Following are the main pages

1. Login page
2. Admin dashboard page
    a. Add student page
    b. Add teacher page
    c. Manage student page
    d. Manage teacher page
3. Student dashboard page
    a. Question display
    b. Report
4. Teacher dashboard page
    a. Assign topics/sub-topics
    b. Report page

Flow of pages represented in a table format:

Role	Initial Page	Navigates To	Action/Functionality
Admin	Login Page → Admin Dashboard	Add Student Page	Admin adds a new student with details (name, email, grade, etc.)
		Add Teacher Page	Admin adds a new teacher with details (name, email, subject, etc.)
		Manage Students	Admin can edit student email or remove a student
		Manage Teachers	Admin can edit teacher email, subject, or remove a teacher
Teacher	Login Page → Teacher Dashboard	Assign Topics	Teacher assigns topics to a specific student or an entire class
		Reports Page	Teacher views student-wise or class-wise performance
	Reports Page → View Student Report	Detailed performance report for an individual student	
Student	Login Page → Student Dashboard	Adaptive Question Page	Student sees assigned questions and solves them adaptively
		Reports Page	Student can view total questions attempted, correct answers, and progress

User Login Page Requirements

Overview

The application should have a simple and intuitive login page that allows three types of users to sign in: **Admin, Student, and Teacher**. Each user type should have access to different functionalities after logging in.

Page Design

The login page should contain:

- A clean and minimalistic interface.
- A form with:
  - **Username** (Text Input)
  - **Password** (Password Input)
  - **User Type Selection** (Dropdown with options: Admin, Student, Teacher)
  - **Login Button**
- A **Forgot Password?** link for password recovery.
- Proper validation for empty fields and incorrect login attempts.

Functionality

1. **User Authentication**
   - Validate the credentials against the database.
   - Display appropriate error messages for invalid login attempts.
   - Redirect the user to their respective dashboard upon successful login.
2. **User Role-Based Navigation**
   - **Admin**: Redirects to the Admin Dashboard.
   - **Student**: Redirects to the Student Dashboard.
   - **Teacher**: Redirects to the Teacher Dashboard.
3. **Forgot Password Feature**
   - Clicking on Forgot Password? Should open a form to enter the registered email address.
   - An email with a password reset link should be sent to the user.
   - The user should be able to create a new password using the reset link.
   - Implement password security policies (e.g., minimum length, special characters, etc.).
   - Show appropriate success or error messages based on the entered email.
4. **Security Measures**
   - Passwords should be securely hashed.

    - Prevent multiple failed login attempts (account lockout after a certain number of failures).

Admin Dashboard Requirements

Overview

The **Admin Dashboard** is a restricted-access page where only the **Admin** user can manage **Teachers** and **Students**. The Admin will have exclusive privileges to **add, remove, and reset passwords** for users. The credentials for the Admin account will be hardcoded in the application.

Admin Authentication

1. **Hardcoded Credentials**
    - **Email**: `admin@mad.com`
    - **Password**: `admin123`
2. The Admin must log in using these credentials to access the dashboard.

Dashboard Layout & Features

- **Left Pane (Navigation Menu)**
    - A dropdown menu that includes:
        - **Add Teacher**
        - **Add Student**
        - **Manage Teachers**
        - **Manage Students**
    - A **Logout** button placed below the dropdown.
- **Main Panel (Content Area)**
    - **Add Teacher**
        - Add Name, Email, Subject, Set password.

    - **Manage Teachers**
        - Search text box to search teacher name
        - Display teacher list as table
        - Reset passwords
        - Delete teacher
    - **Add Student**
        - Add Name, Email, Grade, Age, Set password.

- **Manage Students**
    - Search text box to search student name
    - Display student list as table
    - Reset passwords
    - Delete student

Add Teacher Page Requirements

Overview

The **Add Teacher Page** is accessible only to the **Admin**. When the Admin selects **"Add New Teacher"** from the dashboard menu, the **Add Teacher Form** should be displayed in the right pane. This form allows the Admin to enter details for a new teacher and save them to the database.

Page Layout & Functionality

1. **Navigation**
    - When the Admin clicks **"Add New Teacher"**, the **Add Teacher Page** should be displayed in the right pane.
2. **Form Fields**
    - **Teacher Name** (Text Input) – Required
    - **Email** (Text Input) – Required, must be unique
    - **Subject** (Dropdown or Text Input) – Required
    - **Password** (Password Input) – Required
3. **Data Storage**
    - On clicking **"Add Teacher"**, the entered details must be stored in the **"User"** collection in the database.
    - The **LoginID** for the teacher must be set as the **email**.
    - The **User Type** must be set to `"teacher"` by default.
    - The **Creation Timestamp** must be stored as the exact time when the record is created.
4. **Validations**
    - Ensure the email is unique (check if it already exists in the database before adding).
    - Validate email format.
    - Ensure the password meets security requirements (e.g., minimum 8 characters).
    - Display error messages for missing/invalid inputs.
5. **Success & Error Handling**
    - Show a success message upon successful addition.
    - If an error occurs (e.g., duplicate email), display an appropriate error message.

Add Student Page Requirements

Overview

The **Add Student Page** is accessible only to the **Admin**. When the Admin selects **"Add New Student"** from the dashboard menu, the **Add Student Form** should be displayed in the right pane. This form allows the Admin to enter student details and save them to the database.

Page Layout & Functionality

1. **Navigation**
   - When the Admin clicks **"Add New Student"**, the **Add Student Page** should be displayed in the right pane.
2. **Form Fields**
   - **Student Name** (Text Input) – Required
   - **Email** (Text Input) – Required, must be unique
   - **Password** (Password Input) – Required
   - **Grade** (Dropdown or Text Input) – Required
   - **Gender** (Dropdown: Male/Female/Other) – Required
   - **School Name** (Text Input) – Required
3. **Data Storage (Upon Clicking "Add Student")**
   1. The student details must be stored in **User Collection** (Stores login and role-based information)
4. **Validation Rules**
   - Ensure email is unique before adding a new student.
   - Display error messages for missing or invalid inputs.
5. **Success & Error Handling**
   - Show a success message upon successful addition.
   - Display an error message if a duplicate email is found or validation fails.

Manage Teacher Page Requirements

Overview

The **Manage Teacher Page** allows the **Admin** to view, update, and remove teachers through an intuitive and interactive interface. This page displays a table listing all teachers, where each field is **editable in place**, ensuring seamless modifications and updates to the database.

Page Layout & Functionality

1. **Navigation**
   - When the Admin selects **"Manage Teachers"**, the **Manage Teacher Page** should be displayed in the right pane.
2. **Displayed Data (Table Format)**
   - The page should show a table listing all teachers with the following columns:
     - **Teacher Name**
     - **Email**
     - **Subject**
     - **Class**
     - **Actions** (Edit, Save, Remove)
3. **Inline Editing Feature**
   - Each field (except **Actions**) will be **editable**.
   - Clicking the **edit option (pencil icon or direct inline edit)** allows the Admin to modify the value.
   - Once editing is complete:
     - A **Save** button updates the data in the UI and MongoDB.
     - A **Cancel** button reverts the field to its previous value.
4. **Updating MongoDB on Save**
   - Only **modified fields** should be updated in the database.
   - A **success message** should be displayed upon update.
   - Error handling should prevent updates with invalid values (e.g., incorrect email format).
5. **Remove Teacher**
   - The last column contains a **Remove (Trash Icon)** button.
   - Clicking **Remove** triggers a **confirmation dialog** before deletion.
   - Once confirmed, the teacher is removed from both the table and MongoDB.
6. **Validation & Error Handling**
   - Ensure **valid email format** when updating email.
   - Ensure **subject field is not left empty**.
   - Display a **confirmation popup** before deleting a teacher.
   - Show **success messages** upon successful updates/removals.

Security Considerations

- Ensure **only Admin users** can access this page.
- Use **server-side validation** to prevent unauthorized updates or deletions.
- Prevent accidental deletion with a **confirmation prompt**.
- Implement **role-based access control** to restrict teacher modifications.

Manage Student Page Requirements

Overview

The **Manage Student Page** allows the **Admin** to view, update, and remove student records through an interactive and user-friendly interface. This page displays a **table format** where the Admin can **edit student email inline** and **remove students from the database**.

Page Layout & Functionality

1. **Navigation**
   - When the Admin selects **"Manage Students"**, the **Manage Student Page** should be displayed in the right pane.
2. **Displayed Data (Table Format)**
   - The page should show a table listing all students with the following columns:
     - **Student Name**
     - **Email**
     - **Grade**
     - **School Name**
     - **Actions** (Edit, Save, Remove)
3. **Inline Editing Feature**
   - The **Email** field will be **editable** for updates.
   - Clicking the **edit option (pencil icon or direct inline edit)** allows the Admin to modify the email address.
   - Once editing is complete:
     - A **Save** button updates the email in the UI and MongoDB.
     - A **Cancel** button reverts the field to its previous value.
4. **Updating MongoDB on Save**
   - Only **modified email addresses** should be updated in the **User collection**.
   - A **success message** should be displayed upon successful update.
   - Error handling should prevent updates with invalid values (e.g., incorrect email format).
5. **Remove Student**

    - The last column contains a **Remove (Trash Icon)** button.
    - Clicking **Remove** triggers a **confirmation dialog** before deletion.
6. Once confirmed, the student is removed from both the **User collection**
7. **Password Management**
    - There is **no "Set Student Password"** option on this page.
    - Student passwords are set only during the **Add Student process**.
8. **Validation & Error Handling**
    - Ensure **valid email format** when updating.
    - Display a **confirmation popup** before deleting a student.
    - Show **success messages** upon successful updates/removals.

Security Considerations

- Ensure **only Admin users** can access this page.
- Use **server-side validation** to prevent unauthorized updates or deletions.
- Prevent accidental deletion with a **confirmation prompt**.
- Implement **role-based access control** to restrict student modifications.

Teacher Dashboard Page Requirements

Overview

The **Teacher Dashboard** allows teachers to manage topic assignments efficiently. Teachers can **assign topics/sub-topics** in two ways:

1. **Student-wise** – Assign a topic to an individual student.
2. **Class-wise** – Assign a topic to an entire class.

This ensures flexibility in how topics are assigned based on student needs.

Page Layout & Functionality

1. **Navigation**
    - When a teacher logs in, they are directed to the **Teacher Dashboard**.
    - A section for **Assign Topics** should be prominently displayed.
2. **Assignment Methods**
    - Teachers can assign topics using either:
        - **Student-wise Assignment**
        - **Class-wise Assignment**
3. **Student-wise Assignment**
    - A dropdown or search box allows the teacher to select a **student** from their assigned class.

    - A **topic selection dropdown** allows choosing the subject, topic, and sub-topic.
    - A **"Confirm Assignment"** button saves the assignment to the database.
4. **Class-wise Assignment**
    - A dropdown allows the teacher to select a **class**.
    - A **topic selection dropdown** allows choosing the subject, topic, and sub-topic.
    - A **"Confirm Assignment"** button saves the assignment for all students in the selected class.

- Assignments should be retrieved and displayed on the **Teacher Dashboard**.

5. **Validation & Error Handling**
    - **Ensure a student or class is selected before confirming an assignment.**
    - **Ensure a topic/sub-topic is selected before saving.**
    - **Display a success message upon assignment.**
6. **Security Considerations**
    - **Only authenticated teachers should access this page.**
    - **Teachers can only assign topics to students or classes assigned to them.**
    - **Ensure data integrity by preventing duplicate assignments.**

Student Dashboard Page Requirements

Overview

The **Student Dashboard** provides a structured view of assigned and completed topics while allowing students to answer questions adaptively. The interface is divided into two sections:

- **Left Pane**: Displays **Assigned** and **Completed** topics/sub-topics.
- **Right Pane**: Displays **one adaptive question at a time**, along with **answer options, solution, and correctness feedback**.

Page Layout & Functionality

1. **Navigation**
    - When a student logs in, they are directed to the **Student Dashboard**.
    - The left pane provides **an overview of their learning progress**.
2. **Left Pane (Progress Tracking)**
    - **Assigned Topics/Sub-topics**:
        - Lists topics/sub-topics assigned by the teacher.
        - Clicking a topic loads the corresponding question in the **right pane**.
    - **Completed Topics/Sub-topics**:
        - Displays topics the student has successfully completed.
3. **Right Pane (Adaptive Question Page)**

    - **Displays only one question at a time** based on the **assigned topic/sub-topic**.
    - **Question Structure**:
        - **Question Statement** (retrieved from the database).
        - **Answer Options** (if multiple-choice).
        - **Solution** (revealed after answering or upon request).
        - **Correct/Incorrect Feedback** after submission.
4. **Adaptive Learning Logic**
    - The **next question** is determined based on the student's response:
        - **Correct Answer** → Moves to a more challenging question.
        - **Incorrect Answer** → Displays a question of the same or lower difficulty.
5. **Database Updates**
    - The **StudentProgress Collection** should store learning data with fields
    - Updates should be made in **real-time** as students answer questions.
6. **Validation & Error Handling**
    - Ensure students can only access **topics assigned to them**.
    - Ensure **only one question is displayed at a time**.
    - Prevent skipping questions unless answered.

Security Considerations

- Only **authenticated students** should access this page.
- Students should only see **questions assigned to them**.
- Prevent **manipulation of adaptive learning logic** by ensuring backend validation.

Reports Page Requirements

Overview

The **Reports Page** provides insights into student performance. It has different views based on the user role:

- **Student Report Page**: Displays performance metrics after completing a sub-topic.
- **Teacher Report Page**: Allows viewing **Student-wise** and **Class-wise** reports.

Page Layout & Functionality

1. Student Report Page

- **Access**: Students can view their own report **only after completing every sub-topic**.
- **Metrics Displayed**:
  - **Total Number of Questions Displayed**
  - **Total Number of Questions Correct**
- **Report Visibility**: The report should update dynamically after each sub-topic is completed.

Database Updates (Student Progress Tracking)

---

2. Teacher Report Page

- **Access**: Teachers can view reports **student-wise and class-wise**.
- **Reports Table Structure (Class-Wise View)**:
  - **Student Name**
  - **Class**
  - **Subject**
  - **View Report** (Button to see student-specific details)