PATUAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY

COURSE CODECCE 224

-

SUBMITTED TO:

Prof. Dr. Md Samsuzzaman

Department of Computer and Communication Engineering

Faculty of Computer Science and Engineering

SUBMITTED BY:

Mohammed Sakib Hasan

ID: 2102052

Registration No: 10179

Faculty of Computer Science and Engineering

Date of submission: 15 June, 2025

CSE Study Room – A Student-Centric Learning & Career Portal

Table of Contents

1. Introduction

The **CSE Study Room** is a web-based platform designed to support Computer Science & Engineering students in learning and career development. It allows users to solve coding challenges, track progress, view correct answers, access career roadmaps, apply for jobs, connect with mentors, and join study groups. Built with PHP and MySQL, the platform integrates key features in one place to make learning more interactive, practical, and career-focused.

2. Objective

This project focuses on creating a centralized academic and career development platform tailored for Computer Science & Engineering (CSE) students. The core objective is to provide an interactive environment where students can:

- Solve coding challenges and receive instant feedback,
- Track learning progress and scores across topics,
- Access curated career roadmaps for various tech roles,

- Receive mentorship and apply for relevant job opportunities.

3. Features

- **User Authentication**
  Users can register, log in, and manage their sessions securely.
- **Coding Challenges**
  Practice questions with real-time feedback, retry options, and correct answer reveals.
- **Progress Dashboard**
  Visual track of solved problems, categories, and performance stats.
- **Course & Resource Hub**
  Browse curated courses, programming resources, and tips.
- **Mentorship & Study Groups**
  Request mentorship, join study groups, and learn collaboratively.
- **Jobs & Career Roadmaps**
  Explore job listings, upload CVs, and follow structured career guides. •
  Forum & Questions
  Ask questions, answer peer queries, and engage in topic discussions.
- **Admin Panel**
  Manage users, courses, mentorships, reports, and site content.
- **CV Builder**
  Generate professional resumes using templates and previews.

4. Technology:

Layer	**Technology**
Frontend	HTML, CSS & JavaScript
Backend	PHP
Database	MySQL
Authentication	Session Storage
Hosting	Localhost
Version Control	Git

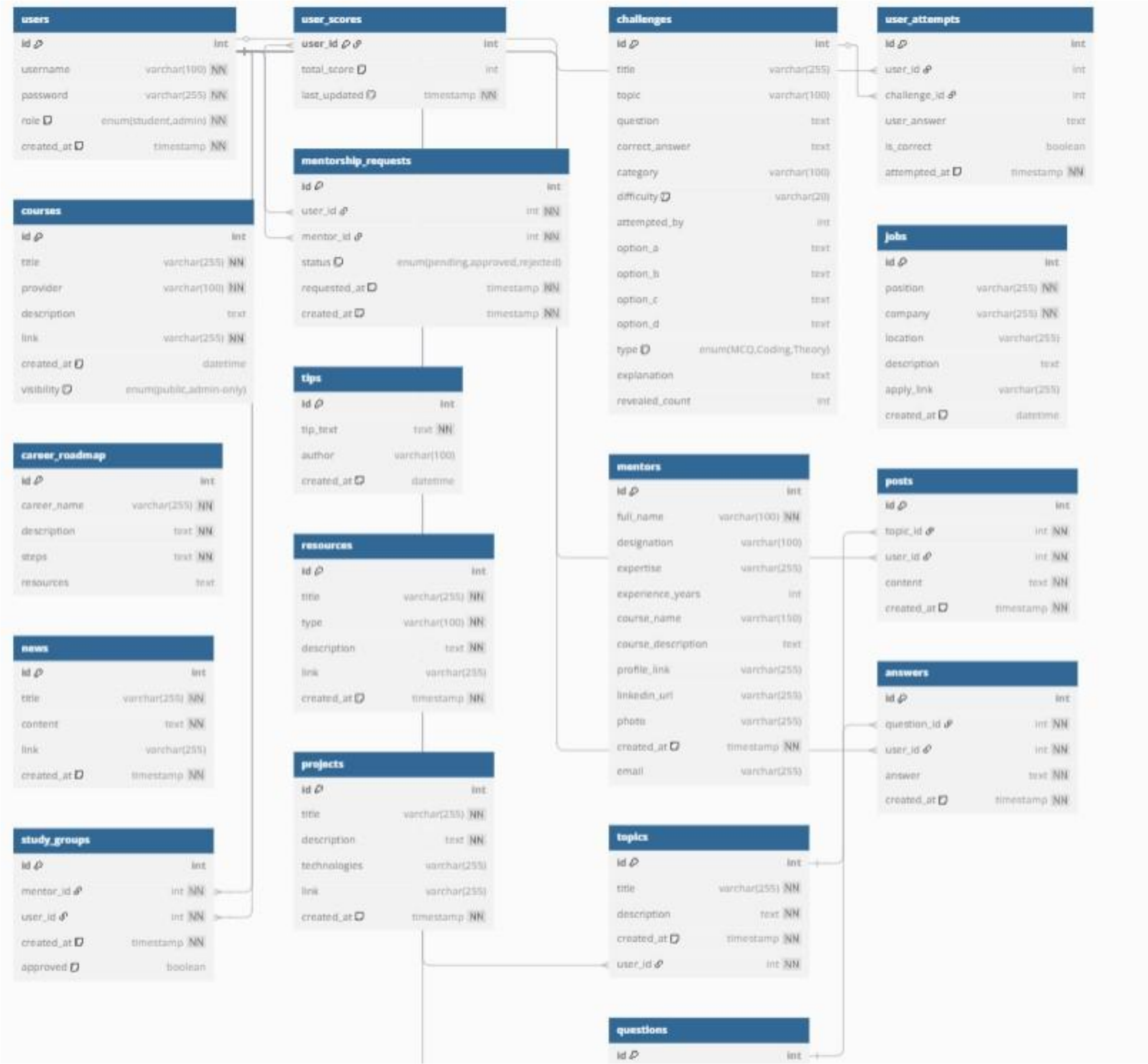
CI/CD	GitHub

5. Database Design

The system uses a **relational database (MySQL/MariaDB)** to store and manage all user, content, and activity data. Key tables and their roles:

- **users**: Stores user credentials, roles (admin/student), and registration details.
- **challenges**: Contains practice problems, correct answers, explanations, and metadata like attempts and category.
- **user_attempts**: Logs each user's submission, correctness, and timestamps.
- **courses, resources, projects**: Host learning content, links, and descriptions for structured learning.
- **mentors, mentorship_requests, study_groups**: Handle mentorship matching and group learning.
- **questions, answers, topics, posts**: Enable Q&A functionality and forum discussions.
- **jobs & cv**: Support job applications, CV building, and career development.
- **tips, news**: Provide motivational tips and updates.
- **user_scores**: Tracks performance for leaderboards and stats.

6.**ER Diagram**:



users		
id	int	
username	varchar(100)	NN
password	varchar(255)	NN
role	enum(student,admin)	NN
created_at	timestamp	NN

user_scores		
user_id	int	
total_score	int	
last_updated	timestamp	NN

challenges	
id	int
title	varchar(255)
topic	varchar(100)
question	text
correct_answer	text
category	varchar(100)
difficulty	varchar(20)
attempted_by	int
option_a	text
option_b	text
option_c	text
option_d	text
type	enum(MCQ,Coding,Theory)
explanation	text
revealed_count	int

user_attempts		
id	int	
user_id	int	
challenge_id	int	
user_answer	text	
is_correct	boolean	
attempted_at	timestamp	NN

mentorship_requests		
id	int	
user_id	int	NN
mentor_id	int	NN
status	enum(pending,approved,rejected)	
requested_at	timestamp	NN
created_at	timestamp	NN

courses		
id	int	
title	varchar(255)	NN
provider	varchar(100)	NN
description	text	
link	varchar(255)	NN
created_at	datetime	
visibility	enum(public,admin-only)	

jobs		
id	int	
position	varchar(255)	NN
company	varchar(255)	NN
location	varchar(255)	
description	text	
apply_link	varchar(255)	
created_at	datetime	

tips		
id	int	
tip_text	text	NN
author	varchar(100)	
created_at	datetime	

career_roadmap		
id	int	
career_name	varchar(255)	NN
description	text	NN
steps	text	NN
resources	text	

mentors		
id	int	
full_name	varchar(100)	NN
designation	varchar(100)	
expertise	varchar(255)	
experience_years	int	
course_name	varchar(150)	
course_description	text	
profile_link	varchar(255)	
linkedin_url	varchar(255)	
photo	varchar(255)	
created_at	timestamp	NN
email	varchar(255)	

posts		
id	int	
topic_id	int	NN
user_id	int	NN
content	text	NN
created_at	timestamp	NN

resources		
id	int	
title	varchar(255)	NN
type	varchar(100)	NN
description	text	NN
link	varchar(255)	
created_at	timestamp	NN

news		
id	int	
title	varchar(255)	NN
content	text	NN
link	varchar(255)	
created_at	timestamp	NN

answers		
id	int	
question_id	int	NN
user_id	int	NN
answer	text	NN
created_at	timestamp	NN

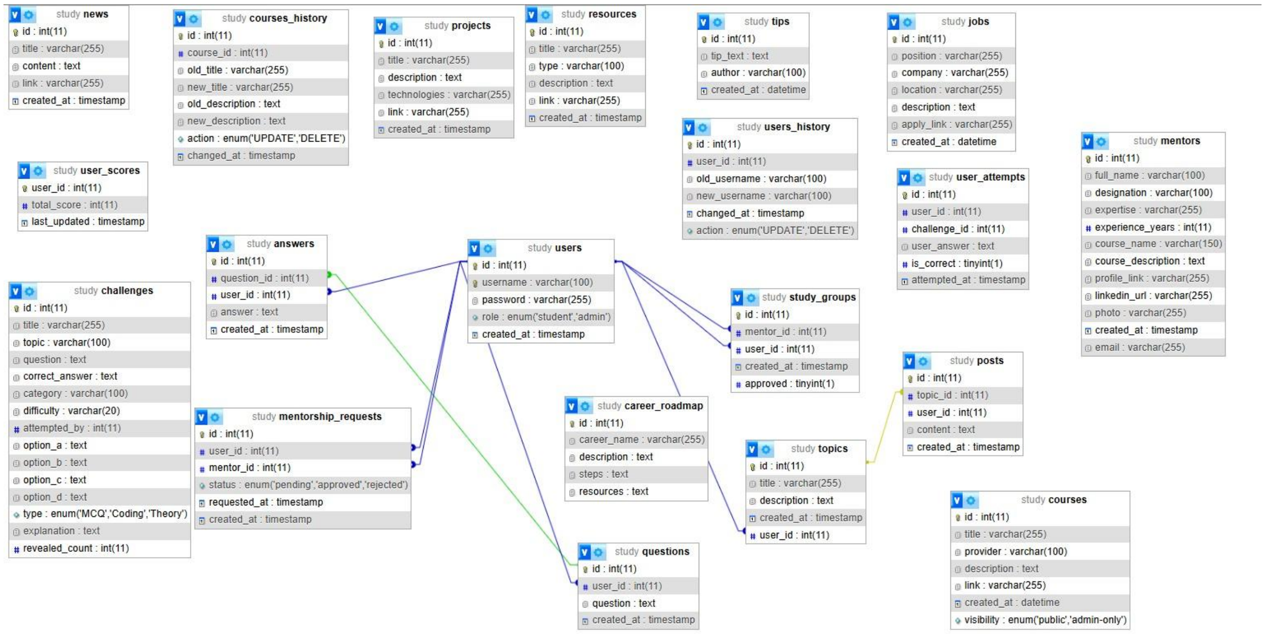
projects		
id	int	
title	varchar(255)	NN
description	text	NN
technologies	varchar(255)	
link	varchar(255)	
created_at	timestamp	NN

study_groups		
id	int	
mentor_id	int	NN
user_id	int	NN
created_at	timestamp	NN
approved	boolean	

topics		
id	int	
title	varchar(255)	NN
description	text	NN
created_at	timestamp	NN
user_id	int	NN

questions	
id	int

7.Schema Diagram:



8.Database Implementation:

Database Creation:

CREATE DATABASE IF NOT EXISTS study;
USE study;

Table Creation:

-- DATABASE CREATION CREATE DATABASE IF NOT EXISTS study; USE study;

-- TABLE: users

```
CREATE TABLE `users` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(100) NOT NULL,
  `password` varchar(255) NOT NULL,
  `role` enum('student','admin') NOT NULL DEFAULT 'student',
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  UNIQUE KEY `username` (`username`)
);

-- TABLE: user_scores
CREATE TABLE `user_scores` (
  `user_id` int(11) NOT NULL,
  `total_score` int(11) DEFAULT 0,
  `last_updated` timestamp NOT NULL DEFAULT current_timestamp() ON
UPDATE current_timestamp(),
  PRIMARY KEY (`user_id`)
);

-- TABLE: challenges
CREATE TABLE `challenges` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) DEFAULT NULL,
  `topic` varchar(100) DEFAULT NULL,
  `question` text DEFAULT NULL,
  `correct_answer` text DEFAULT NULL,
  `category` varchar(100) DEFAULT NULL,
  `difficulty` varchar(20) DEFAULT 'Easy',
  `attempted_by` int(11) DEFAULT 0,
  `option_a` text DEFAULT NULL,
  `option_b` text DEFAULT NULL,
  `option_c` text DEFAULT NULL,
  `option_d` text DEFAULT NULL,
  `type` enum('MCQ','Coding','Theory') DEFAULT 'Coding',
  `explanation` text DEFAULT NULL,
  `revealed_count` int(11) DEFAULT 0,
  PRIMARY KEY (`id`)
);
```

```
-- TABLE: user_attempts
CREATE TABLE `user_attempts` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) DEFAULT NULL,
  `challenge_id` int(11) DEFAULT NULL,
`user_answer` text DEFAULT NULL,
  `is_correct` tinyint(1) DEFAULT NULL,
  `attempted_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);

-- TABLE: courses
CREATE TABLE `courses` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `provider` varchar(100) NOT NULL,
  `description` text DEFAULT NULL,
  `link` varchar(255) NOT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  `visibility` enum('public','admin-only') DEFAULT 'public',
  PRIMARY KEY (`id`)
);

-- TABLE: mentorship_requests
CREATE TABLE `mentorship_requests` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `mentor_id` int(11) NOT NULL,
  `status` enum('pending','approved','rejected') DEFAULT 'pending',
  `requested_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  FOREIGN KEY (`user_id`) REFERENCES `users` (`id`),
  FOREIGN KEY (`mentor_id`) REFERENCES `users` (`id`)
);
```

-- TABLE: mentors

```
CREATE TABLE `mentors` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `full_name` varchar(100) NOT NULL,
  `designation` varchar(100) DEFAULT NULL,
  `expertise` varchar(255) DEFAULT NULL,
  `experience_years` int(11) DEFAULT NULL,
  `course_name` varchar(150) DEFAULT NULL,
  `course_description` text DEFAULT NULL,
  `profile_link` varchar(255) DEFAULT NULL,
  `linkedin_url` varchar(255) DEFAULT NULL,
  `photo` varchar(255) DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `email` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
);
```

-- TABLE: jobs

```
CREATE TABLE `jobs` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `position` varchar(255) NOT NULL,
  `company` varchar(255) NOT NULL,
  `location` varchar(255) DEFAULT NULL,
  `description` text DEFAULT NULL,
  `apply_link` varchar(255) DEFAULT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);
```

-- TABLE: career_roadmap

```
CREATE TABLE `career_roadmap` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `career_name` varchar(255) NOT NULL,
  `description` text NOT NULL,
  `steps` text NOT NULL,
  `resources` text DEFAULT NULL,
```

```sql
  PRIMARY KEY (`id`)
);

-- TABLE: tips
CREATE TABLE `tips` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `tip_text` text NOT NULL,
  `author` varchar(100) DEFAULT NULL,
  `created_at` datetime DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);

-- TABLE: topics
CREATE TABLE `topics` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `description` text NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `user_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (`user_id`) REFERENCES `users`(`id`) ON DELETE CASCADE
);

-- TABLE: posts
CREATE TABLE `posts` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `topic_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `content` text NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  FOREIGN KEY (`topic_id`) REFERENCES `topics`(`id`) ON DELETE CASCADE
);

-- TABLE: news
CREATE TABLE `news` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `content` text NOT NULL,
  `link` varchar(255) DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);

-- TABLE: resources
CREATE TABLE `resources` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `type` varchar(100) NOT NULL,
  `description` text NOT NULL,
  `link` varchar(255) DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);

-- TABLE: questions
CREATE TABLE `questions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `question` text NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
);

-- TABLE: answers
CREATE TABLE `answers` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `question_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `answer` text NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`),
  FOREIGN KEY (`question_id`) REFERENCES `questions` (`id`) ON DELETE CASCADE,
  FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
);

-- TABLE: study_groups
CREATE TABLE `study_groups` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `mentor_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `approved` tinyint(1) DEFAULT 0,
  PRIMARY KEY (`id`),
  FOREIGN KEY (`mentor_id`) REFERENCES `users`(`id`),
  FOREIGN KEY (`user_id`) REFERENCES `users`(`id`)
);

-- TABLE: projects
CREATE TABLE `projects` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `description` text NOT NULL,
  `technologies` varchar(255) DEFAULT NULL,
  `link` varchar(255) DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  PRIMARY KEY (`id`)
);
```

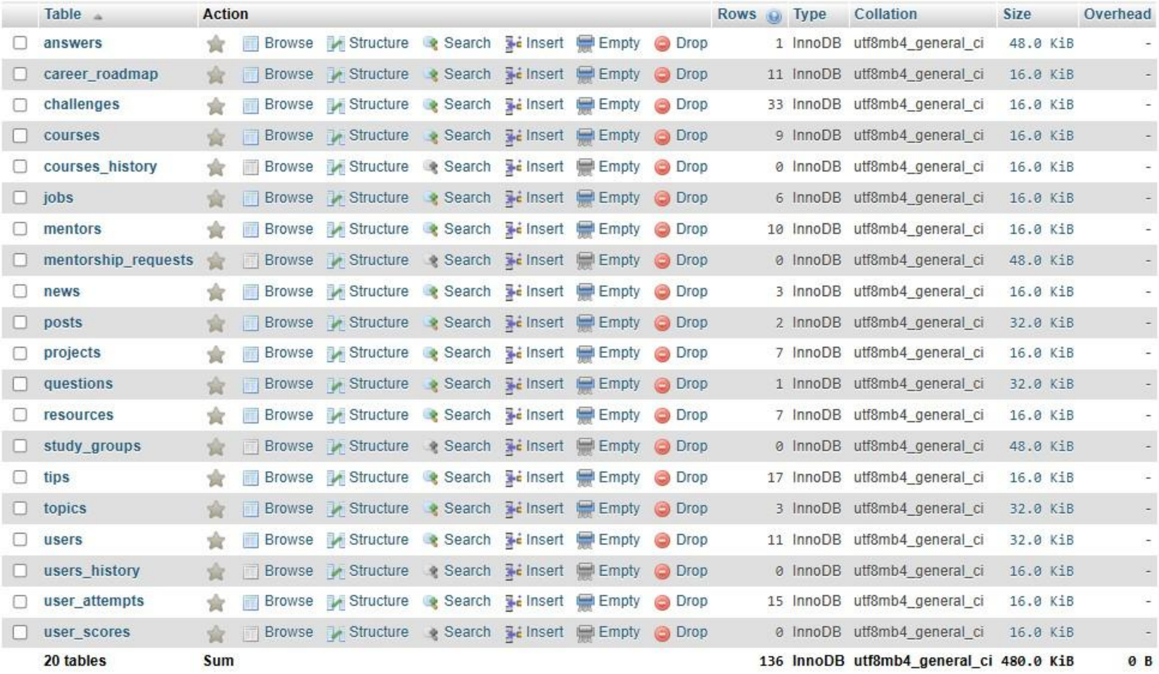


	Table	Action							Rows	Type	Collation	Size	Overhead
☐	answers	☆	Browse	Structure	Search	Insert	Empty	Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	career_roadmap	☆	Browse	Structure	Search	Insert	Empty	Drop	11	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	challenges	☆	Browse	Structure	Search	Insert	Empty	Drop	33	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	courses	☆	Browse	Structure	Search	Insert	Empty	Drop	9	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	courses_history	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	jobs	☆	Browse	Structure	Search	Insert	Empty	Drop	6	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	mentors	☆	Browse	Structure	Search	Insert	Empty	Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	mentorship_requests	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	news	☆	Browse	Structure	Search	Insert	Empty	Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	posts	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	projects	☆	Browse	Structure	Search	Insert	Empty	Drop	7	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	questions	☆	Browse	Structure	Search	Insert	Empty	Drop	1	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	resources	☆	Browse	Structure	Search	Insert	Empty	Drop	7	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	study_groups	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	48.0 KiB	-
☐	tips	☆	Browse	Structure	Search	Insert	Empty	Drop	17	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	topics	☆	Browse	Structure	Search	Insert	Empty	Drop	3	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	users	☆	Browse	Structure	Search	Insert	Empty	Drop	11	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	users_history	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	user_attempts	☆	Browse	Structure	Search	Insert	Empty	Drop	15	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	user_scores	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	20 tables	Sum							136	InnoDB	utf8mb4_general_ci	480.0 KiB	0 B

Insertion in table:

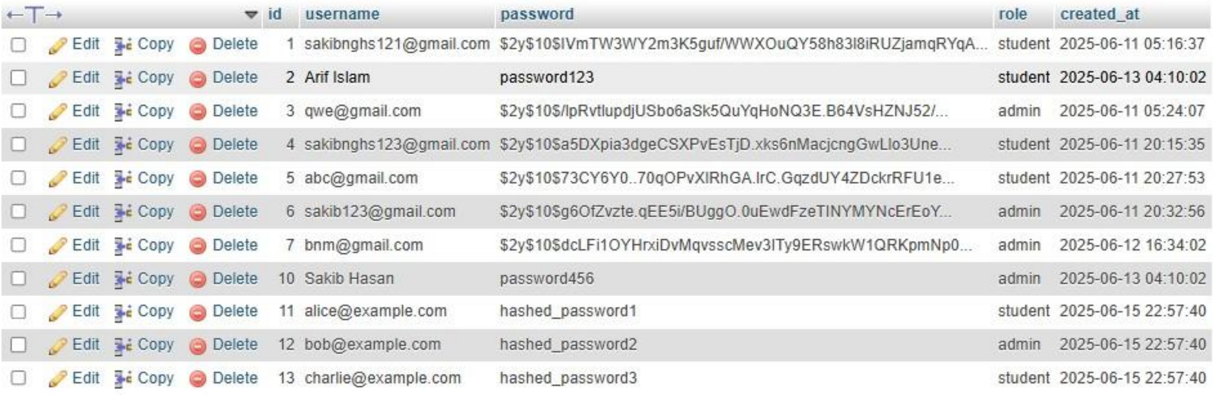
Insertion in table: `users`

INSERT INTO `users` (`id`, `username`, `password`, `role`, `created_at`) VALUES

(1, 'sakibnghs121@gmail.com', '...', 'student', NOW()),

(2, 'Arif Islam', 'password123', 'student', NOW()),

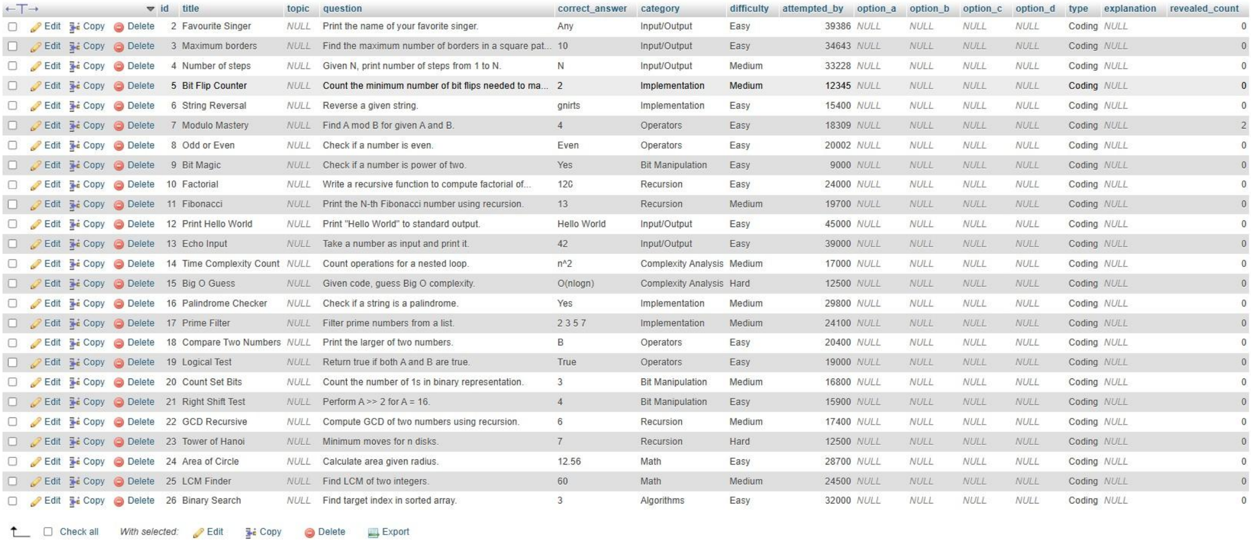
(3, 'qwe@gmail.com', '...', 'admin', NOW());



←T→				id	username	password	role	created_at
☐	Edit	Copy	Delete	1	sakibnghs121@gmail.com	$2y$10$IVmTW3WY2m3K5guf/WWXOuQY58h83I8iRUZjamqRYqA...	student	2025-06-11 05:16:37
☐	Edit	Copy	Delete	2	Arif Islam	password123	student	2025-06-13 04:10:02
☐	Edit	Copy	Delete	3	qwe@gmail.com	$2y$10$/lpRvtlupdjUSbo6aSk5QuYqHoNQ3E.B64VsHZNJ52/...	admin	2025-06-11 05:24:07
☐	Edit	Copy	Delete	4	sakibnghs123@gmail.com	$2y$10$a5DXpia3dgeCSXPvEsTjD.xks6nMacjcngGwLlo3Une...	student	2025-06-11 20:15:35
☐	Edit	Copy	Delete	5	abc@gmail.com	$2y$10$73CY6Y0..70qOPvXIRhGA.IrC.GqzdUY4ZDckrRFU1e...	student	2025-06-11 20:27:53
☐	Edit	Copy	Delete	6	sakib123@gmail.com	$2y$10$g6OfZvzte.qEE5i/BUggO.0uEwdFzeTINYMYNcErEoY...	admin	2025-06-11 20:32:56
☐	Edit	Copy	Delete	7	bnm@gmail.com	$2y$10$dcLFi1OYHrxiDvMqvsscMev3ITy9ERswkW1QRKpmNp0...	admin	2025-06-12 16:34:02
☐	Edit	Copy	Delete	10	Sakib Hasan	password456	admin	2025-06-13 04:10:02
☐	Edit	Copy	Delete	11	alice@example.com	hashed_password1	student	2025-06-15 22:57:40
☐	Edit	Copy	Delete	12	bob@example.com	hashed_password2	admin	2025-06-15 22:57:40
☐	Edit	Copy	Delete	13	charlie@example.com	hashed_password3	student	2025-06-15 22:57:40

Insertion in table: `challenges`

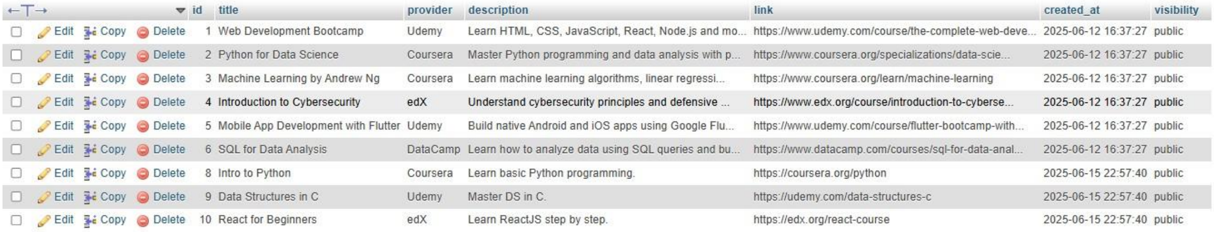
```
INSERT INTO `challenges` (`id`, `title`, `question`,
`correct_answer`, `category`, `difficulty`, `type`)
VALUES
(1, 'Print Hello World', 'Print "Hello World" to standard
output.', 'Hello World', 'Input/Output',
'Easy', 'Coding'),
(2, 'Check Even Number', 'Check if a number is even.',
'Even', 'Operators', 'Easy', 'Coding');
```



←T→	id	title	topic	question	correct_answer	category	difficulty	attempted_by	option_a	option_b	option_c	option_d	type	explanation	revealed_count
☐ Edit Copy Delete	2	Favourite Singer	NULL	Print the name of your favorite singer.	Any	Input/Output	Easy	39386	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	3	Maximum borders	NULL	Find the maximum number of borders in a square pat...	10	Input/Output	Easy	34643	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	4	Number of steps	NULL	Given N, print number of steps from 1 to N.	N	Input/Output	Medium	33228	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	5	Bit Flip Counter	NULL	Count the minimum number of bit flips needed to ma...	2	Implementation	Medium	12345	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	6	String Reversal	NULL	Reverse a given string.	gnirts	Implementation	Easy	15400	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	7	Modulo Mastery	NULL	Find A mod B for given A and B.	4	Operators	Easy	18309	NULL	NULL	NULL	NULL	Coding	NULL	2
☐ Edit Copy Delete	8	Odd or Even	NULL	Check if a number is even.	Even	Operators	Easy	20002	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	9	Bit Magic	NULL	Check if a number is power of two.	Yes	Bit Manipulation	Easy	9000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	10	Factorial	NULL	Write a recursive function to compute factorial of...	120	Recursion	Easy	24000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	11	Fibonacci	NULL	Print the N-th Fibonacci number using recursion.	13	Recursion	Medium	19700	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	12	Print Hello World	NULL	Print "Hello World" to standard output.	Hello World	Input/Output	Easy	45000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	13	Echo Input	NULL	Take a number as input and print it.	42	Input/Output	Easy	39000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	14	Time Complexity Count	NULL	Count operations for a nested loop.	n^2	Complexity Analysis	Medium	17000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	15	Big O Guess	NULL	Given code, guess Big O complexity.	O(nlogn)	Complexity Analysis	Hard	12500	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	16	Palindrome Checker	NULL	Check if a string is a palindrome.	Yes	Implementation	Medium	29800	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	17	Prime Filter	NULL	Filter prime numbers from a list.	2 3 5 7	Implementation	Medium	24100	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	18	Compare Two Numbers	NULL	Print the larger of two numbers.	B	Operators	Easy	20400	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	19	Logical Test	NULL	Return true if both A and B are true.	True	Operators	Easy	19000	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	20	Count Set Bits	NULL	Count the number of 1s in binary representation.	3	Bit Manipulation	Medium	16800	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	21	Right Shift Test	NULL	Perform A >> 2 for A = 16.	4	Bit Manipulation	Easy	15900	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	22	GCD Recursive	NULL	Compute GCD of two numbers using recursion.	6	Recursion	Medium	17400	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	23	Tower of Hanoi	NULL	Minimum moves for n disks.	7	Recursion	Hard	12500	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	24	Area of Circle	NULL	Calculate area given radius.	12.56	Math	Easy	28700	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	25	LCM Finder	NULL	Find LCM of two integers.	60	Math	Medium	24500	NULL	NULL	NULL	NULL	Coding	NULL	0
☐ Edit Copy Delete	26	Binary Search	NULL	Find target index in sorted array.	3	Algorithms	Easy	32000	NULL	NULL	NULL	NULL	Coding	NULL	0

Insertion in table: `courses`

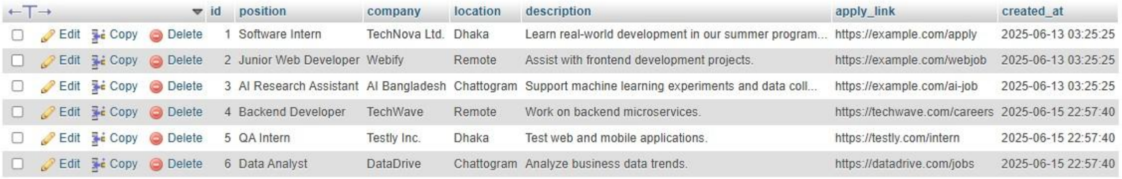
```
INSERT INTO `courses` (`id`, `title`, `provider`,
`description`, `link`, `created_at`, `visibility`)
VALUES
(1, 'Web Development Bootcamp', 'Udemy', 'Learn HTML,
CSS, JS, React, Node.js',
'https://www.udemy.com/course/the-complete-
webdevelopment-bootcamp/', NOW(), 'public');
```



←T→	id	title	provider	description	link	created_at	visibility
☐ Edit Copy Delete	1	Web Development Bootcamp	Udemy	Learn HTML, CSS, JavaScript, React, Node.js and mo...	https://www.udemy.com/course/the-complete-web-deve...	2025-06-12 16:37:27	public
☐ Edit Copy Delete	2	Python for Data Science	Coursera	Master Python programming and data analysis with p...	https://www.coursera.org/specializations/data-scie...	2025-06-12 16:37:27	public
☐ Edit Copy Delete	3	Machine Learning by Andrew Ng	Coursera	Learn machine learning algorithms, linear regressi...	https://www.coursera.org/learn/machine-learning	2025-06-12 16:37:27	public
☐ Edit Copy Delete	4	Introduction to Cybersecurity	edX	Understand cybersecurity principles and defensive ...	https://www.edx.org/course/introduction-to-cyberse...	2025-06-12 16:37:27	public
☐ Edit Copy Delete	5	Mobile App Development with Flutter	Udemy	Build native Android and iOS apps using Google Flu...	https://www.udemy.com/course/flutter-bootcamp-with...	2025-06-12 16:37:27	public
☐ Edit Copy Delete	6	SQL for Data Analysis	DataCamp	Learn how to analyze data using SQL queries and bu...	https://www.datacamp.com/courses/sql-for-data-anal...	2025-06-12 16:37:27	public
☐ Edit Copy Delete	8	Intro to Python	Coursera	Learn basic Python programming.	https://coursera.org/python	2025-06-15 22:57:40	public
☐ Edit Copy Delete	9	Data Structures in C	Udemy	Master DS in C.	https://udemy.com/data-structures-c	2025-06-15 22:57:40	public
☐ Edit Copy Delete	10	React for Beginners	edX	Learn ReactJS step by step.	https://edx.org/react-course	2025-06-15 22:57:40	public

Insertion in table: `jobs`

```
INSERT INTO `jobs` (`id`, `position`, `company`,
`location`, `description`, `apply_link`, `created_at`)
VALUES
(1, 'Software Intern', 'TechNova Ltd.', 'Dhaka', 'Summer
internship.', 'https://example.com/apply', NOW());
```



←T→	id	position	company	location	description	apply_link	created_at
□ Edit Copy Delete	1	Software Intern	TechNova Ltd.	Dhaka	Learn real-world development in our summer program...	https://example.com/apply	2025-06-13 03:25:25
□ Edit Copy Delete	2	Junior Web Developer	Webify	Remote	Assist with frontend development projects.	https://example.com/webjob	2025-06-13 03:25:25
□ Edit Copy Delete	3	AI Research Assistant	AI Bangladesh	Chattogram	Support machine learning experiments and data coll...	https://example.com/ai-job	2025-06-13 03:25:25
□ Edit Copy Delete	4	Backend Developer	TechWave	Remote	Work on backend microservices.	https://techwave.com/careers	2025-06-15 22:57:40
□ Edit Copy Delete	5	QA Intern	Testly Inc.	Dhaka	Test web and mobile applications.	https://testly.com/intern	2025-06-15 22:57:40
□ Edit Copy Delete	6	Data Analyst	DataDrive	Chattogram	Analyze business data trends.	https://datadrive.com/jobs	2025-06-15 22:57:40

Insertion in table: `resources`

```
INSERT INTO `resources` (`id`, `title`, `type`,
`description`, `link`, `created_at`) VALUES
(1, 'Learn JavaScript', 'Tutorial', 'JS programming from
scratch.', 'https://www.learn-js.org/', NOW());
```

Insertion in table: `tips`

```
INSERT INTO `tips` (`id`, `tip_text`, `author`,
`created_at`) VALUES
(1, 'Practice daily. Consistency beats intensity.',
'Arif', NOW());
```

Insertion in table: `mentors`

```
INSERT INTO `mentors` (`id`, `full_name`, `designation`,
`expertise`, `experience_years`, `course_name`,
`created_at`) VALUES
(1, 'John Doe', 'Senior Software Engineer', 'Web
Development, PHP', 8, 'Advanced Web Programming', NOW());
```

Insertion in table: `projects`

```
INSERT INTO `projects` (`id`, `title`, `description`,
`technologies`, `link`, `created_at`) VALUES
(1, 'Student Attendance System', 'Tracks attendance via
facial recognition.', 'Python, OpenCV, MySQL',
'https://github.com/yourusername/attendance-system',
NOW());
```

Insertion in table: news

```
INSERT INTO `news` (`id`, `title`, `content`, `link`,
`created_at`) VALUES
(1, 'New AI Research Breakthrough', 'AI model improves
NLU significantly.', 'https://example.com/ai-research',
NOW());
```

Insertion in table: questions

```
INSERT INTO `questions` (`id`, `user_id`, `question`,
`created_at`) VALUES
(1, 1, 'What is polymorphism in OOP?', NOW());
```

Insertion in table: answers

```
INSERT INTO `answers` (`id`, `question_id`, `user_id`,
`answer`, `created_at`) VALUES
(1, 1, 1, 'Polymorphism allows methods to do different
things based on the object.', NOW());
```

Insertion in table: user_attempts

```
INSERT INTO `user_attempts` (`id`, `user_id`,
`challenge_id`, `user_answer`, `is_correct`,
`attempted_at`) VALUES
(1, 1, 1, 'Hello', 0, NOW());
```

Insertion in table: `user_scores`

```
INSERT INTO `user_scores` (`user_id`, `total_score`,
`last_updated`) VALUES
(1, 10, NOW());
```

Tables with Joinable Relationships

Here are the key relationships that naturally support JOIN operations:

1. answers - questions & users

```
SELECT a.answer, q.question, u.username
FROM answers a
JOIN questions q ON a.question_id = q.id
JOIN users u ON a.user_id = u.id;
```

2. mentorship_requests - users (as user and mentor)

```
SELECT mr.id, u1.username AS student, u2.username AS mentor, mr.status
FROM mentorship_requests mr
JOIN users u1 ON mr.user_id = u1.id
JOIN users u2 ON mr.mentor_id = u2.id;
```

3. study_groups - users (as mentor and user)

```
SELECT sg.id, mentor.username AS mentor_name, student.username AS
student_name FROM study_groups sg
JOIN users mentor ON sg.mentor_id = mentor.id
JOIN users student ON sg.user_id = student.id;
```

4. posts - topics

```
SELECT p.content, t.title
FROM posts p
JOIN topics t ON p.topic_id = t.id;
```

5. questions - users

```
SELECT q.question, u.username
FROM questions q
JOIN users u ON q.user_id = u.id;
```

6. topics - users

```
SELECT t.title, u.username
FROM topics t
JOIN users u ON t.user_id = u.id;
```

9.Triggers & Security

Trigger: `before_courses_update`

Table: `courses`
Purpose: Log course title/description changes into `courses_history`

```
DELIMITER $$

CREATE TRIGGER `before_courses_update`
BEFORE UPDATE ON `courses`
FOR EACH ROW
BEGIN
  INSERT INTO courses_history (
course_id, old_title, new_title,
old_description, new_description,
action   ) VALUES (
    OLD.id, OLD.title, NEW.title,
    OLD.description, NEW.description,
    'UPDATE'
  );
END$$
```

```
DELIMITER ;
```

Trigger: `before_jobs_delete`

Table: `jobs`
Purpose: Log deleted job info into `jobs_log` before deletion

```
DELIMITER $$

CREATE TRIGGER `before_jobs_delete`
BEFORE DELETE ON `jobs`
FOR EACH ROW
BEGIN
  INSERT INTO jobs_log (
job_id, position, company,
action, deleted_at
  ) VALUES (
    OLD.id, OLD.position, OLD.company,
    'DELETE', NOW()
  );
END$$

DELIMITER ;

CREATE TABLE `jobs_log` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
  `job_id` INT,
  `position` VARCHAR(255),
  `company` VARCHAR(255),
  `action` ENUM('DELETE'),
  `deleted_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP );
```

Trigger: `before_users_update`

Table: `users`

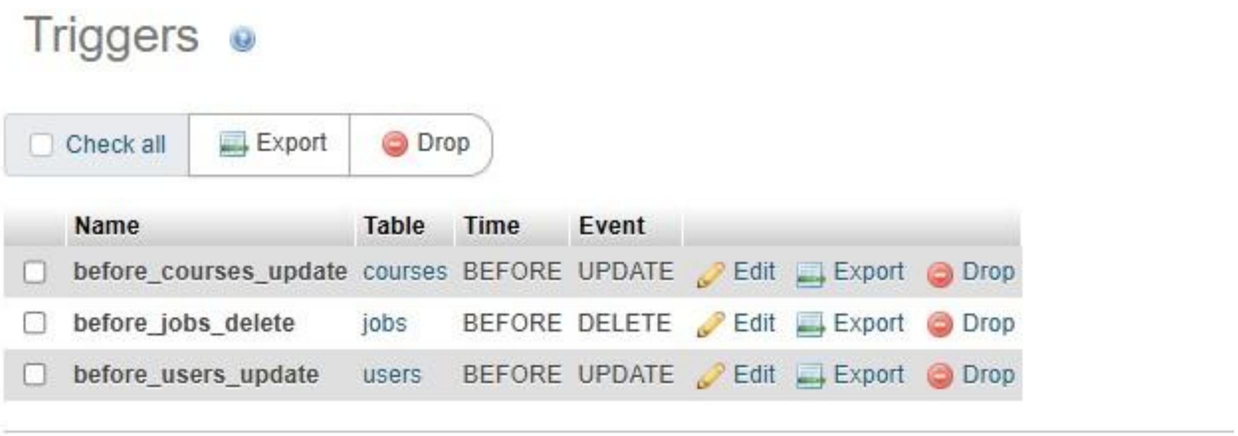
Purpose: Log username updates into `users_history`

```
DELIMITER $$

CREATE TRIGGER `before_users_update`
BEFORE UPDATE ON `users`
FOR EACH ROW
BEGIN
  INSERT INTO users_history (
    user_id, old_username, new_username, action
  ) VALUES (
    OLD.id, OLD.username, NEW.username, 'UPDATE'
  );
END$$

DELIMITER ;
```

```
CREATE TABLE `users_history` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
  `user_id` INT,
  `old_username` VARCHAR(100),
  `new_username` VARCHAR(100),
  `changed_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  `action` ENUM('UPDATE','DELETE')
);
```



	Name	Table	Time	Event			
□	before_courses_update	courses	BEFORE	UPDATE	Edit	Export	Drop
□	before_jobs_delete	jobs	BEFORE	DELETE	Edit	Export	Drop
□	before_users_update	users	BEFORE	UPDATE	Edit	Export	Drop

10. Testing

To ensure the reliability and performance of the platform, various testing methods were applied during development:

1. Functional Testing

- Verified that all modules (courses, projects, questions, jobs, etc.) perform expected tasks.
- Tested user roles (admin, student) for correct permission handling.
- Ensured CRUD operations work on all major entities (users, mentors, posts, etc.).

2. Database Testing

- Checked all **foreign key relationships** via join queries.
- Validated **trigger execution** (e.g., before_users_update, before_jobs_delete).
- Inserted dummy data into all tables and verified data integrity.

3. UI/UX Testing

- Ensured responsiveness across browsers and screen sizes.
- Validated form input handling, error messages, and alerts.
- Tested navigation between pages (e.g., login, dashboard, profile, etc.).

4. Security Testing

- Confirmed **authentication** and **role-based access control**.
- Checked for **SQL injection protection** in input fields.
- Passwords stored securely using **hashing** (e.g., bcrypt).

5. Performance Testing

- Loaded the dashboard with sample data to assess rendering speed.
- Tested query response time for complex JOIN operations.
- Reviewed time taken to load media assets (images, videos).

11. Future Scope

The current version of the CSE Study Room platform serves as a foundational tool for students, educators, and developers. However, there is significant potential for future enhancements:

1. Real-Time Communication

- Integrate live chat and video mentoring using WebRTC or third-party APIs (e.g., Zoom SDK).

2. Mobile Application

- Develop Android/iOS apps for better accessibility and user experience.

3. Gamification

- Introduce badges, leaderboards, and rewards to encourage active participation.

4. AI-Based Recommendation

- Recommend personalized courses, mentors, and challenges based on user activity and interests.

5. Advanced Analytics

- Dashboard for mentors/admins to monitor user performance, course popularity, and system metrics.

6. Resume & Portfolio Builder

- Provide tools to auto-generate resumes and project portfolios from user activities.

7. API Integration

- Support external systems with RESTful APIs for user authentication, data sync, etc.

8. Multilingual Support

- Enable the platform to support multiple languages for wider reach.

12.Conclusion:

The **CSE Study Room** project successfully integrates various academic and career development tools into a unified platform. It offers features such as course management, coding challenges, job listings, mentorship, and discussion forums — all tailored to empower students and developers.

Through its database-backed design and modular architecture, the system ensures scalability, usability, and future expansion. This project not only demonstrates strong backend and frontend integration but also reflects real-world software development practices.

Overall, it lays a solid foundation for a dynamic learning ecosystem and opens up opportunities for further innovation in education technology.