Task API Documentation

API documentation for task management application. This API allows users to register, log in, manage tasks, and perform various operations on them.

Authentication

To access the endpoints provided by this API, users need to authenticate using JSON Web Tokens (JWT). I have implemented Simple JWT.

Base URL: **http://127.0.0.1:8000/api/**

URL: **/register/**
Method: **POST**
Authentication: **None**
Request Body: Requires a JSON object with "email", "username", and "password" fields.
Response: Upon successful registration, returns user details with a status code of 201 (Created).
Eg.

```
{
  "username": "akash",
  "password": "1234",
  "email": "akash@gmail.com"
}
```

URL: **/login/**
Method: **POST**
Authentication: **None**
Request Body: Requires a JSON object with "username" and "password" fields.
Response: Upon successful login, returns a JWT token pair (access token and refresh token).

```
{
  "username": "akash",
  "password": "1234",
}
```

Task Management

URL: **/tasks/**
Method: **GET**
JWT Authentication: **Required**
Response: Returns a list of all tasks with details such as id, title, description, due_date, status, and owner.

URL: **/tasks/**
Method: **POST**
JWT Authentication: **Required**
Request Body: Requires a JSON object with "title", "description", "due_date", and "status" fields.
Response: Upon successful creation, returns the details of the newly created task.
Eg.

```
{
  "title": "akashhello",
  "description": "working",
  "due_date": "2024-02-14",
  "status": "Completed"
}
```

Bearer Token:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNzA3OT
QyNTAxLCJpYXQiOjE3MDc5NDIyMDEsImp0aSI6ImYzZDAzZGIxYWU0MzQzNmY5MzkyODVi
OTA0YTQxYTRlIiwidXNlcl9pZCI6NH0.nZUhXMCSBMiP-gakd4vVzIKAQMYVMiFGdWZ-p_i2NtI

UPDATE AND DELETE

URL: **/task/<int:pk>/**
Method: **GET, PUT, PATCH, DELETE**
JWT Authentication: **Required**
Permission : **IsOwner Required**
Response:
GET: Returns details of a specific task identified by its ID.
PUT, PATCH: Updates the details of a specific task identified by its ID.
DELETE: Deletes a specific task identified by its ID.
Error Handling

PUT Example:

```
{
  "title": "Welcome",
  "description": "It is working",
  "due_date": "2024-02-14",
  "status": "In Progress"
}
```

Bearer Token:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNzA3OT
QyNTAxLCJpYXQiOjE3MDc5NDIyMDEsImp0aSI6ImYzZDAzZGIxYWU0MzQzNmY5MzkyODVi
OTA0YTQxYTRlIiwidXNlcl9pZCI6NH0.nZUhXMCSBMiP-gakd4vVzIKAQMYVMiFGdWZ-p_i2NtI
```

In case of errors, the API returns appropriate HTTP status codes along with error messages in the response body. Common error scenarios include:

400 Bad Request: Invalid request data or missing required fields.
401 Unauthorized: Authentication credentials were not provided or are invalid.
403 Forbidden: User does not have permission to access the resource.
404 Not Found: Resource not found.
500 Internal Server Error: Unexpected server error.