Project Management System

Objective

The objective of this project is to develop a Project Management System using Laravel. The system will allow users to register, log in, create projects, manage tasks under projects, update task statuses, and generate task reports. The system will be implemented as RESTful APIs with JWT authentication.

Technology Stack

- **Backend Framework:** Laravel
- **Database:** MySQL (using XAMPP)
- **Authentication:** JWT Token-based authentication
- **API Testing Tool:** Postman
- **Version Control:** GitHub

Features

1. User Management

- Users can register using an email and password.
- Users can log in and receive a JWT token for authentication.
- Users can only access their own projects and tasks.

2. Project Management

- Users can create, update, and delete projects.
- Each project is private to the user who created it.

3. Task Management

- Users can create tasks under their projects.
- Users can update the task status (Pending, In Progress, Completed).
- Users can add daily remarks for each task.

4. Task Reports

- Users can fetch project reports showing:

- Task details
- Status history
- Daily remarks

Database Design

Tables:

1. **Users Table**
   - id (Primary Key)
   - name
   - email (Unique Index)
   - password
   - created_at
   - updated_at
2. **Projects Table**
   - id (Primary Key)
   - user_id (Foreign Key: users)
   - name
   - description
   - created_at
   - updated_at
3. **Tasks Table**
   - id (Primary Key)
   - project_id (Foreign Key: projects)
   - name
   - status (ENUM: Pending, In Progress, Completed)
   - created_at
   - updated_at
4. **Task Remarks Table**
   - id (Primary Key)
   - task_id (Foreign Key: tasks)
   - remark
   - created_at
   - updated_at

API Endpoints

1. Authentication

Register a new user

- **POST - http://localhost:8000/api/register**

  Request Body:

  {

  "name": "ABC",

  "email": "abc@example.com",

  "password": "password123"

  }

  Response:

  {

  "message": "User registered successfully",

  }

- **POST - http://localhost:8000/api/login** - Authenticate and generate JWT token

  Request Body:

  {

  "email": "abc@example.com",

  "password": "password123"

  }

  Response:

  {

  "message": "Login successful",

  }

2. Project Management

2.1 .POST - **/api/projects** - Create a new project

Request Body:

```
{
    "title": "Website Development",
    "description": "Building a website for a client"
}
```

Response:

```
{
"id": 1,
    "title": "Website Development",
    "description": "Building a website for a client",
    "user_id": 1
}
```

2.2 .GET - **/api/projects** **-** Get All Projects of Logged-in User

Response:

```
 {
        "id": 1,
        "title": "Website Development",
        "description": "Building a website for a client",
        "user_id": 1
    }
```

2.3 .PUT - **/api/projects/1** - Update a Project

Request Body (JSON):

```
{
   "title": "Updated Project Title",
   "description": "Updated description"
}
```

Response (200 OK):

```
{
   "id": 1,
   "title": "Updated Project Title",
   "description": "Updated description",
   "user_id": 1
}
```

2.4 .DELETE - /api/projects/1 - Delete a Project

Response (200 OK):

```
{
   "message": "Project deleted"
}
```

3. Task Management

- Tasks are linked to a project, so make sure you create a project first.
- Use the JWT token in the header for all requests.

3.1 Create a Task Under a Project

POST /api/projects/1/tasks

Request Body (JSON):

```
{
  "title": "Design Homepage",
  "description": "Create the homepage UI",
  "status": "Pending"
}
```

Response (201 Created):

```
{
  "id": 1,
  "title": "Design Homepage",
  "description": "Create the homepage UI",
  "status": "Pending",
  "project_id": 1,
  "user_id": 1
}
```

3.2 Get All Tasks Under a Project

GET - /api/projects/1/tasks

Response (200 OK):

```
[
  {
    "id": 1,
    "title": "Design Homepage",
    "description": "Create the homepage UI",
    "status": "Pending",
```

```
    "project_id": 1,
    "user_id": 1
}
```


3.3 Update a Task

PUT /api/tasks/1

Request Body (JSON):


```
{
  "title": "Update Homepage Design",
  "description": "Make the homepage more responsive",
  "status": "In Progress"
}
```


Response (200 OK):


```
{
  "id": 1,
  "title": "Update Homepage Design",
  "description": "Make the homepage more responsive",
  "status": "In Progress",
  "project_id": 1,
  "user_id": 1
}
```


3.4 Delete a Task

DELETE /api/tasks/1

Response (200 OK):

```
{
    "message": "Task deleted"
}
```

4. Logout API

POST /api/logout

Response:

```
{
    "message": "Successfully logged out"
}
```

5. Task Management

POST /api/tasks/1/remarks

Request Body (JSON):

```
{
    "remark": "Worked on UI fixes",
    "date": "2025-03-08"
}
```

6. Fetch Project Report

Users can fetch project reports showing:

- Task details
- Status history
- Daily remarks

GET /api/projects/1/report

Response:


```
{
  "project_id": 1,
  "tasks": [
    {
      "id": 1,
      "name": "Build API",
      "status": "In Progress",
      "remarks": [
        {
          "id": 1,
          "remark": "Worked on UI fixes",
          "date": "2025-03-08"
        },
        {
          "id": 2,
          "remark": "Completed API testing",
          "date": "2025-03-09"
        }
      ]
    }
  ]
```

}

7. Task Management

- Users can update the task status (Pending, In Progress, Completed).
- Users can add daily remarks for each task.

POST /api/tasks/1/status

```
{
    "status": "Completed"
}
```