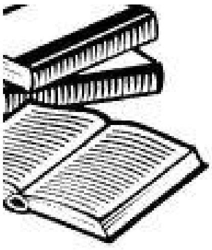
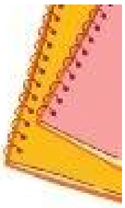
1. TECHNOLOGY AND PLATFORM

TITLE OF THE PROJECT





PROJECT SCOPE AND OBJECTIVE

- **User Management**
  - Student Access: View personal info, mark-sheets, attendance, and mentor details.
  - Admin Access (Faculty):Manage student details, academic records, attendance, and mentors.
- **Data Management**
  - Store and update personal info, academic records, and attendance.
- **Security and Permissions**

- Students view only their details.
- Admins control data changes.

➢ **Communication and Notifications**

- Display mentor details.
- Notify users of updates.

➢ **Reporting and Analytics**

- Generate performance and attendance reports.
- Analyze trends.

➢ **User Interface and Experience**

- User-friendly dashboards.
- Mobile-friendly design.

➢ **Scalability and Maintenance**

- Handle growth in users and data.
- Regular updates and maintenance.

OVERVIEW OF TECHNOLOGY / PLATFORM

➢ **Technology Used**:

- Front end:MyMento uses simple web technologies like HTML, CSS, and JavaScript to create a user-friendly interface. This makes it easy for students, teachers, and parents to access the platform on any device (computer, phone, or tablet).
- Back end:The back end, which handles data and processes, is built with tools like PHP, Python, MYsql. This part makes sure everything runs smoothly in the background.
- Database:Information such as student records and attendance is stored in a secure database like MySQL or PostgreSQL. This keeps the data organised and safe.

➢ **Platform**:

- Web-Based:MyMento is a website, so you can use it on any device with a web browser—no need to install anything! It works on laptops, smartphones, and tablets.

- **Integration**:
  - MyMento can connect to other tools or services, like sending emails for notifications, through APIs (special connections for different systems to work together).
- **Security**:
  - The platform uses strong security measures like encryption and restricted access to ensure that only authorized users can view or change data

FUTURE ENHANCEMENT OF TECHNOLOGY/ PLATFORM

- **Mobile App Development**:Developing a dedicated mobile app for MyMento can provide a more seamless experience for students, teachers, and parents. The app would allow users to access all features on the go with better notifications and offline access.
- **AI-Powered Analytics**:Integrating artificial intelligence (AI) to provide advanced analytics could help track student performance more efficiently. AI can offer personalised insights, predict student progress, and suggest improvements based on trends in academic performance.
- **Automated Mentorship Matching**:Future enhancements could include an automated system to match students with the most suitable mentors based on their academic records, interests, and career goals, ensuring a more personalised mentoring experience.
- **Integration with Learning Management Systems (LMS)**:MyMento can be integrated with popular LMS platforms like Moodle or Google Classroom to provide a one-stop solution for both mentorship and learning management. This would streamline the management of coursework, assignments, and mentoring in a single platform.
- **Gamification Features:**Adding gamification elements, such as rewards, badges, and leaderboards for attendance and performance, could motivate students to stay engaged and active in their academic journey.
- **Multi-Language Support**:Expanding MyMento to support multiple languages will make it accessible to a wider range of users, catering to diverse linguistic backgrounds and global institutions.

- **Enhanced Communication Tools**:Introducing more communication tools like video conferencing, real-time chat, or discussion forums within MyMento will improve interaction between students, teachers, and mentors, making the platform a complete solution for educational engagement.
- **Customisable Dashboards**:Allowing users to customise their dashboards, based on their roles (student, teacher, parent), would enhance the user experience, making it easier to access the most relevant information at a glance.

2. COMPANY OVERVIEW

NAME AND CONTACT DETAILS OF COMPANY/INSTITUTE

- **Name :-** S.V. INSTITUTE OF COMPUTER STUDIES
- **Logo :-**

- **Contact Details:-**
  - Phone Number :- 7096309000 and 7096308000

HISTORY OF COMPANY/INSTITUTE

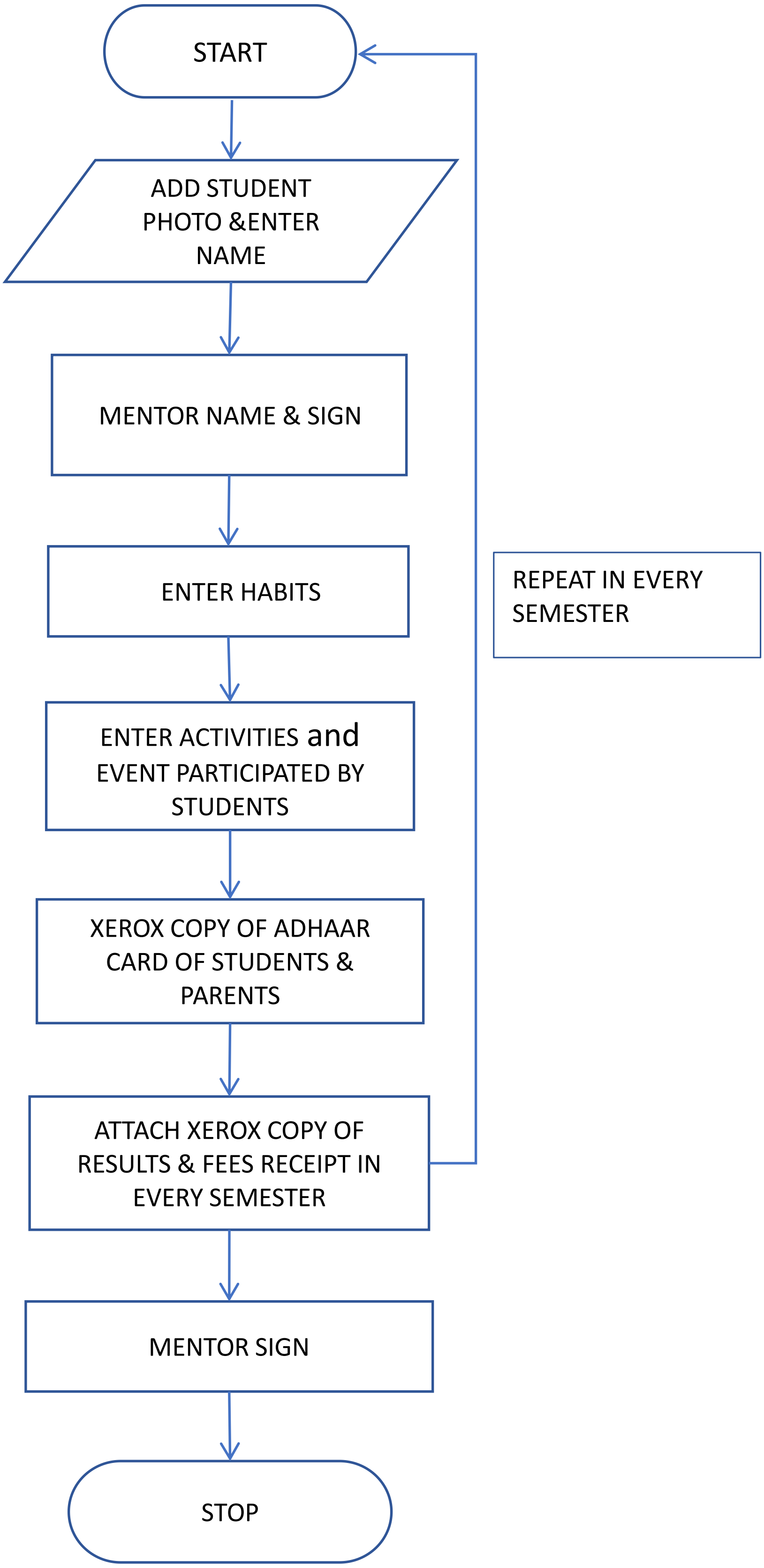
- The SV INSTITUTE OF COMPUTER STUDIES (SVICS), located in Gandhinagar, is an integral part of KADI SARVA VISHWAVIDYALAYA (KSV). Established with the goal of offering state-of-the-art education in computer studies, SVICS provides various undergraduate and postgraduate programs focusing on information technology and computer science. Notably, the institute offers an integrated BCA + MCA course, allowing students to pursue both degrees in a combined program.
- SVICS emphasises practical learning through a curriculum designed to keep pace with industry trends, preparing students to become skilled professionals in the IT sector. The institute also encourages industry interactions, internships, and research projects, enabling students to gain hands-on experience in real-world settings.

3. STUDY OF EXISTING SYSTEM

DRAWBACK OF EXISTING SYSTEM

- **Time-Consuming**: Entering and retrieving data manually takes a significant amount of time, especially when handling a large number of students. This can slow down processes like report generation or updating records.
- **Prone to Errors**: Manual entry increases the likelihood of human errors, such as misplacing documents, incorrect data entry, or omissions, which can compromise the accuracy of records.
- **Difficult to Update**: Updating physical records can be cumbersome, especially when changes need to be made frequently. Each update might require manual corrections or new entries, leading to inconsistencies.
- **Storage Space Issues**: As the number of students grows, storing all the physical documents becomes a challenge. Physical records take up space and require proper organization to avoid clutter and confusion.
- **Security Risks**: Physical documents are vulnerable to theft, loss, or damage due to accidents such as fires, floods, or mishandling. Securing sensitive data physically is more challenging compared to encryption or digital security methods.
- **Limited Accessibility**: Accessing data requires physically being present at the storage location, which limits the flexibility for faculty or administration to access information remotely or outside working hours.
- **No Backup**: Unlike digital systems, physical records lack an automatic backup mechanism. If records are lost or damaged, they cannot be easily restored.
- **Inefficiency in Data Sharing**: Sharing physical documents between departments or individuals can be slow and cumbersome, often requiring duplication and increasing the risk of misplacing information.
- **Environmental Impact**: The need for paper, printing, and storage contributes to resource wastage and has a negative impact on the environment due to paper consumption.

CURRENT WORKING METHOD OF SYSTEM (DIAGRAM)


START
ADD STUDENT PHOTO &ENTER NAME
MENTOR NAME & SIGN
ENTER HABITS
ENTER ACTIVITIES and EVENT PARTICIPATED BY STUDENTS
XEROX COPY OF ADHAAR CARD OF STUDENTS & PARENTS
ATTACH XEROX COPY OF RESULTS & FEES RECEIPT IN EVERY SEMESTER
MENTOR SIGN
STOP
REPEAT IN EVERY SEMESTER

EXISTING SYSTEM FLOW

(STUDY OF ENTIRE SYSTEM)

The current system uses physical mentoring books managed by mentors (admins). It requires physical storage, manual data entry, and in-person access, making it prone to human error and data loss. Communication and updates are manual, with no automated notifications. Generating reports and analyzing trends is time-consuming. Accessibility is limited, and scalability is challenging as student numbers grow. Maintenance involves physical upkeep and protection. Transitioning to a digital system like Mymento would improve accuracy, efficiency, and accessibility.

(ADVANTAGES)

- **Simplicity and Accessibility**: Physical records are easy to manage and access without the need for technical expertise or digital tools. Anyone familiar with traditional record-keeping can quickly retrieve information.
- **No Dependency on Technology**: Since it is a physical system, there’s no risk of data loss due to system crashes, hacking, or technological failures like software bugs or hardware breakdowns.
- **Personal Interaction**: A physical record-keeping system often allows for more personal involvement by faculty, which can help in understanding the student’s needs and tracking their progress closely.
- **Data Control**: Faculty has direct control over the data, ensuring that sensitive student information is handled with care and protected from unauthorized access that could happen in a digital system.
- **Immediate Reference**: In cases where digital systems are not practical or are under maintenance, physical books allow for immediate referencing of key student details without delay.
- **Low Initial Cost**: Maintaining physical records can sometimes be more cost-effective in institutions where setting up or maintaining a digital infrastructure is expensive.
- **Offline Availability**: Physical records are accessible even in areas with limited or no internet connectivity, making them reliable in environments where digital infrastructure is weak.

4. STUDY OF PROPOSED SYSTEM

PROPOSED SYSTEM TYPE

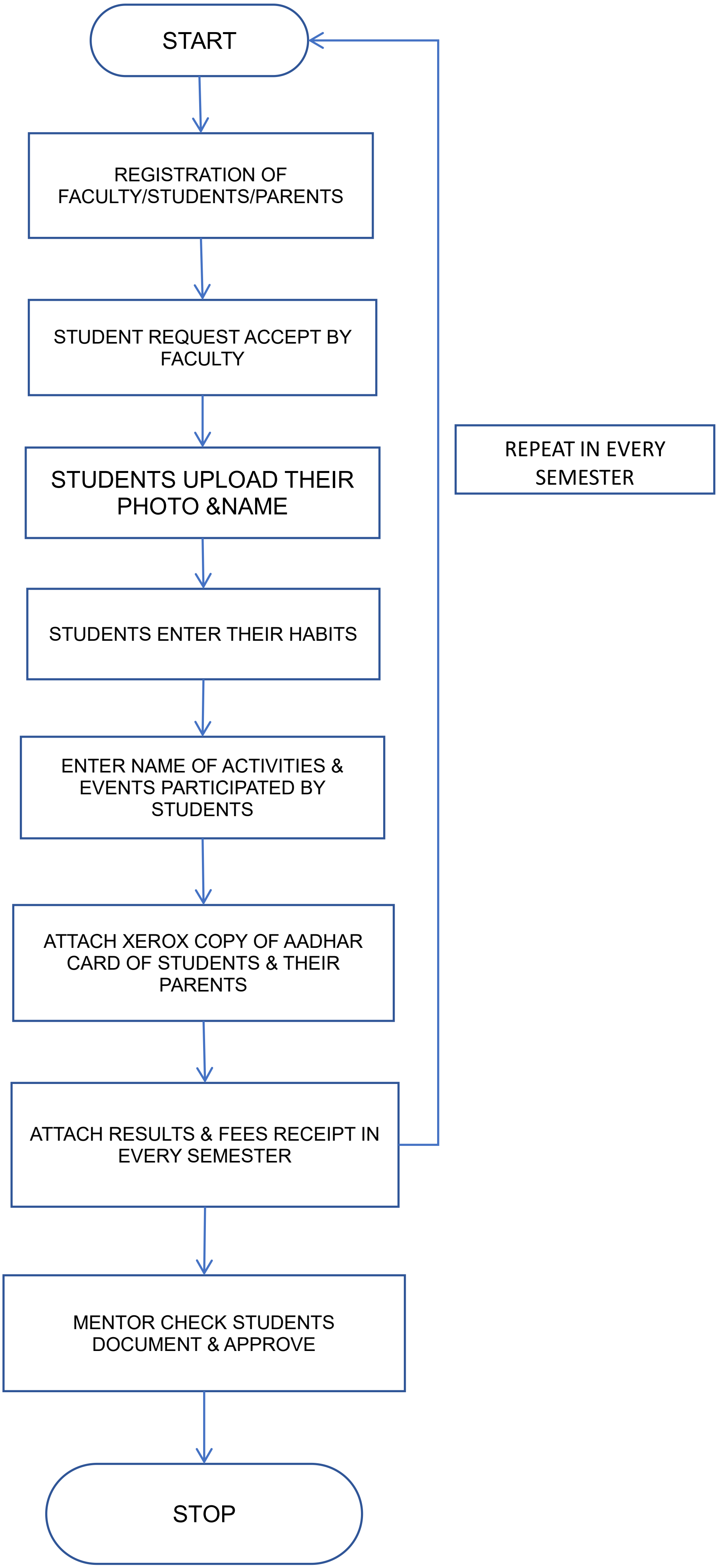
- The proposed system for MyMento is a web-based application designed to streamline mentoring processes and academic management. As a cloud-based platform, it operates over the internet, accessible through any browser on devices like computers, tablets, or smartphones.

Key Characteristics:

1. Centralized Data Management: All user data, such as student information, mentoring logs, and attendance, is stored and managed on a central server, ensuring easy access and updates.
2. Real-time Communication: The system supports real-time notifications and updates, allowing mentors and students to stay informed about attendance, mentoring schedules, and other academic-related activities.
3. Scalability: Being web-based, MyMento can easily scale to handle an increasing number of users, whether students, mentors, or administrators, without the need for major infrastructure upgrades.

In summary, MyMento is a modern, cloud-based solution, offering flexibility, accessibility, and efficient management for educational institutions.

PROPOSED SYSTEM FLOW(DIAGRAM)


START
REGISTRATION OF FACULTY/STUDENTS/PARENTS
STUDENT REQUEST ACCEPT BY FACULTY
STUDENTS UPLOAD THEIR PHOTO &NAME
STUDENTS ENTER THEIR HABITS
ENTER NAME OF ACTIVITIES & EVENTS PARTICIPATED BY STUDENTS
ATTACH XEROX COPY OF AADHAR CARD OF STUDENTS & THEIR PARENTS
ATTACH RESULTS & FEES RECEIPT IN EVERY SEMESTER
MENTOR CHECK STUDENTS DOCUMENT & APPROVE
STOP
REPEAT IN EVERY SEMESTER

(ADVANTAGES)

- **Efficiency**: Automated data entry and updates save time and reduce manual work.
- **Accuracy**: Reduces human errors in record-keeping.
- **Accessibility**: Provides 24/7 access to information for students and faculty.
- **Security**: Ensures data integrity with controlled access.
- **Scalability**: Capable of handling growing data and user numbers

(DISADVANTAGES)

- **Initial Setup Cost**: High development and implementation expenses.
- **Learning Curve**: Faculty and students may need time to adapt.
- **Data Security Risks**: Potential vulnerabilities to cyberattacks.
- **Dependence on Internet**: Requires a stable connection for access.
- **Maintenance**: Requires regular updates and ongoing system upkeep.
- **Privacy Concerns**: Risk of unauthorized access to sensitive student information.

(MODULE OF PROJECT)

- **Login**:-
  - Input:User credentials (username and password).
  - Process:Verify the credentials with the stored database information.
  - Output:Access granted to the system or prompt to retry.
- **Registration:-**
  - Input:Student information (name, email, student ID, etc.).
  - Process:Authenticate the information and create a new account in the database.
  - Output:Registration successful or retry message.
- **Attendance Tracking**:-
  - Input:Student attendance details (date, class, status).
  - Process:Record the attendance in the database and update records.
  - Output:Attendance data saved or error message.
- **Performance Tracking:-**
  - Input:Academic scores and progress reports.
  - Process:Update and store the performance data for each student.
  - Output:Generate performance reports or notification for missing data.
- **Notification**:-
  - Input:Details of low attendance, mentoring book updates, or other system alerts.
  - Process:Send notifications to users about low attendance or mentor updates.

- Output:Notifications sent successfully or retry.

5.FEASIBILITY STUDY

OPERATIONAL FEASIBILITY STUDY

- **User Training**:Training sessions for faculty and students to ease the transition from physical to digital systems.
- **Support**:A dedicated support team for technical issues and troubleshooting.
- **Routine Maintenance**:Regular system updates and maintenance

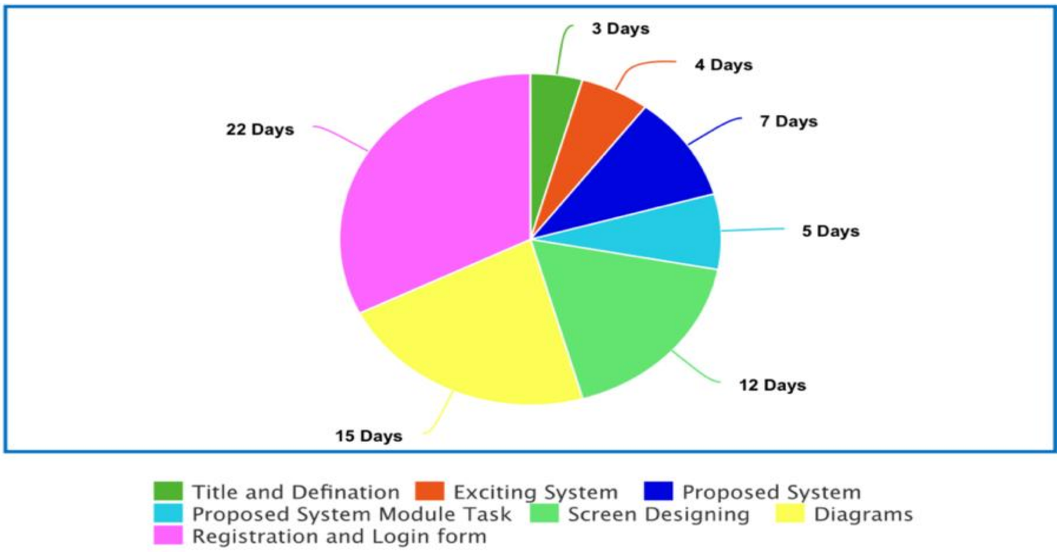
Technical Feasibility Study

- **Technology Stack**: The proposed system will use modern web technologies (e.g., HTML, CSS, JavaScript) and a robust backend (e.g. Python) with a secure database (e.g., MySQL).
- **Infrastructure**: The system will be hosted on reliable cloud platforms (e.g., AWS, Azure) with scalability features.
- **Integration**: The system will integrate with existing academic systems and databases.

Economical Feasibility Study

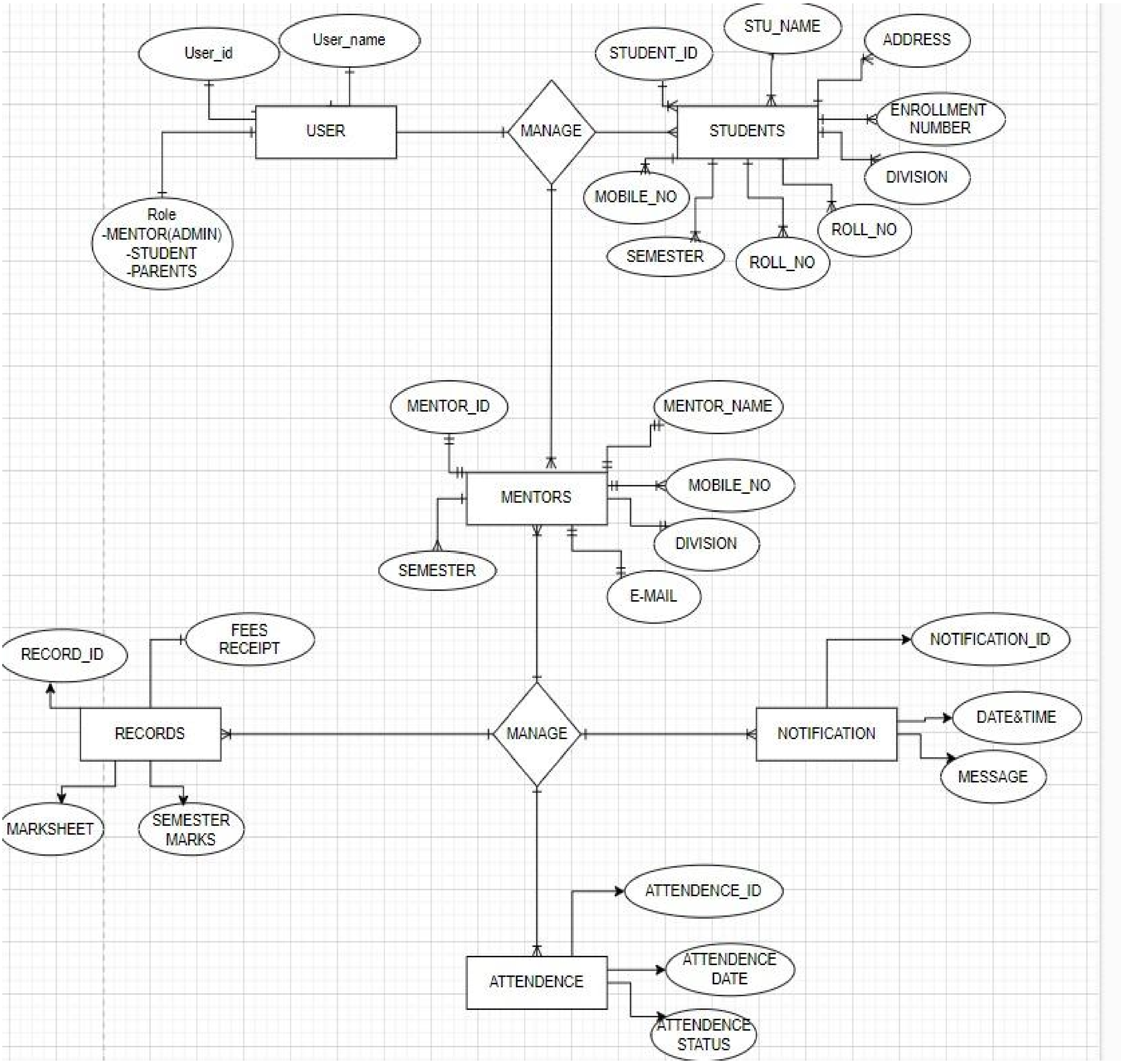
- **Tangible Benefits**:
- **Cost Savings**: Reduced paperwork and administrative tasks.
- **Cloud Hosting**: Lowers the need for physical servers and IT infrastructure.
- **Efficiency**: Automation reduces human errors and improves resource allocation.
- **Intangible Benefits**:
- **Student Engagement**: Enhanced communication and engagement through automated notifications.
- **Data-Driven Decisions**: Analytics improve decision-making on student performance.
- **Scalability**: Easily handles growth, ensuring future readiness.
- **Institutional Reputation**: Boosts the image of the institution with modern technology.

SCHEDULE FEASIBILITY STUDY(TIME LINE CHART)



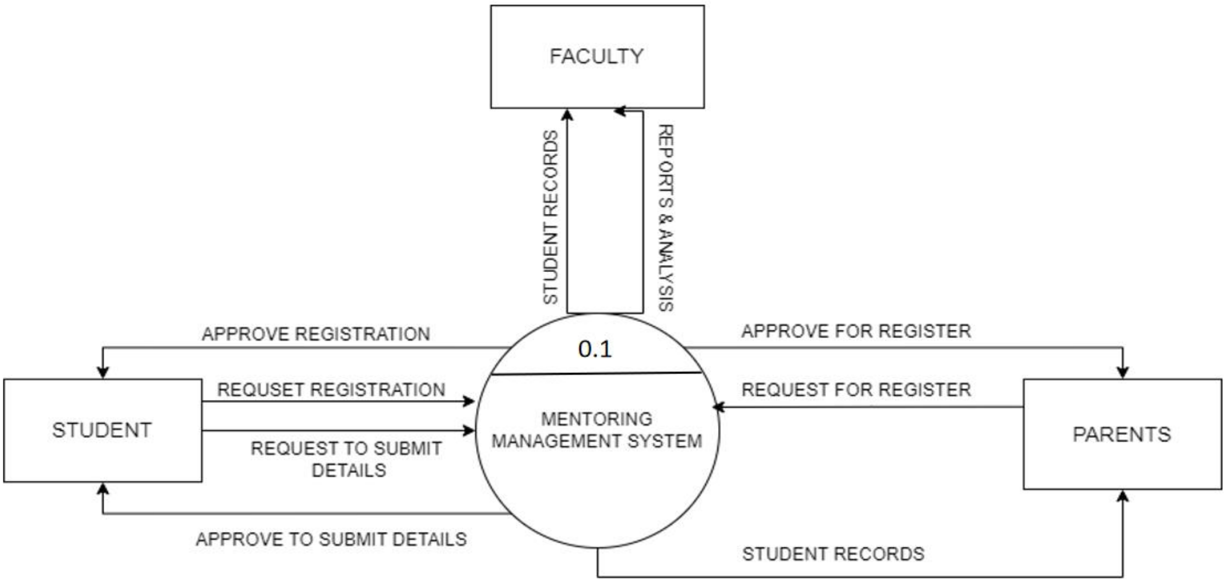
6.SYSTEM ANALYSIS AND DESIGN

E-R Diagram

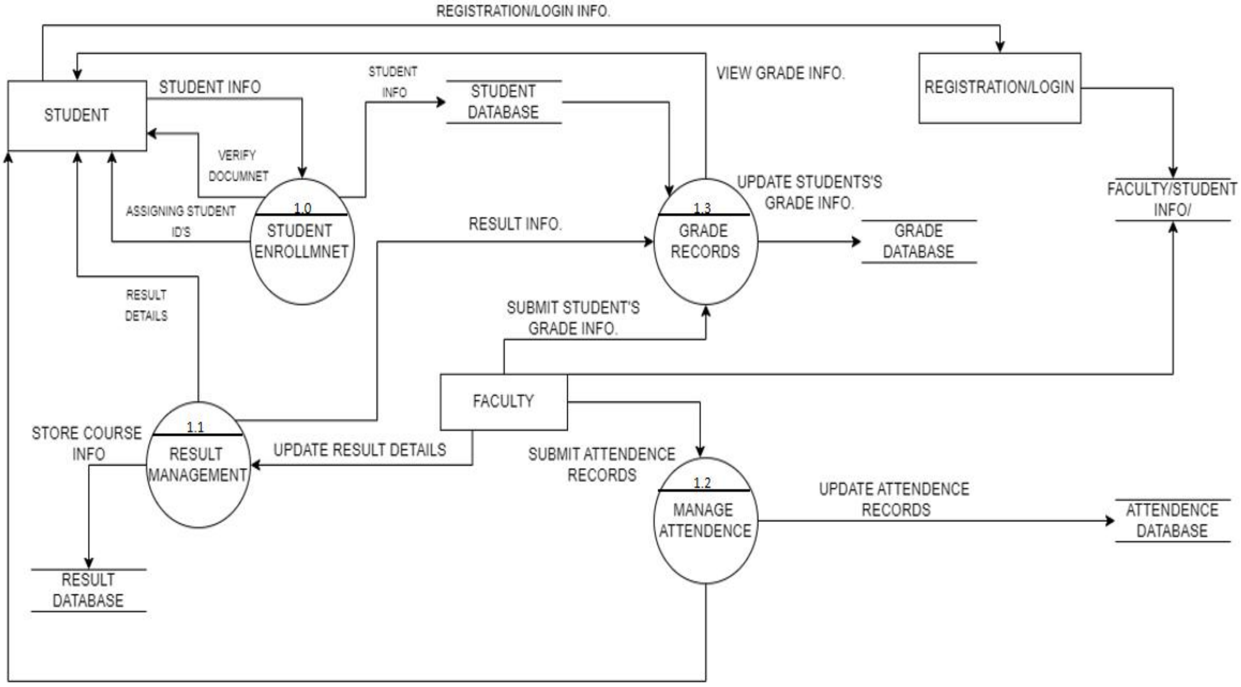


DFDS

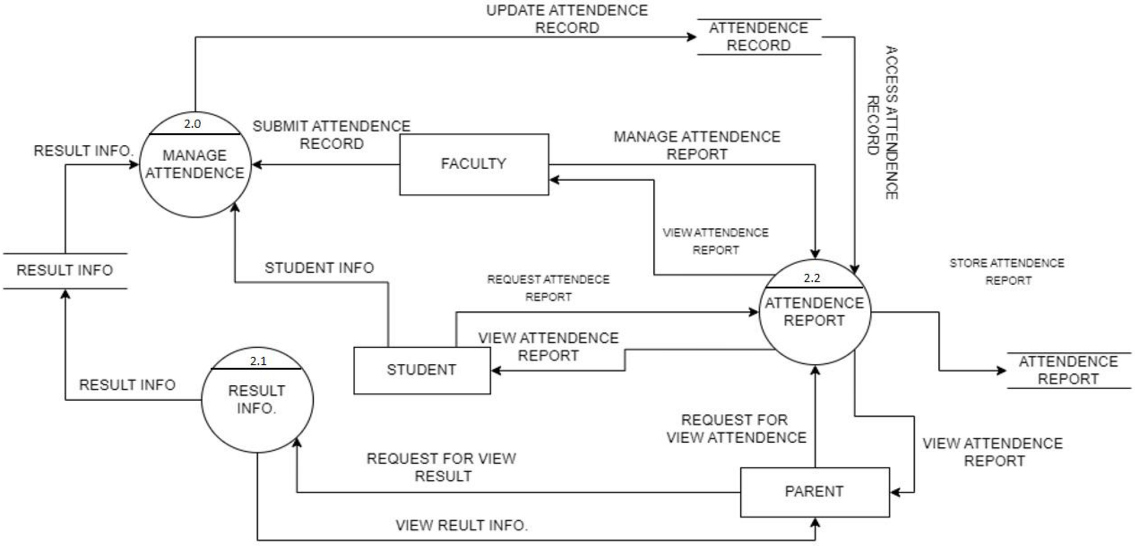
(CONTEXT LEVEL)



(1ST LEVEL DFD)

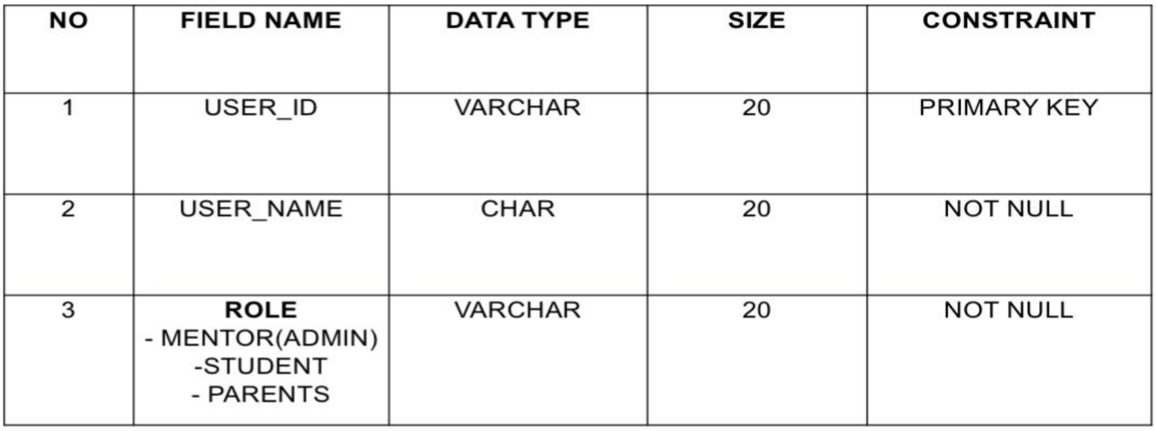


(2ND Level DFD)



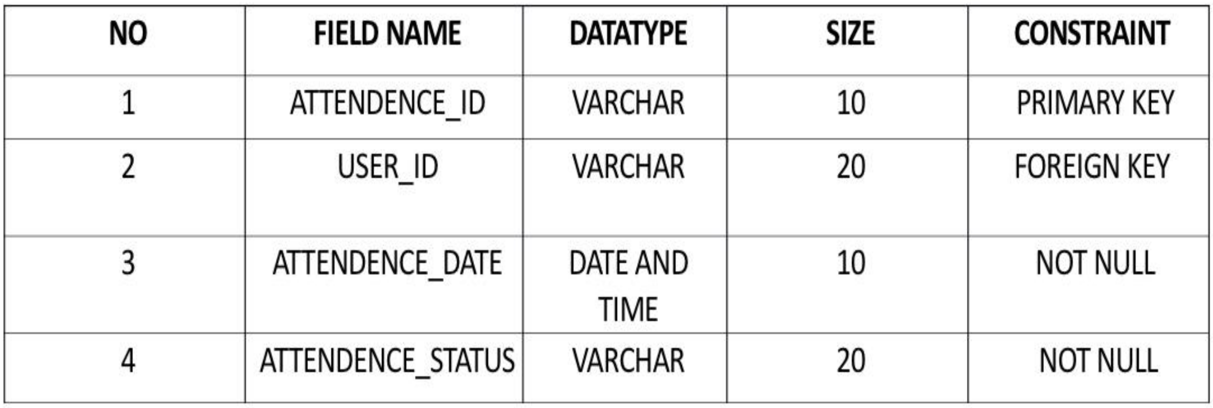
DATA DICTIONARY

(1.USER TABLE)



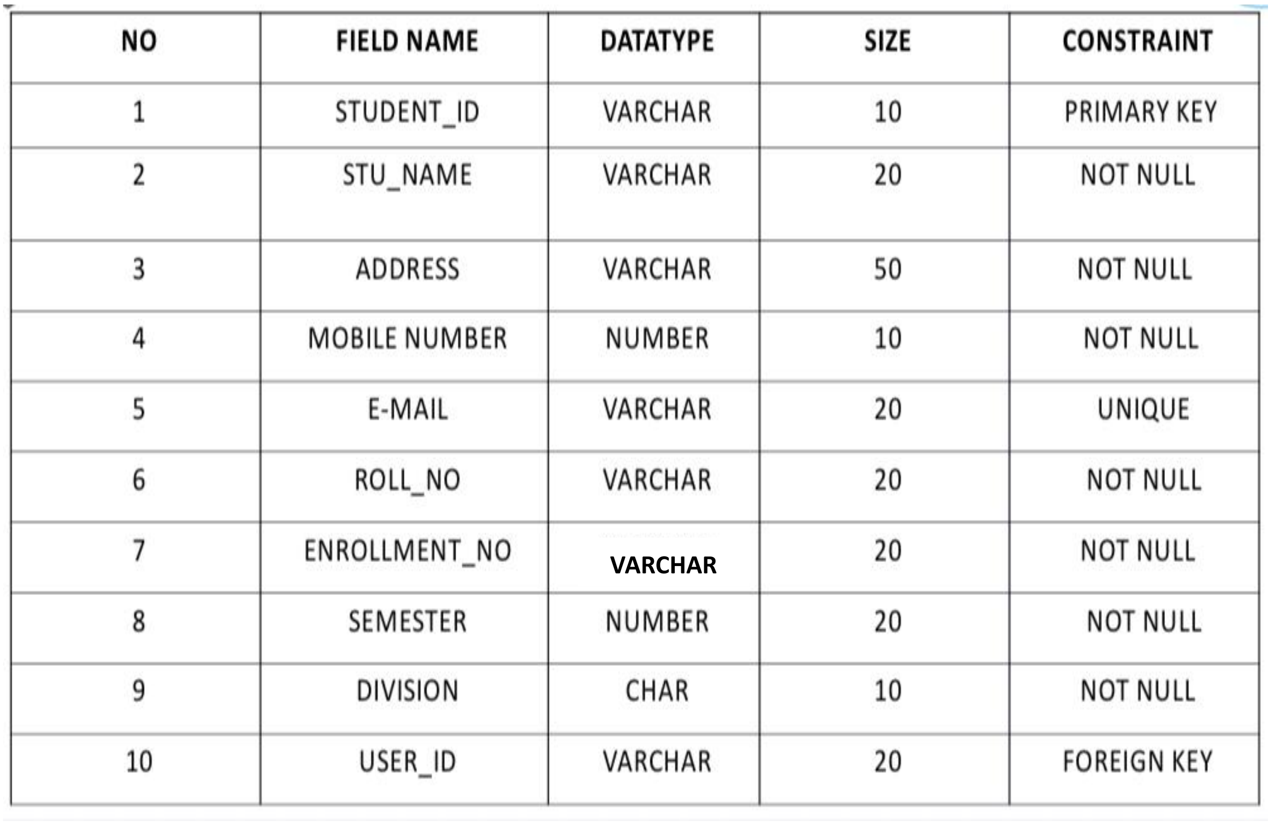
NO	FIELD NAME	DATA TYPE	SIZE	CONSTRAINT
1	USER_ID	VARCHAR	20	PRIMARY KEY
2	USER_NAME	CHAR	20	NOT NULL
3	**ROLE** - MENTOR(ADMIN) -STUDENT - PARENTS	VARCHAR	20	NOT NULL

(2.ATTENDANCE TABLE)



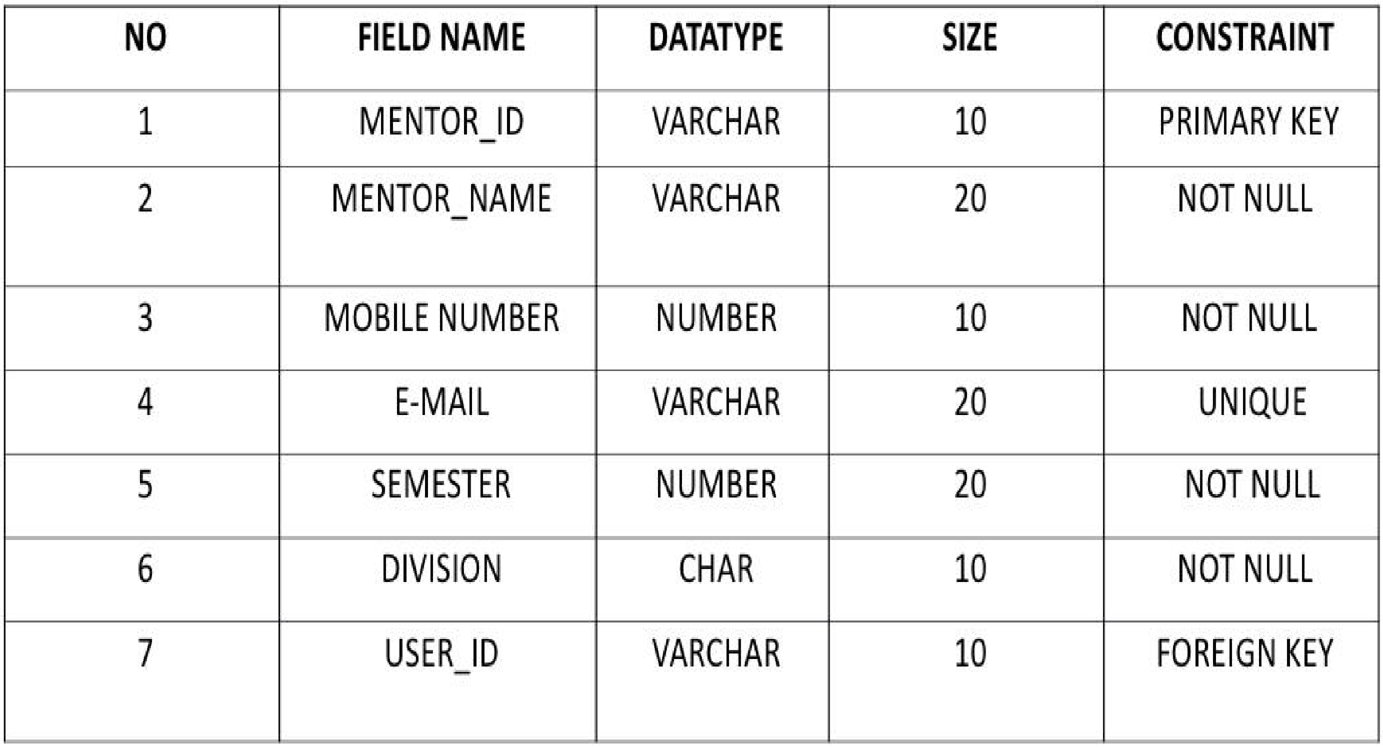
NO	FIELD NAME	DATATYPE	SIZE	CONSTRAINT
1	ATTENDENCE_ID	VARCHAR	10	PRIMARY KEY
2	USER_ID	VARCHAR	20	FOREIGN KEY
3	ATTENDENCE_DATE	DATE AND TIME	10	NOT NULL
4	ATTENDENCE_STATUS	VARCHAR	20	NOT NULL

(3.STUDENT TABLE)



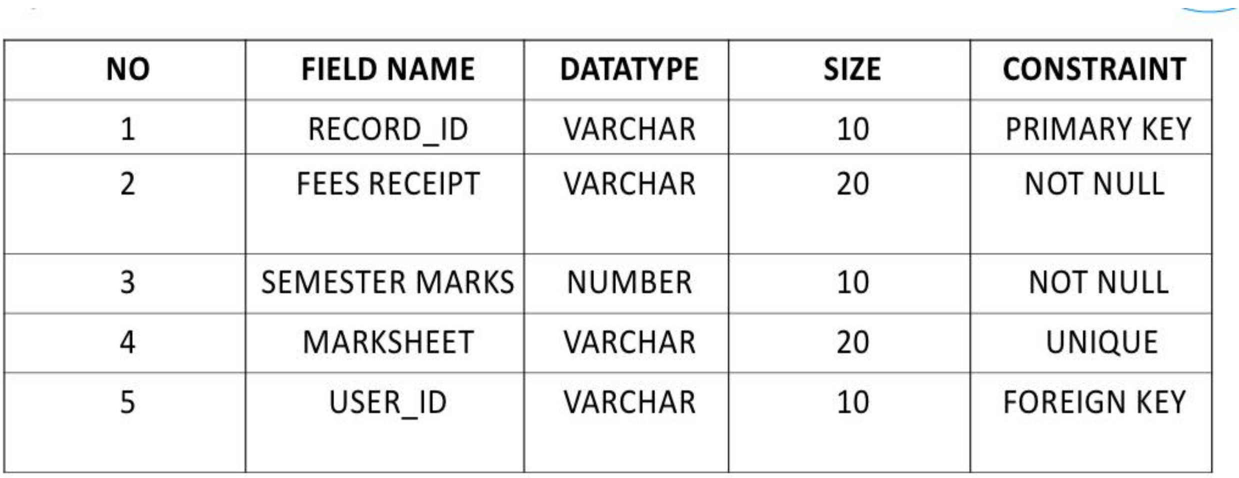
NO	FIELD NAME	DATATYPE	SIZE	CONSTRAINT
1	STUDENT_ID	VARCHAR	10	PRIMARY KEY
2	STU_NAME	VARCHAR	20	NOT NULL
3	ADDRESS	VARCHAR	50	NOT NULL
4	MOBILE NUMBER	NUMBER	10	NOT NULL
5	E-MAIL	VARCHAR	20	UNIQUE
6	ROLL_NO	VARCHAR	20	NOT NULL
7	ENROLLMENT_NO	VARCHAR	20	NOT NULL
8	SEMESTER	NUMBER	20	NOT NULL
9	DIVISION	CHAR	10	NOT NULL
10	USER_ID	VARCHAR	20	FOREIGN KEY

(4. MENTOR TABLE)



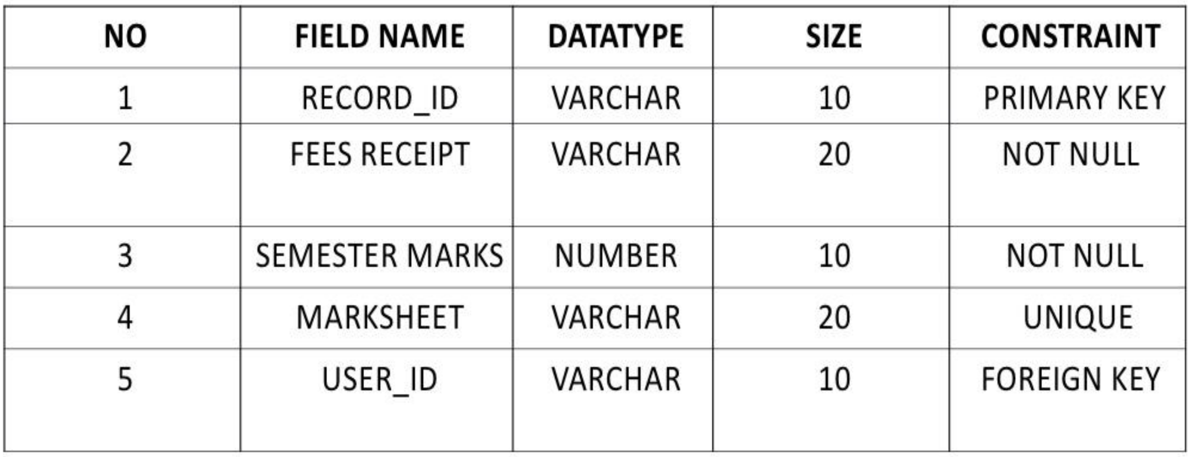
NO	FIELD NAME	DATATYPE	SIZE	CONSTRAINT
1	MENTOR_ID	VARCHAR	10	PRIMARY KEY
2	MENTOR_NAME	VARCHAR	20	NOT NULL
3	MOBILE NUMBER	NUMBER	10	NOT NULL
4	E-MAIL	VARCHAR	20	UNIQUE
5	SEMESTER	NUMBER	20	NOT NULL
6	DIVISION	CHAR	10	NOT NULL
7	USER_ID	VARCHAR	10	FOREIGN KEY

(5. RECORD TABLE)



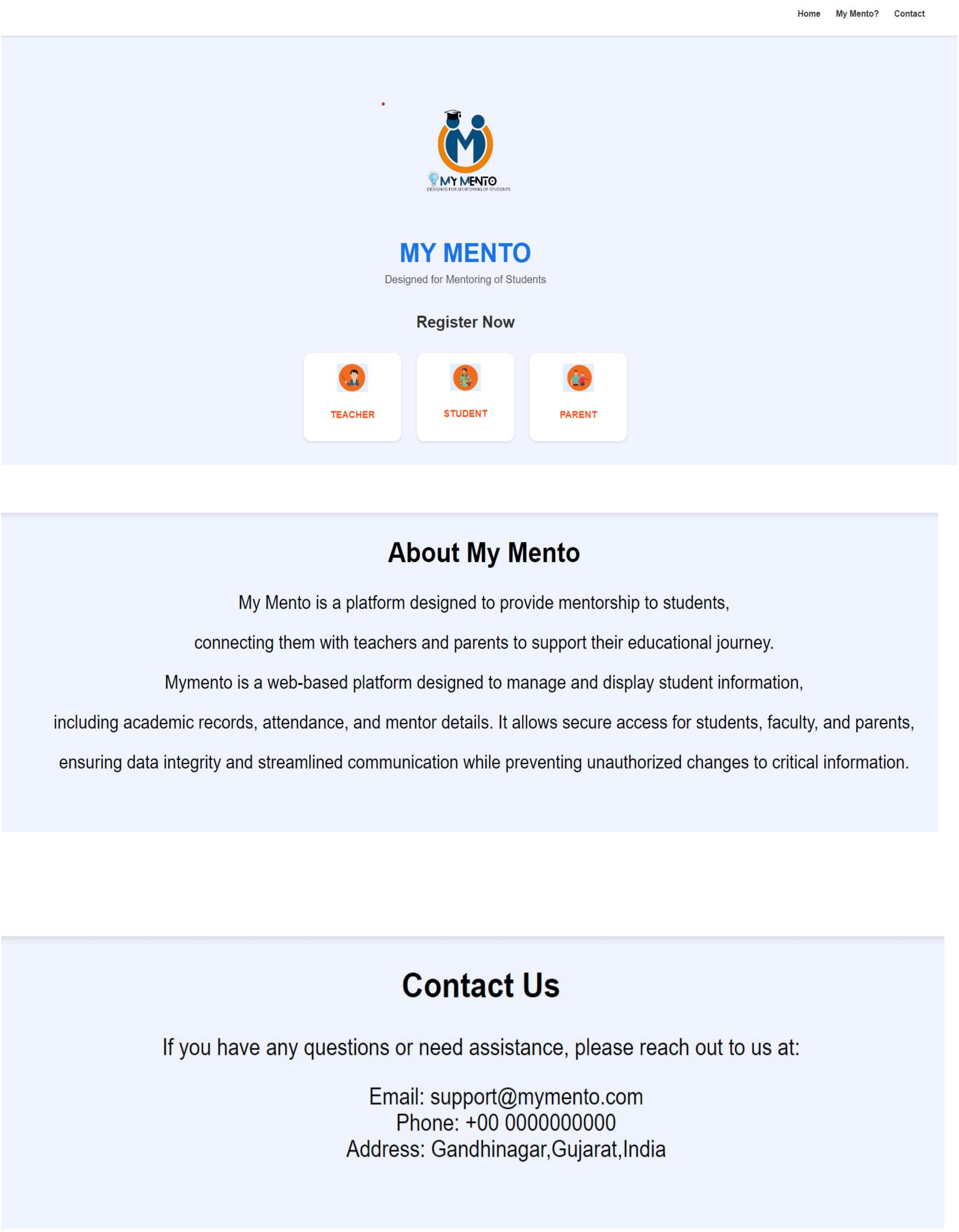
NO	FIELD NAME	DATATYPE	SIZE	CONSTRAINT
1	RECORD_ID	VARCHAR	10	PRIMARY KEY
2	FEES RECEIPT	VARCHAR	20	NOT NULL
3	SEMESTER MARKS	NUMBER	10	NOT NULL
4	MARKSHEET	VARCHAR	20	UNIQUE
5	USER_ID	VARCHAR	10	FOREIGN KEY

(6. NOTIFICATION TABLE)



NO	FIELD NAME	DATATYPE	SIZE	CONSTRAINT
1	RECORD_ID	VARCHAR	10	PRIMARY KEY
2	FEES RECEIPT	VARCHAR	20	NOT NULL
3	SEMESTER MARKS	NUMBER	10	NOT NULL
4	MARKSHEET	VARCHAR	20	UNIQUE
5	USER_ID	VARCHAR	10	FOREIGN KEY

PAGE LAYOUT



MY MENTO

Designed for Mentoring of Students

Register Now

About My Mento

My Mento is a platform designed to provide mentorship to students,

connecting them with teachers and parents to support their educational journey.

Mymento is a web-based platform designed to manage and display student information,

including academic records, attendance, and mentor details. It allows secure access for students, faculty, and parents,

ensuring data integrity and streamlined communication while preventing unauthorized changes to critical information.

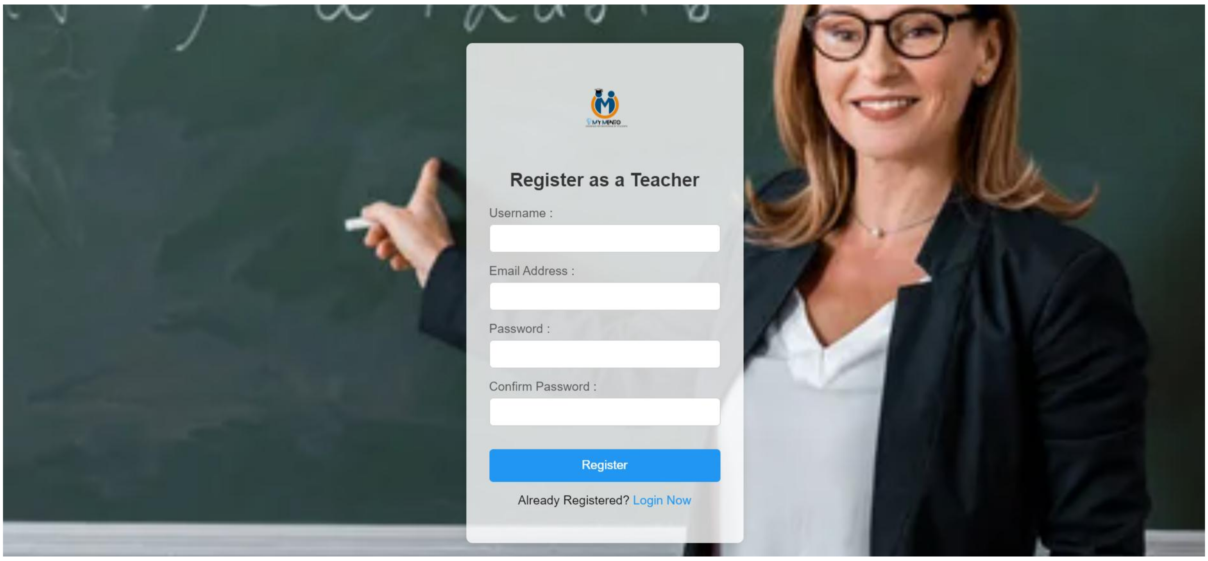
Contact Us

If you have any questions or need assistance, please reach out to us at:

Email: support@mymento.com
Phone: +00 0000000000
Address: Gandhinagar,Gujarat,India

7.REGISTRARION AND LOGIN FORM
(WITH CODING)

- Registration as a Teacher:-



Register as a Teacher

Username :

Email Address :

Password :

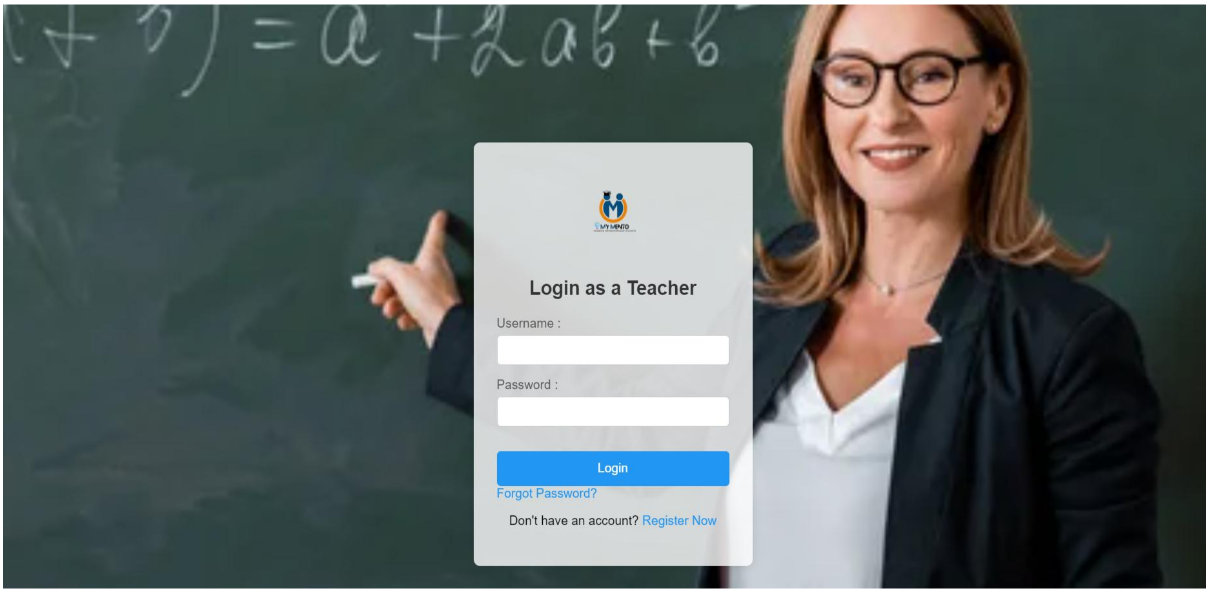
Confirm Password :

Code :-

```
 <!DOCTYPE html>
<html lang="en">
<head>
    <title>Teacher Registration</title>
    <link rel="stylesheet" href="mohit.css">
    <script>
        // Function to handle registration
        function handleRegistration() {
            const username = document.getElementById("username").value;
            const email = document.getElementById("email").value;
            const password = document.getElementById("password").value;
            const confirmPassword = document.getElementById("confirm-password").value;
            const confirmError = document.getElementById("confirm-error");

            // Clear any previous error messages
            confirmError.textContent = '';
            // Check if all fields are filled
            if (!username || !email || !password || !confirmPassword) {
                alert("Please fill in all fields!");
                return false; // Prevent form submission if fields are not filled
            }

            // Check if passwords match
            if (password !== confirmPassword) {
                confirmError.textContent = 'Passwords do not match!'; // Show error message
                return false; // Prevent form submission if passwords don't match
            }

            // If all checks pass
            alert("Registration Successful!");
            window.location.href = "dashboard.html"; // Redirect to the dashboard
            return false; // Prevent actual form submission and page reload
        }
    </script>
</head>
<body>
    <div class="register-container">
        <div class="register-box">
            <div class="register-header">
                <img src="MOHIT.png" alt="Logo" class="logo">
                <h2>Register as a Teacher</h2>
            </div>
            <form action = "registerteacher.php" method="post" id="register-form" onsubmit="return handleRegistration();">
                <div class="form-group">
                    <label for="username">Username :</label>
                    <input type="text" id="username" name="username" required>
                </div>
                <div class="form-group">
```

```
          <label for="email">Email Address :</label>
          <input type="email" id="email" name="email" required
              pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.com$"
              title="Please enter a valid email address with a .com domain (e.g.,
user@example.com)">
        </div>
        <div class="form-group">
          <label for="password">Password :</label>
          <input type="password" id="password" name="password" required>
        </div>
        <div class="form-group">
          <label for="confirmpassword">Confirm Password :</label>
          <input type="password" id="confirmpassword" name="confirmpassword"
required>
          <div id="confirm-error" class="error" style="color: red;"></div>
        </div>
        <button type="submit" class="register-btn">Register</button>
      </form>

      <p class="login-link">Already Registered? <a href="loginteacher.html">Login
Now</a></p>
    </div>
  </div>
</body>
</html>
```

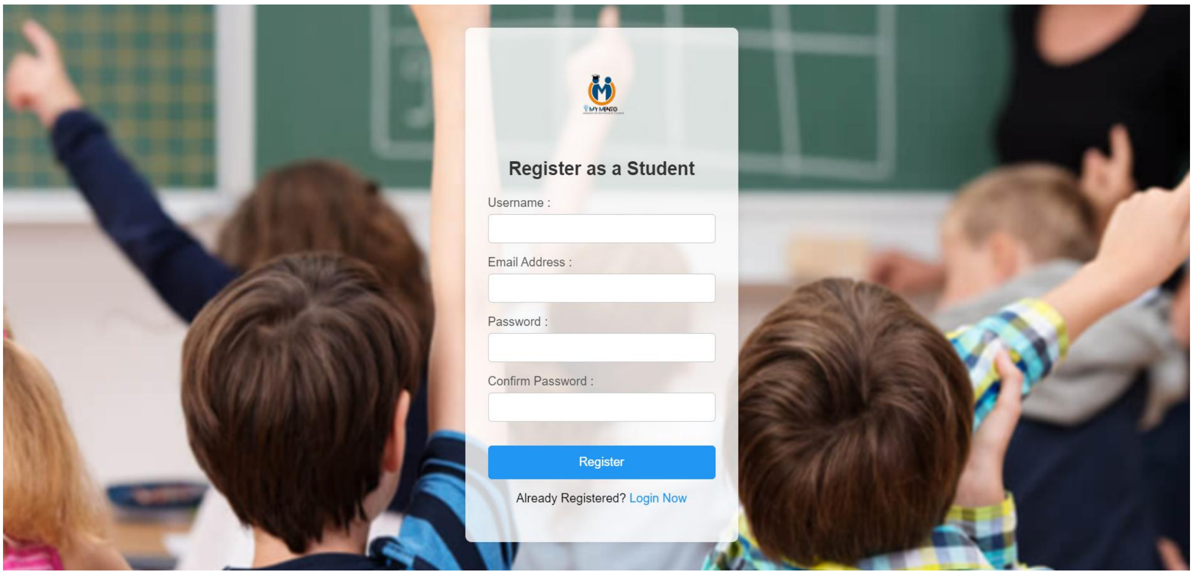
- Login as a Teacher:-



Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Teacher Login</title>
    <link rel="stylesheet" href="mohit.css">
    <script>

        function handleLogin() {
            const username = document.getElementById("login-username").value;
            const password = document.getElementById("login-password").value;
            const confirmPassword = document.getElementById("confirmPassword").value;
            const confirmError = document.getElementById("confirmError");

                        confirmError.textContent = '';

            if (!username || !password) {
                alert("Please fill in all fields!");
                return false;              }
            alert("Login Successful!");
            window.location.href = "dashboard.html";
            return false;
        }
        function handleForgotPassword(event) {
            event.preventDefault();
            alert("Redirecting to password recovery page...");

            return false;
        }
    </script>
</head>
<body>
    <div class="register-container">
        <div class="register-box">
            <div class="register-header">
                <img src="mohit.png" alt="Logo" class="logo">
                <h2>Login as a Teacher</h2>
            </div>
            <form id="login-form" onsubmit="return handleLogin();">
                <div class="form-group">
                    <label for="login-username">Username :</label>
                <input type="text" id="login-username" name="login-username" required>
                </div>
                <div class="form-group">
                    <label for="login-password">Password :</label>
        <input type="password" id="login-password" name="login-password" required>
                </div>

                <button type="submit" class="login-btn">Login</button>
            </form>
            <div>
               <a href="#" id="forgotPassword" onclick="handleForgotPassword(event);">Forgot
Password?</a>
            </div>
            <p class="register-link">Don't have an account? <a
href="registerteacher.html">Register Now</a></p>
         </div>
    </div>
</body> </html>
```

➢ Registration as a Student

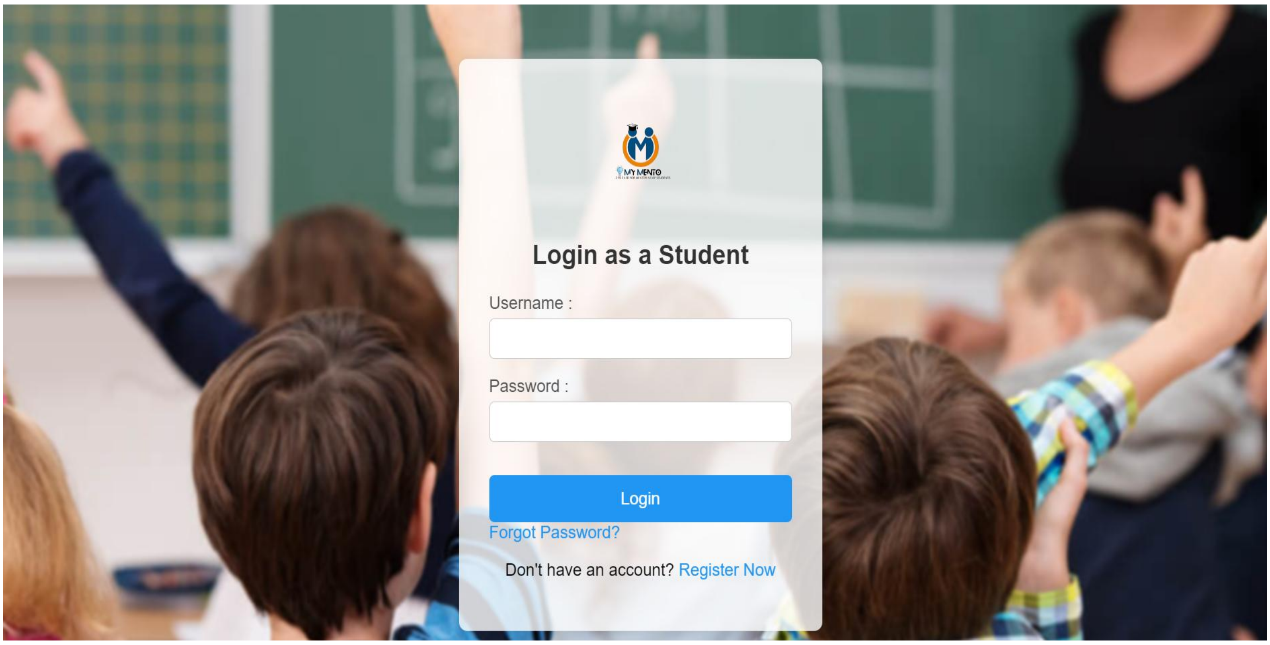


Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Student Registration</title>
    <link rel="stylesheet" href="dev.css">
    <script>
        // Function to handle registration
        function handleRegistration() {
            const username = document.getElementById("username").value;
            const email = document.getElementById("email").value;
            const password = document.getElementById("password").value;
            const confirmPassword = document.getElementById("confirm-password").value;
            const confirmError = document.getElementById("confirm-error");

            // Clear any previous error messages
            confirmError.textContent = '';

            // Check if all fields are filled
            if (!username || !email || !password || !confirmPassword) {
                alert("Please fill in all fields!");
                return false; // Prevent form submission if fields are not filled
            }

            // Check if passwords match
            if (password !== confirmPassword) {
                confirmError.textContent = 'Passwords do not match!'; // Show error message
                return false; // Prevent form submission if passwords don't match
            }

            // If all checks pass
            alert("Registration Successful!");
            window.location.href = "dashboard.html"; // Redirect to the dashboard
            return false; // Prevent actual form submission and page reload
        }
    </script>
</head>
<body>
    <div class="register-container">
        <div class="register-box">
            <div class="register-header">
                <img src="mohit.png" alt="Logo" class="logo">
                <h2>Register as a Student</h2>
            </div>
```

```
      <form id="register-form" onsubmit="return handleRegistration();">
        <div class="form-group">
          <label for="username">Username :</label>
          <input type="text" id="username" name="username" required>
        </div>
        <div class="form-group">
          <label for="email">Email Address :</label>
          <input type="email" id="email" name="email" required>
        </div>
        <div class="form-group">
          <label for="password">Password :</label>
          <input type="password" id="password" name="password" required>
        </div>
        <div class="form-group">
          <label for="confirm-password">Confirm Password :</label>
          <input type="password" id="confirm-password" name="confirm-password" required>
          <div id="confirm-error" class="error" style="color: red;"></div> <!-- Error message for password mismatch -->
        </div>
        <button type="submit" class="register-btn">Register</button>
      </form>

      <p class="login-link">Already Registered? <a href="loginstudent.html">Login Now</a></p>
    </div>
  </div>
</body>
</html>
```

- **Login as a Student**



Login as a Student

Username :

Password :

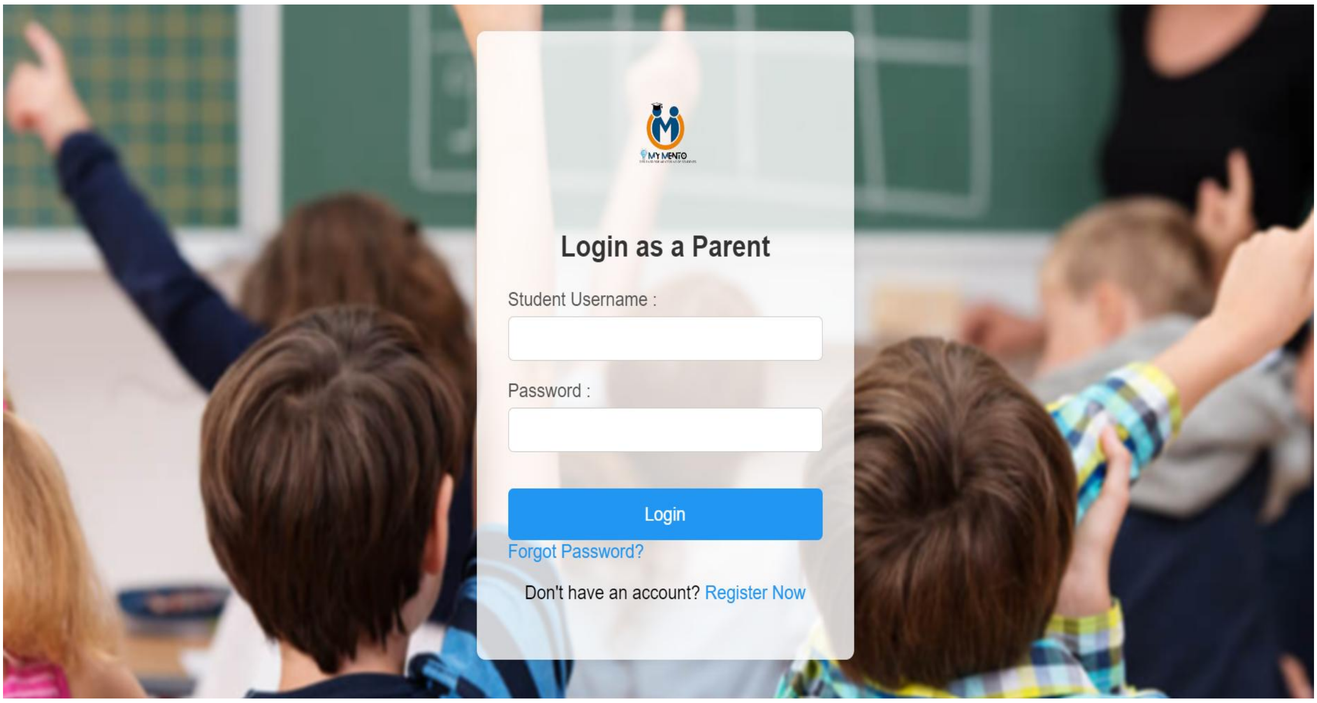
Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Student Login</title>
  <link rel="stylesheet" href="dev.css">
  <script>
    // Function to handle login
    function handleLogin() {
      const username = document.getElementById("login-username").value;
      const password = document.getElementById("login-password").value;
      const confirmPassword = document.getElementById("confirmPassword").value;
      const confirmError = document.getElementById("confirmError");

      // Clear previous error message
      confirmError.textContent = '';
```

```
            // Check if username and password are filled
            if (!username || !password) {
                alert("Please fill in all fields!"); // Show alert for empty fields
                return false; // Prevent form submission if fields are not filled
            }



            // If all checks pass
            alert("Login Successful!"); // Show success alert
            window.location.href = "dashboard.html"; // Redirect to the dashboard
            return false; // Prevent actual form submission and page reload
        }

        // Function to handle forgot password link click
        function handleForgotPassword() {
            alert("Redirecting to password recovery page..."); // Replace with actual logic
            return false; // Prevent default anchor behavior
        }
    </script>
</head>
<body>
    <div class="register-container">
        <div class="register-box">
            <div class="register-header">
                <img src="mohit.png" alt="Logo" class="logo">
                <h2>Login as a Student</h2>
            </div>
            <form id="login-form" onsubmit="return handleLogin();">
                <div class="form-group">
                    <label for="login-username">Username :</label>
                    <input type="text" id="login-username" name="login-username" required>
                </div>
                <div class="form-group">
```

```
          <label for="login-password">Password :</label>
          <input type="password" id="login-password" name="login-password" required>
        </div>

        <button type="submit" class="login-btn">Login</button>
      </form>
      <div>
        <a href="#" id="forgotPassword" onclick="return handleForgotPassword();">Forgot Password?</a>
      </div>
      <p class="register-link">Don't have an account? <a href="registerstudent.html">Register Now</a></p>
    </div>
  </div>
</body>
</html>
```

- **Login as a Parent**

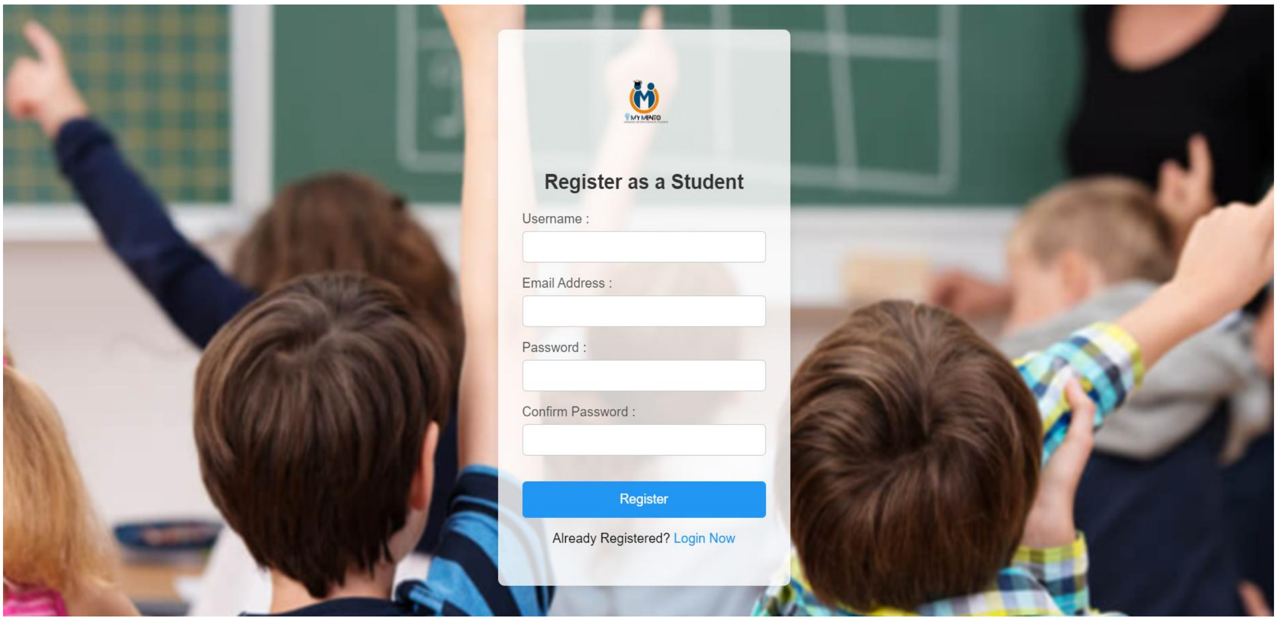


Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Parent Login</title>
   <link rel="stylesheet" href="dev.css">
   <script>
      // Function to handle login
      function handleLogin() {
         const username = document.getElementById("login-username").value;
         const password = document.getElementById("login-password").value;

         // Check if username and password are filled
         if (!username || !password) {
            alert("Please fill in all fields!"); // Show alert for empty fields
            return false; // Prevent form submission if fields are not filled
         }

         // If all checks pass
         alert("Login Successful!"); // Show success alert
         window.location.href = "dashboard.html"; // Redirect to the dashboard
         return false; // Prevent actual form submission and page reload
      }

      // Function to handle forgot password link click
      function handleForgotPassword() {
         alert("Redirecting to password recovery page..."); // Replace with actual logic
         return false; // Prevent default anchor behavior
      }
   </script>
</head>
<body>
```

```
    <div class="register-container">
        <div class="register-box">
            <div class="register-header">
                <img src="mohit.png" alt="Logo" class="logo">
                <h2>Login as a Parent</h2>
            </div>
            <form id="login-form" onsubmit="return handleLogin();">
                <div class="form-group">
                    <label for="login-username">Student Username :</label>
                    <input type="text" id="login-username" name="login-username" required>
                </div>
                <div class="form-group">
                    <label for="login-password">Password :</label>
                    <input type="password" id="login-password" name="login-password" required>
                </div>
                <button type="submit" class="login-btn">Login</button>
            </form>
            <div>
                <a href="#" id="forgotPassword" onclick="return
handleForgotPassword();">Forgot Password?</a>
            </div>
            <p class="register-link">Don't have an account? <a
href="registerstudent.html">Register Now</a></p>
        </div>
    </div>
</body>
</html>
```

➢ Registration as a Parent



Register as a Student

Username :

Email Address :

Password :

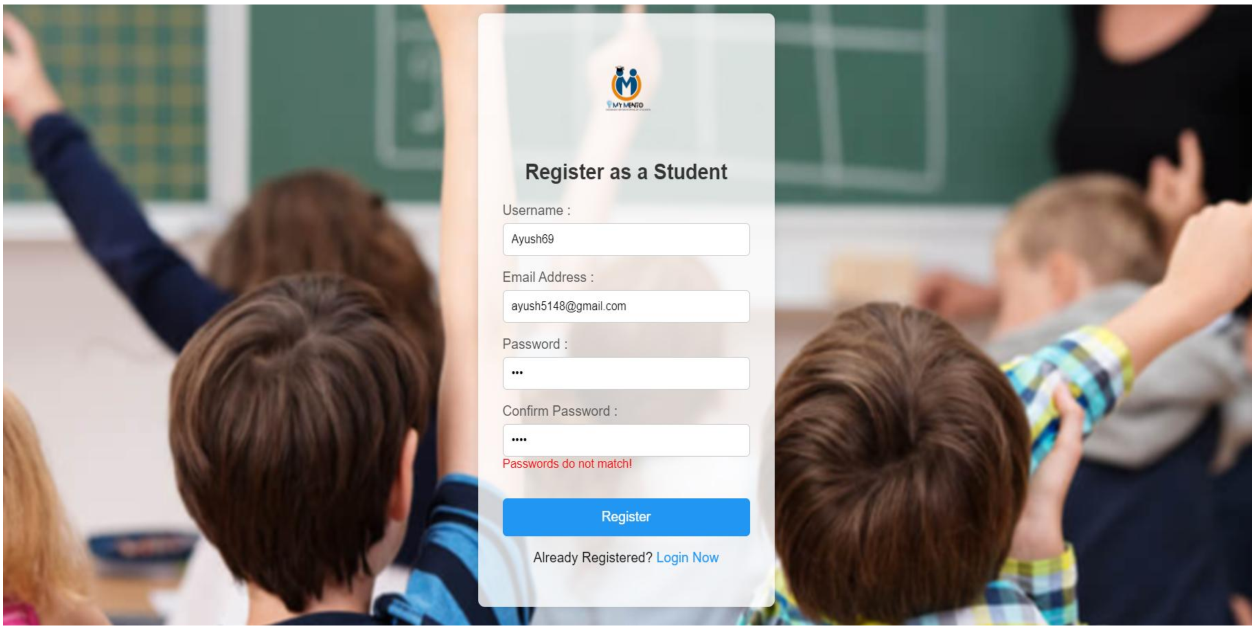
Confirm Password :

Code:-

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Student Registration</title>
  <link rel="stylesheet" href="dev.css">
  <script>
    // Function to handle registration
    function handleRegistration() {
      const username = document.getElementById("username").value;
      const email = document.getElementById("email").value;
      const password = document.getElementById("password").value;
      const confirmPassword = document.getElementById("confirm-password").value;
      const confirmError = document.getElementById("confirm-error");

      // Clear any previous error messages
      confirmError.textContent = '';

      // Check if all fields are filled
      if (!username || !email || !password || !confirmPassword) {
          alert("Please fill in all fields!");
          return false; // Prevent form submission if fields are not filled
      }

      // Check if passwords match
      if (password !== confirmPassword) {
          confirmError.textContent = 'Passwords do not match!'; // Show error message
          return false; // Prevent form submission if passwords don't match
      }

      // If all checks pass
      alert("Registration Successful!");
      window.location.href = "dashboard.html"; // Redirect to the dashboard
      return false; // Prevent actual form submission and page reload
    }
  </script>
</head>
<body>
  <div class="register-container">
    <div class="register-box">
      <div class="register-header">
      <img src="mohit.png" alt="Logo" class="logo">
      <h2>Register as a Student</h2>
      </div>
      <form id="register-form" onsubmit="return handleRegistration();">
      <div class="form-group">
  <label for="username">Username :</label>
          <input type="text" id="username" name="username" required>
      </div>
      <div class="form-group">
          <label for="email">Email Address :</label>
          <input type="email" id="email" name="email" required>
      </div>
      <div class="form-group">
```

```
            <label for="password">Password :</label>
            <input type="password" id="password" name="password" required>
          </div>
          <div class="form-group">
            <label for="confirm-password">Confirm Password :</label>
            <input type="password" id="confirm-password" name="confirm-password" required>
            <div id="confirm-error" class="error" style="color: red;"></div> <!-- Error message for password mismatch -->
          </div>
          <button type="submit" class="register-btn">Register</button>
        </form>

        <p class="login-link">Already Registered? <a href="loginstudent.html">Login Now</a></p>
      </div>
    </div>
</body>
</html>
```

VALIDATORS



Register as a Student

Username : Ayush69

Email Address : ayush5148@gmail.com

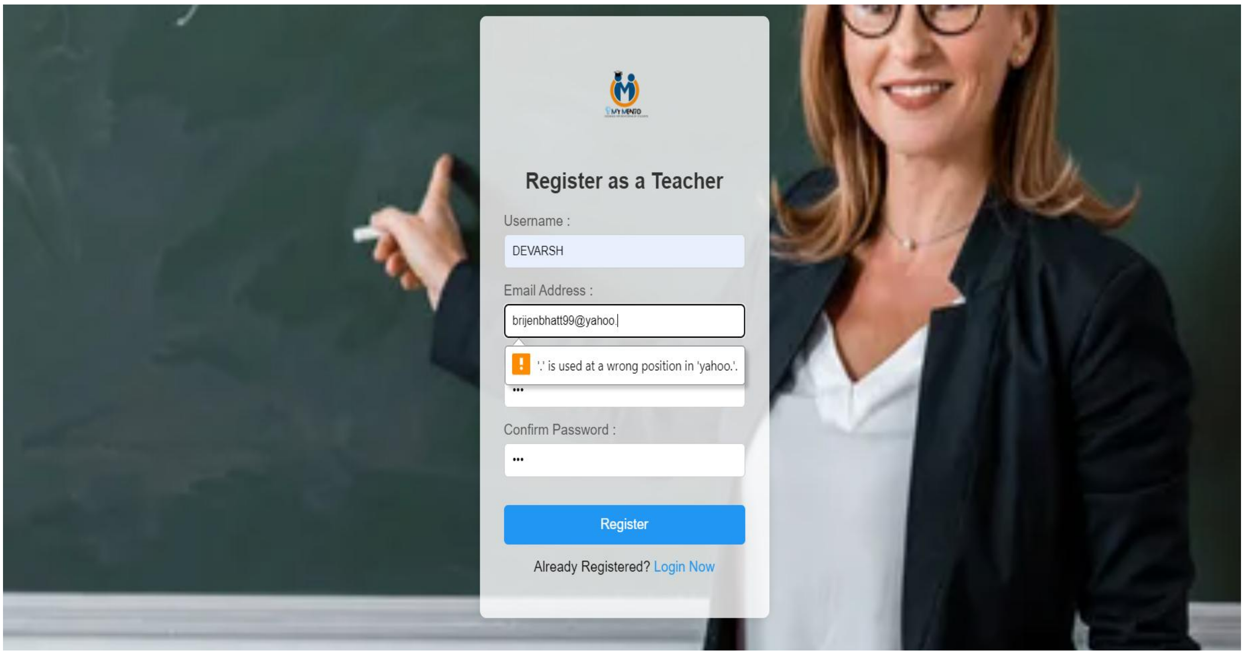
Password : •••

Confirm Password : ••••

Passwords do not match!

Already Registered? Login Now

- The password confirmation validator in MyMento ensures that users enter the same password in both the “Password” and “Confirm Password” fields during registration. If the passwords don’t match, the system immediately displays a warning message, such as “Passwords do not match”, prompting the user to correct the mistake. This validation improves the user experience by preventing errors and enhancing security, ensuring that users create accounts with the intended password, thereby avoiding future login issues.



Register as a Teacher

Username : DEVARSH

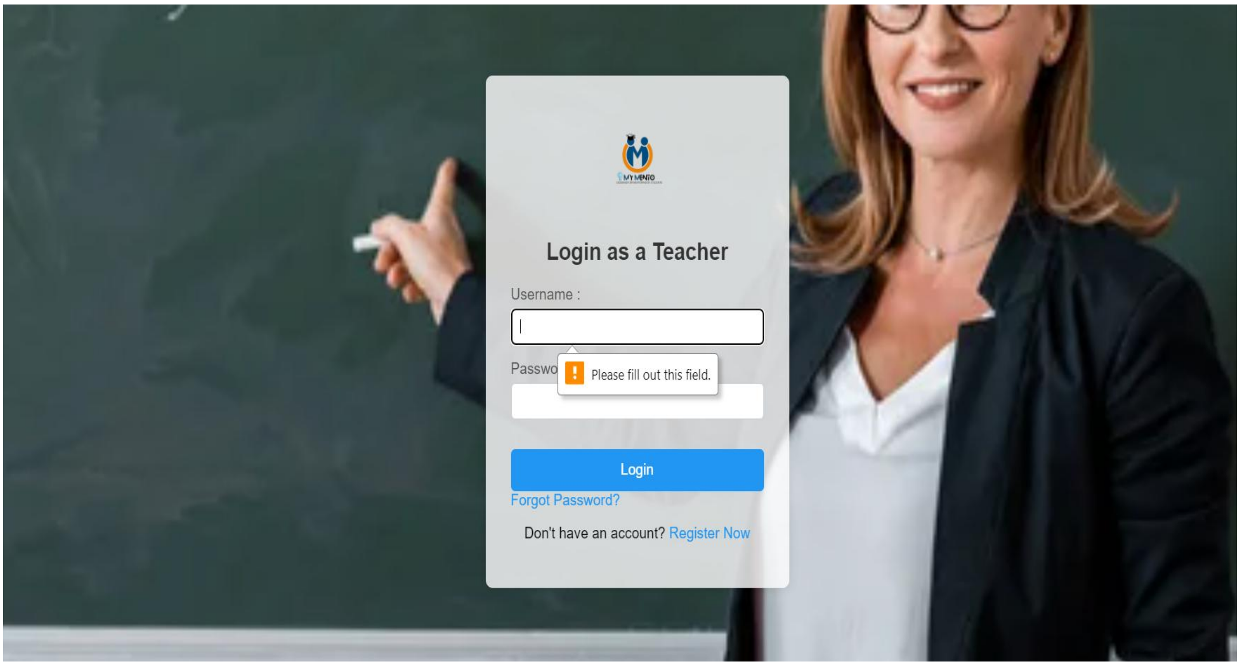
Email Address : brijenbhatt99@yahoo.

'.' is used at a wrong position in 'yahoo.'.

•••

Confirm Password : •••

- In MyMento, when users enter an email address in an incorrect format, such as placing “.com” in the wrong position or using an invalid structure, the system employs a regular expression validator to catch these mistakes. This validator checks the entered email against a specific pattern (e.g., name@domain.com) to ensure it follows standard email formatting rules. If the input doesn’t match the expected pattern, the system displays an error message, prompting the user to correct the email format before proceeding. This helps prevent invalid email addresses from being registered, ensuring proper communication with users.



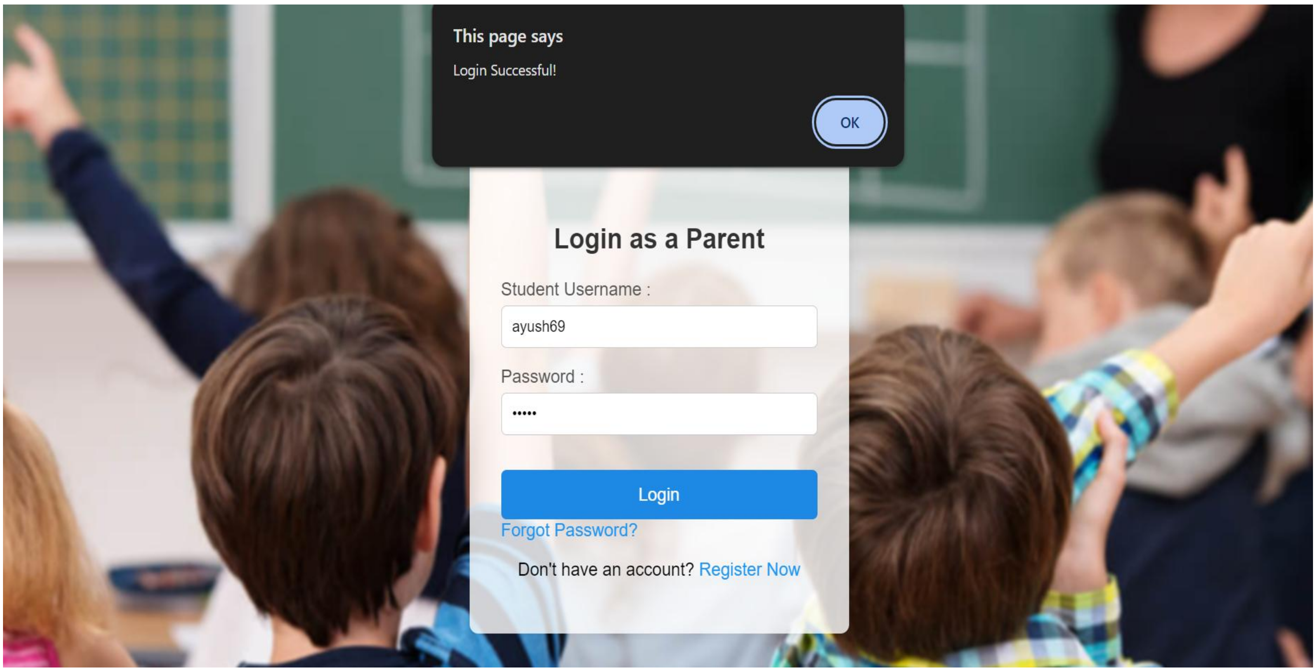
Login as a Teacher

Username :

Passwo

Please fill out this field.

- In MyMento, when a user tries to submit the form without filling in a required field, the system uses an “Required Field Validator” to ensure no mandatory fields are left blank. If a required field is empty, the validator triggers an error message, such as “Please fill out this field”, prompting the user to provide the necessary information. This validator ensures that all essential data is entered before form submission, maintaining data completeness and avoiding incomplete registrations.



This page says

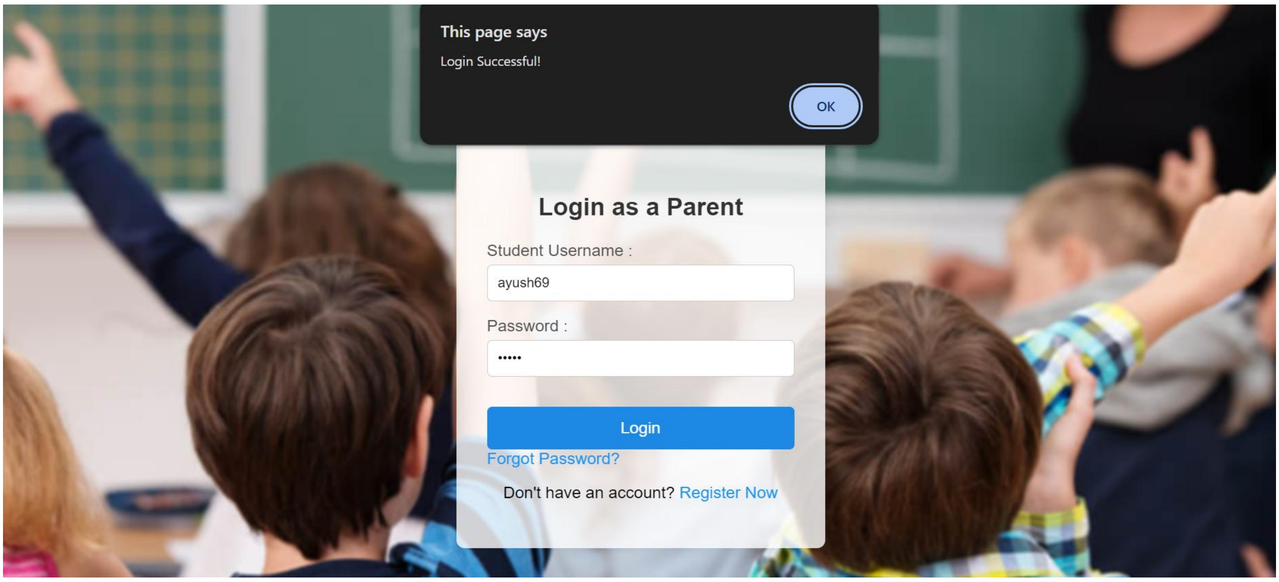
Login Successful!

Login as a Parent

Student Username : ayush69

Password : •••••

- In MyMento, after successful registration, a pop-up message appears saying “This page says: Registration Successful” to inform the user that the process is complete. This feedback is typically handled using a JavaScript Validator or Success Message Validator. Once all the input fields pass their respective validations (e.g., username, email format, password match), the system triggers this pop-up. This validator ensures that only when all the data is correctly entered and validated, the user receives a success message, improving the user experience by confirming their registration was completed without issues.



This page says

Login Successful!

Login as a Parent

Student Username : ayush69

Password : •••••

Don't have an account? Register Now

- In MyMento, after successfully logging in, a pop-up message appears saying “Login Successful” to confirm that the user has been authenticated. This is typically managed by a JavaScript Validator or Login Success Validator. Once the login credentials (username and password) are validated against the database and confirmed correct, the system triggers this pop-up to notify the user of the successful login. This ensures that the user receives immediate feedback, confirming their access to the system, and improving overall user experience by clearly indicating that the login process was completed smoothly.