Sure! Let's go through your **manage\_studentsadmin.php** file in detail. This file is responsible for **managing student records** in the admin panel of your IPHS Campus Management System. It supports **listing, adding, and deleting students**, and uses **PHP with MySQLi**, **Bootstrap**, and a modal form.

**🔐 1. Session and Role Check**

session\_start();

include 'auth.php';

checkRole('admin');

include 'db\_connect.php';

* session\_start() begins the session so you can access session variables (e.g., $\_SESSION['role']).
* auth.php likely contains checkRole() to ensure only an **admin** can access this page.
* db\_connect.php establishes the MySQL connection using $conn.

**✅ 2. Add Student Functionality**

if ($\_SERVER["REQUEST\_METHOD"] == "POST" && isset($\_POST['add\_student'])) {

This block runs when the **"Add Student"** form is submitted.

**🔸 a. Input Fields**

$fullname = trim($\_POST['fullname']);

$email = trim($\_POST['email']);

$password = password\_hash($\_POST['password'], PASSWORD\_BCRYPT);

$class = trim($\_POST['class']);

$guardian\_name = trim($\_POST['guardian\_name']);

* Retrieves and sanitizes inputs from the form.
* Password is **hashed securely** with PASSWORD\_BCRYPT.

**🔸 b. Check for Email Uniqueness**

$check\_email = $conn->prepare("SELECT id FROM users WHERE email = ?");

* Ensures no duplicate user emails.

**🔸 c. Insert into users and students tables**

INSERT INTO users (fullname, email, password, role)

INSERT INTO students (user\_id, class, guardian\_name)

* Adds to the users table first (assigns role = student).
* Then adds the linked data to the students table using the returned user\_id.

**🔍 3. Fetch All Students**

$students\_query = "... JOIN users ON students.user\_id = users.id"

* Combines student data with user details using an SQL JOIN.

**❌ 4. Delete Student**

if (isset($\_GET['delete'])) {

* Gets the student ID from the URL (?delete=ID)
* Finds the corresponding user\_id from the students table
* Deletes both from students and then from users (to avoid foreign key constraints).

**📄 5. HTML Structure**

**✅ Head Section**

Includes Bootstrap, Font Awesome, Chart.js, and your admin\_dashboard.css:

<link rel="stylesheet" href="...bootstrap.min.css">

<link rel="stylesheet" href="...font-awesome.min.css">

**📚 6. Sidebar**

<div class="sidebar" id="sidebarMenu">

<img src="../images/logo.jpg">

<a href="...">Dashboard</a>

...

</div>

* Fixed sidebar for admin navigation.
* Contains links to key management pages (students, courses, exams, fees, etc.).

**👥 7. Main Content**

<h2>Manage Students</h2>

<button data-bs-toggle="modal">➕ Add Student</button>

* Page header with "Add Student" button that triggers a modal.

**📊 8. Students Table**

<table>

<thead> ... </thead>

<tbody>

<?php while ($student = $students\_result->fetch\_assoc()): ?>

<tr>

<td><?= $student['id'] ?></td>

...

</tr>

<?php endwhile; ?>

</tbody>

</table>

* Displays student data in a table using a PHP while loop.
* Uses htmlspecialchars() for security to prevent XSS.

**➕ 9. Add Student Modal**

<form method="POST">

<input type="text" name="fullname">

...

</form>

* Modal popup with form fields for full name, email, password, class, and guardian name.
* On submit, it triggers the PHP add student block above.

**⚙️ 10. Scripts**

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

* Loads Bootstrap JS to enable the modal and other components.

**✅ Summary**

| **Feature** | **Description** |
| --- | --- |
| ✅ Secure Role Check | Admin-only access using checkRole() |
| ✅ Add Student | Uses a modal form, checks for duplicate emails, and hashes passwords |
| ✅ Delete Student | Deletes from both students and users safely |
| ✅ List Students | Uses SQL JOIN to show full student info |
| ✅ Responsive UI | Built with Bootstrap 5 |
| ✅ Security | Passwords are hashed, and inputs are escaped |

Would you like me to **refactor this into components**, **convert it into a PWA**, or **optimize it for accessibility/SEO** next?