**Student Management System**

A **Student Management System (SMS)** is a software application that allows educational institutions to manage various processes related to students, courses, tasks, enrollment, and teachers. It helps organize and streamline the tasks that take place in an academic environment.

Here's an overview of the key components in the system (student, course, task, enrollment, and teacher) and how these entities interact with one another in sequence. This documentation will explain how each part works within the system.

### 1. **Student Management System Overview**

* **Purpose:** To manage student-related data, including personal details, courses enrolled in, assigned tasks, and grades.
* **Modules:**
  1. **Student Module**
  2. **Course Module**
  3. **Task Module**
  4. **Enrollment Module**
  5. **Teacher Module**

### 2. **Modules and Functions:**

#### A. **Student Module:**

* **Purpose:** To manage student records, including personal details (name, email, contact information, etc.), enrollment status, and their progress.
* **Key Operations:**
  + **Add Student:** Allow the admin or authorized user to enter student details into the system.
  + **View Student Details:** Show the student’s personal information, courses enrolled in, grades, and tasks assigned.
  + **Update Student Information:** Edit student details (e.g., contact info, grade changes).
  + **Delete Student:** Remove a student's record from the system (e.g., in case of dropout or transfer).
* **Example Workflow:**
  1. Admin logs in.
  2. Admin adds a new student with name, email, contact, etc.
  3. The student is now available in the system for further actions.

#### B. **Course Module:**

* **Purpose:** Manages all available courses, including course details (name, description, credits, etc.), and ensures students can enroll in them.
* **Key Operations:**
  + **Add Course:** Admin adds a new course to the system.
  + **View Course List:** Display a list of available courses for students and teachers.
  + **Update Course Information:** Modify course details like schedule, description, or credits.
  + **Delete Course:** Remove a course from the system if no longer offered.
* **Example Workflow:**
  1. Admin adds a course with details such as course name, code, and description.
  2. The course becomes available for students to enroll in.

#### C. **Task Module:**

**Purpose:** Assign and track tasks, assignments, exams, and other academic activities.

**Key Operations:**

* + **Add Task/Assignment:** Teachers can add assignments or tasks for students to complete within a given timeframe.
  + **View Task List:** Students and teachers can see all the tasks assigned.
  + **Update Task:** Teachers can update or change task requirements or deadlines.
  + **Submit Task:** Students can submit completed tasks.

**Example Workflow:**

* 1. A teacher adds a new assignment or task for a course.
  2. Students are notified, and they can submit their work.
  3. Teachers grade the submitted work and enter the grades into the system.

#### D. **Enrollment Module:**

* **Purpose:** Manages the process of students enrolling in courses, updating enrollment status, and keeping track of active students.
* **Key Operations:**
  + **Enroll Student in Course:** Students can select courses they want to enroll in, based on availability and prerequisites.
  + **View Enrollment:** Shows which courses a student is enrolled in.
  + **Update Enrollment Status:** Change the enrollment status (e.g., drop a course, add a new course).

#### E. **Teacher Module:**

**Purpose:** Manages teacher-related data, including personal details and their assigned courses.

**Key Operations:**

* + **Add Teacher:** Admin adds new teachers to the system with details such as name, contact info, and assigned courses.
  + **Assign Teacher to Course:** Teachers are linked to the courses they will teach.
  + **View Teacher’s Assigned Courses:** Displays which courses a teacher is responsible for.
  + **Grade Assignments/Tasks:** Teachers are responsible for grading students' work.

### 3. **How Modules Work Together:**

#### Sequence of Operations:

**Student enrolls in a course (Enrollment Module).**

* 1. The student selects a course they want to join, and their enrollment is recorded in the system.
  2. Enrollment status is updated in the system.

**Teacher is assigned to the course (Teacher Module).**

* 1. The teacher is linked to the specific course they will teach.

**Teacher creates tasks or assignments for the course (Task Module).**

* 1. The teacher assigns tasks for students in the course. This could include homework, projects, quizzes, etc.

**Students complete tasks (Task Module).**

* 1. Students submit their completed tasks/assignments, which are then reviewed by the teacher.

**Teacher grades tasks (Task Module).**

* 1. Teachers assign grades to the completed tasks, which are updated in the student’s record.

### 4. **Data Flow & Interaction:**

* **Student → Enrollment → Course:** A student enrolls in one or more courses.
* **Teacher → Course:** Teachers are assigned to courses and create tasks.
* **Course → Task → Student:** Teachers create tasks for students in the course. Students submit these tasks.
* **Task → Teacher → Grade:** Teachers grade tasks submitted by students, and grades are stored in the student’s record.
* **Enrollment Status → Course → Reports:** Admin can generate reports to track course enrollments, student progress, and teacher assignments.

### 5. **Database Schema Example:**

To implement the system, a relational database schema could look like this:

#### Tables:

* **Students** (student\_id, name, email, contact\_number, dob,address)
* **Courses** (course\_id, course\_name, description, description,)
* **Teachers** (teacher\_id, name, email)
* **Tasks** (task\_id, course\_id, task\_name, due\_date, )
* **Enrollments** (enrollment\_id, student\_id, course\_id, enrollment\_date)
* **Task\_Submissions** (student\_id, teacher\_id, title, description, due\_date,status, created\_at,updated\_at )

#### Relationships:

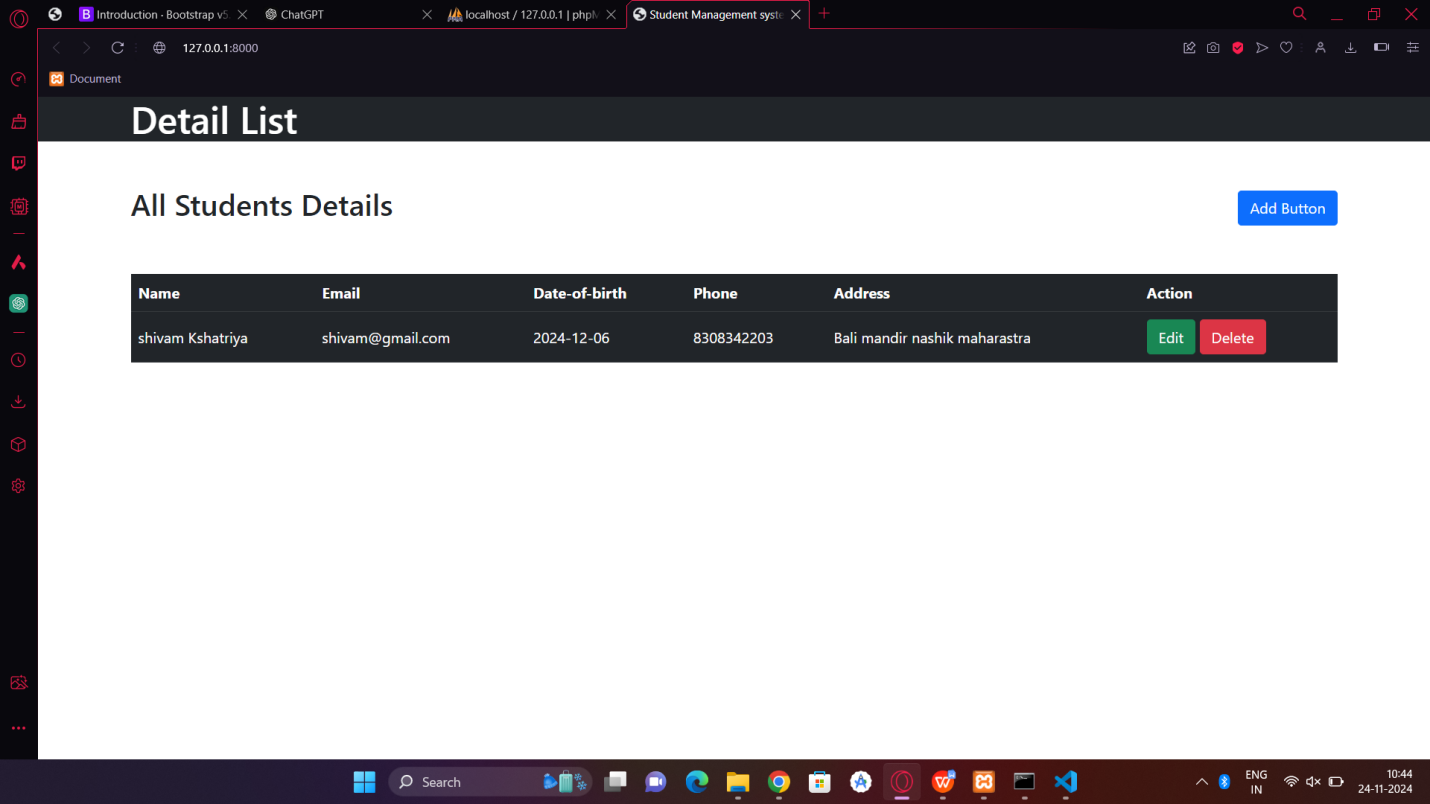
* **Student → Enrollment → Course:** A student can be enrolled in multiple courses.
* **Teacher → Course:** A teacher is assigned to one or more courses.
* **Course → Task → Student:** Tasks are linked to a course and assigned to students in that course.

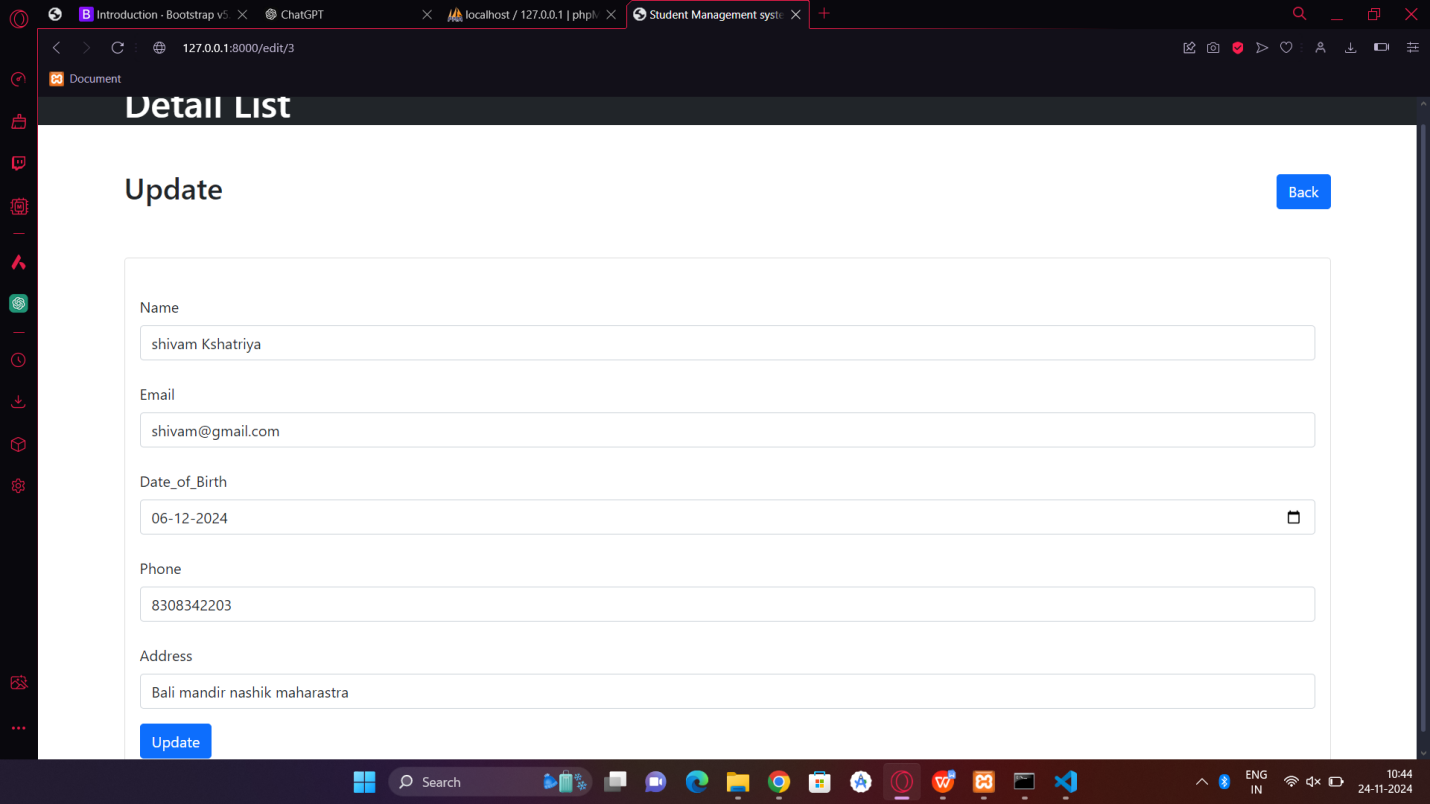
### 6. **Possible Technologies:**

* **Frontend:** HTML, CSS, bootstrap.
* **Backend:** Php , Laravel.
* **Database:** MySQL.

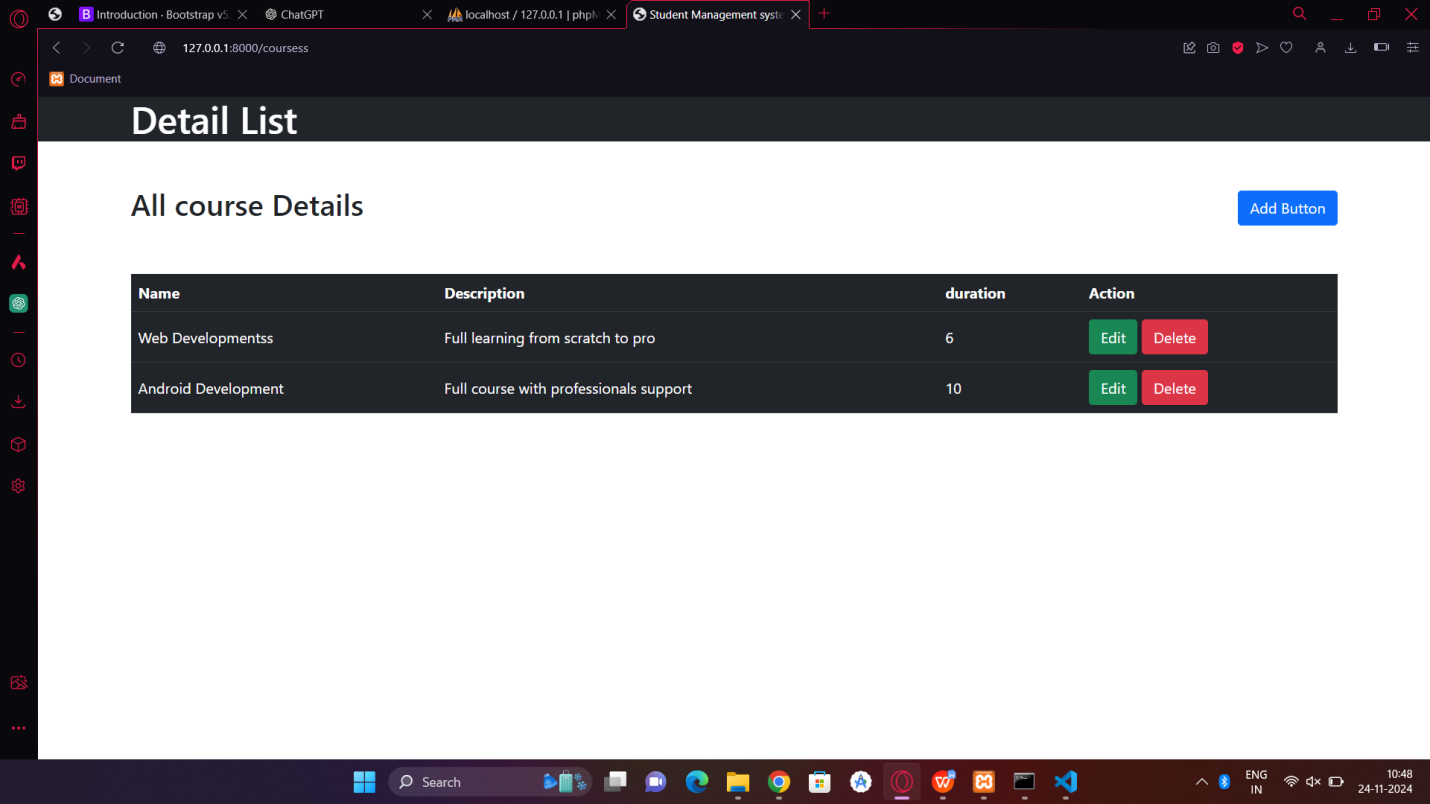
1. **How to Setup:**
2. Here we have to run the command PHP artisan migrate ,to creating the tables that are made in project.
3. Add the value to each database using the forms that are design for each main modules like students,courses and task.
4. All the modules are not integrated on the single dashboard sidebar ,so to run each them by using the home page of that module
5. To reuse these code you can simply add the extra text filed input and one more column in that database.
6. Make sure that You want to change the design use the bootstrap version 5.0.2 so that it can easy to apply the changes.
7. And also use the Laravel version 10.0.3 and PHP version 8.1.25 which is stable to recreate it.
8. **Screenshots:**

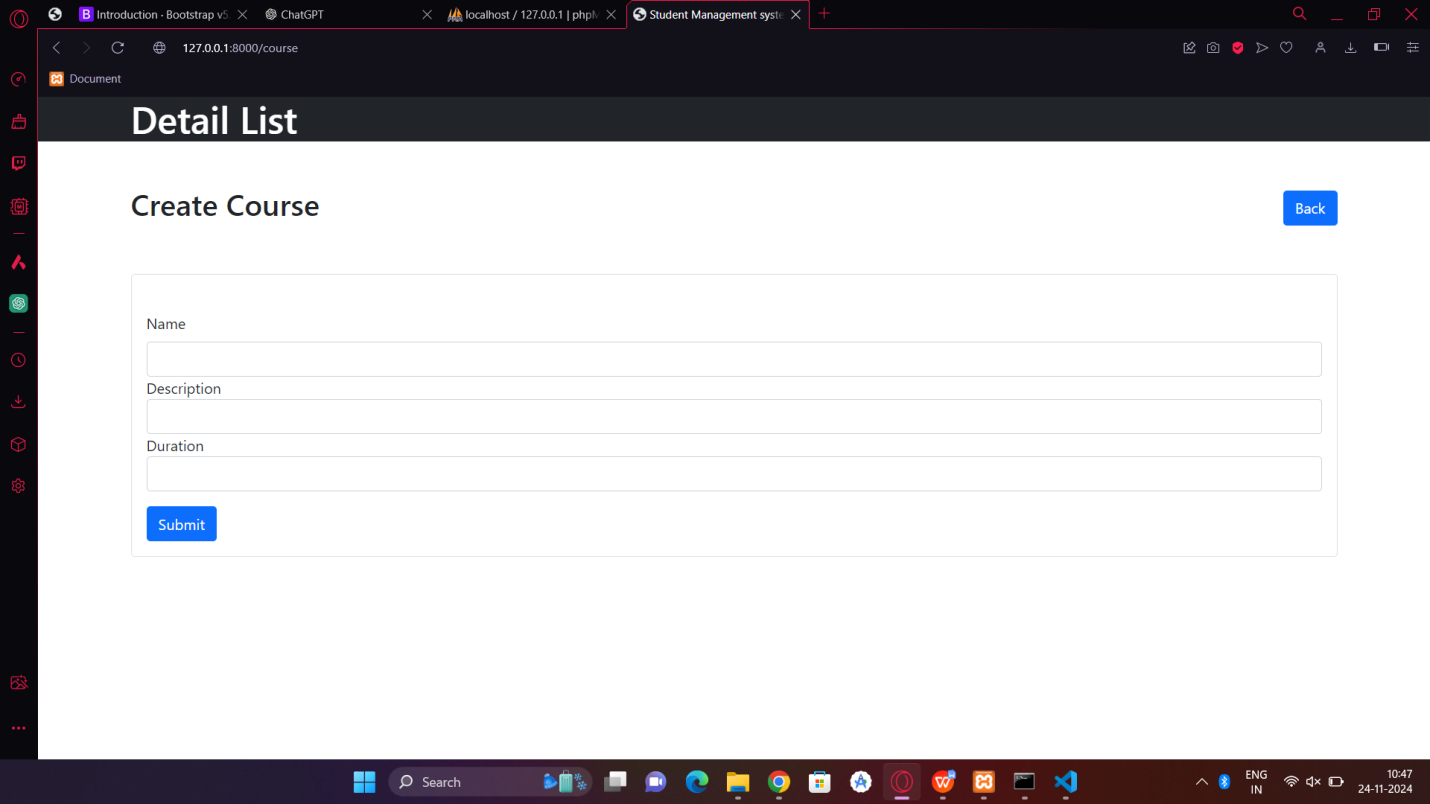
**1.students**





1. **courses:**

****

****