API Documentation

Date: 06-18-2024

Author: Team ProManager

Table of Contents

Project Management API

1. Get All Projects

Title: Get All Projects
Description: Retrieve all projects from the database.
Method: GET
Endpoint: /apiv1/project/getallprojects
Request Headers: Content-Type: application/json
Response Codes:

- 200 OK: Successfully retrieved all projects.
- 500 Internal Server Error: Failed to retrieve projects.

Response Body:


```
{
  "projects": [
    {
      "project_id": "1",
      "projectname": "Create Educational Platform",
      "owner_id": 1,
      "active": true,
      "description": "An educational platform for online learning.",
      "created_on": "2023-01-01T00:00:00.000Z",
      "updated_on": "2023-01-01T00:00:00.000Z",
      "status": "OPEN",
      "type": "Platform"
    }
  ]
}
```


2. **Get Project by ID**

Title: Get Project by ID
Description: Retrieve a specific project by its ID.
Method: GET
Endpoint: /apiv1/project/getIdWiseProject/{id}
Request Headers: Content-Type: application/json
Path Parameters: id (String): The ID of the project to retrieve.
Response Codes:

- 200 OK: Successfully retrieved the project.
- 404 Not Found: Project not found.
- 500 Internal Server Error: Failed to retrieve the project.

Response Body:

```
{
  "project": [
    {
      "project_id": "1",
      "projectname": "Create Educational Platform",
      "name": "Pranjal",
      "active": true,
      "description": "An educational platform for online learning.",
      "created_on": "2023-01-01T00:00:00.000Z",
      "updated_on": "2023-01-01T00:00:00.000Z",
      "status": "OPEN",
      "type": "Platform"
    }
  ]
}
```


3. Add Project

Title: Add Project
Description: Add a new project to the database.
Method: POST
Endpoint: /apiv1/project/addprojects/
Request Headers: Content-Type: application/json
Request Body:


```
{
  "projects": [
    {
      "project_id": "2",
      "projectname": "New Project",
      "owner_id": 2,
      "description": "Description of the new project.",
      "created_on": 1672531200000,
      "updated_on": 1672531200000,
      "status": "OPEN",
      "type": "NewType",
      "active": true
    }
  ]
}
```

Response Codes:
- 200 OK: Successfully added the project.
- 400 Bad Request: Invalid input data.
- 500 Internal Server Error: Failed to add the project.

Response Body:

```
{
  "Response": "OK"
}
```

4. Delete Project

Title: Delete Project
Description: Delete a specific project by its ID.
Method: DELETE
Endpoint: /apiv1/project/deleteproject/{id}
Request Headers: Content-Type: application/json
Path Parameters: id (String): The ID of the project to delete.
Response Codes:
- 200 OK: Successfully deleted the project.
- 404 Not Found: Project not found.
- 500 Internal Server Error: Failed to delete the project.

Response Body:

```
{
  "Response": "Success"
}
```

5. Edit Project

Title: Edit Project
Description: Edit the details of an existing project.
Method: POST
Endpoint: /apiv1/project/editProject
Request Headers: Content-Type: application/json
Request Body:

```
{
  "project_id": "1",
  "projectname": "Updated Project",
  "owner_id": 1,
  "description": "Updated description.",
  "updated_on": 1672531200000,
  "status": "IN_PROGRESS",
  "type": "UpdatedType"
}
```

Response Codes:

- 200 OK: Successfully edited the project.
- 400 Bad Request: Invalid input data.
- 404 Not Found: Project not found.
- 500 Internal Server Error: Failed to edit the project.

Response Body:

```
{
  "Response": "Success"
}
```

Task Management API

1. Get All Tasks for a Project

Title: Get All Tasks for a Project
Description: Retrieve all tasks for a specific project from the database.
Method: GET
Endpoint: /apiv1/task/project/getalltasks/{projectid}
Request Headers: Content-Type: application/json
Path Parameters: projectid (String): The ID of the project whose tasks are to be retrieved.
Response Codes:

- 200 OK: Successfully retrieved all tasks for the project.
- 500 Internal Server Error: Failed to retrieve tasks.

Response Body:


```
{
  "tasks": [
    {
      "project_id": "1",
      "projectname": "Create Educational Platform",
      "task_id": "101",
      "task_name": "Design Database",
      "description": "Design the database schema",
      "status": "IN_PROGRESS",
      "priority": "High",
      "due_date": "2023-01-10T00:00:00.000Z",
      "assigned_user": "John Doe"
    }
  ]
}
```


2. Add Tasks

Title: Add Tasks
Description: Add new tasks to a project in the database.
Method: POST
Endpoint: /apiv1/task/project/addtasks
Request Headers: Content-Type: application/json
Request Body:

```
{
  "tasks": [
    {
      "task_id": "102",
      "project_id": "1",
      "task_name": "Develop API",
      "description": "Develop the REST API endpoints",
      "status": "OPEN",
      "priority": "Medium",
      "assigned_user_id": 2,
      "due_date": 1672531200000,
      "created_on": 1672531200000,
      "updated_on": 1672531200000
    }
  ]
}
```


Response Codes:

- 200 OK: Successfully added the tasks.
- 400 Bad Request: Invalid input data.
- 500 Internal Server Error: Failed to add the tasks.

Response Body:


```
{
  "Response": "OK"
}
```


3. Delete Task

Title: Delete Task
Description: Delete a specific task by its ID.
Method: DELETE
Endpoint: /apiv1/task/project/deletetask/{id}
Request Headers: Content-Type: application/json
Path Parameters: id (String): The ID of the task to delete.
Response Codes:

- 200 OK: Successfully deleted the task.
- 404 Not Found: Task not found.
- 500 Internal Server Error: Failed to delete the task.

Response Body:


```
{
  "Response": "Success"
}
```

4. Edit Task

Title: Edit Task
Description: Edit the details of an existing task.
Method: POST
Endpoint: /apiv1/task/project/editTask
Request Headers: Content-Type: application/json
Request Body:

```
{
  "task_id": "101",
  "task_name": "Updated Task Name",
  "project_id": "1",
  "description": "Updated description",
  "status": "IN_PROGRESS",
  "priority": "High",
  "assigned_user_id": 1,
  "due_date": 1672531200000,
  "updated_on": 1672531200000
}
```

Response Codes:

- 200 OK: Successfully edited the task.
- 400 Bad Request: Invalid input data.
- 404 Not Found: Task not found.
- 500 Internal Server Error: Failed to edit the task.

Response Body:

```
{
  "Response": "Success"
}
```

Comment Management API

1. Get All Comments for a Project

Title: Get All Comments for a Project
Description: Retrieve all comments for a specific project from the database.
Method: GET
Endpoint: /api/comments/getallcomments/{projectId}
Request Headers: Content-Type: application/json
Path Parameters: projectId (int): The ID of the project whose comments are to be retrieved.
Response Codes:

- 200 OK: Successfully retrieved all comments for the project.
- 500 Internal Server Error: Failed to retrieve comments.

Response Body:

```
{
  "comments": [
    {
      "id": 1,
      "projectId": 101,
      "comments": "This is a comment",
      "userId": 1,
      "createdOn": "2023-01-10T00:00:00.000Z"
    }
  ]
}
```

2. Add Comment

Title: Add Comment
Description: Add a new comment to the specified project.
Method: POST
Endpoint: /api/comments/addcomment
Request Headers: Content-Type: application/json
Request Body:

```
{
  "projectId": 101,
  "comments": "This is a new comment",
  "userId": 2,
  "createdOn": 1672531200000
}
```

Response Codes:

- 200 OK: Successfully added the comment.
- 400 Bad Request: Invalid input data.
- 500 Internal Server Error: Failed to add the comment.

Response Body:


```
{
  "comment": {
    "id": 2,
    "projectId": 101,
    "comments": "This is a new comment",
    "userId": 2,
    "createdOn": "2023-01-10T00:00:00.000Z"
  }
}
```


3. Delete Comment

Title: Delete Comment
Description: Delete a specific comment by its ID.
Method: DELETE
Endpoint: /api/comments/deletecomment/{id}
Request Headers: Content-Type: application/json
Path Parameters: id (int): The ID of the comment to delete.
Response Codes:

- 200 OK: Successfully deleted the comment.
- 404 Not Found: Comment not found.
- 500 Internal Server Error: Failed to delete the comment.

Response Body:


```
{
  "response": "Success"
}
```


4. Edit Comment

Title: Edit Comment
Description: Edit the details of an existing comment.
Method: POST
Endpoint: /api/comments/editcomment
Request Headers: Content-Type: application/json
Request Body:

```
{
  "id": 1,
  "projectId": 101,
  "comments": "Updated comment",
  "userId": 1,
  "createdOn": 1672531200000
}
```

Response Codes:

- 200 OK: Successfully edited the comment.
- 400 Bad Request: Invalid input data.
- 404 Not Found: Comment not found.
- 500 Internal Server Error: Failed to edit the comment.

Response Body:

```
{
  "comment": {
    "id": 1,
    "projectId": 101,
    "comments": "Updated comment",
    "userId": 1,
    "createdOn": "2023-01-10T00:00:00.000Z"
  }
}
```

User Project Management API

1. Get All User Projects

Title: Get All User Projects
Description: Retrieve all projects associated with a specific user.
Method: GET
Endpoint: /apiv1/user/getalluserprojects/{user_id}
Request Headers: Content-Type: application/json
Path Parameters: user_id (String): The ID of the user whose projects are to be retrieved.
Response Codes:

- 200 OK: Successfully retrieved all user projects.
- 500 Internal Server Error: Failed to retrieve user projects.

Response Body:


```
{
  "user_projects": [
    {
      "project_id": "1",
      "projectname": "Project Name",
      "project_description": "Description of the project",
      "project_owner": "Owner Name",
      "project_owner_email": "owner@example.com",
      "project_created_on": "2023-01-10T00:00:00.000Z",
      "project_updated_on": "2023-01-15T00:00:00.000Z",
      "project_status": "Open",
      "project_type": "Type",
      "project_active": true
    }
  ]
}
```


2. Get All Project Members

Title: Get All Project Members
Description: Retrieve all members associated with a specific project.
Method: GET
Endpoint: /api/project/getallprojectmembers/{project_id}
Request Headers: Content-Type: application/json
Path Parameters:

- project_id (String): The ID of the project whose members are to be retrieved.
  Response Codes:
- 200 OK: Successfully retrieved all project members.
- 500 Internal Server Error: Failed to retrieve project members.

Response Body:

```
{
  "project_members": [
    {
      "member_id": "1",
      "member_name": "Member Name",
      "member_role": "Role",
      "member_email": "member@example.com",
      "joined_on": "2023-01-10T00:00:00.000Z"
    }
  ]
}
```

3. Add Project Member

Title: Add Project Member
Description: Add a new member to a specific project.
Method: POST
Endpoint: /api/project/addprojectmember
Request Headers: Content-Type: application/json
Request Body:

```
{
  "project_member": [
    {
      "PROJECT_USER_ID": 1,
      "PROJECT_ID": "Project ID",
      "USER_ID": 1,
      "ROLE": "COLLABORATOR",
      "created_at": 1672531200000,
      "updated_at": 1672531200000
    }
  ]
}
```

Response Codes:

- 200 OK: Successfully added the project member.
- 400 Bad Request: Invalid input data.
- 500 Internal Server Error: Failed to add the project member.

Response Body:

```
{
  "Response": "OK"
}
```