Criterion C

Technology used:

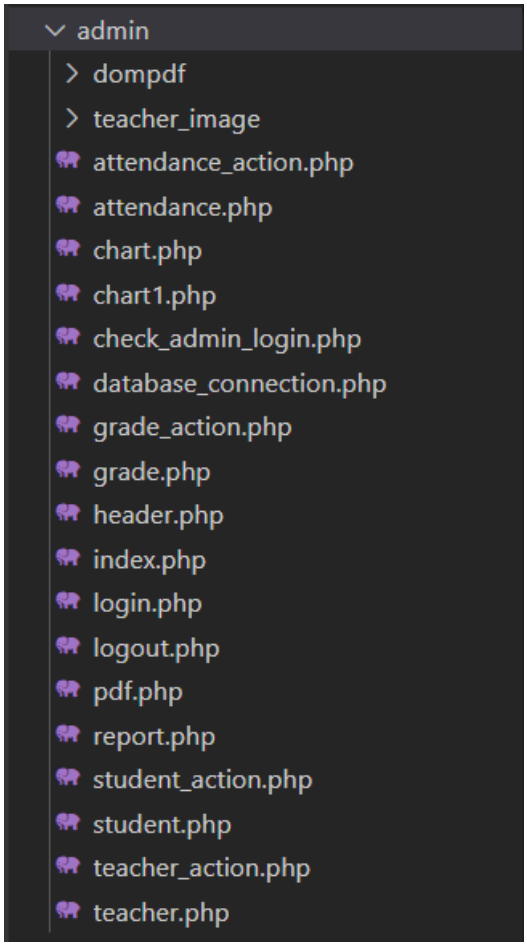
Number	Programming languages and frameworks	Description
1	PHP (Hypertext Processor)	PHP was used for developing server-side scripts, which communicate with the front-end and database.
2	HTML (Hypertext markup language)	HTML was used to design the front-end screens.
3	CSS (Cascading style sheets)	I used CSS to style the HTML pages. (Font, colours, iconography, spacing and positioning)
4	Bootstrap (Responsive framework)	Used for making the application responsive to all devices. (Desktop, Tablet, Mobile)
5	Java script	Java script was used for form validations and DOM (Document object model) events.
6	jQuery (Java Script library)	Used for minification of the Java Script code.
7	AJAX	AJAX was used to submit data to the server asynchronously without interfering with the display and behaviour of the existing page.
8	MySQL	MySQL was used as a database to store the data.
9	XAMPP Server	XAMPP server was used to create a local server to run the application.

List of techniques used:

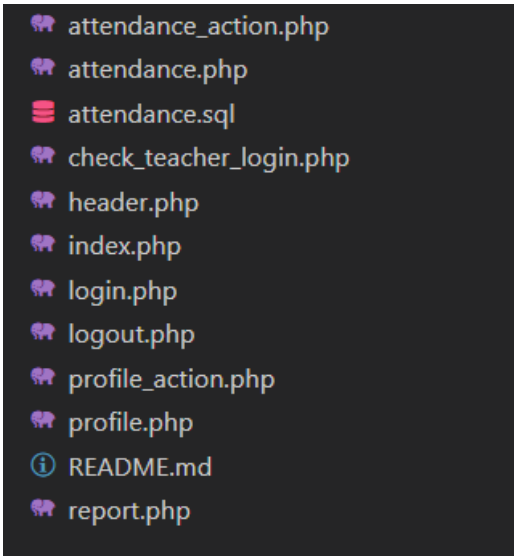
- For loops
- Conditional statements (if, else)
- Arrays
- Variables
- Linear searching
- Functions
- Error-handling
- GUI tabs and popups
- PDF Reporting

Structure of project:

Admin functionalities:

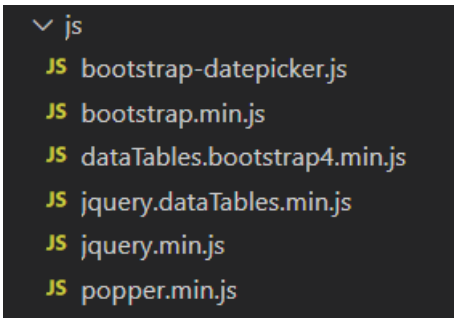


Teacher functionalities:

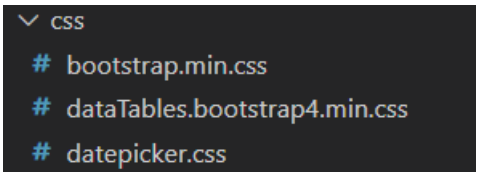


All functionalities were coded in each individual file. For example, teacher_action.php was used to update, delete, and insert teacher information.

Java Script:



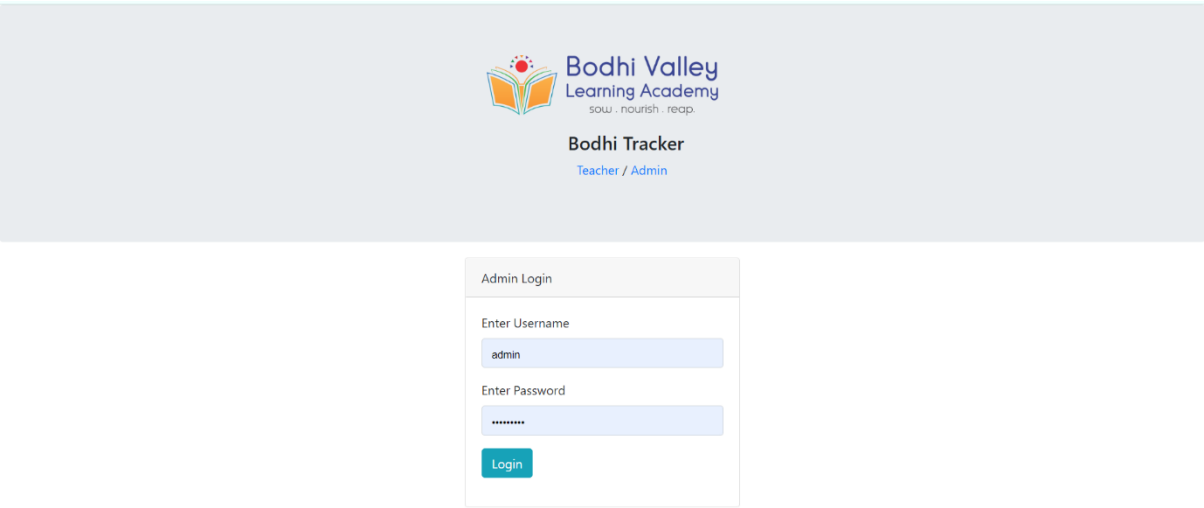
CSS:



All Java Script files are located in the JS folder and all style sheets are in the CSS folder.

Teacher login page:

This is the login page of the proposed system which consists of three buttons. In order to go to the desired screen the user has to login with the valid username and password.

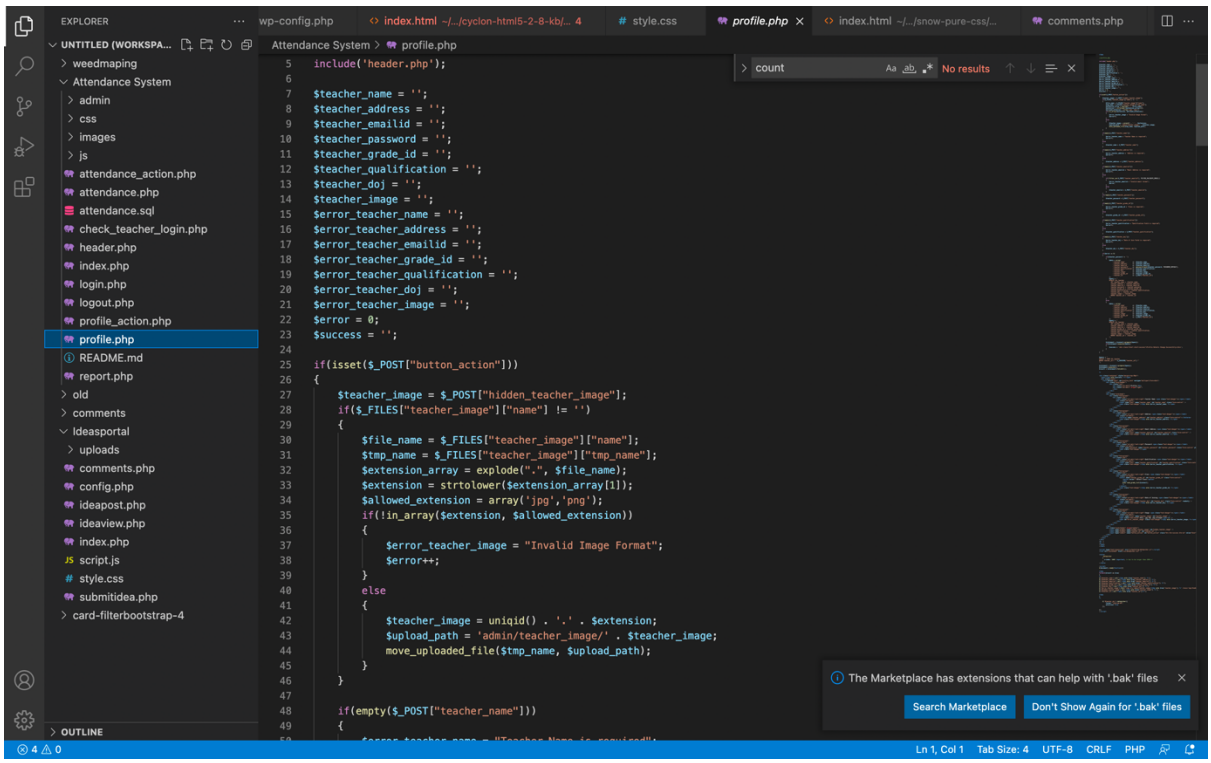


Back-end:

The information is collected from the input fields (username and password), and then stored into Java Script variables. When the login button is clicked, an Ajax post method is initiated, and data is moved to the back-end PHP.

```
<script>
$(document).ready(function(){
  $('#teacher_login_form').on('submit', function(event){
    event.preventDefault();
    $.ajax({
      url:"check_teacher_login.php",
      method:"POST",
      data:$(this).serialize(),
      dataType:"json",
      beforeSend:function(){
        $('#teacher_login').val('Validate...');
        $('#teacher_login').attr('disabled','disabled');
      },
      success:function(data)
      {
        if(data.success)
        {
          location.href="index.php";
        }
```

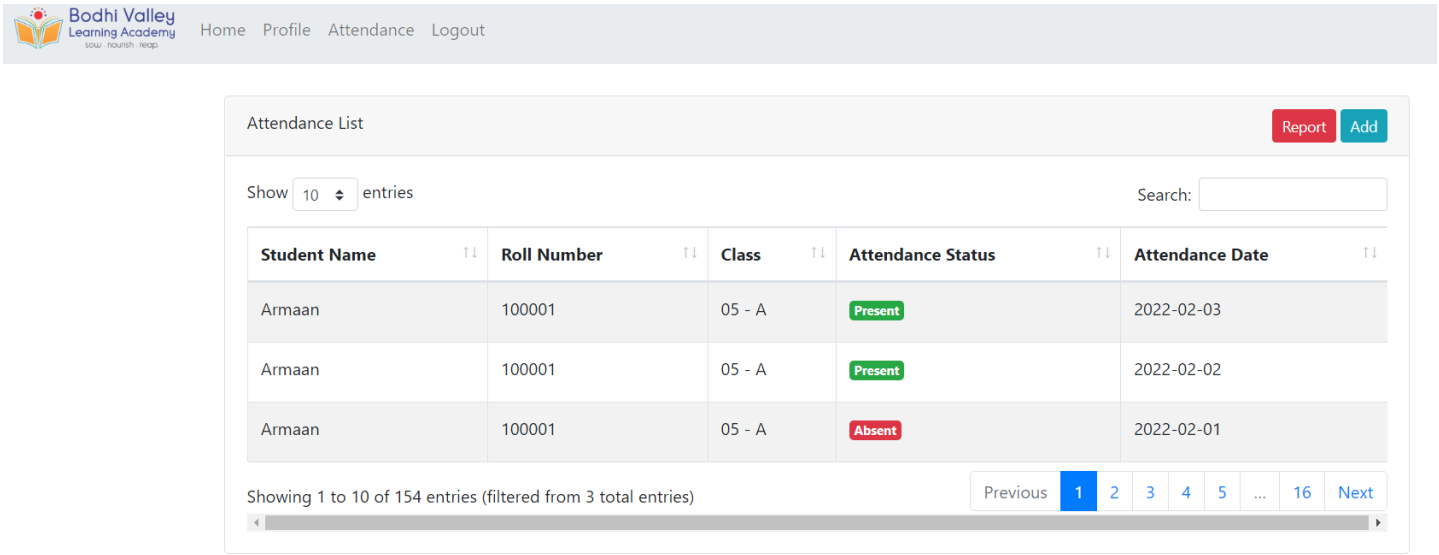
The HTTP request sent by Ajax is collected by PHP and stored into variables. After which, PHP connects to the MySQL database, and verifies the email address and password sent from the front end. If the email address and password match the PHP sends back a success response through Ajax to the front-end. If the username and password do not match, it will send an error message to the front-end.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	teacher_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	teacher_name	varchar(150)	latin1_swedish_ci		No	None			Change Drop More
3	teacher_address	text	latin1_swedish_ci		No	None			Change Drop More
4	teacher_emailid	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
5	teacher_password	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
6	teacher_qualification	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
7	teacher_doj	date			No	None			Change Drop More
8	teacher_image	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
9	teacher_grade_id	int(11)			No	None			Change Drop More

This is the MySQL database table used to store teacher information. The login email id and password is verified with the teacher_emailid and teacher_password columns.

Student Attendance table:



On this page the teacher can search, view, and add attendance for their respective students. In addition to that, the teacher can also generate reports of each individual student's attendance.

Storing student attendance to MySQL database:

```
{
    for($count = 0; $count < count($student_id); $count++)
    {
        $data = array(
            ':student_id'           =>  $student_id[$count],
            ':attendance_status'    =>  $_POST["attendance_status".$student_id[$count].""],
            ':attendance_date'      =>  $attendance_date,
            ':teacher_id'           =>  $_SESSION["teacher_id"]
        );

        $query = "
        INSERT INTO tbl_attendance
        (student_id, attendance_status, attendance_date, teacher_id)
        VALUES (:student_id, :attendance_status, :attendance_date, :teacher_id)
        ";
        $statement = $connect->prepare($query);
        $statement->execute($data);
    }
    $output = array(
        'success'       =>  'Data Added Successfully',
    );
}
```

Here, all the student attendance is collected and stored into an array. After which, it is looped through a for loop and inserted into MySQL database using INSERT query. Once the data, is inserted we get a success response ‘Data Added Successfully’ through Ajax to the front-end.

Student Attendance Table:

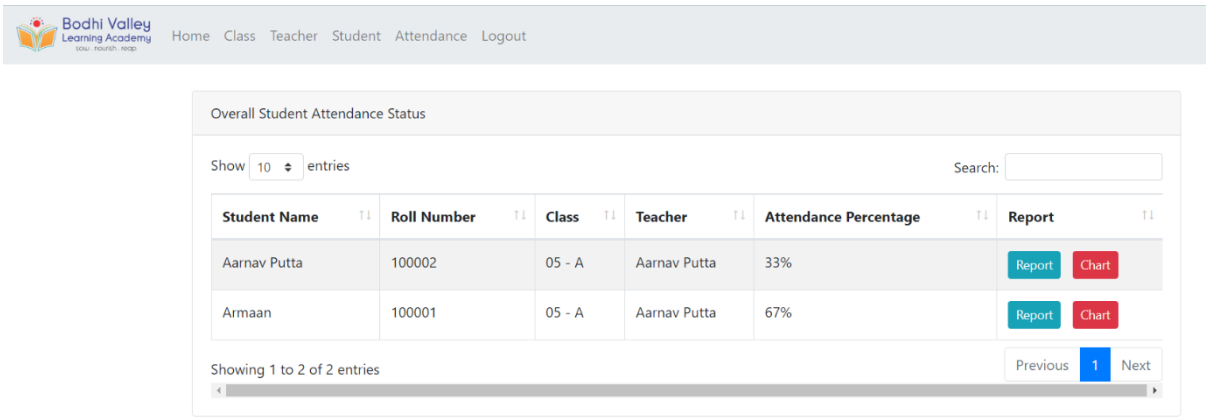
+ Options

	attendance_id	student_id	attendance_status	attendance_date	teacher_id
Edit Copy Delete	1	27	Present	2020-08-23	7
Edit Copy Delete	2	28	Present	2020-08-23	7
Edit Copy Delete	3	29	Absent	2020-08-23	7
Edit Copy Delete	4	30	Present	2020-08-23	7
Edit Copy Delete	5	31	Present	2020-08-23	7
Edit Copy Delete	6	27	Absent	2020-08-24	7
Edit Copy Delete	7	28	Present	2020-08-24	7
Edit Copy Delete	8	29	Present	2020-08-24	7
Edit Copy Delete	9	30	Absent	2020-08-24	7
Edit Copy Delete	10	31	Present	2020-08-24	7
Edit Copy Delete	11	27	Present	2020-08-25	7
Edit Copy Delete	12	28	Absent	2020-08-25	7
Edit Copy Delete	13	29	Present	2020-08-25	7
Edit Copy Delete	14	30	Present	2020-08-25	7
Edit Copy Delete	15	31	Present	2020-08-25	7
Edit Copy Delete	16	27	Present	2020-08-26	7
Edit Copy Delete	17	28	Present	2020-08-26	7
Edit Copy Delete	18	29	Present	2020-08-26	7

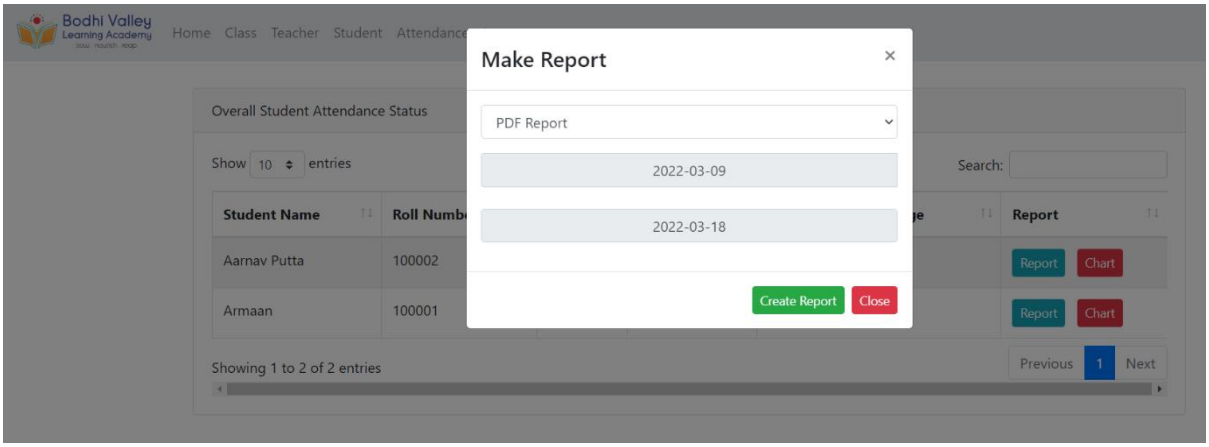
This table stores the student attendance status information, student ID, attendance date, and teacher ID.

Attendance report generator:

This feature enables the user to retrieve a detailed PDF and pie chart report of an individual student's attendance during a specific time period.



After clicking on the report option, you will get a pop-up as mentioned in the below screen that enables you to select a specific date range.



PDF Report:

Attendance Report

Student Name	Armaan
Roll Number	100001
Class	05 - A

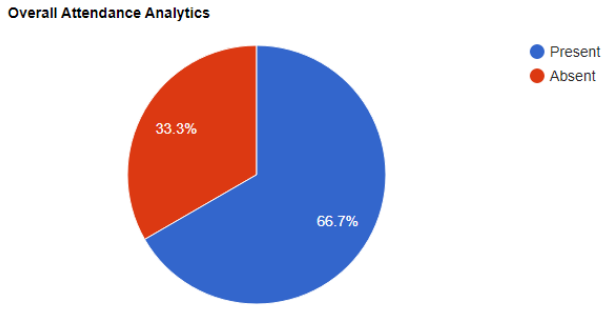
Attendance Details

Attendance Date	Attendance Status
2022-02-01	Absent
2022-02-02	Present
2022-02-03	Present

Chart Report:

Attendance Chart

Student Name	Armaan
Class	05 - A
Teacher Name	Aarnav Putta
Time Period	2022-02-01 to 2022-03-03



Date	Attendance Status
2022-02-01	Absent
2022-02-02	Present
2022-02-03	Present

Back-end report generating logic:

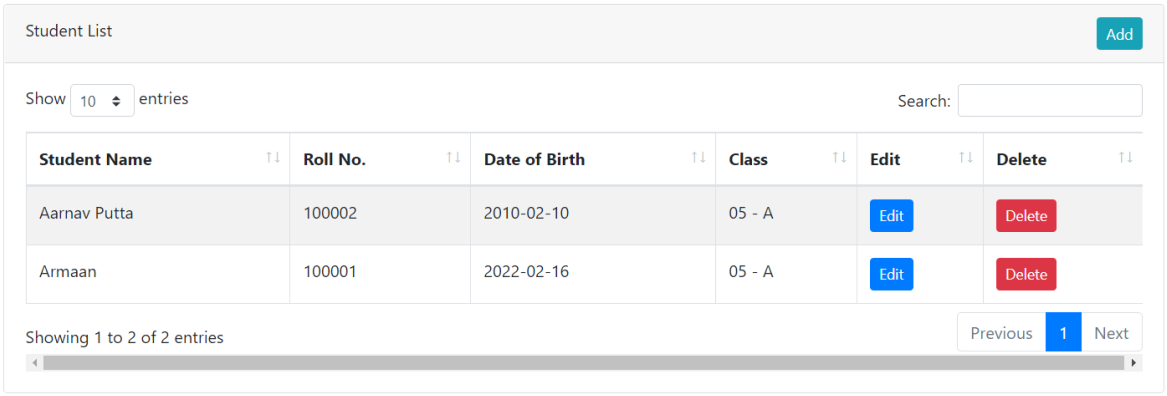
admin > report.php

```php
<?php

//report.php

if(isset($_GET["action"]))
{
    include('database_connection.php');
    require_once 'pdf.php';
    session_start();
    $output = '';
    if($_GET["action"] == 'attendance_report')
    {
        if(isset($_GET["grade_id"], $_GET["from_date"], $_GET["to_date"]))
        {
            $pdf = new Pdf();
            $query = "
            SELECT tbl_attendance.attendance_date FROM tbl_attendance
            INNER JOIN tbl_student
            ON tbl_student.student_id = tbl_attendance.student_id
            WHERE tbl_student.student_grade_id = '".$_GET["grade_id"]."'
            AND (tbl_attendance.attendance_date BETWEEN '".$_GET["from_date"]."' AND '".$_GET["to_date"]."')
            GROUP BY tbl_attendance.attendance_date
            ORDER BY tbl_attendance.attendance_date ASC
            ";
            $statement = $connect->prepare($query);
            $statement->execute();
            $result = $statement->fetchAll();
            $output .= '
                <style>
                @page { margin: 20px; }
```

Once the ‘Create Report’ button is clicked, Ajax sends a request to PHP. After which, based on the date range PHP runs the MySQL query and retrieves the data from the MySQL database. As the data is received, a response is sent back to Ajax. The file chart.php contains all the pie diagram logic and the file pdf.php contains all the logic for generating the pdf report.

Adding Student details:

To add student details, click on student option in the menu and below screen will be displayed.



Clicking on the add option enables the admin to add new students to the database. After which, student information can be edited using the edit button in the list. Students can also be deleted from the database using the delete button in the student list.

Back-end:

Adding student data:

```php
if($_POST["action"] == 'Add')
{
    $data = array(
        ':student_name'     =>  $student_name,
        ':student_roll_number'  =>  $student_roll_number,
        ':student_dob'      =>  $student_dob,
        ':student_grade_id' =>  $student_grade_id
    );
    $query = "
    INSERT INTO tbl_student
    (student_name, student_roll_number, student_dob, student_grade_id)
    VALUES (:student_name, :student_roll_number, :student_dob, :student_grade_id)
    ";

    $statement = $connect->prepare($query);
    if($statement->execute($data))
    {
        $output = array(
            'success'       =>  'Data Added Successfully',
        );
    }
```

Ajax sends a request to PHP to store the student record into MySQL database.

Editing student data:

```php
if($_POST["action"] == "Edit")
{
    $data = array(
        ':student_name'         =>  $student_name,
        ':student_roll_number'  =>  $student_roll_number,
        ':student_dob'          =>  $student_dob,
        ':student_grade_id'     =>  $student_grade_id,
        ':student_id'           =>  $_POST["student_id"]
    );
    $query = "
    UPDATE tbl_student
    SET student_name = :student_name,
    student_roll_number = :student_roll_number,
    student_dob = :student_dob,
    student_grade_id = :student_grade_id
    WHERE student_id = :student_id
    ";
    $statement = $connect->prepare($query);
    if($statement->execute($data))
    {
        $output = array(
            'success'       =>  'Data Edited Successfully',
        );
    }
}
```

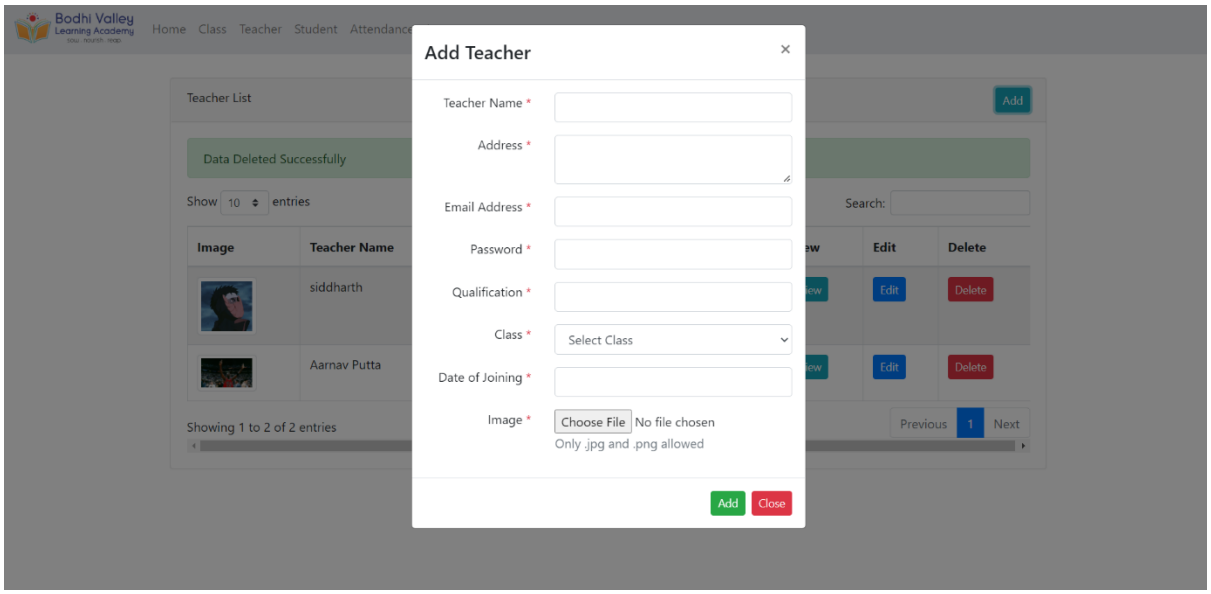
Ajax sends the student ID as a request to PHP. PHP runs the query with the student ID to update the information in MySQL database.

Deleting Student data:

```php
if($_POST["action"] == "delete")
{
    $query = "
    DELETE FROM tbl_student
    WHERE student_id = '".$_POST["student_id"]."'
    ";
    $statement = $connect->prepare($query);
    if($statement->execute())
    {
        echo 'Data Delete Successfully';
    }
}
```

Ajax sends the student ID as a request to PHP. PHP runs the query with the student ID to delete the information of the student in the MySQL database.

Adding Teacher details:



This feature enables the admin to create a teacher profile for the teacher to login and update student attendance.

Adding Teacher data:

```php
if($_POST["action"] == 'Add')
{
    $data = array(
        ':teacher_name'         =>  $teacher_name,
        ':teacher_address'      =>  $teacher_address,
        ':teacher_emailid'      =>  $teacher_emailid,
        ':teacher_password'     =>  password_hash($teacher_password, PASSWORD_DEFAULT),
        ':teacher_qualification'    =>  $teacher_qualification,
        ':teacher_doj'          =>  $teacher_doj,
        ':teacher_image'        =>  $teacher_image,
        ':teacher_grade_id'     =>  $teacher_grade_id
    );
    $query = "
    INSERT INTO tbl_teacher
    (teacher_name, teacher_address, teacher_emailid, teacher_password, teacher_qualification, teacher_doj, teacher_image, tea
    SELECT * FROM (SELECT :teacher_name, :teacher_address, :teacher_emailid, :teacher_password, :teacher_qualification, :teac
    WHERE NOT EXISTS (
        SELECT teacher_emailid FROM tbl_teacher WHERE teacher_emailid = :teacher_emailid
    ) LIMIT 1
    ";
    $statement = $connect->prepare($query);
    if($statement->execute($data))
    {
        if($statement->rowCount() > 0)
        {
            $output = array(
                'success'       =>  'Data Added Successfully',
            );
        }
        else
        {
            $output = array(
                'error'                 =>  true,
                'error_teacher_emailid' =>  'Email Already Exists'
            );
```

Ajax sends a request to PHP to store the teacher information along with login credentials into MySQL database.

Editing Teacher Data:

```php
if($_POST["action"] == "Edit")
{
    $data = array(
        ':teacher_name'     =>  $teacher_name,
        ':teacher_address'  =>  $teacher_address,
        ':teacher_qualification'    =>  $teacher_qualification,
        ':teacher_doj'      =>  $teacher_doj,
        ':teacher_image'    =>  $teacher_image,
        ':teacher_grade_id' =>  $teacher_grade_id,
        ':teacher_id'       =>  $_POST["teacher_id"]
    );
    $query = "
    UPDATE tbl_teacher
    SET teacher_name = :teacher_name,
    teacher_address = :teacher_address,
    teacher_grade_id = :teacher_grade_id,
    teacher_qualification = :teacher_qualification,
    teacher_doj = :teacher_doj,
    teacher_image = :teacher_image
    WHERE teacher_id = :teacher_id
    ";
    $statement = $connect->prepare($query);
    if($statement->execute($data))
    {
        $output = array(
            'success'       =>  'Data Edited Successfully',
        );
```

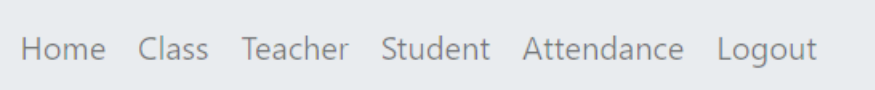
Ajax sends teacher username as a request to PHP. PHP runs the query with the username to update the information in the MySQL database.

Deleting Teacher information:

```php
if($_POST["action"] == "delete")
{
    $query = "
    DELETE FROM tbl_teacher
    WHERE teacher_id = '".$_POST["teacher_id"]."'
    ";
    $statement = $connect->prepare($query);
    if($statement->execute())
    {
        echo 'Data Deleted Successfully';
    }
}
```

Ajax sends the teacher username as a request to PHP. PHP runs the query with the teacher username to delete the information of the teacher in the MySQL database.

Logout functionality:



Clicking on the logout button will redirect the user to the login page. Along with that the user session will be terminated.

Back-end:

```php
<?php

//logout.php

session_start();

session_destroy();

header('location:login.php');

?>
```

The user's session will be terminated, after which they will be redirected to login.php.

Word count: 922