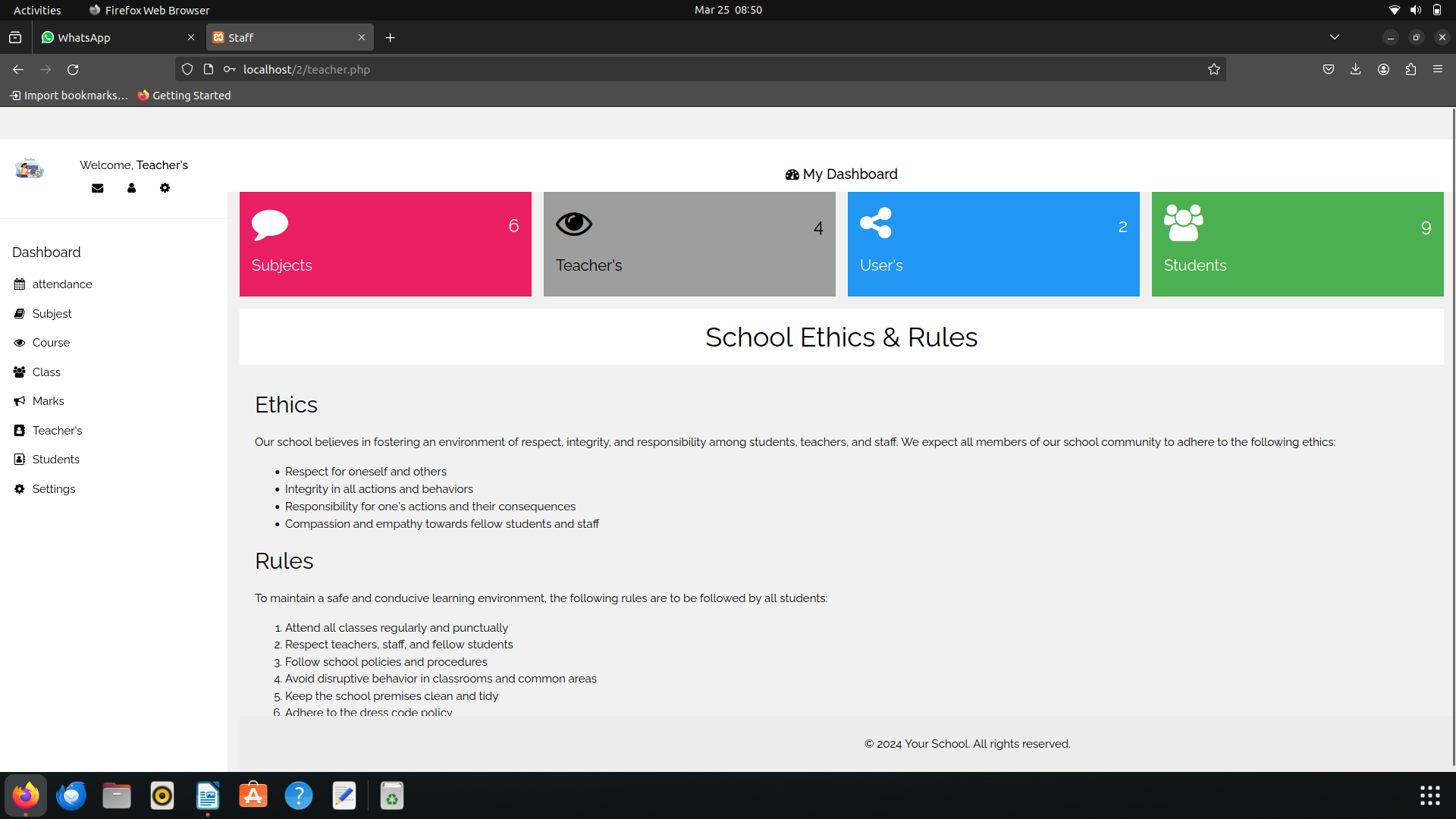
The back end is implemented using Oracle which is used to design the databases. Oracle is an open-source relational database management system (RDBMS). The SQL phrase stands for structured query.

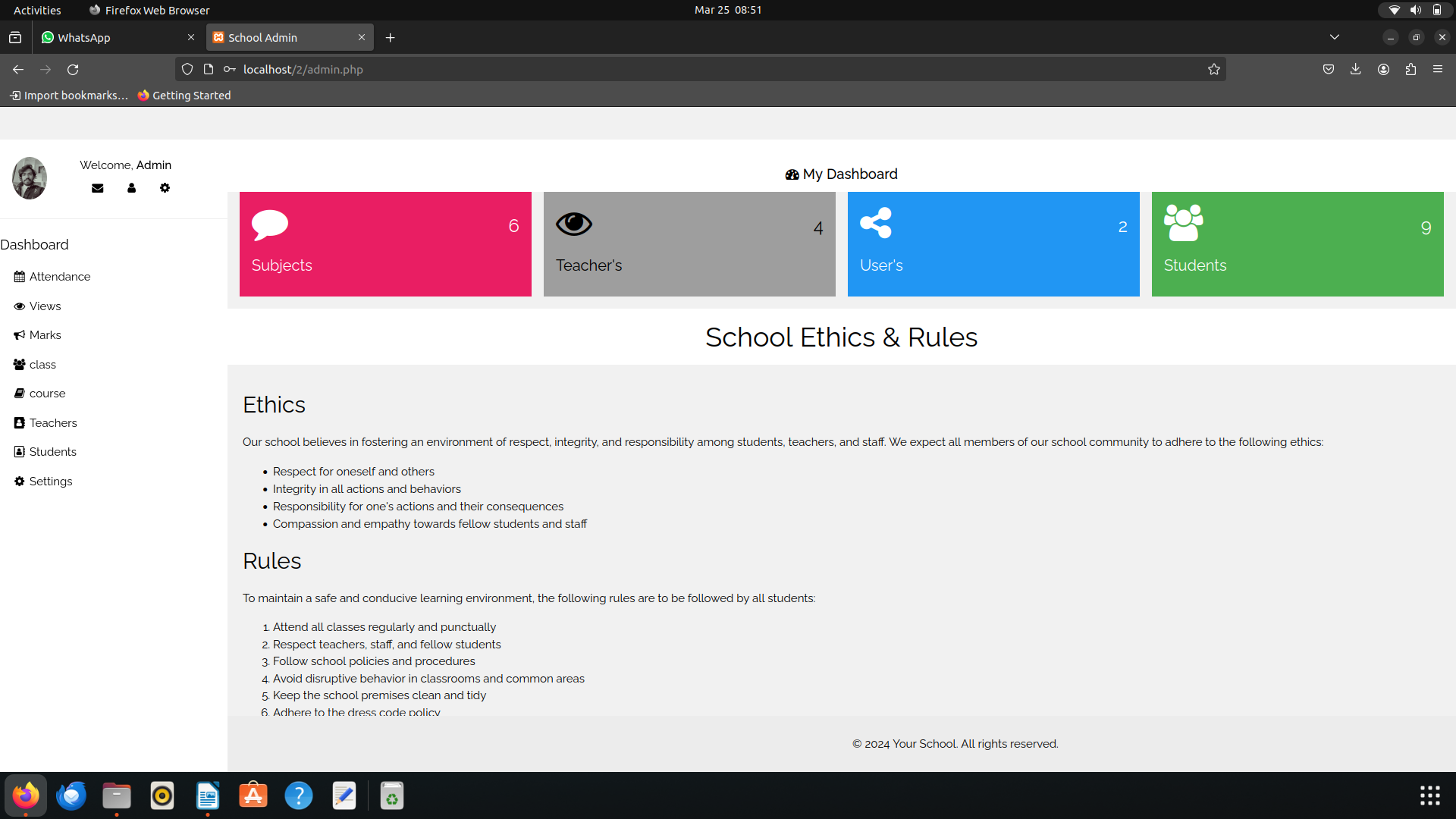
An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a [server](https://docs.oracle.com/cd/B19306_01/server.102/b14220/glossary.htm#i432724) reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery**.**

**CHAPTER 3**

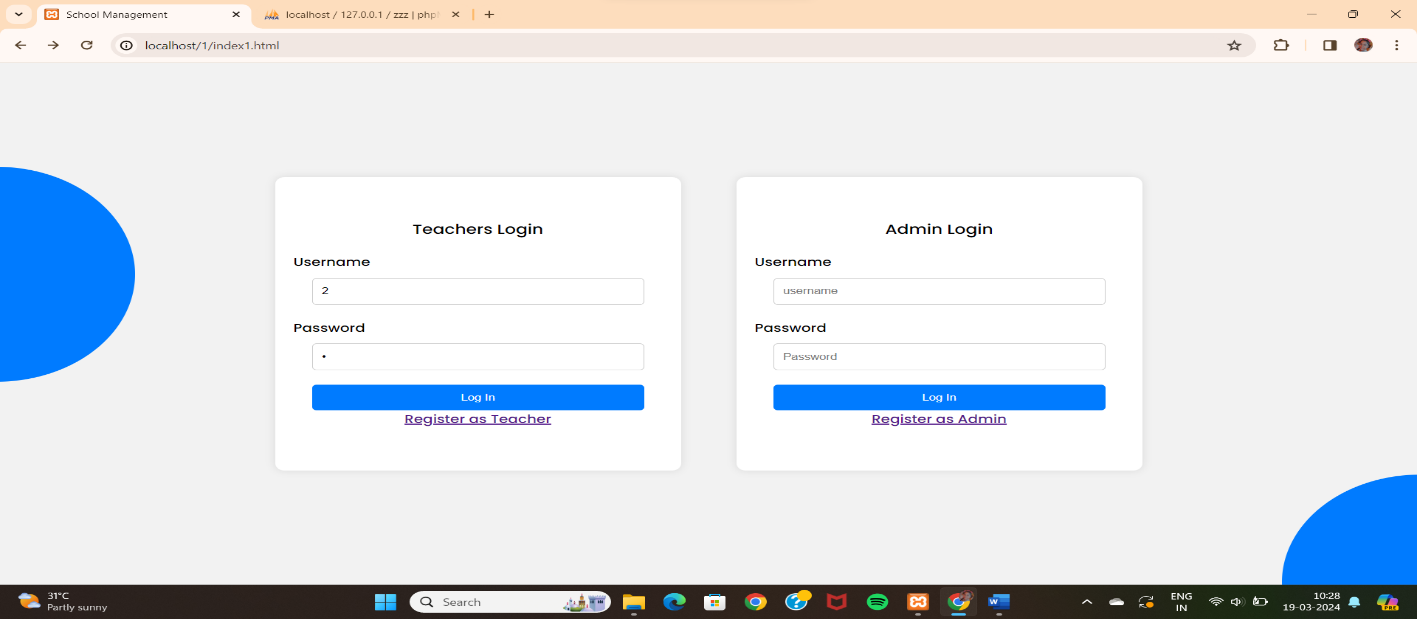
**USER INTERFACES**

**Home Page:**

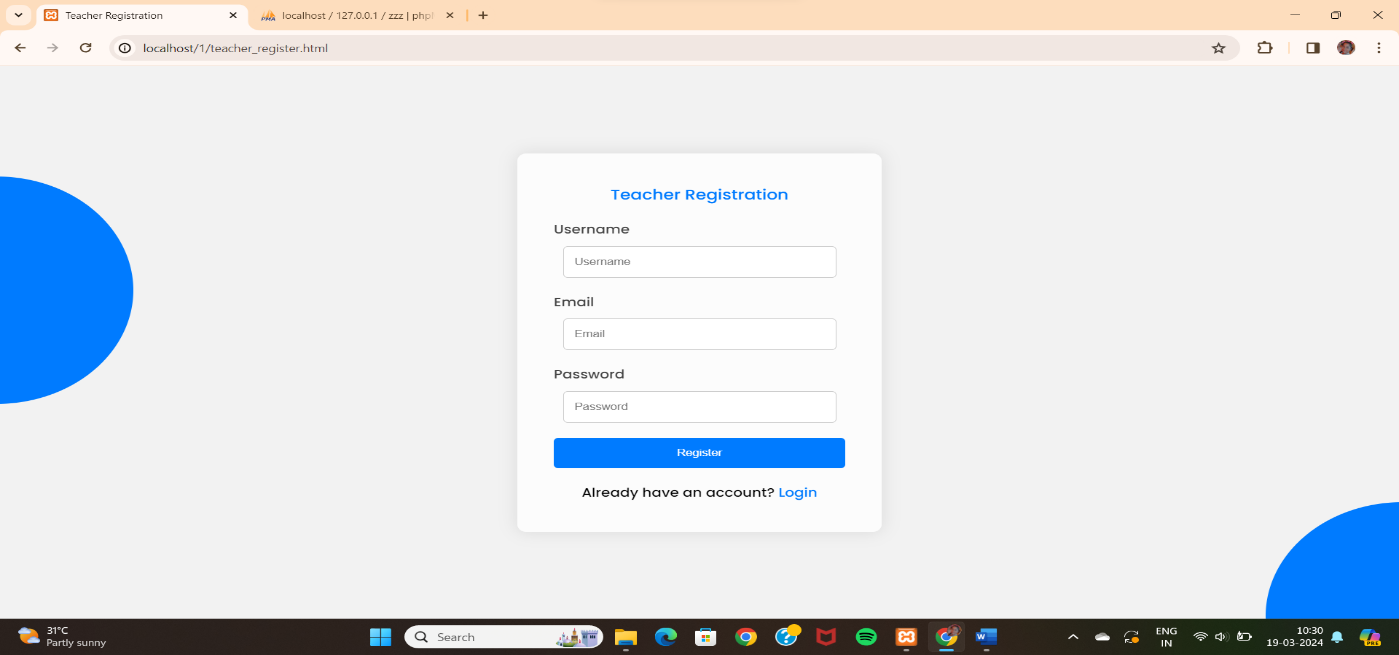




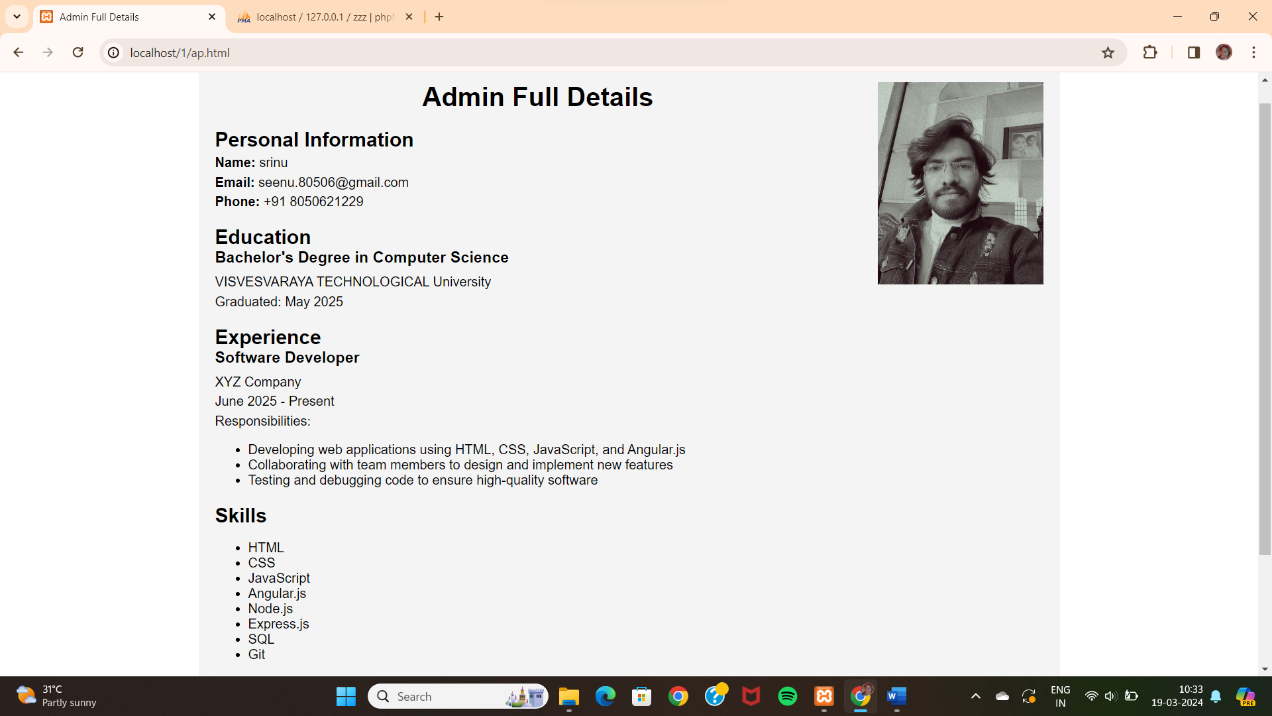
**User login page:**



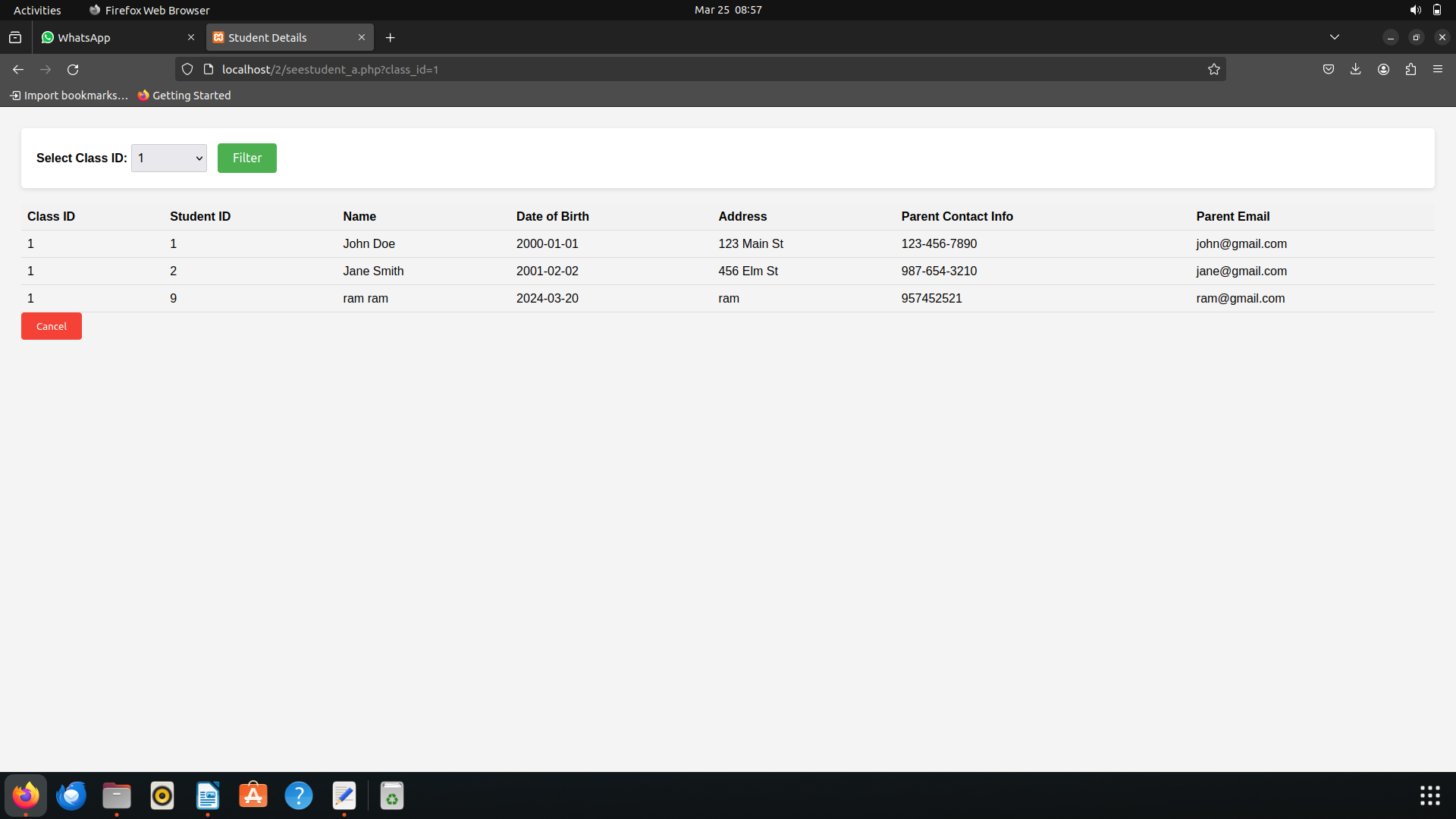
**Registration page:**



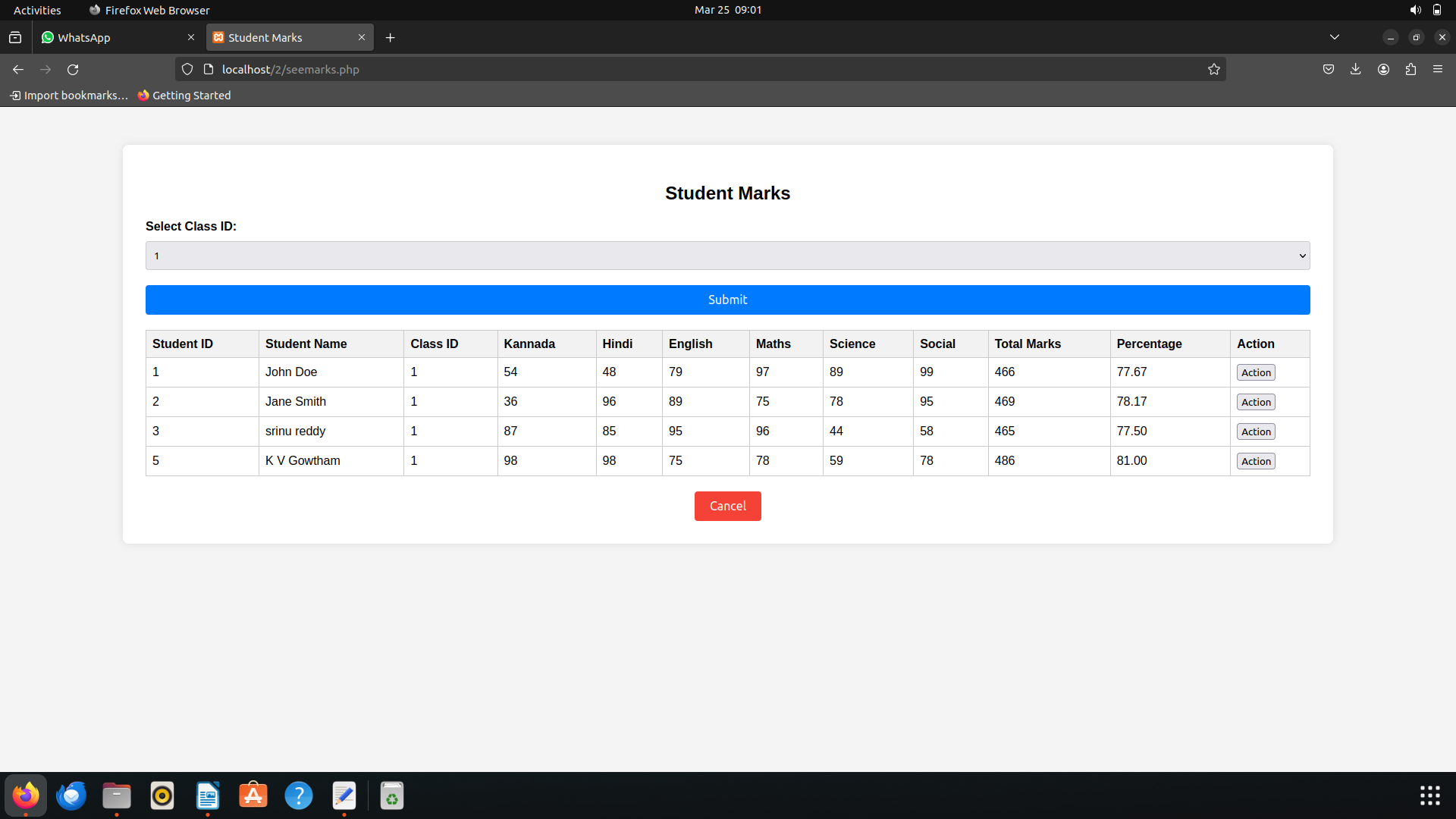
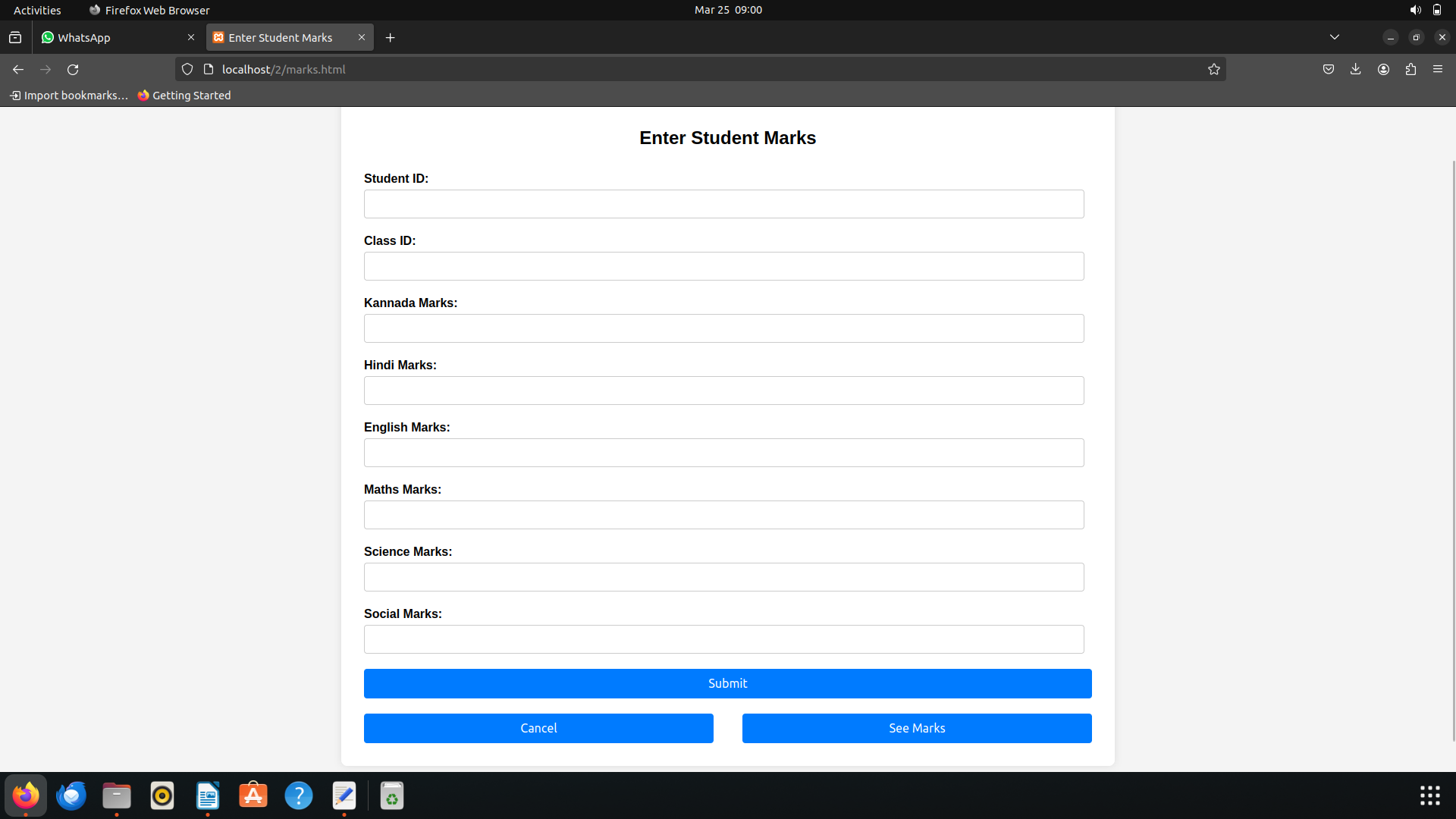
**Admin Profile:**



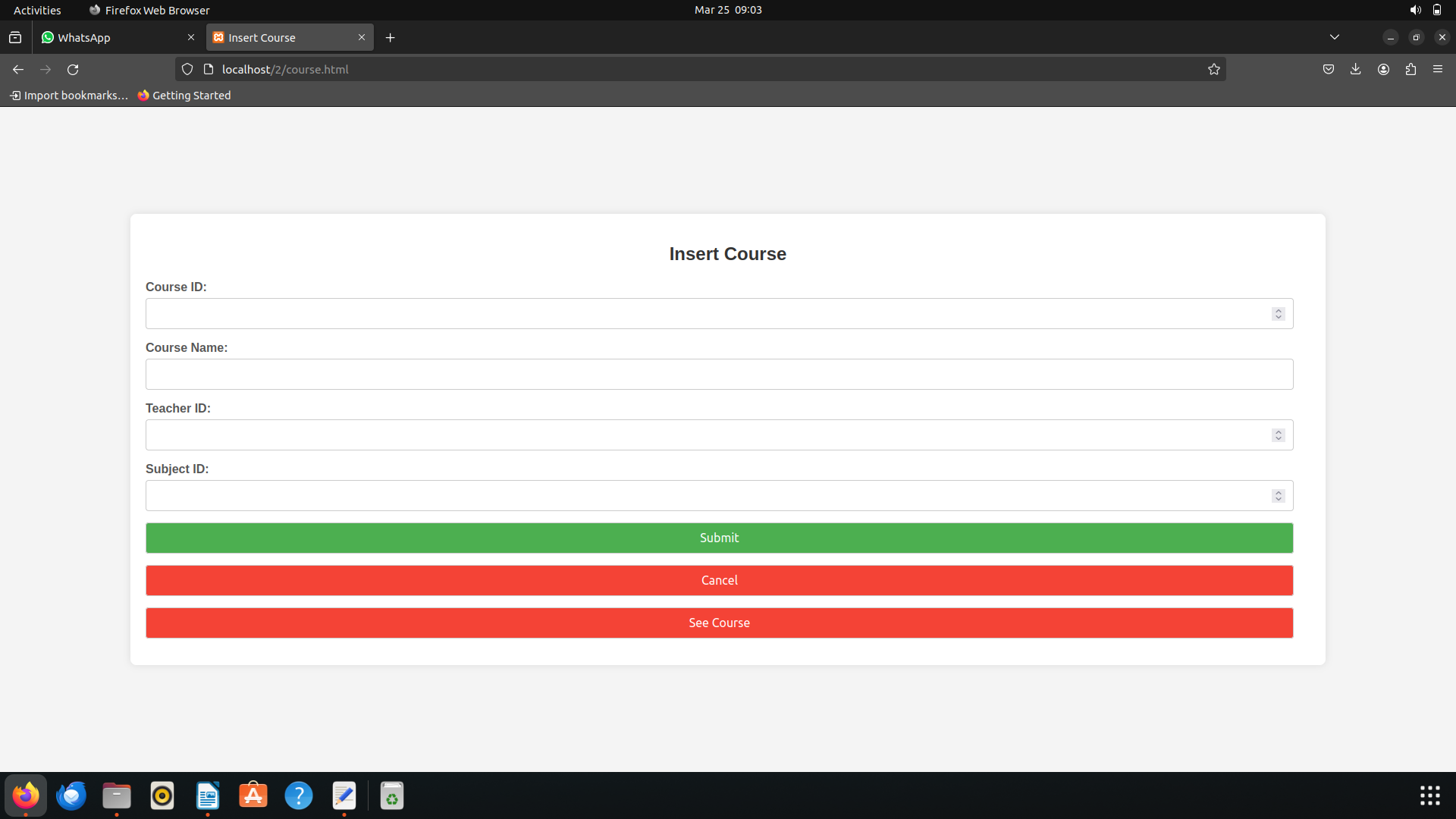
**Student table:**



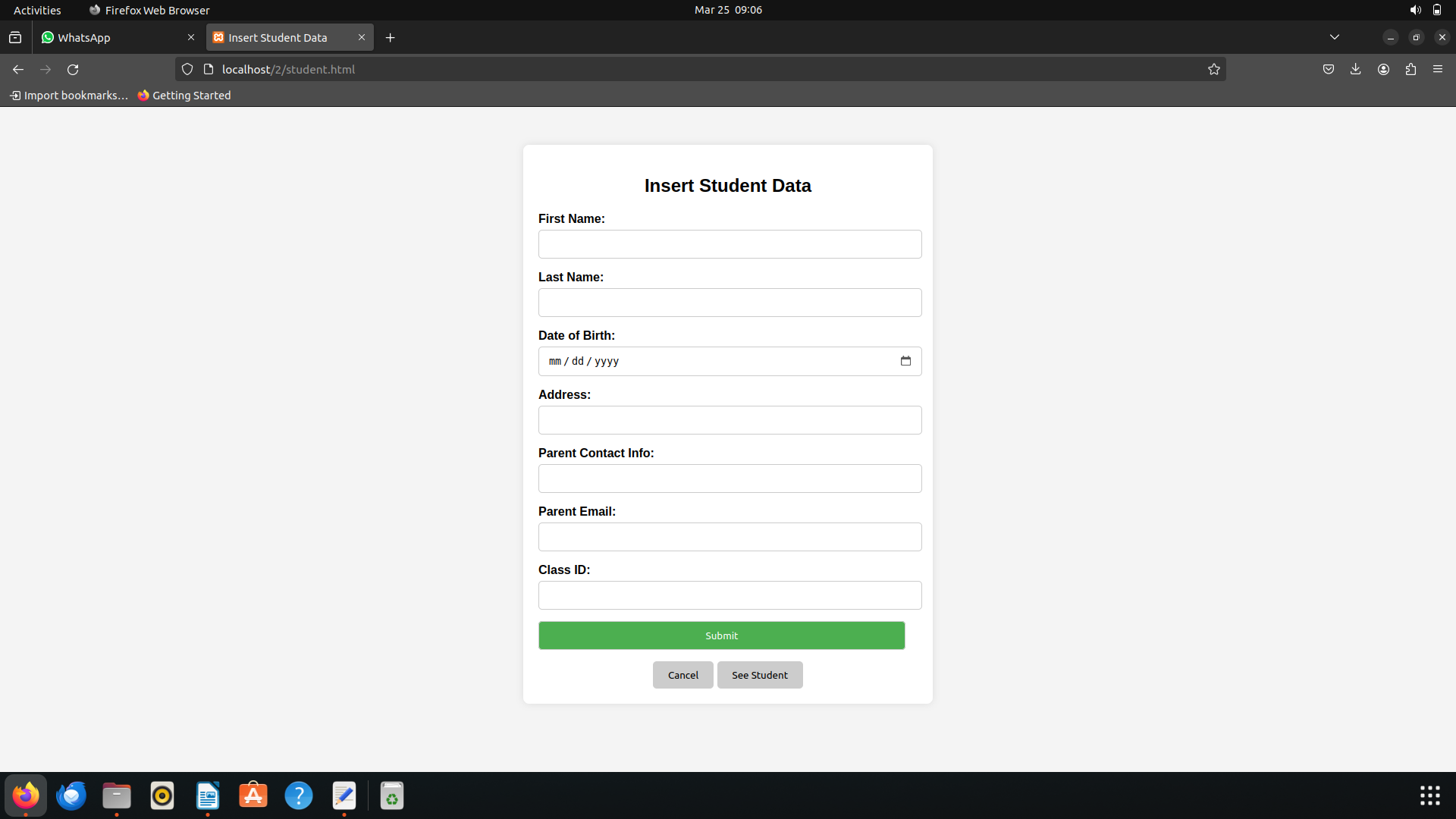
Student marks table:

**** 

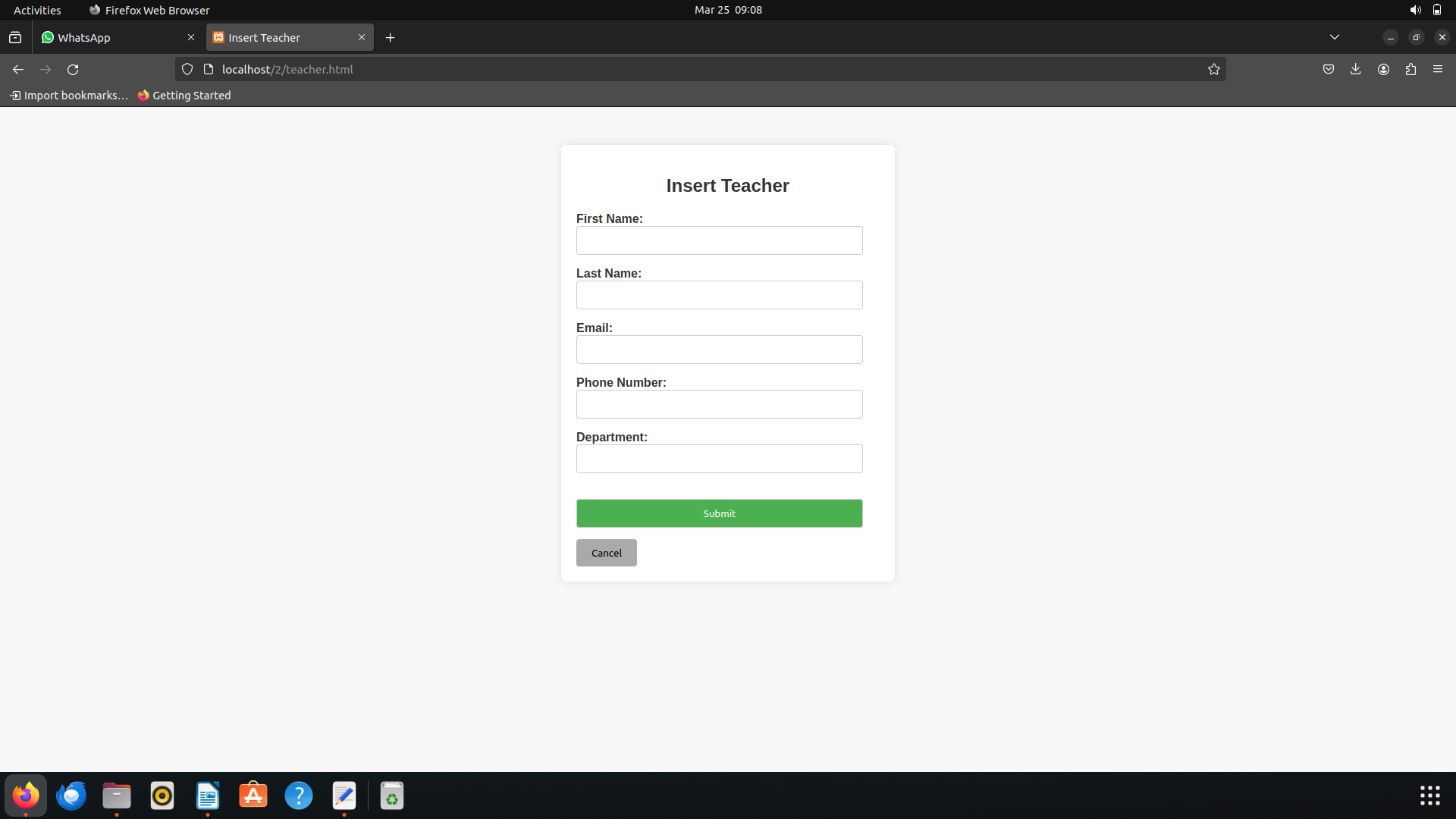
Insert course table:



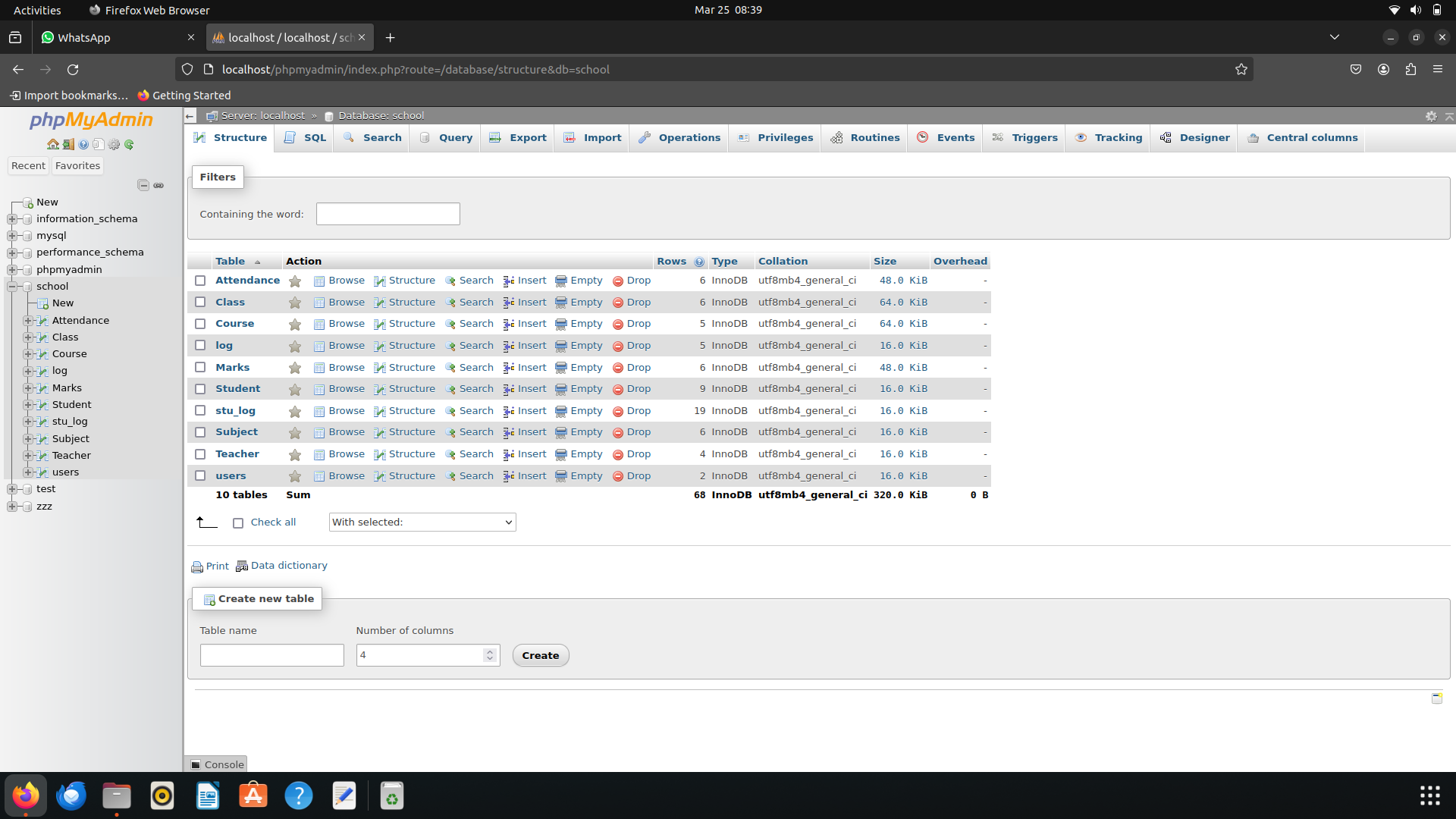
Insert into student table:



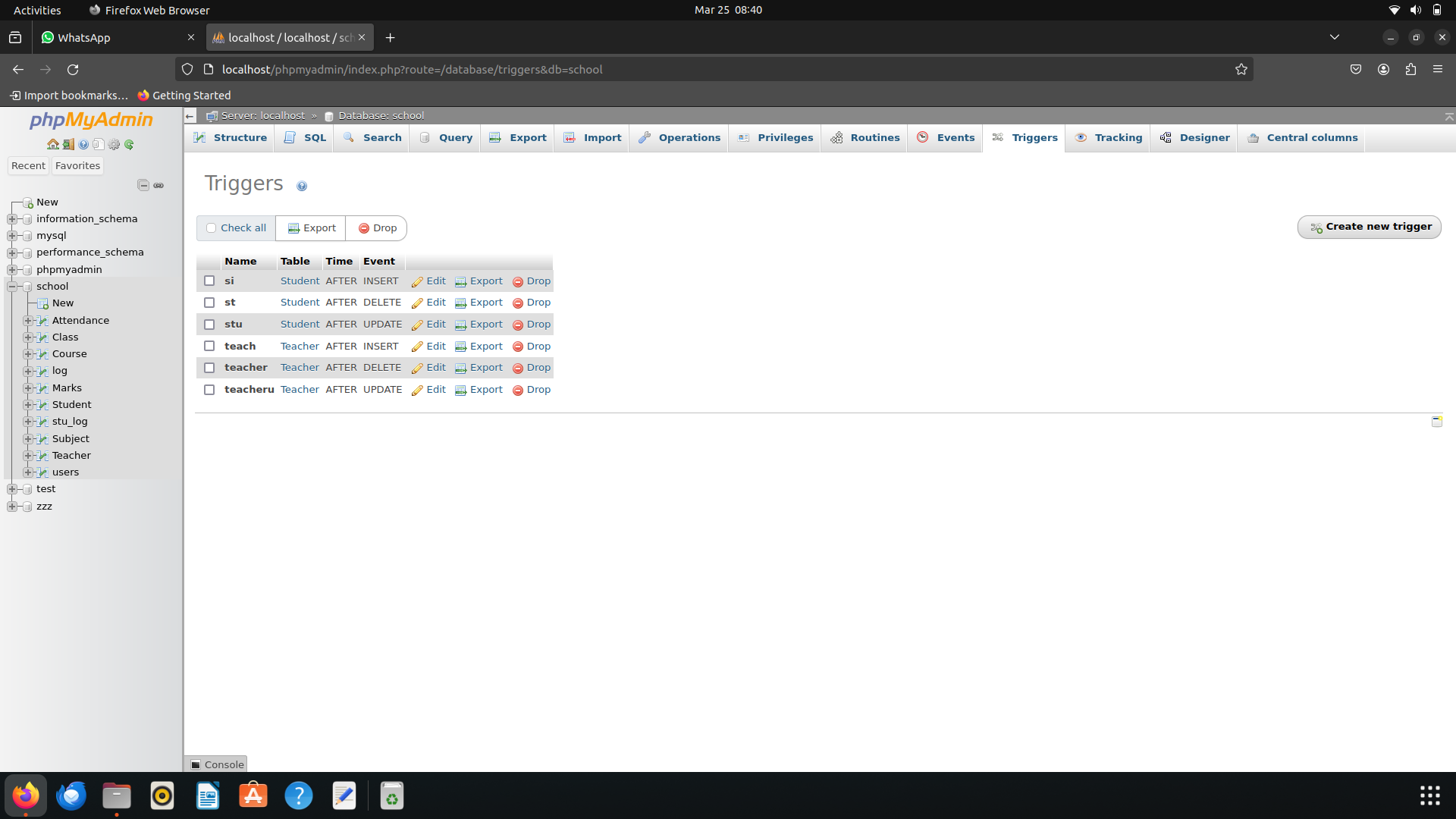
Insert Into Teacher Table:

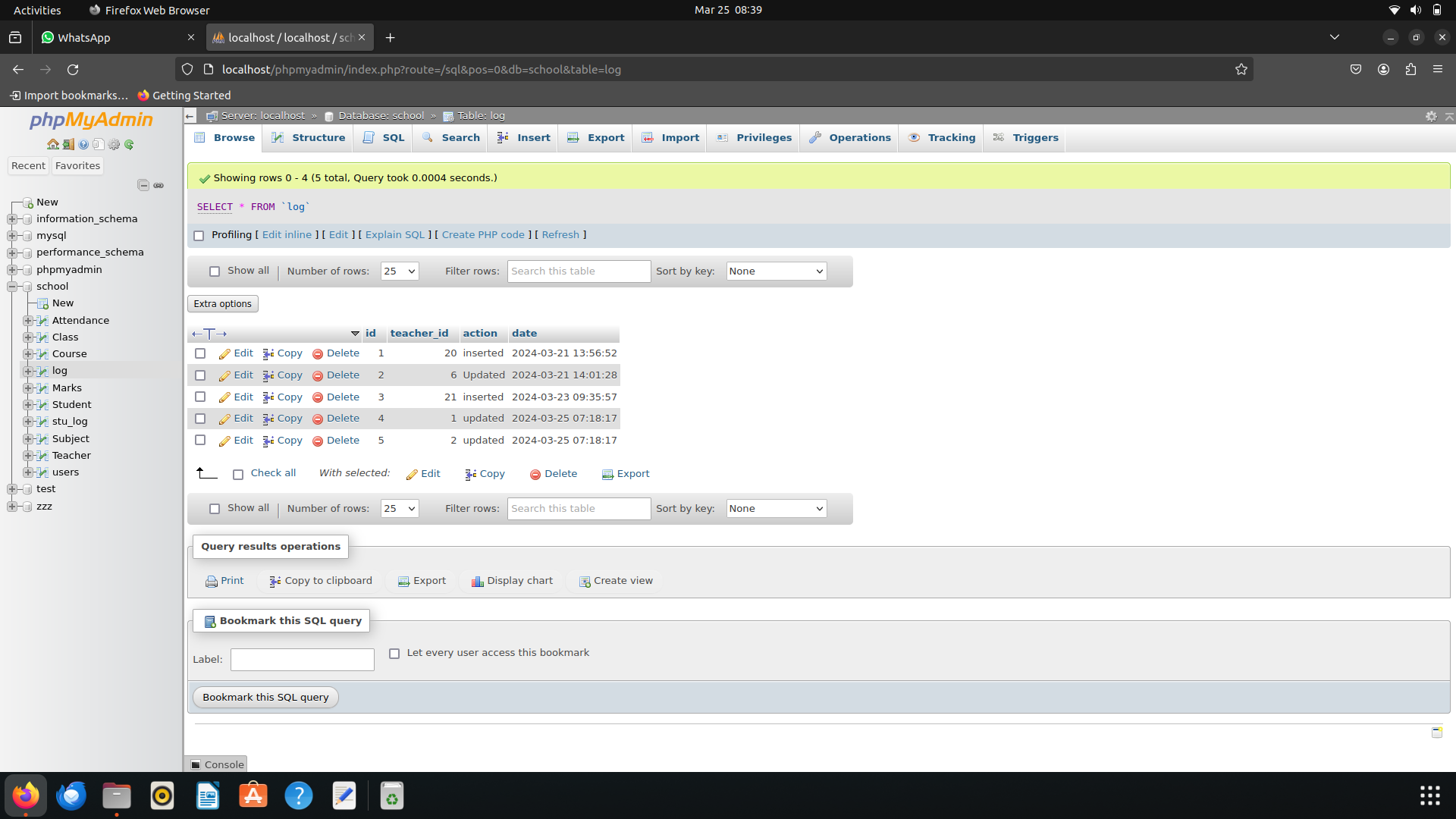


XMAPP SERVER TABLE INTERFACE:



XMAPP SERVER TRIGGER TABLE INTERFACE:





**CHAPTER 5**

**SOURCE CODE**

<!DOCTYPE html>

<html lang="en">

<head>

<!-- Design by foolishdeveloper.com -->

<title>School Management</title>

<link rel="preconnect" href="https://fonts.gstatic.com">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&display=swap" rel="stylesheet">

<!--Stylesheet-->

<style>

body {

margin: 0;

padding: 0;

font-family: 'Poppins', sans-serif;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

background-color: #f2f2f2;

overflow: hidden;

}

.background {

position: absolute;

width: 100%;

height: 100%;

z-index: -1;

}

.shape {

position: absolute;

width: 300px;

height: 300px;

background-color: #007bff;

border-radius: 50%;

}

.shape:first-child {

top: 20%;

left: -10%;

}

.shape:last-child {

bottom: -20%;

right: -10%;

}

.form-container-wrapper {

display: flex;

justify-content: center;

align-items: center;

flex-wrap: wrap; /\* Added \*/

gap: 20px; /\* Added gap between tables \*/

}

.form-container {

flex: 1; /\* Make both containers take equal width \*/

max-width: 400px; /\* Limit maximum width of containers \*/

padding: 40px 20px; /\* Adjusted padding for increased height \*/

margin: 20px; /\* Added margin \*/

background-color: #fff;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

overflow: hidden;

text-align: center;

}

.form-container form {

margin-bottom: 20px;

}

.form-container h3 {

margin-bottom: 20px;

font-weight: 500;

}

.form-container label {

display: block;

margin-bottom: 10px;

font-weight: 500;

text-align: left;

}

.form-container input[type="text"],

.form-container input[type="password"] {

width: calc(100% - 40px);

padding: 10px;

margin-bottom: 20px;

border-radius: 5px;

border: 1px solid #ccc;

box-sizing: border-box;

}

.form-container button {

width: calc(100% - 40px);

padding: 10px 20px;

background-color: #007bff;

color: #fff;

border: none;

border-radius: 5px;

cursor: pointer;

transition: background-color 0.3s;

}

.form-container button:hover {

background-color: #0056b3;

}

.signup-link {

text-align: center;

}

.signup-link a {

color: #007bff;

text-decoration: none;

transition: color 0.3s;

}

.signup-link a:hover {

color: #0056b3;

}

</style>

</head>

<body>

<div class="background">

<div class="shape"></div>

<div class="shape"></div>

</div>

<div class="form-container-wrapper">

<div class="form-container">

<form id="teachers-login" action="login.php" method="POST">

<h3>Teachers Login</h3>

<label for="teachers-username">Username</label>

<input type="text" placeholder="username" id="teachers-username" name="username">

<label for="teachers-password">Password</label>

<input type="password" placeholder="Password" id="teachers-password" name="password">

<button type="submit">Log In</button>

<a href="teacher\_register.html">Register as Teacher</a>

</form>

</div>

<div class="form-container">

<form id="admin-login" action="login.php" method="POST">

<h3>Admin Login</h3>

<label for="admin-username">Username</label>

<input type="text" placeholder="username" id="admin-username" name="username">

<label for="admin-password">Password</label>

<input type="password" placeholder="Password" id="admin-password" name="password">

<button type="submit">Log In</button>

<a href="admin\_register.html">Register as Admin</a>

</form>

</div>

</div>

<script>

document.getElementById("admin-login").addEventListener("submit", function(event) {

event.preventDefault(); // prevent form submission

var username = document.getElementById("admin-username").value;

var password = document.getElementById("admin-password").value;

// Directly check username and password

if (username === "1" && password === "1") {

window.location.href = "admin.php"; // Redirect to admin.php on successful login

} else {

alert("Invalid username or password");

}

});

</script>

</body>

</html>

**CHAPTER 6**

**CONCLUSION**

SCHOOL DATABASE MANAGEMENT SYSTEM successfully designed and implemented Administration management database which helps us in administrating the data user for managing the tasks performed by staff, students.

The project successfully used various functionalities of Python django and also create the fully functional database management system for offline school database. Using Oracle RDBMS and sqlite as the database is highly beneficial as it is free to download, popular and can be easily customized. The data stored in the oracle database can easily be retrieved and manipulated according to the requirements with basic knowledge of SQL.

The project will helps to a school DBMS optimizes resource allocation, streamlines administrative processes, and ensures data security and privacy compliance. This project will also helps to understand concepts of web development and MVC model and object oriented programming as well as frontend features like html, css and java script.

**REFERENCES:9**

[1] Elmasri and Navathe, “Fundamentals of Database Systems” , 3/e, Addison - Wesley, 2001

[2] A Silberschaltz, H.F. Korth, and S sudarshan, “Database System Concepts”, 3/e, Tata

Mcgraw Hill,1997

[3] Thomas M. Connolly, Carolyn E. Begg, “Database Systems & Practical Approach to Design

Implementation and Management”, 4/e,Addison – Wesley, 2005

[4] W3Schools ([https://www.w3schools.com/xmapp/](https://www.w3schools.com/django/))