# LABORATORY EXERCISE 2

# DATABASE DESIGN AND MIGRATION

**Learning Objectives**

By the end of this laboratory exercise, students should be able to:

* Design a database schema for a Learning Management System (LMS).
* Identify and define core database tables and relationships.
* Implement database migrations using CodeIgniter’s migration feature.
* Seed the database with sample data for testing purposes.
* Use GitHub for version control and tracking database changes.

**Prerequisite student experiences and knowledge**

Before starting this exercise, students should have:

* Basic knowledge of relational database concepts.
* Familiarity with MySQL database creation and management.
* Understanding of CodeIgniter framework basics.
* Experience using Git and GitHub for project version control.
* Ability to run and configure a local web server (XAMPP/WAMP/LAMP).

**Background**

This laboratory focuses on creating the database backbone for the LMS project. The process includes identifying core tables, defining their structure, setting up migrations for automated schema creation, and seeding data for testing. Using migrations ensures version-controlled database changes that can be easily shared among team members through GitHub.

**Materials/Resources**

* **Personal Computer with Internet Access**
* **XAMPP/WAMP/LAMP server installed**
* **CodeIgniter Framework (latest version)**
* **Bootstrap (via CDN or downloaded files)**
* **Visual Studio Code or any code editor**
* **Git and GitHub Account**
* **Web Browser (Chrome, Firefox, etc.)**

**Laboratory Activity**

**Step 1: Identify Core Tables**

List the primary tables for the LMS:

1. users – stores student, instructor, and admin information.
2. courses – contains course details.
3. enrollments – manages student enrollment in courses.
4. lessons – stores lesson content linked to courses.
5. quizzes – contains quiz questions linked to lessons.
6. submissions – stores quiz submissions and results.

**Step 2: Create Database**

1. Open phpMyAdmin.
2. Create a new database named: lms\_lastname
3. Do not create tables manually — migrations will handle this.

**Step 3: Enable CodeIgniter Migrations**

1. In `application/config/migration.php`, set:

$config['migration\_enabled'] = TRUE;

$config['migration\_type'] = 'sequential';

1. In `application/config/config.php`, set your database connection details.

**Step 4: Create Migration Files**

1. In the terminal, navigate to your project folder.
2. Create migration files for each table:

php spark make:migration CreateUsersTable

php spark make:migration CreateCoursesTable

php spark make:migration CreateEnrollmentsTable

php spark make:migration CreateLessonsTable

php spark make:migration CreateQuizzesTable

php spark make:migration CreateSubmissionsTable

1. Define the schema in each migration file.

**Step 5: Run Migrations**

Run the following command:

* php spark migrate

**Step 6: Seed Sample Data**

1. Create a seeder: **php spark make:seeder UserSeeder**
2. Add sample users (students, instructors, admin).
3. Run the seeder: **php spark db:seed UserSeeder**

**Step 7: Push to GitHub**

1. Stage and commit your migration and seeder files:

**git add .**

**git commit -m "Added database migrations and seeders"**

**git push**

Output / Results

* Screenshot of successful migration in phpMyAdmin.
* Screenshot of seeded sample data in phpMyAdmin.
* Screenshot of GitHub repository with migration files.

**QUESTIONS:**

1. Why are migrations important in database development?

* Mgs, particularly if more than one developer is working on the same appligrations are useful in database construction since they offer a structured means of handling changes to the database schema. With applications expanding, developers will have to add new tables, change columns, or change data relations. With no migrations, all these changes will have to be performed manually, and this can result in inconsistencies and buication

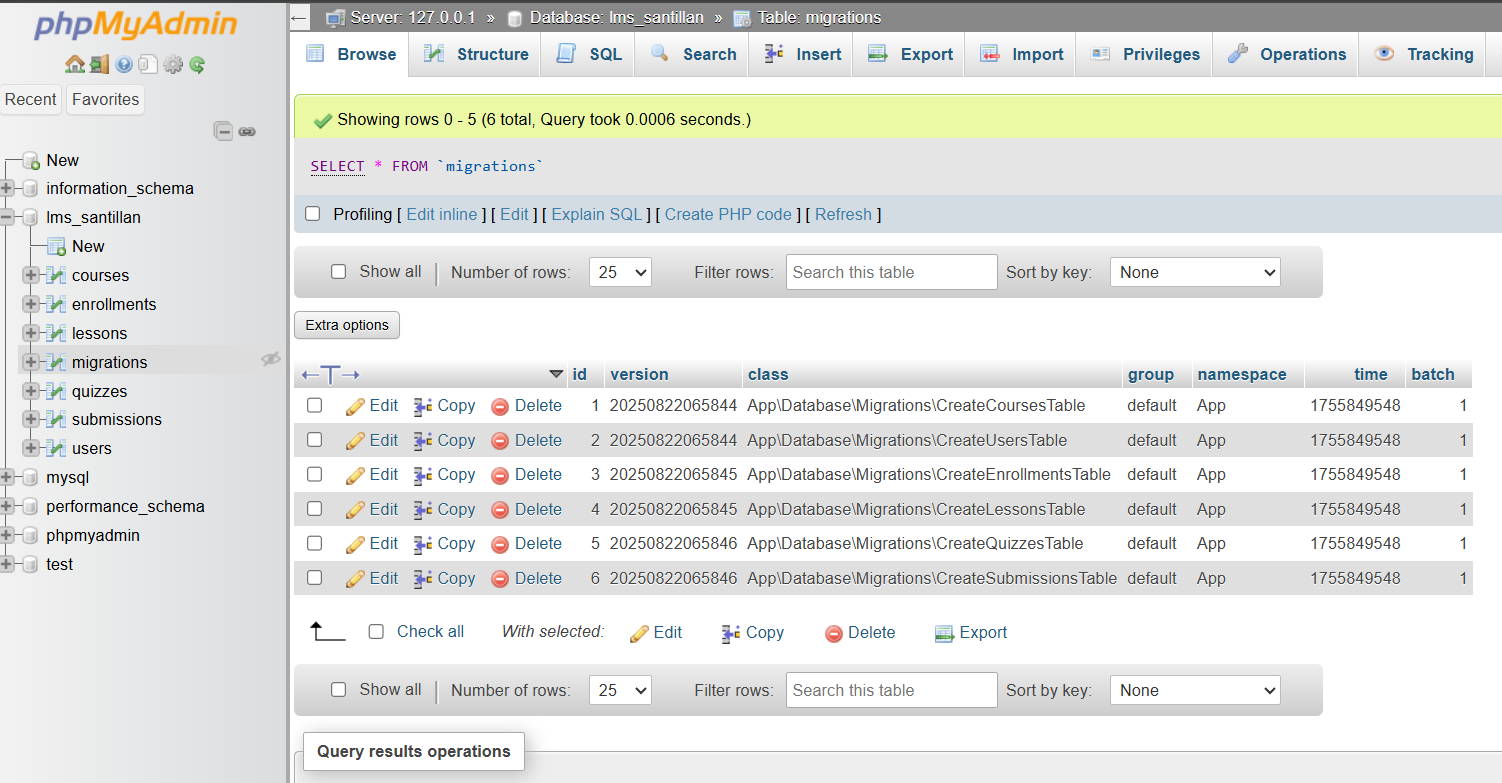
1. How does using GitHub with migrations help in team projects?

* Working with GitHub using migrations significantly enhances teamwork in group projects as all developers work with the same database schema. In big applications, the database tends to be constantly revised in terms of adding new tables, changing relationships, or resizing columns. When such modifications are monitored through migrations and posted on GitHub, all team members can simply pull the latest changes and apply them to their local databases. This avoids mistakes that come when another developer's database is different from yours.

1. What is the advantage of seeding sample data during development?

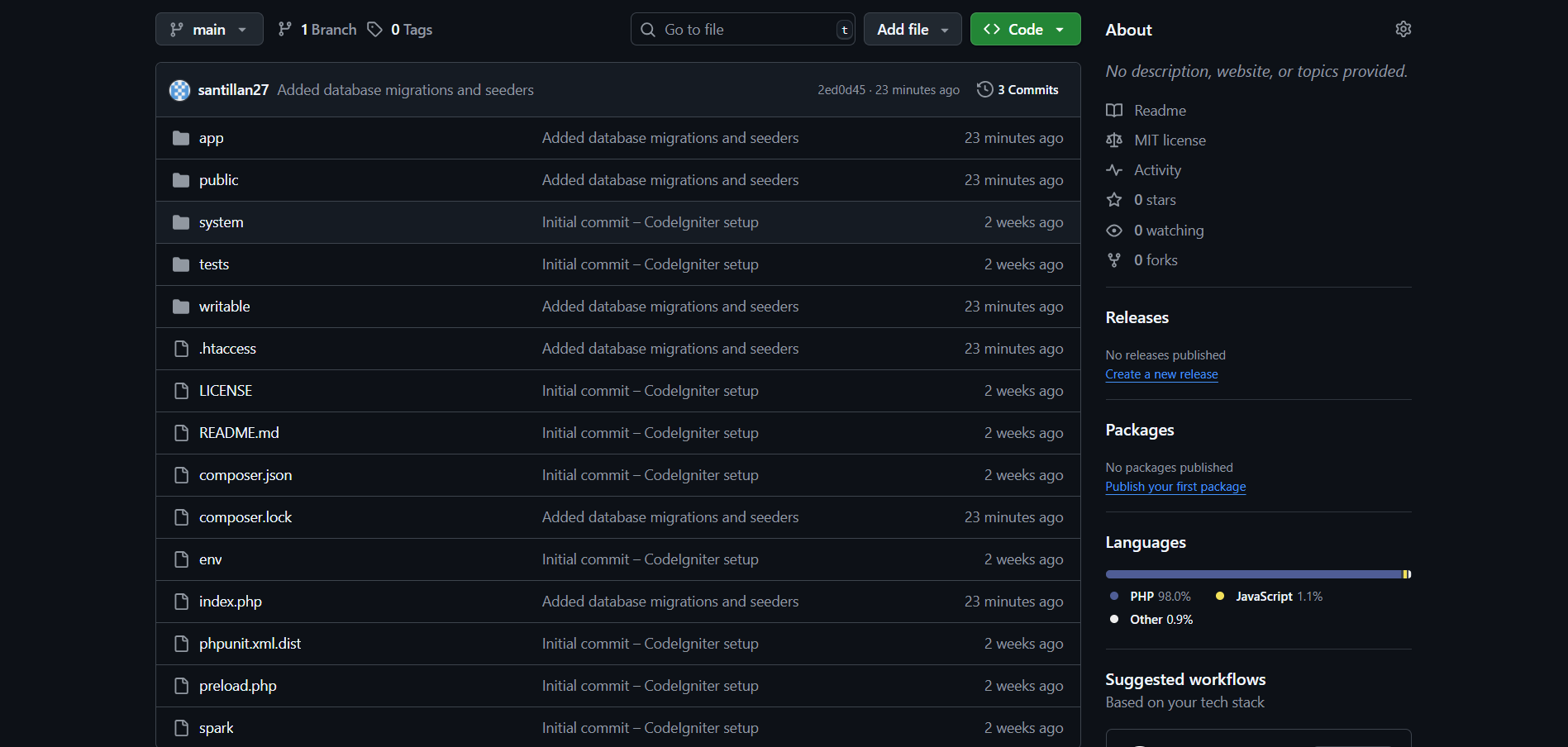
* The benefit of seeding sample data in development is that developers have access to a realistic dataset with which they can develop and test an application. Without sample data, developers would have to manually add records to the database every time, which is time-consuming and error-prone. With seeders, useful test data—users, products, or transactions—can be automatically created and consistently applied in all development environments.

**Output / Results**



A screenshot of a computer

AI-generated content may be incorrect.



**Conclusion**

migration is not simply a matter of individuals relocating from one location to another, it is a dynamic force that reshapes societies, economies, and cultures. Though it has challenges, including social integration, cultural diversity, and economic stress, it also brings opportunities for development, diversity, and innovation. Migrants also enrich their host and home nations through work, knowledge, and cultural transfer, demonstrating that mobility has long been at the heart of development. In the end, migration must be regarded not only as a phenomenon to be controlled but as a reality to be comprehended and accepted with empathy, equity, and responsibility. By solving its challenges and optimizing its gains, migration can be a bridge to more resilient, integrated societies in a more interconnected world.