
E-Learning Platform

Empowering Education Through Technology

Features:

- **Course Management**: Enables instructors to create, update, and manage courses.
- **Student Enrollment**: Students can enrol in courses and provide feedback.
- **Secure User Authentication**: Ensures secure login and role-based access (instructor, student).

Technologies Used:

- **Backend**: MySQL, Node.js, Express
- **Frontend**: React, HTML, CSS
- **Authentication**: JWT-based security
- **Database**: MySQL

Developed By:

- Vishnu Srinivas – PES1UG22CS720
- Vidwan KV – PES1UG22CS687

Purpose

This product implements the server-side database software for managing and tracking an e-learning platform, designed for use by educators and administrators. The platform allows institutions to host the software on their own servers to monitor and manage educational services for students. This

product handles high-level aspects of running the e-learning system, including course management, user enrollment, and providing intuitive interfaces for both students and instructors.

Description

The **E-Learning Platform** is a web-based system designed to enhance the learning experience for both students and instructors. It provides a comprehensive set of tools to manage courses, track student progress, and offer interactive learning opportunities. Through this platform, students can access a wide range of educational content, participate in courses, and provide feedback on their learning experiences.

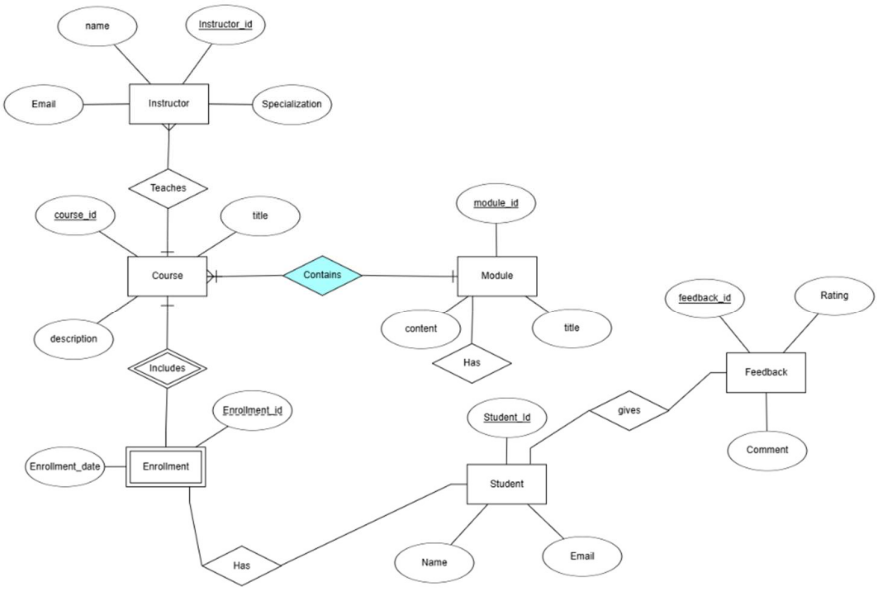
The platform is structured to facilitate easy course creation and management by instructors, enabling them to upload materials, create quizzes, assign tasks, and monitor student performance in real-time. Students can enroll in courses based on their interests, interact with instructors, and track their learning journey through detailed progress reports.

Key features of the platform include:

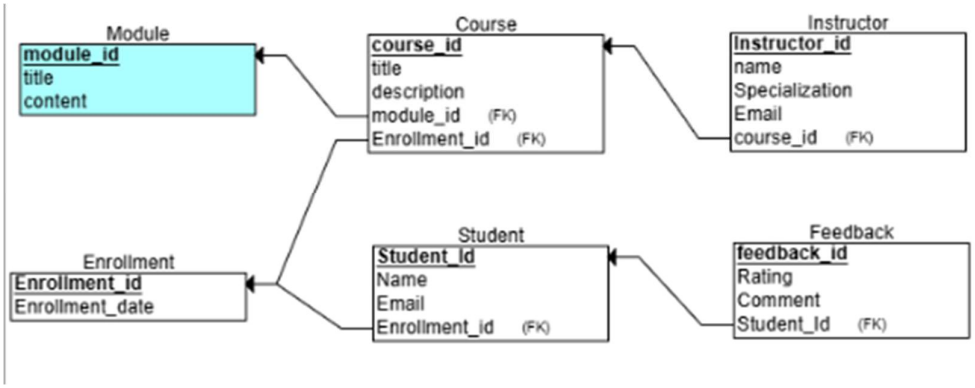
- **Course Management**: Instructors can create and manage various courses, add multimedia content, and engage students with discussion forums and assignments.
- **Student Enrollment and Progress Tracking**: Students can browse, enroll, and track their progress throughout their course. They can review materials, complete assignments, and participate in assessments.
- **Feedback Mechanism**: Both students and instructors can provide valuable feedback on courses, enhancing future course offerings and educational quality.
- **Role-based Access Control**: The platform includes multiple user roles, such as students, instructors, and administrators, each with specific access rights.
- **Interactive Learning Tools**: The platform integrates multimedia resources such as videos, presentations, and live lectures to enrich the learning experience.

With its easy-to-use interface and powerful back-end infrastructure, the **E-Learning Platform** aims to make education more accessible, flexible, and engaging for all learners, regardless of location or device.

ER – DIAGRAM



RELATIONAL SCHEMA



DDL COMMANDS:

```
CREATE DATABASE ELEARNING;
USE ELEARNING;

CREATE TABLE USERS (
  ID INT AUTO_INCREMENT PRIMARY KEY,
  USERNAME VARCHAR(50) NOT NULL UNIQUE,
  PASSWORD VARCHAR(255) NOT NULL,
  EMAIL VARCHAR(100) NOT NULL UNIQUE,
  ROLE ENUM('STUDENT', 'INSTRUCTOR', 'ADMIN') NOT NULL
);

CREATE TABLE INSTRUCTORS (
  INSTRUCTOR_ID INT AUTO_INCREMENT PRIMARY KEY,
  USER_ID INT,
  USERNAME VARCHAR(50) NOT NULL UNIQUE,
  PASSWORD VARCHAR(255) NOT NULL,
  EMAIL VARCHAR(100) NOT NULL UNIQUE,
  FOREIGN KEY (USER_ID) REFERENCES USERS(ID)
);

CREATE TABLE STUDENTS (
  STUDENT_ID INT AUTO_INCREMENT PRIMARY KEY,
  USER_ID INT,
  USERNAME VARCHAR(50) NOT NULL UNIQUE,
  PASSWORD VARCHAR(255) NOT NULL,
  EMAIL VARCHAR(100) NOT NULL UNIQUE,
  FOREIGN KEY (USER_ID) REFERENCES USERS(ID)
);

CREATE TABLE COURSES (
  ID INT AUTO_INCREMENT PRIMARY KEY,
  TITLE VARCHAR(100) NOT NULL,
```

```
    DESCRIPTION TEXT,
    INSTRUCTOR_ID INT,
    FOREIGN KEY (INSTRUCTOR_ID) REFERENCES INSTRUCTORS(INSTRUCTOR_ID)
);

CREATE TABLE MODULES (
    ID INT AUTO_INCREMENT PRIMARY KEY,
    TITLE VARCHAR(100) NOT NULL,
    CONTENT TEXT,
    COURSE_ID INT,
    DESCRIPTION TEXT,
    FOREIGN KEY (COURSE_ID) REFERENCES COURSES(ID)
);

CREATE TABLE ENROLLMENTS (
    USER_ID INT,
    COURSE_ID INT,
    ENROLLMENT_DATE DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (USER_ID, COURSE_ID),
    FOREIGN KEY (USER_ID) REFERENCES USERS(ID),
    FOREIGN KEY (COURSE_ID) REFERENCES COURSES(ID)
);

CREATE TABLE FEEDBACK (
    ID INT AUTO_INCREMENT PRIMARY KEY,
    USER_ID INT,
    COURSE_ID INT,
    RATING INT NOT NULL,
    COMMENT TEXT,
    CREATED_AT TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (USER_ID) REFERENCES USERS(ID),
```

```
  FOREIGN KEY (course_id) REFERENCES courses(id)
);
```

→TRIGGER

Trigger for validating course title and description

```
CREATE TRIGGER validate_course_data_insert
BEFORE INSERT ON courses
FOR EACH ROW
BEGIN
  IF LENGTH(NEW.title) < 5 THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Course title must be at least 5 characters
long.';
  END IF;

  IF LENGTH(NEW.description) < 10 THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Course description must be at least 10
characters long.';
  END IF;
END;


DELIMITER $$
```

Trigger for checking duplicate usernames

```
CREATE TRIGGER check_duplicate_username
BEFORE INSERT ON users
FOR EACH ROW
BEGIN
  -- Check if the username already exists
  DECLARE duplicate_username INT;

  SELECT COUNT(*)
  INTO duplicate_username
  FROM users
  WHERE username = NEW.username;

  -- If a duplicate username exists, raise an error
  IF duplicate_username > 0 THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = CONCAT('Duplicate entry for username: ', NEW.username);
  END IF;
END $$

DELIMITER ;
```

→Function

Function returns enrolled courses for specified user.

```
CREATE DEFINER=`root`@`localhost` FUNCTION `get_courses_for_user`(user_id INT)
RETURNS JSON DETERMINISTIC
BEGIN
 DECLARE course_list JSON;

 SELECT JSON_ARRAYAGG(
  JSON_OBJECT(
   'course_id', c.id,
   'course_title', c.title,
   'course_description', c.description
  )
 ) INTO course_list
 FROM courses c
 JOIN enrollments e ON c.id = e.course_id
 WHERE e.user_id = user_id;

 RETURN IFNULL(course_list, JSON_ARRAY());
END
```

→Procedure

Procedure returns courses associated with that instructor

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_courses_by_instructor`(
  IN instructorId INT
)
BEGIN
  SELECT id AS course_id,
      title AS course_title,
      description AS course_description
  FROM courses
  WHERE instructor_id = instructorId;
END
```

→Queries

Aggregate Query

Returns the number of students enrolled in each course(count)

```
SELECT courses.title, COUNT(enrollments.user_id) AS student_count
FROM courses
LEFT JOIN enrollments ON courses.id = enrollments.course_id
GROUP BY courses.title;
```

Join Query

Returns instructor made courses

```
SELECT courses.title, users.username
FROM courses
```

```
INNER JOIN USERS ON COURSES.INSTRUCTOR_ID = USERS.ID
WHERE USERS.ROLE = 'INSTRUCTOR';
```

Join Query

Returns the course taken by a student

```
SELECT USERS.USERNAME AS STUDENT, COURSES.TITLE
FROM ENROLLMENTS
INNER JOIN USERS ON ENROLLMENTS.USER_ID = USERS.ID
INNER JOIN COURSES ON ENROLLMENTS.COURSE_ID = COURSES.ID
WHERE USERS.ROLE = 'STUDENT';
```

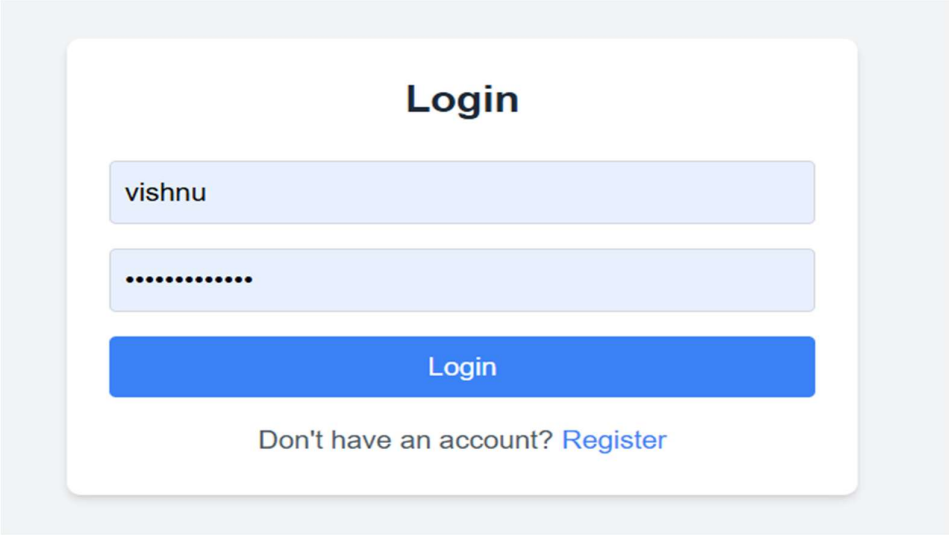
Nested Query

Returns the most enrolled course

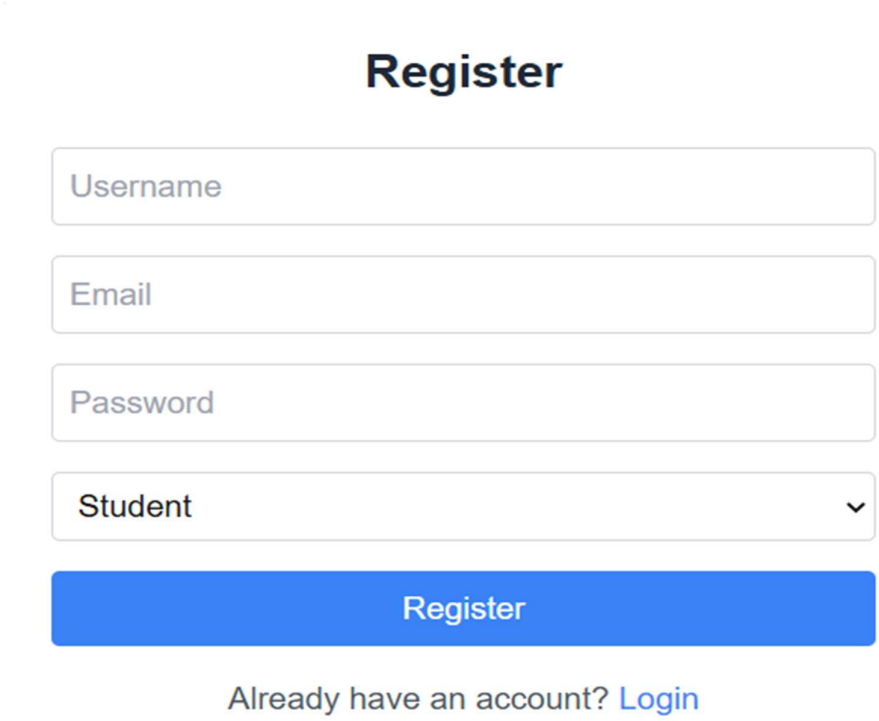
```
SELECT TITLE
FROM COURSES
WHERE ID = (
  SELECT COURSE_ID
  FROM ENROLLMENTS
  GROUP BY COURSE_ID
  ORDER BY COUNT(USER_ID) DESC
  LIMIT 1
);
```

List of Functionalities:

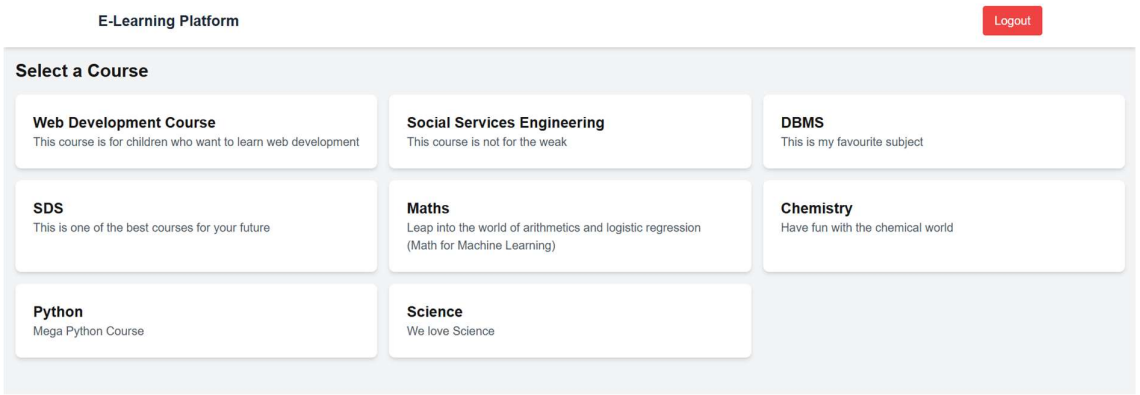
→Login



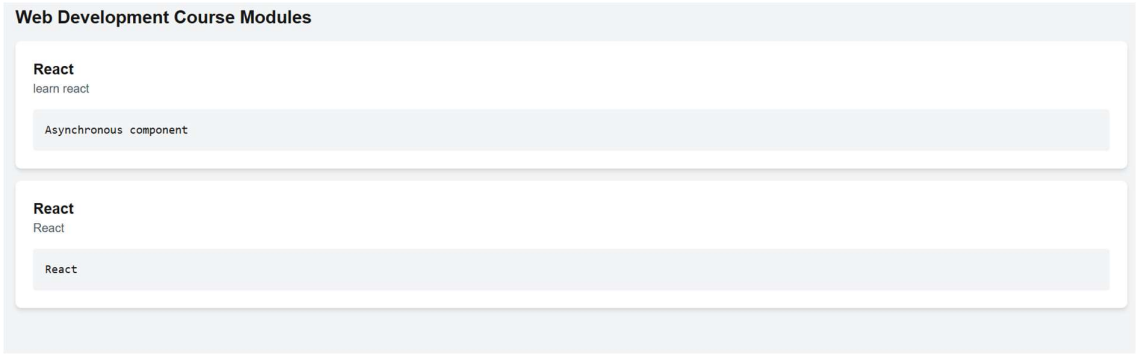
→Register



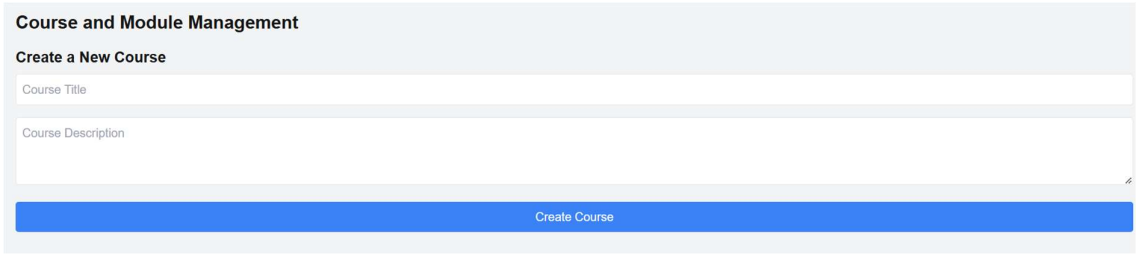
Select Module



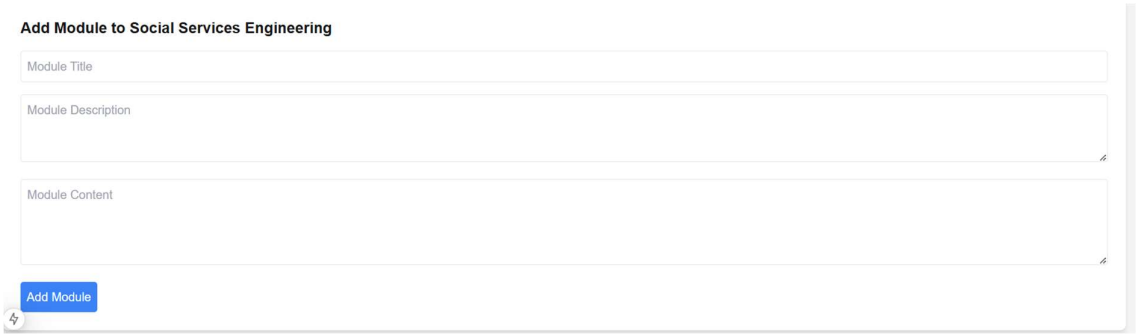
Module



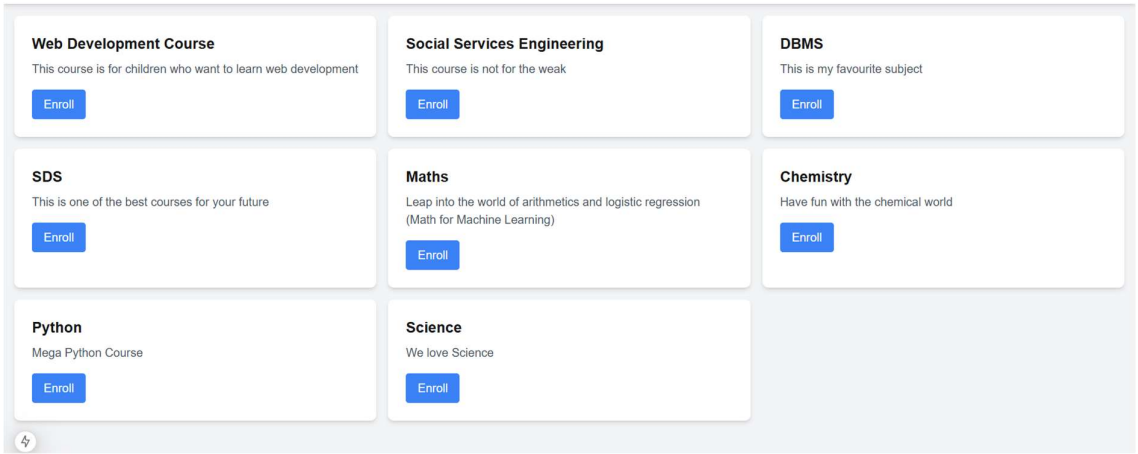
Create Course



Adding Module to a course



Enroll (For Students)



Crud Operations

Create operation listed above

Read operation

```
mysql> select * from users;
+----+-------------+--------------------------------------------------------------+-----------------------------------+------------+
| id | username    | password                                                     | email                             | role       |
+----+-------------+--------------------------------------------------------------+-----------------------------------+------------+
|  1 | vishnu      | $2b$12$uc0EV72ft.YpgYaCE5MZiuZRbsXUqj6OvwtosoICYscE6n2eCLN1m | vishnusrinivas00@gmail.com        | student    |
|  3 | vishnu12    | $2b$12$vFQIYkIFvIDm/67Bddn5dudn26ZE.O0rOzqr.ssLPGEwGvOu/0y9S | vishnusrinivas100@gmail.com       | instructor |
|  4 | vishnu123   | $2b$12$IZtGlJCJosj0/HLRaIowz.aVm1Hm2WbOCmbKAq6jG4GknpxBuYksu | vishnusrin43ivas00@gmail.com      | student    |
|  5 | vishnu124   | $2b$12$4mGVJLAeX9cusZSa2EcDaeH4BAkNCP57UOVFhE7b2xVhRz./zEw3e | vishnusrin43iv32as00@gmail.com    | student    |
|  6 | vishnu1244  | $2b$12$EvVtzVr295UIrSEYjBJCHOhyeNyWmnMUNhqsuORcy1.6LTnJbtz6e | vishnusrin43ivse32as00@gmail.com  | instructor |
|  9 | vishnu322   | $2b$12$eQX.9hYB6.dR/uOt8iXMSeH8hN81omon6R0GS1Rehu9T49goNHx3u | vishnusrin43iv3223e32as00@gmail.com | instructor |
| 11 | vishnu322e  | $2b$12$b7/.ZbJM.8uMAyORSQPE8O8Vo0cPE0NDA4.kxd2PAq8hbU.TUSqAy | vishnusrin43iv3es223e32as00@gmail.com | student    |
| 13 | vishnu123445 | $2b$12$uDZvMEtcG3/8eDARD93mWOB/LWSYndsxEiZizx0u.sXx2WJKNBUMa | vishnusriniva3223s00@gmail.com    | student    |
| 14 | shiyo       | $2b$12$V89McOI59qfyVJWOEHLMGe4x.DXnIOGmpMLfKfO3kz6HkkAOi1aiK | vanivila12@gmai                   | instructor |
+----+-------------+--------------------------------------------------------------+-----------------------------------+------------+
9 rows in set (0.00 sec)

mysql>
```

```
mysql> select * from enrollments;
+---------+-----------+---------------------+
| user_id | course_id | enrollment_date     |
+---------+-----------+---------------------+
|       1 |         1 | 2024-11-19 10:47:31 |
|       1 |         7 | 2024-11-19 14:16:48 |
|       1 |         8 | 2024-11-19 14:02:28 |
|       1 |         9 | 2024-11-19 14:02:24 |
+---------+-----------+---------------------+
4 rows in set (0.00 sec)

mysql>
```

Update option

```
mysql> UPDATE courses
    -> SET description = 'Learn the fundamentals of web development and build interactive websites with ease.'
    -> WHERE id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from courses;
+----+---------------------------+-----------------------------------------------------------------------------------+---------------+
| id | title                     | description                                                                       | instructor_id |
+----+---------------------------+-----------------------------------------------------------------------------------+---------------+
|  1 | Web Development Course    | Learn the fundamentals of web development and build interactive websites with ease. |             3 |
|  2 | Social Services Engineering | This course is not for the weak                                                 |             3 |
|  3 | DBMS                      | This is my favourite subject                                                      |             3 |
|  4 | SDS                       | This is one of the best courses for your future                                   |             3 |
|  5 | Maths                     | Leap into the world of arithmetics and logistic regression (Math for Machine Learning) |             3 |
|  7 | Chemistry                 | Have fun with the chemical world                                                  |             3 |
|  8 | Python                    | Mega Python Course                                                                |            14 |
|  9 | Science                   | We love Science                                                                   |             3 |
+----+---------------------------+-----------------------------------------------------------------------------------+---------------+
8 rows in set (0.00 sec)
```

Delete operation

```
DELETE FROM courses
WHERE title = 'Chemistry';
```