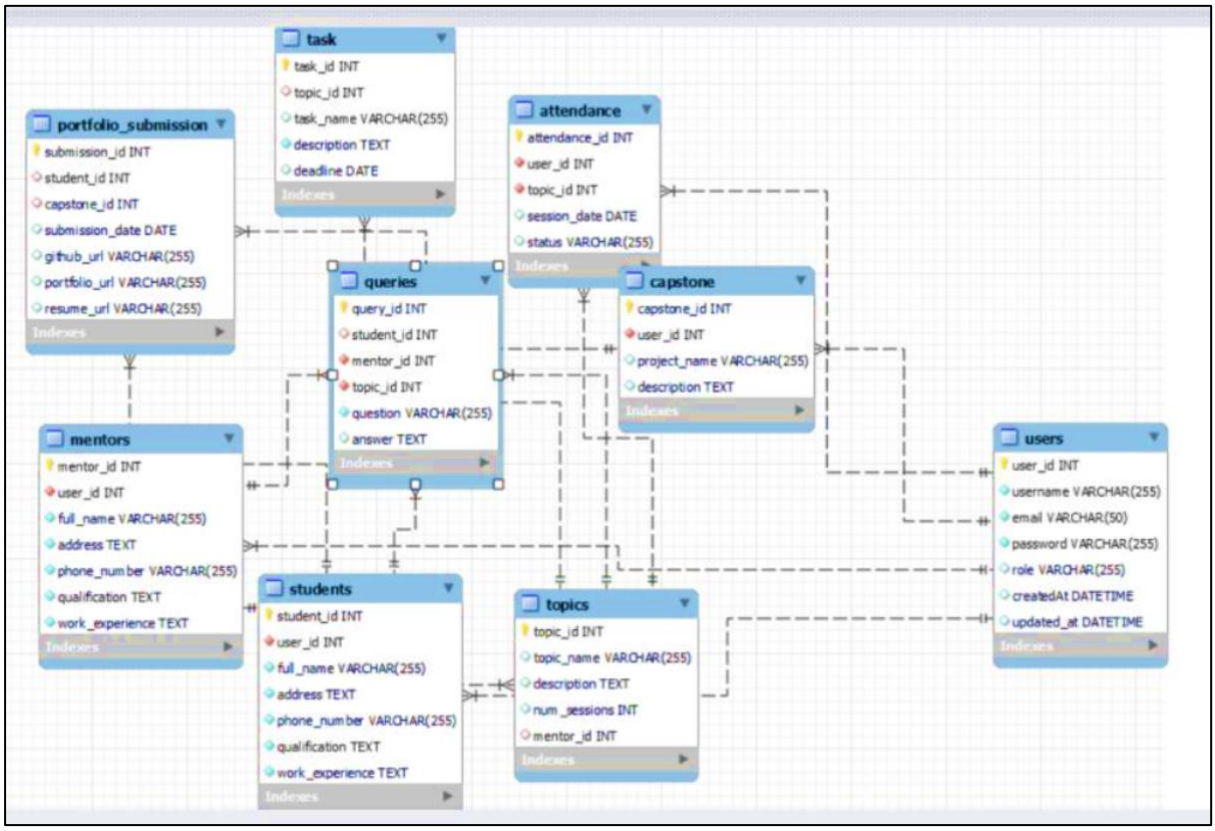
Design DB model for Guvi Zen class



The Guvi Zen class management system

- Users: Records information about registered users, including their username, email, password, role, and creation/update timestamps.
- Students: Stores details of enrolled students such as their full name, address, phone number, qualification, and work experience.
- Mentors: Holds information about mentors guiding students, including their full name, address, phone number, qualification, and work experience.
- Topics: Defines subjects or areas of focus within the curriculum, with attributes like topic name, description, and the mentor responsible for each topic.
- Tasks: Specifies assignments or tasks assigned to students within a particular topic, including details such as task name, description, and deadline.
- Attendance: Tracks the attendance of users in topic sessions, recording session dates and attendance status.
- Capstones: Represents comprehensive projects undertaken by students, capturing project name, description, and student details.

- Queries: Facilitates communication between students and mentors, storing questions and corresponding answers along with topic and user details.
- Portfolio Submissions: Stores submissions of student portfolios and capstone projects, including submission dates and URLs for GitHub, portfolio, and resume.

Code:

```
-- Database Creation:

CREATE DATABASE IF NOT EXISTS GuviDB;


-- Tables Creation :

USE GuviDB;

CREATE TABLE users (
    user_id INT PRIMARY KEY,
    username VARCHAR(255) NOT NULL,
    email VARCHAR(50) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    role ENUM('student', 'mentor') NOT NULL,
    createdAt DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at DATETIME DEFAULT NULL
);




CREATE TABLE students (
    student_id INT PRIMARY KEY,
    user_id INT NOT NULL,
```

```
    full_name VARCHAR(255) NOT NULL,
    address TEXT NOT NULL,
    phone_number VARCHAR(20) NOT NULL,
    qualification TEXT NOT NULL,
    work_experience TEXT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

CREATE TABLE mentors (
    mentor_id INT PRIMARY KEY,
    user_id INT NOT NULL,
    full_name VARCHAR(255) NOT NULL,
    address TEXT NOT NULL,
    phone_number VARCHAR(20) NOT NULL,
    qualification TEXT NOT NULL,
    work_experience TEXT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);



CREATE TABLE topics (
    topic_id INT PRIMARY KEY,
    topic_name VARCHAR(255),
    description TEXT,
    num_sessions INT,
```

```
    mentor_id INT,
    FOREIGN KEY (mentor_id) REFERENCES mentors(mentor_id)
);
CREATE TABLE tasks (
    task_id INT PRIMARY KEY,
    topic_id INT,
    task_name VARCHAR(255),
    description TEXT NOT NULL,
    deadline DATE,
    FOREIGN KEY (topic_id) REFERENCES topics(topic_id)
);
CREATE TABLE attendance (
    attendance_id INT PRIMARY KEY,
    user_id INT NOT NULL,
    topic_id INT NOT NULL,
    session_date DATE,
    status ENUM('present', 'absent') NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    FOREIGN KEY (topic_id) REFERENCES topics(topic_id) );
CREATE TABLE capstones (
    capstone_id INT PRIMARY KEY,
    user_id INT NOT NULL,
    project_name VARCHAR(255),
    description TEXT,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

```
CREATE TABLE queries (
    query_id INT PRIMARY KEY,
    student_id INT,
    mentor_id INT,
    topic_id INT,
    question VARCHAR(255) NOT NULL,
    answer TEXT,
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (mentor_id) REFERENCES mentors(mentor_id),
    FOREIGN KEY (topic_id) REFERENCES topics(topic_id)
);
```

```
CREATE TABLE portfolio_submissions (
    submission_id INT PRIMARY KEY,
    student_id INT,
    capstone_id INT,
    submission_date DATE,
    github_url VARCHAR(255),
    portfolio_url VARCHAR(255),
    resume_url VARCHAR(255),
```

```
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (capstone_id) REFERENCES capstones(capstone_id)
);
```

Task Completed …!