

[Project Report]

Project Title:
EduQuery - Educational Query & Resource Management System

Submitted by: Utsho Kumar Dey
Student ID: 11230321321
Email Address: utshokumerdey@gmail.com

Course Title: Software Development I
Course Code: CSE 2216
Section: 4D

Submitted to:
Shovon Mandal [SM]
Lecturer,
Northern University of Business and Technology Khulna

Date of Submission: 21.08.2025

Contents

2. Abstract:	1
3. Introduction:	1
4. Literature Review:	1
6. Implementation:	5
7. Results and Analysis:	6
8. Discussion:	11
9. Conclusion:	11
10. References:	12

Table of Figure:

Figure 1: Initial Entry & Authentication	2
Figure 2: Role-Based Dashboards	3
Figure 3: User-Specific Actions	4
Figure 4: Create database connection	5
Figure 5: Core logic of the application	5
Figure 6: Logging <i>ensures</i>	6
Figure 7: Registration Page	6
Figure 8: Login Page	7
Figure 9: Teacher Dashboard	7
Figure 10: Manage Assignments (Teacher)	7
Figure 11: Manage Results (Teacher)	8
Figure 12: Student Dashboard	8
Figure 13: View Assignments (Student)	8
Figure 14: View Results (Student)	9
Figure 15: Admin Dashboard	9
Figure 16: Manage Users (Admin)	9
Figure 17: Edit User (Admin)	10
Figure 18: Manage Notices (Admin)	10
Figure 19: Database View (phpMyAdmin)	11

Student Management System

2. Abstract:

EduQuery is a web-based Educational Query & Resource Management System designed to streamline communication and resource sharing between students, teachers, and administrators in academic institutions. The system enables role-based access, allowing students to view notices, submit assignments, and check results; teachers to post notices, upload results, and manage assignments; and administrators to oversee the entire system. Developed using PHP, MySQL, HTML, CSS, JavaScript, and Bootstrap, EduQuery offers a responsive design, secure authentication, and a scalable architecture. The project enhances accessibility, reduces paperwork, and fosters an organized academic environment.

3. Introduction:

Background:

In many educational institutions, managing academic activities like assignment submissions, result publication, and notice distribution is often handled manually or through disjointed systems. This leads to inefficiencies, delays, and a lack of transparency. EduQuery aims to provide a centralized platform to manage these processes efficiently.

Problem Statement:

Existing solutions are either too complex for small to mid-sized institutions or lack role-specific functionalities. There is a need for a simple, accessible, and responsive platform that ensures secure role-based access while maintaining ease of use.

Objectives:

- Develop a user-friendly web-based academic management platform.
- Implement secure role-based authentication for Students, Teachers, and Admins.
- Enable students to view notices, submit assignments, and check results.
- Allow teachers to manage assignments, upload results, and post notices.
- Provide admins with full control over user management and monitoring.
- Store all academic data in a secure MySQL database.

Scope:

The system focuses on academic communication and management within an institution. It does not cover financial transactions, detailed analytics, or AI-based learning assistance in its current version. Future expansions may integrate these features.

4. Literature Review:

Several educational management systems exist, such as Moodle and Google Classroom, which offer comprehensive functionalities. However, these platforms often require significant technical expertise and resources to manage, making them unsuitable for smaller institutions. EduQuery is designed to be lightweight yet powerful, bridging the gap between complexity and usability.

References:

- Kumar, A. (2025). *Student Management System using PHP and MySQL*. PHPGurukul.
- Bootstrap Documentation. <https://getbootstrap.com/docs>
- GeeksforGeeks – PHP & MySQL Tutorials. <https://www.geeksforgeeks.org>

5. Methodology:

Approach:

EduQuery follows a modular design approach, ensuring each role (Admin, Teacher, Student) has its own dedicated dashboard and functionalities.

Algorithms:

- **Role-Based Access Control (RBAC)** for authentication.
- **CRUD Operations** for managing assignments, notices, and results.
- **Password Hashing** for secure user authentication.

Tools and Technologies:

- **Frontend:** HTML5, CSS3, JavaScript, Bootstrap
- **Backend:** PHP
- **Database:** MySQL (via XAMPP)

Experimental Design:

The system was tested locally using XAMPP. Functional testing was carried out for all modules.

1.Initial Entry & Authentication

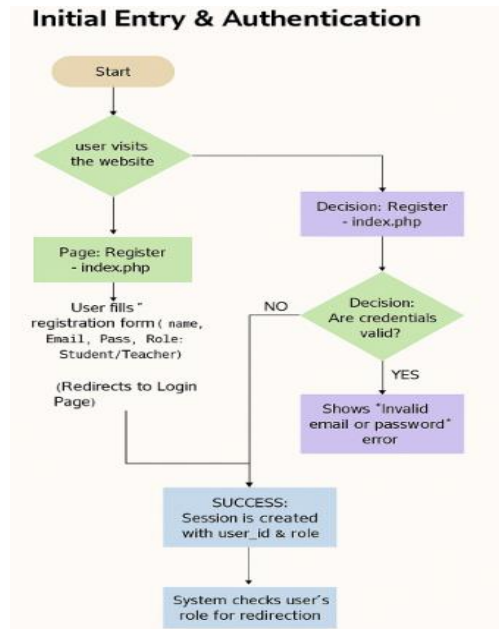


Figure 1: Initial Entry & Authentication

Description: This flowchart outlines the process a user follows when first accessing the website. The flow begins with the user visiting the site. The system then checks whether the user is already registered. If not, the user is redirected to the registration page (index.php), where they must provide necessary details including name, email, password, and role (student or teacher). Upon successful registration, the user is redirected to the login page.

If the user is already registered, they are taken directly to the login page (login.php). After entering their credentials, the system validates them against the database. If the credentials are incorrect, an error message is displayed ("Invalid email or password") and the user remains on the login page.

If the credentials are correct, a session is created with the user's ID and role information. Finally, the system checks the user's role and redirects them to the appropriate dashboard based on their role (admin, teacher, or student).

2. Role-Based Dashboards

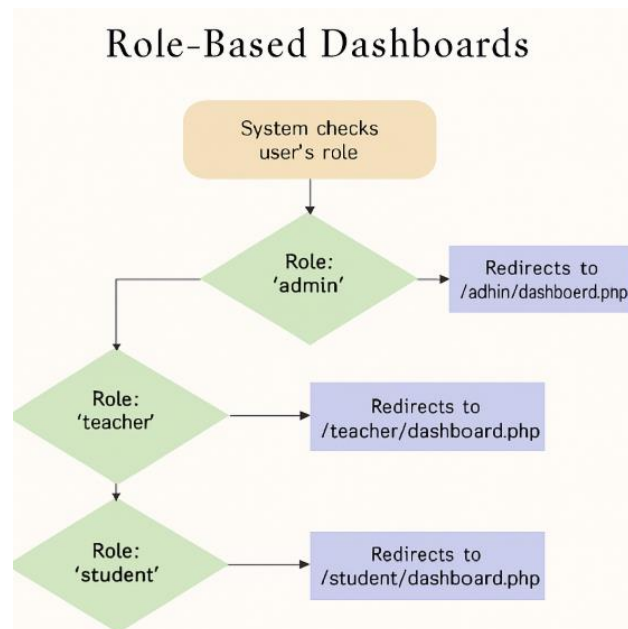


Figure 2: Role-Based Dashboards

Description: This flowchart describes the redirection process based on a user's role after a successful login and session creation. Once the system identifies the logged-in user's role, it performs a conditional check to determine the appropriate dashboard.

- If the role is **admin**, the system redirects the user to /admin/dashboard.php.
- If the role is **teacher**, the redirection goes to /teacher/dashboard.php.
- If the role is **student**, the user is taken to /student/dashboard.php.

This role-based branching ensures that each type of user is provided with a personalized interface and tools relevant to their responsibilities within the system.

3. User-Specific Actions

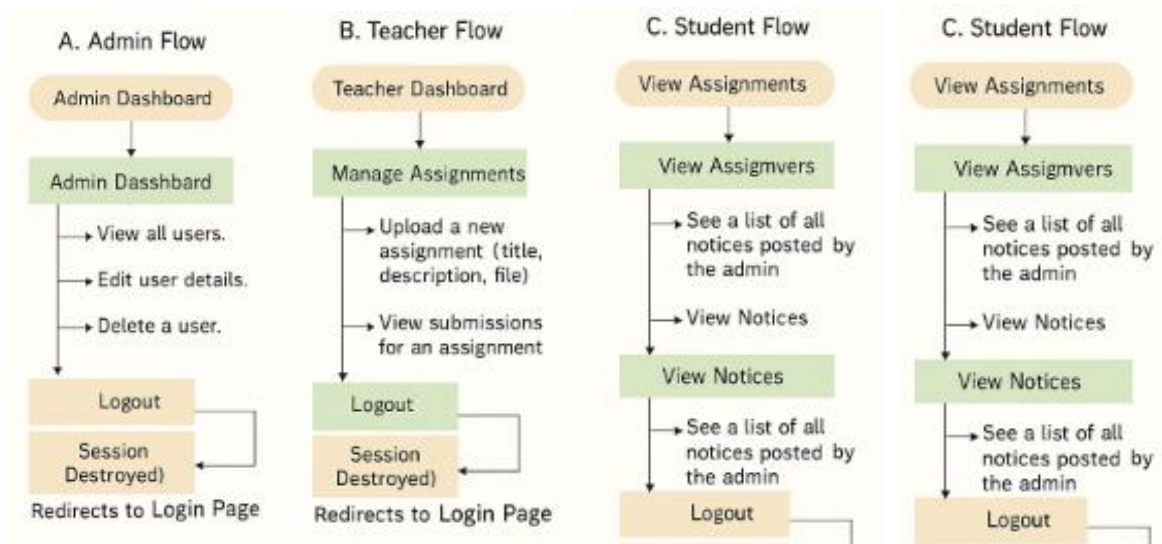


Figure 3: User-Specific Actions

Description: This comprehensive flowchart breaks down the available features and actions for each user role: **Admin**, **Teacher**, and **Student**.

A. Admin Flow

The admin dashboard allows administrative control over the system. Admins can:

- Manage Users: view all users, edit user details, and delete users.
- Manage Notices: post new notices, view existing notices, and delete them.
- Logout: Ends the session and redirects the user to the login page.

B. Teacher Flow

The teacher dashboard is designed to handle academic content and results. Teachers can:

- Manage Assignments: upload new assignments (title, description, file) and view submissions.
- Manage Results: upload results for individual students by entering student ID, subject, and marks.
- View Notices: see all notices posted by the admin.
- Logout: Ends the session and redirects to the login page.

C. Student Flow

The student dashboard focuses on learning materials and academic performance. Students can:

- View Assignments: access all published assignments, download files, and submit their work.
- View Results: check their personal academic results.
- View Notices: read all public notices posted by the admin.
- Logout: Ends the session and redirects to the login page.

This user-specific segmentation ensures that each role has tailored functionality, making the system organized, secure, and easy to use.

6. Implementation:

System Architecture:

The system uses a three-tier architecture:

1. **Presentation Layer** (HTML, CSS, JS, Bootstrap)
2. **Application Layer** (PHP)
3. **Data Layer** (MySQL)

Components:

- **Login & Authentication Module**
- **Admin Dashboard**
- **Teacher Dashboard**
- **Student Dashboard**
- **Database Management**

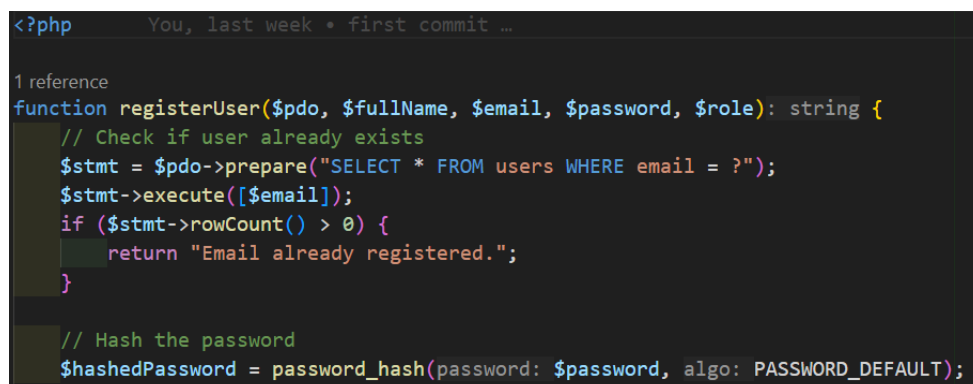
Code Excerpts:



```
includes > db.php > ...
You, last week | 1 author (You)
1  <?php
2  $host = 'localhost';
3  $db = 'eduquery';
4  $user = 'root';
5  $pass = '';
6
7  try {
8      $pdo = new PDO(dsn: "mysql:host=$host;dbname=$db", username: $user, password: $pass);
9      $pdo->setAttribute(attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
10 } catch (PDOException $e) {
11     die("Could not connect to the database $db :". $e->getMessage());
12 }
```

Figure 4: Create database connection

Description: This file is critically important. It handles the database connection, which is essential for the entire application to function. Without it, no data can be stored or retrieved



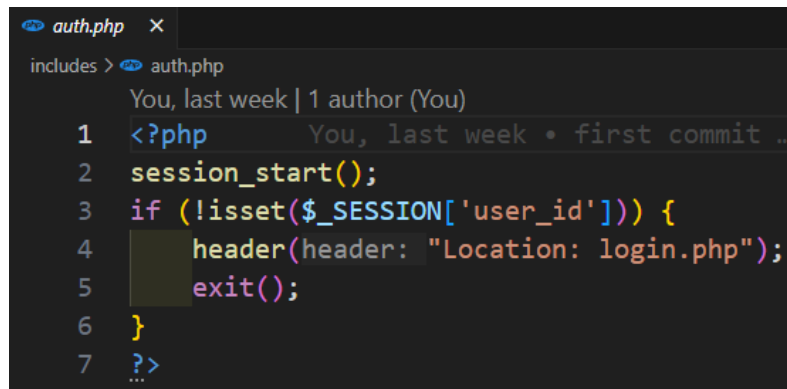
```
<?php You, last week * first commit ...
1 reference
function registerUser($pdo, $fullName, $email, $password, $role): string {
    // Check if user already exists
    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
    $stmt->execute([$email]);
    if ($stmt->rowCount() > 0) {
        return "Email already registered.";
    }

    // Hash the password
    $hashedPassword = password_hash(password: $password, algo: PASSWORD_DEFAULT);
}
```

Figure 5: Core logic of the application

Description: It contains the core business logic of the application, including user registration, login, and functions for all user roles (admin, teacher, student). It's the heart of the application's functionality. One of the key security features of this project is password hashing. Instead of storing plain text passwords, the system uses the PHP `password_hash()` function to securely

hash user passwords before saving them in the database. This ensures that even if the database is compromised, the actual passwords remain protected and cannot be directly retrieved.



```
auth.php x
includes > auth.php
You, last week | 1 author (You)
1 <?php You, last week • first commit ...
2 session_start();
3 if (!isset($_SESSION['user_id'])) {
4     header(header: "Location: login.php");
5     exit();
6 }
7 ...
```

Figure 6: Logging ensures

Description: This file is important for security. It ensures that only logged-in users can access protected pages.

7. Results and Analysis:

Results:

- Successfully implemented role-based dashboards.
- All CRUD operations function correctly.
- Responsive UI tested on desktop and mobile browsers.

Analysis:

The system meets its intended objectives, with smooth navigation and clear separation of roles. Future updates may integrate live chat and push notifications.

Snippet of this project-

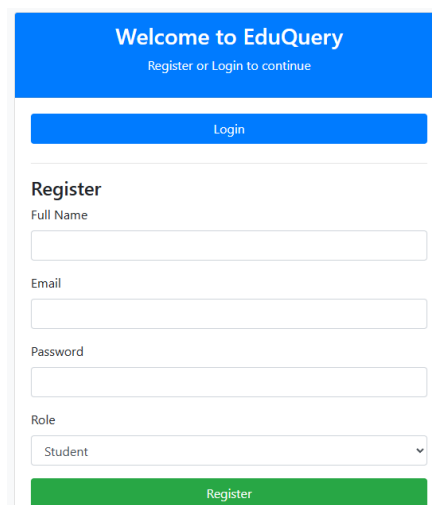
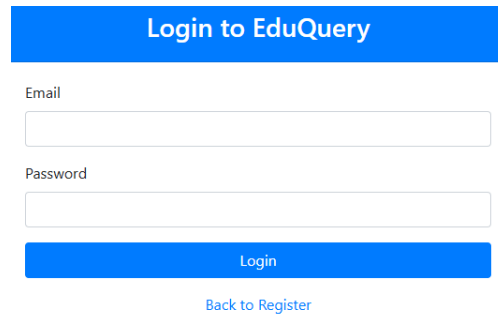


Figure 7: Registration Page

Description: This page allows new users to create an account by entering their full name, email, password, and selecting a role (Student, Teacher, or Admin).



Login to EduQuery

Email

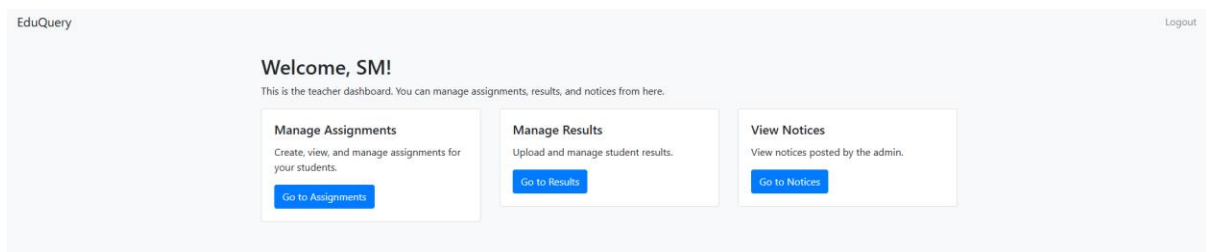
Password

Login

[Back to Register](#)

Figure 8: Login Page

Description: Users can log in to the system by providing their registered email and password. A link is available to return to the registration page.



EduQuery Logout

Welcome, SM!
This is the teacher dashboard. You can manage assignments, results, and notices from here.

Manage Assignments
Create, view, and manage assignments for your students.

[Go to Assignments](#)

Manage Results
Upload and manage student results.

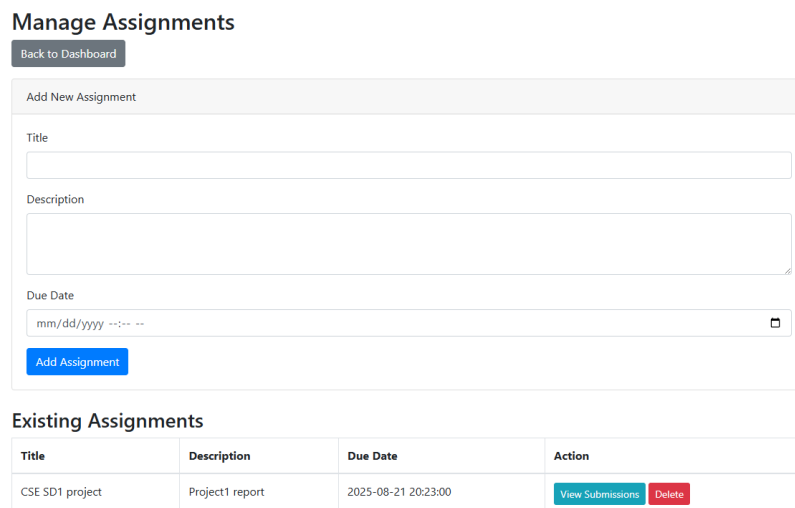
[Go to Results](#)

View Notices
View notices posted by the admin.

[Go to Notices](#)

Figure 9: Teacher Dashboard

Description: After login, teachers can access their dashboard with options to manage assignments, view submissions, and manage results.



Manage Assignments

[Back to Dashboard](#)

Add New Assignment

Title

Description

Due Date

[Add Assignment](#)

Existing Assignments

Title	Description	Due Date	Action
CSE SD1 project	Project1 report	2025-08-21 20:23:00	View Submissions Delete

Figure 10: Manage Assignments (Teacher)

Description: Teachers can create new assignments by entering a title, description, and due date. Existing assignments are listed with edit and delete options.

Manage Results

Back to Dashboard

Add New Result

Student

Utsho

Subject

Marks

Grade

Add Result

Published Results

Student Name	Subject	Marks	Grade	Action
--------------	---------	-------	-------	--------

Figure 11: Manage Results (Teacher)

Description: Teachers can publish results by selecting a student, entering subject, marks, and grade. A list of published results is displayed below.

EduQuery

Logout

Welcome, ak!

This is your student dashboard. You can view assignments, submit your work, check results, and see notices from here.

View Assignments

View and submit your assignments.

Go to Assignments

View Results

Check your academic results.

Go to Results

View Notices

View notices from the admin and your teachers.

Go to Notices

Figure 12: Student Dashboard

Description: Students see their dashboard with options to view assignments, submit work, and view results.

View Assignments

Back to Dashboard

CSE SD1 project

Posted by: SM

Project1 report

Due Date:

2025-08-21 20:23:00

Submit Your Work

Choose File

No file chosen

Submit

Figure 13: View Assignments (Student)

8

Description: Students can view assignment details, including project name, description, due date, and an option to submit the work.

View Results

[Back to Dashboard](#)

Subject	Marks	Grade	Published By	Date
---------	-------	-------	--------------	------

Figure 14: View Results (Student)

Description: Displays the results published by teachers for the logged-in student, showing subject, marks, grade, publisher, and date.

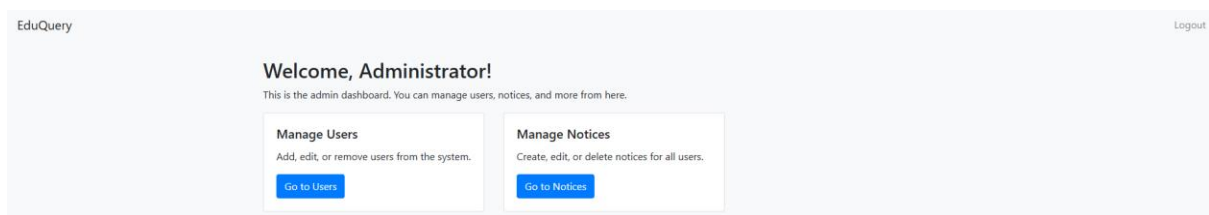


Figure 15: Admin Dashboard

Description: Admins have access to manage users, view system data, and manage notices.

Manage Users

[Back to Dashboard](#)

ID	Full Name	Email	Role	Action
1	Admin User	admin@eduquery.com	admin	Edit Delete
2	Utsho	utshokumerdey2@gmail.com	student	Edit Delete
3	Utsho1	utshokumerdey@gmail.com	student	Edit Delete
4	Utsho1	utshokumerdey1@gmail.com	student	Edit Delete
5	a	a@gmail.com	student	Edit Delete
6	ab	ab@gmail.com	student	Edit Delete
7	m	m@gmail.com	student	Edit Delete
8	y	y@gmail.com	student	Edit Delete
9	SM	SM@gmail.com	teacher	Edit Delete
10	r	r@gmail.com	student	Edit Delete
11	ak	ak@gmail.com	student	Edit Delete
12	Administrator	admin@example.com	admin	Edit Delete

Figure 16: Manage Users (Admin))

Description: Admins can view all registered users with their full name, email, and role. Options are available to edit or delete users.

Edit User

Full Name

Email

Role

Figure 17: Edit User (Admin)

Description: Allows the admin to edit an existing user's details, including full name, email, and role.

Manage Notices

Add New Notice

Title

Content

Existing Notices

Title	Content	Posted By	Date	Action
Final Exam	Coming soon	Administrator	2025-08-07 01:06:02	<input type="button" value="Delete"/>

Figure 18: Manage Notices (Admin)

Description: Admins can create and post new notices. Existing notices are displayed with options to delete them.

Showing rows 0 - 10 (11 total, Query took 0.0002 seconds)

```
SELECT * FROM `users`
```

Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	id	full_name	email	password	role	created_at
<input type="checkbox"/>	1	Admin User	admin@eduquery.com	\$2y\$10\$N.RSP3iGpe5L5WjL00/s..Cf0vtYJbS0dG3vj08i8s...	admin	2025-07-21 00:49:27
<input type="checkbox"/>	2	Utsho	utshokumerdey2@gmail.com	\$2y\$10\$0..oYPN0ZsdDTPpkw0n90uXUfEuscwMQ50PIWxD/sn1...	student	2025-07-21 00:51:49
<input type="checkbox"/>	3	Utsho1	utshokumerdey@gmail.com	\$2y\$10\$Bo9YA8zKM5/NSH21Kp3..ZxPv.KrQZ7CraZ2Ar5r8...	student	2025-07-21 00:55:53
<input type="checkbox"/>	4	Utsho1	utshokumerdey1@gmail.com	\$2y\$10\$XGzv8PloWCZQR4vdWYBdguEtrYumVPBV5TKhdzGo37...	student	2025-07-21 00:58:58
<input type="checkbox"/>	5	a	a@gmail.com	\$2y\$10\$tpxD1i561pWBijGdWnSo7OPKRvum.xSPAnDYkL7zZx3...	student	2025-07-21 01:04:41
<input type="checkbox"/>	6	ab	ab@gmail.com	\$2y\$10\$sh8nD/Fi3Beth52rbEcwODnkV1b3jT7bt0W3GHijV...	student	2025-07-21 01:12:57
<input type="checkbox"/>	7	m	m@gmail.com	\$2y\$10\$K/OwgBVieuDJA6jaG6Y8.lu/40ThjJ/zVyb2luGCH...	student	2025-07-21 01:17:53
<input type="checkbox"/>	8	y	y@gmail.com	\$2y\$10\$WfgOaGdMCiY.7i8nRCgyjOrX2w/v0irdT2E..QvcK...	student	2025-08-01 21:51:33
<input type="checkbox"/>	9	SM	SM@gmail.com	\$2y\$10\$Ukhkw6lID3radMSqRjRGGeT.pnhVtoSdQF2u8gpye...	teacher	2025-08-01 22:12:38
<input type="checkbox"/>	10	r	r@gmail.com	\$2y\$10\$bepDCXfurV06.SzS9NgCcuH0w/WFeYMGz9wMtnbP7kf...	student	2025-08-01 23:10:43
<input type="checkbox"/>	11	ak	ak@gmail.com	\$2y\$10\$ng.at51e6G2JldoP5uReCJZnKp2ujrN5Q5nAy82C...	student	2025-08-06 14:30:48

Figure 19: Database View (phpMyAdmin)

Description: Shows the `users` table in the database with fields for `id`, `full name`, `email`, `password` (hashed), `role`, and `created at` timestamp.

8. Discussion:

Interpretation:

EduQuery addresses the key problems of fragmented academic communication by providing a single platform.

Challenges:

- Initial database structuring took longer than expected.
- Managing session states for multiple roles required careful handling.

Limitations:

- Currently limited to local hosting; no cloud deployment in this version.

9. Conclusion:

Summary:

EduQuery successfully delivers a functional educational management platform tailored for academic institutions.

Contributions:

- A centralized dashboard for Students, Teachers, and Admins.
- Streamlined assignment submission and result viewing.

Future Work:

- Add notification system.
- Mobile app integration.
- Cloud hosting for remote access.

10. References:

- [1] W3Schools.com – Java HashMap Reference.
https://www.w3schools.com/java/java_ref_hashmap.asp
- [2] Bootstrap Documentation. <https://getbootstrap.com/docs>
- [3] PHP.net Manual. <https://www.php.net/manual/en/>
- [4] GeeksforGeeks – PHP & MySQL Tutorials. <https://www.geeksforgeeks.org>