# Project Management System API Documentation

## Authentication APIs

All authentication endpoints are publicly accessible. JWT token is required for accessing protected routes.

### Register

**Endpoint:** `POST /api/register`

**Description:** Register a new user.

#### Request Body

```
{
  "name": "John Doe",
  "email": "john@example.com",
  "password": "password123"
}
```

#### Success Response

**Status:** `200 OK`

```
{
  "status": true,
  "message": "User registered successfully",
  "data": {
    "user": {
      "id": 1,
      "name": "John Doe",
      "email": "john@example.com"
    },
    "token": "JWT_TOKEN"
  }}
```

**Error Response**

**Status:** `422 Unprocessable Entity` (Validation Errors)

```
{
  "status": false,
  "message": "Validation error",
  "errors": {
    "email": ["The email has already been taken."]
  }
}
```

## Login

**Endpoint:** `POST /api/login`

**Description:** Login user and get JWT token.

**Request Body**

```
{
  "email": "john@example.com",
  "password": "password123"
}
```

**Success Response**

**Status:** `200 OK`

```
{
  "status": true,
  "message": "Login successful",
  "data": {
    "token": "JWT_TOKEN"
  }}
```

**Error Response**

**Status:** `401 Unauthorized`

```
{
  "status": false,
  "message": "Invalid credentials"
}
```

### Get Authenticated User

**Endpoint:** `GET /api/user`
**Auth Required:** Yes (JWT Token)

**Headers:**

```
Authorization: Bearer {JWT_TOKEN}
```

**Success Response**

**Status:** `200 OK`

```
{
  "status": true,
  "data": {
    "id": 1,
    "name": "John Doe",
    "email": "john@example.com"
  }
}
```

**Error Response**

**Status:**

```
401 Unauthorized
```

```
{
  "status": false,
  "message": "Unauthorized"
}
```

## JWT Error Responses

| Scenario | HTTP Status | Message |
|---|---|---|
| Missing Token | 401 | Token is missing |
| Invalid Token | 401 | Token is invalid |
| Expired Token | 401 | Token has expired |
| Unauthenticated User | 401 | Unauthorized |

### List Projects

**Endpoint:** `GET /api/projects`
**Description:** Returns all projects created by the authenticated user.

**Success Response**

```
{
  "status": true,
  "message": "Projects fetched successfully",
  "data": [
    {
      "id": 1,
      "user_id": 1,
      "name": "Project A",
      "description": "Description here",
      "created_at": "...",
      "updated_at": "..."
    },
    ...
  ]
}
```

## Create Project

**Endpoint:** `POST /api/projects/store`
**Description:** Creates a new project under the authenticated user.

**Request Body**

```
{
  "name": "New Project",
  "description": "Optional description"
}
```

**Success Response**

```
{
  "status": true,
  "message": "Project created successfully",
  "data": {
    "id": 1,
    "user_id": 1,
    "name": "New Project",
    "description": "Optional description",
    "created_at": "...",
    "updated_at": "..."
  }
}
```

**Validation Error**

```
{
  "status": false,
  "message": "Validation failed",
  "errors": {
    "name": ["The name field is required."]
  }
}
```

## Update Project

**Endpoint:** `PUT /api/projects/update/{id}`
**Description:** Updates a project belonging to the authenticated user.

**Request Body**

```
{
  "name": "Updated Project",
  "description": "Updated description"
}
```

**Success Response**

```
{
  "status": true,
  "message": "Project updated successfully",
  "data": {
    "id": 1,
    "user_id": 1,
    "name": "Updated Project",
    "description": "Updated description",
    "created_at": "...",
    "updated_at": "..."
  }
}
```

**Not Found / Unauthorized**

```
{
  "status": false,
  "message": "Project not found or unauthorized"
}
```

## Delete Project

**Endpoint:** `DELETE /api/projects/delete/{id}`
**Description:** Deletes a specific project that belongs to the authenticated user.

**Success Response**

```
{
  "status": true,
  "message": "Project deleted successfully"
}
```

**Not Found / Unauthorized**

```
{
  "status": false,
  "message": "Project not found or unauthorized"
}
```

## Project Table Schema

The `projects` table structure:

| Column | Type | Description |
|---|---|---|
| `id` | BIGINT | Primary key (auto-increment) |
| `user_id` | BIGINT | Foreign key → `users.id` |
| `name` | STRING | Project name |
| `description` | TEXT | Project description (nullable) |
| `created_at` | TIMESTAMP | Record creation timestamp |
| `updated_at` | TIMESTAMP | Record update timestamp |

**Foreign Key Constraint:**

- `user_id` references `users(id)` with `onDelete('cascade')`

### Create a Task

**URL**: `POST /api/projects/tasks/{projectId}`
**Auth**: Required
**Params (Path)**:

- `projectId` — ID of the project the task belongs to.

**Body (JSON)**:

```
{
  "title": "Task Title",
  "description": "Optional description"
}
```

**Response** `201 Created`

```
{
  "status": true,
  "message": "Task created successfully",
  "data": {
    "id": 1,
    "project_id": 5,
    "title": "Task Title",
    "description": "Optional description",
    "status": "Pending",
    "created_at": "...",
    "updated_at": "..."
  }
}
```

### Update a Task

**URL**: `PUT /api/tasks/update/{id}`
**Auth**: Required
**Params (Path)**:

- `id` — ID of the task to update.

**Body (JSON)**:

```
{
  "title": "Updated Task Title",
  "description": "Updated description",
  "status": "In Progress"  // Optional - must be "Pending", "In
Progress", or "Completed"
}
```

**Response** `200 OK`

```
{
  "status": true,
  "message": "Task updated successfully",
  "data": {
    ...
  }
}
```

### Delete a Task

**URL**: `DELETE /api/tasks/delete/{id}`
**Auth**: Required
**Params (Path)**:

- `id` — ID of the task to delete.

**Response** `200 OK`

```
{
  "status": true,
  "message": "Task deleted successfully"
}
```

### Add Task Remark & Update Status

**URL**: `POST /api/tasks/remark/{id}`
**Auth**: Required
**Params (Path)**:

- `id` — ID of the task to remark on.

**Body (JSON)**:

```
{
  "status": "Completed",  // Required - must be "Pending", "In
Progress", or "Completed"
```

```
  "remark": "This is the final update"
}
```

**Response** `200 OK`

```
{
  "status": true,
  "message": "Status updated with remark successfully"
}
```

# Project Report

### Get Project Report with Tasks & Remarks

**URL**: `GET /api/projects/report/{id}`
**Auth**: Required
**Params (Path)**:

- `id` — ID of the project.

**Response** `200 OK`

```
{
  "status": true,
  "message": "Project details fetched successfully",
  "data": {
    "project": {
      "id": 1,
      "name": "Project Name",
      "description": "Project Description"
    },
    "tasks": [
      {
        "task_id": 2,
        "title": "Task 1",
        "description": "Details...",
        "status": "Completed",
        "remarks": [
          {
```

```
          "status": "In Progress",
          "remark": "Started working",
          "date": "2024-04-01"
        },
        {
          "status": "Completed",
          "remark": "Finished the work",
          "date": "2024-04-02"
        }
      ]
    }
  ]
 }
}
```