Authentication and Security

- This API uses JWT (JSON Web Token) for authentication. To access protected endpoints, you must include the JWT token in the `Authorization` header of each request.
- Tokens expire after 24 hours. You can retrieve a new token by logging in with your credentials.

Task Management API

- The Task Management API allows users to create, update, delete, and retrieve tasks. It supports user authentication and role-based access control. This API is built using RESTful principles and returns data in JSON format.

1. CREATE
    - User Registration
        i. Endpoint: `/user/register`
        ii. Method: `POST`
        iii. Description: Registers a new user.
        iv. Authentication: Not required.
        v. Request Body:

```
{
  "userName": "string",
  "password": "string",
  "email": "string",
  "phoneNum": "string"
}
```

        vi. Response:{“User register successfully”}
    -
2. GET
    - Login(Fetching JWT Token)
        i. Endpoint: `/user/login`
        ii. Method: `GET`
        iii. Description: Logins user.
        iv. Request Params:
            1. Username:string
            2. Password:string
        v. Response :

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5..."
}
```

- Get Task
    i. Endpoint: `/Task/getTask`
    ii. Method: `GET`
    iii. Description:Get a todo item.
    iv. Request Params:
        1. userId:long
        2. taskId:long
    v. Response :


```
{
 "task": "Office work",
 "descriptionStatus": "Complete cashfree payment",
 "priority": "high",
 "dueDate": "2024-11-17",
 "createdTs": "2024-09-28T11:32:45.260+00:00",
 "updatedTs": "2024-09-28T11:32:45.260+00:00",
 "userId": 3,
 "status": "inprogress"
}
```


- Get All Task
    i. Endpoint: `/Task/getAllTask`
    ii. Method: `GET`
    iii. Description : get all tasks by userid.
    iv. Request Params:
        1. userId:long
    v. Response :


```
{
    {
        "task": "Office work",
        "descriptionStatus": "Complete cashfree payment",
        "priority": "high",
        "dueDate": "2024-11-17",
        "createdTs": "2024-09-28T11:32:45.260+00:00",
        "updatedTs": "2024-09-28T11:32:45.260+00:00",
        "userId": 3,
        "status": "inprogress"
    },

    {
```

```
                "task": "Office work",
                "descriptionStatus": "Complete cashfree payment",
                "priority": "high",
                "dueDate": "2024-11-17",
                "createdTs": "2024-09-28T11:32:45.260+00:00",
                "updatedTs": "2024-09-28T11:32:45.260+00:00",
                "userId": 3,
                "status": "inprogress"
            }

        }
```


- Filters
    i. Endpoint: `/Task/filter`
    ii. Method: `GET`
    iii. Description : filtered tasks by status,priority,duedate,userid.
    iv. Request Params:
        1. userId:long
        2. Status:string
        3. Priority:string
        4. duedate:string
- Search
    i. Endpoint: `/Task/search`
    ii. Method: `GET`
    iii. Description :search by inputTerm.
    iv. Request Params:
        1. inputTerm:string

3. PUT
    - UpdateTasks
        i. Endpoint: `/Task/updateTask`
        ii. Method: `PUT`
        iii. Description :update a todo item.
        iv. Request Params:
            1. userId:long
            2. taskId:long
4. DELETE
    - DeleteTasks
        i. Endpoint: `/Task/deleteTask`
        ii. Method: `DELETE`
        iii. Description :delete task by userId and taskId.
        iv. Request Params:
            1. userId:long
            2. taskId:long
        v. Response:

“Task deleted successfully”

Challenges:

- Implemented authentication by generating a JWT token during each sign-up.
- Implemented search and filtering based on the user's status and the priority of todo items.
- Created a Docker image for the application and tracked the container status, ports, and configurations.
- Configured Swagger and integrated it with the application.