Api Documentation for Project Management

1. Register User

- **Endpoint**: `/api/users/register/`
- **Method**: `POST`
- **Description**: Registers a new user by creating an account.
- **Request Headers**:
  - No authentication required.
- **Request Body**:

```
{
    "username": "johndoe",
    "email": "johndoe@example.com",
    "first_name": "John",
    "last_name": "Doe",
    "password": "securepassword"
}
```

- **Response**:
  - **Success (`201 Created`)**:

```
{
    "id": 1,
    "username": "johndoe",
    "email": "johndoe@example.com",
    "first_name": "John",
    "last_name": "Doe"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Username or email already exists."
}
```

- **Notes**:
  - Password is write-only and will not be included in the response.
  - Fields `id` and `password` are handled internally.

2. Login User

- **Endpoint**: `/api/user/login/`
- **Method**: `POST`
- **Description**: Authenticates a user and returns JWT tokens.
- **Request Headers**:

    - No authentication required.
- **Request Body**:

```
{
    "username": "johndoe",
    "password": "securepassword"
}
```

- **Response**:
    - **Success (`200 OK`)**:

```
{
    "refresh": "eyJhbGciOiJIUzI1...",
    "access": "eyJhbGciOiJIUzI1..."
}
```

    - **Error (`401 Unauthorized`)**:

```
{
    "error": "Invalid credentials"
}
```

- **Notes**:
    - JWT tokens (`refresh` and `access`) are returned for further authentication.
    - Use the `access` token for accessing protected endpoints and `refresh` to obtain a new access token.

3. Retrieve User Details

- **Endpoint**: `/api/users/<int:pk>/`
- **Method**: `GET`
- **Description**: Fetches details of a specific user by their ID.
- **Request Headers**:
    - `Authorization: Bearer <access_token>`
- **Request Parameters**:
    - `pk` (path parameter): The ID of the user.
- **Response**:
    - **Success (`200 OK`)**:

```
{
    "id": 1,
    "username": "johndoe",
    "email": "johndoe@example.com",
    "first_name": "John",
    "last_name": "Doe",
    "date_joined": "2025-01-01T12:00:00Z"
}
```

    - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

4. Update User

- **Endpoint**: `/api/users-update/<int:pk>/`
- **Method**: `PUT` or `PATCH`
- **Description**: Updates details of a specific user by their ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the user.
- **Request Body**:

```
{
    "username": "john_updated",
    "email": "john_updated@example.com",
    "first_name": "John",
    "last_name": "Updated"
}
```

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "username": "john_updated",
    "email": "john_updated@example.com",
    "first_name": "John",
    "last_name": "Updated",
    "date_joined": "2025-01-01T12:00:00Z"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Invalid data provided."
}
```

5. Delete User

- **Endpoint**: `/api/user-delete/<int:pk>/`
- **Method**: `DELETE`
- **Description**: Deletes a specific user by their ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the user.

- **Response**:
  - **Success (`204 No Content`)**:
    - No response body.
  - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

6. List All Projects

- **Endpoint**: `/api/projects/`
- **Method**: `GET`
- **Description**: Retrieves a list of all projects.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Response**:
  - **Success (`200 OK`)**:

```
[
    {
        "id": 1,
        "project_id": "PRJ001",
        "name": "Project Alpha",
        "description": "Description of Project Alpha",
        "owner": 3,
        "created_at": "2025-01-01T12:00:00Z"
    },
    {
        "id": 2,
        "project_id": "PRJ002",
        "name": "Project Beta",
        "description": "Description of Project Beta",
        "owner": 5,
        "created_at": "2025-01-02T15:30:00Z"
    }
]
```

7. Create a New Project

- **Endpoint**: `/api/project/create/`
- **Method**: `POST`
- **Description**: Creates a new project.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Body**:

```
{
    "project_id": "PRJ003",
    "name": "Project Gamma",
    "description": "Description of Project Gamma",
    "owner": 3
```

```
}
```

- **Response**:
  - **Success (`201 Created`)**:

```
{
    "id": 3,
    "project_id": "PRJ003",
    "name": "Project Gamma",
    "description": "Description of Project Gamma",
    "owner": 3,
    "created_at": "2025-01-03T10:45:00Z"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Owner ID is required to create a project."
}
```

8. Retrieve a Single Project

- **Endpoint**: `/api/project/<int:pk>/`
- **Method**: `GET`
- **Description**: Fetches details of a specific project by its ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project.
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "project_id": "PRJ001",
    "name": "Project Alpha",
    "description": "Description of Project Alpha",
    "owner": 3,
    "created_at": "2025-01-01T12:00:00Z"
}
```

  - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

9. Update a Project

- **Endpoint**: `/api/project/update/<int:pk>/`
- **Method**: `PUT` or `PATCH`
- **Description**: Updates details of a specific project.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project.
- **Request Body**:

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

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "project_id": "PRJ001",
    "name": "Updated Project Alpha",
    "description": "Updated description of Project Alpha",
    "owner": 3,
    "created_at": "2025-01-01T12:00:00Z"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Invalid data provided."
}
```

10. Delete a Project

- **Endpoint**: `/api/projects/delete/<int:pk>/`
- **Method**: `DELETE`
- **Description**: Deletes a specific project.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project.
- **Response**:
  - **Success (`200 OK`)**:

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

  - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

11. List All Tasks

- **Endpoint**: `/api/task/`
- **Method**: `GET`
- **Description**: Retrieves a list of all tasks.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "message": "List of tasks retrieved successfully",
    "data": [
        {
            "id": 1,
            "title": "Task Alpha",
            "description": "Description of Task Alpha",
            "status": "Pending",
            "priority": "High",
            "assigned_to": 5,
            "project": 3,
            "created_at": "2025-01-01T12:00:00Z",
            "due_date": "2025-01-10"
        },
        {
            "id": 2,
            "title": "Task Beta",
            "description": "Description of Task Beta",
            "status": "Completed",
            "priority": "Medium",
            "assigned_to": 6,
            "project": 4,
            "created_at": "2025-01-02T15:30:00Z",
            "due_date": "2025-01-15"
        }
    ]
}
```

12. Create a New Task

- **Endpoint**: `/api/task/create/`
- **Method**: `POST`
- **Description**: Creates a new task.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Body**:

```
{
```

```
    "title": "Task Gamma",
    "description": "Description of Task Gamma",
    "status": "In Progress",
    "priority": "Low",
    "assigned_to": 7,
    "project": 3,
    "due_date": "2025-01-20"
}
```

- **Response**:
  - **Success (`201 Created`)**:

```
{
    "id": 3,
    "title": "Task Gamma",
    "description": "Description of Task Gamma",
    "status": "In Progress",
    "priority": "Low",
    "assigned_to": 7,
    "project": 3,
    "created_at": "2025-01-03T10:45:00Z",
    "due_date": "2025-01-20"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Project ID is required to create a task."
}
```

13. Retrieve a Single Task

- **Endpoint**: `/api/task/<int:pk>/`
- **Method**: `GET`
- **Description**: Fetches details of a specific task by its ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the task.
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "title": "Task Alpha",
    "description": "Description of Task Alpha",
    "status": "Pending",
    "priority": "High",
    "assigned_to": 5,
    "project": 3,
    "created_at": "2025-01-01T12:00:00Z",
```

```
    "due_date": "2025-01-10"
}
```

- **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

13. Update a Task

- **Endpoint**: `/api/task/update/<int:pk>/`
- **Method**: `PUT` or `PATCH`
- **Description**: Updates details of a specific task.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the task.
- **Request Body**:

```
{
    "title": "Updated Task Alpha",
    "description": "Updated description of Task Alpha",
    "status": "Completed"
}
```

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "title": "Updated Task Alpha",
    "description": "Updated description of Task Alpha",
    "status": "Completed",
    "priority": "High",
    "assigned_to": 5,
    "project": 3,
    "created_at": "2025-01-01T12:00:00Z",
    "due_date": "2025-01-10"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Invalid data provided."
}
```

14. Delete a Task

- **Endpoint**: `/api/task/delete/<int:pk>/`

- **Method**: `DELETE`
- **Description**: Deletes a specific task.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the task.
- **Response**:
  - **Success (`200 OK`)**:

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

  - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

15. List All Comments

- **Endpoint**: `/api/comment/`
- **Method**: `GET`
- **Description**: Retrieves a list of all comments.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "message": "List of Comments retrieved successfully",
    "data": [
        {
            "id": 1,
            "content": "This is a comment",
            "user": 5,
            "task": 3,
            "created_at": "2025-01-01T12:00:00Z"
        },
        {
            "id": 2,
            "content": "Another comment",
            "user": 6,
            "task": 4,
            "created_at": "2025-01-02T15:30:00Z"
        }
    ]
}
```

16. Create a New Comment

- **Endpoint**: `/api/comment/create/`
- **Method**: `POST`
- **Description**: Creates a new comment for a specific task.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Body**:

```
{
    "content": "This is a new comment.",
    "task": 3
}
```

- **Response**:
  - **Success (`201 Created`)**:

```
{
    "id": 3,
    "content": "This is a new comment.",
    "user": 5,
    "task": 3,
    "created_at": "2025-01-03T10:45:00Z"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Task ID is required to create a comment."
}
```

17. Retrieve a Single Comment

- **Endpoint**: `/api/comment/<int:pk>/`
- **Method**: `GET`
- **Description**: Fetches details of a specific comment by its ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the comment.
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "content": "This is a comment",
    "user": 5,
    "task": 3,
    "created_at": "2025-01-01T12:00:00Z"
}
```

- **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

18. Update a Comment

- **Endpoint**: `/api/comment/update/<int:pk>/`
- **Method**: `PUT` or `PATCH`
- **Description**: Updates the content of a specific comment.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the comment.
- **Request Body**:

```
{
    "content": "Updated comment content."
}
```

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "content": "Updated comment content.",
    "user": 5,
    "task": 3,
    "created_at": "2025-01-01T12:00:00Z"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Invalid data provided."
}
```

19. Delete a Comment

- **Endpoint**: `/api/comment/delete/<int:pk>/`
- **Method**: `DELETE`
- **Description**: Deletes a specific comment.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the comment.

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "message": "Comment deleted successfully."
}
```

  - **Error (`404 Not Found`)**:

```
{
    "detail": "Not found."
}
```

20. List All Project Members

- **Endpoint**: `/api/project/member/`
- **Method**: `GET`
- **Description**: Retrieves a list of all project members.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "message": "List of Project Members retrieved successfully",
    "data": [
        {
            "id": 1,
            "project": 5,
            "user": 10,
            "role": "admin"
        },
        {
            "id": 2,
            "project": 6,
            "user": 11,
            "role": "member"
        }
    ]
}
```

21. Create a New Project Member

- **Endpoint**: `/api/project/member/create/`
- **Method**: `POST`
- **Description**: Adds a new member to a project.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Body**:

```
{
    "project": 5,
    "user": 10,
    "role": "member"
}
```

- **Response**:
  - **Success (`201 Created`)**:

    ```
    {
        "id": 3,
        "project": 5,
        "user": 10,
        "role": "member"
    }
    ```

  - **Error (`400 Bad Request`)**:

    ```
    {
        "error": "Project ID is required to create a project member."
    }
    ```

22. Retrieve a Single Project Member

- **Endpoint**: `/api/project/member/<int:pk>/`
- **Method**: `GET`
- **Description**: Fetches details of a specific project member by their ID.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project member.
- **Response**:
  - **Success (`200 OK`)**:

    ```
    {
        "id": 1,
        "project": 5,
        "user": 10,
        "role": "admin"
    }
    ```

  - **Error (`404 Not Found`)**:

    ```
    {
        "detail": "Not found."
    }
    ```

23. Update a Project Member

- **Endpoint**: `/api/project/member/update/<int:pk>/`
- **Method**: `PUT` or `PATCH`
- **Description**: Updates the role or details of a specific project member.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project member.
- **Request Body**:

```
{
    "role": "admin"
}
```

- **Response**:
  - **Success (`200 OK`)**:

```
{
    "id": 1,
    "project": 5,
    "user": 10,
    "role": "admin"
}
```

  - **Error (`400 Bad Request`)**:

```
{
    "error": "Invalid data provided."
}
```

24. Delete a Project Member

- **Endpoint**: `/api/project/member/delete/<int:pk>/`
- **Method**: `DELETE`
- **Description**: Removes a member from a project.
- **Request Headers**:
  - `Authorization: Bearer <access_token>`
- **Request Parameters**:
  - `pk` (path parameter): The ID of the project member.
- **Response**:
  - **Success (`200 OK`)**:

```
{
    "message": "Project Member deleted successfully."
}
```

  - **Error (`404 Not Found`)**:

```
{
```

```
    "detail": "Not found."
}
```

- **400 Bad Request: Validation error in the request payload (e.g., missing or invalid data).**
- **401 Unauthorized: The user is not authenticated.**
- **403 Forbidden: The user is not authorized to perform the action.**
- **404 Not Found: The project member with the specified ID does not exist.**
- **Notes**:
  - JWT tokens (`refresh` and `access`) are returned for further authentication.
  - Use the `access` token for accessing protected endpoints and `refresh` to obtain a new access token.