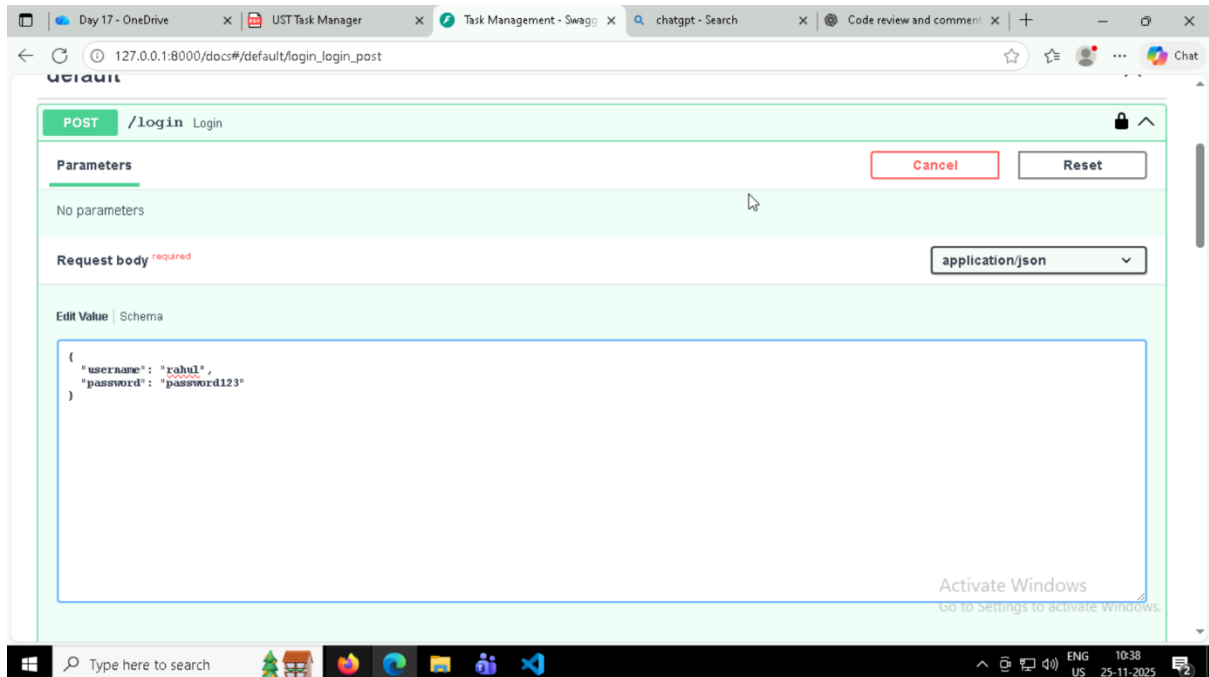
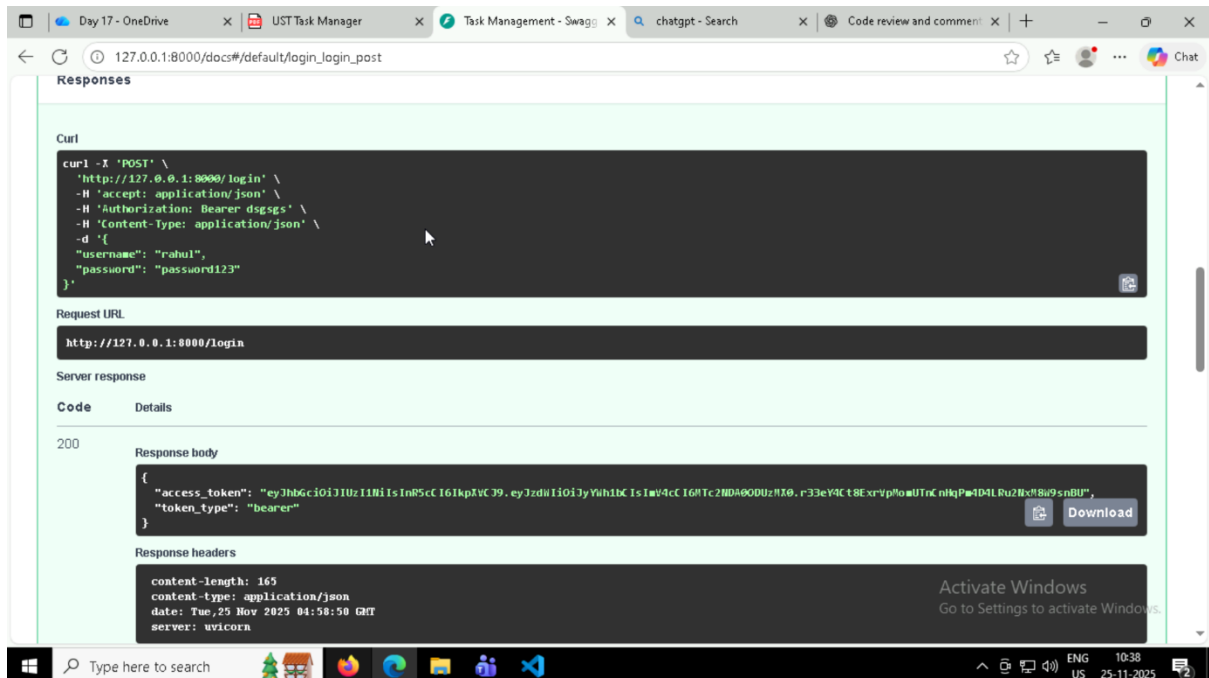


### SCENERIO 1-Successful Login

## Login using username and password

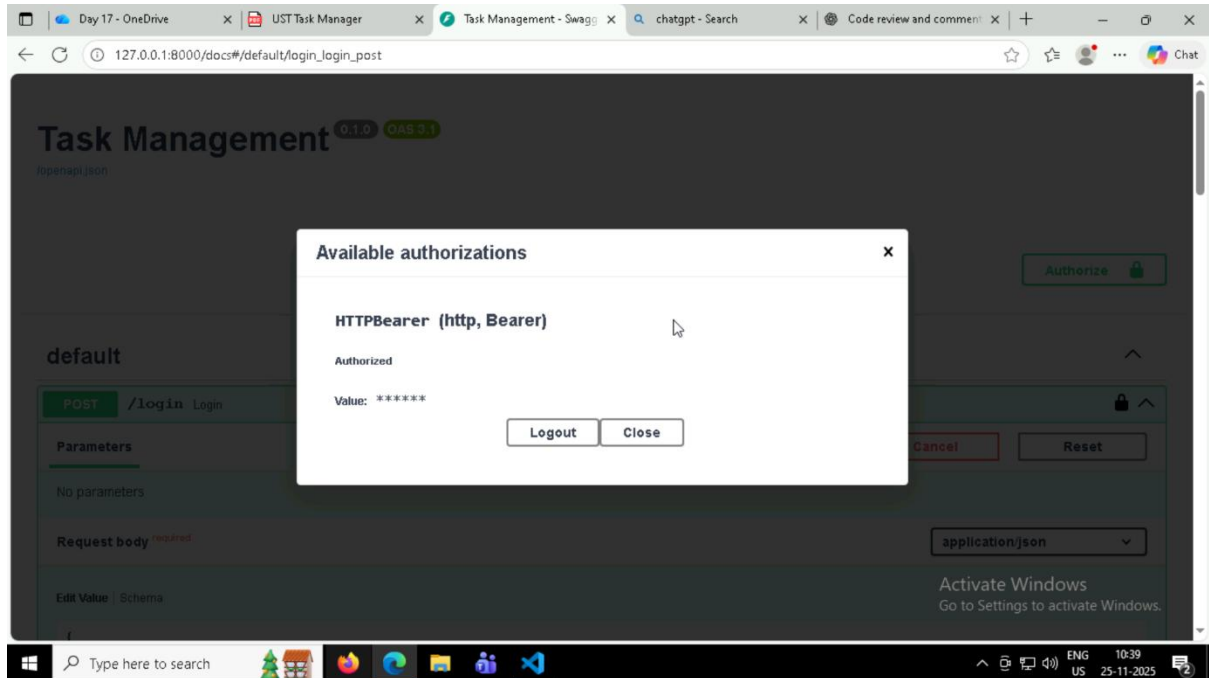


Token Generated if username and password are valid



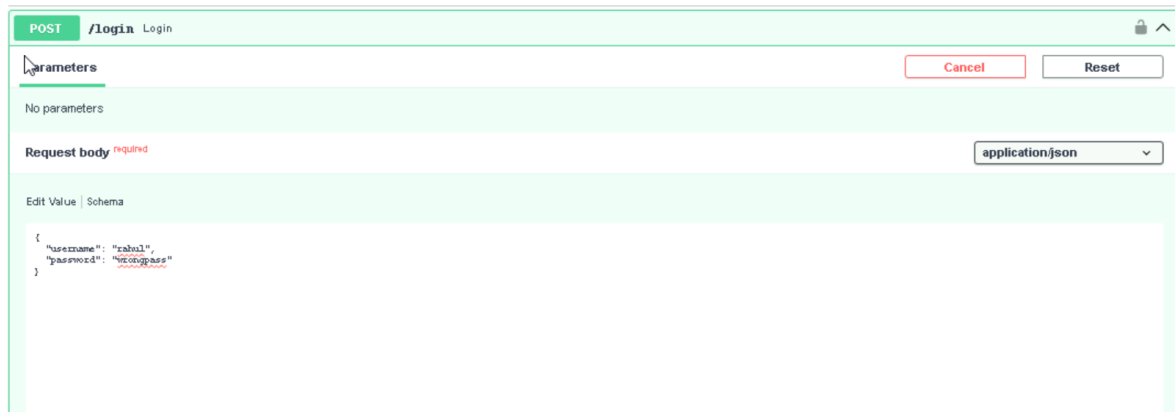
## Authorizing using Token

### Login successful

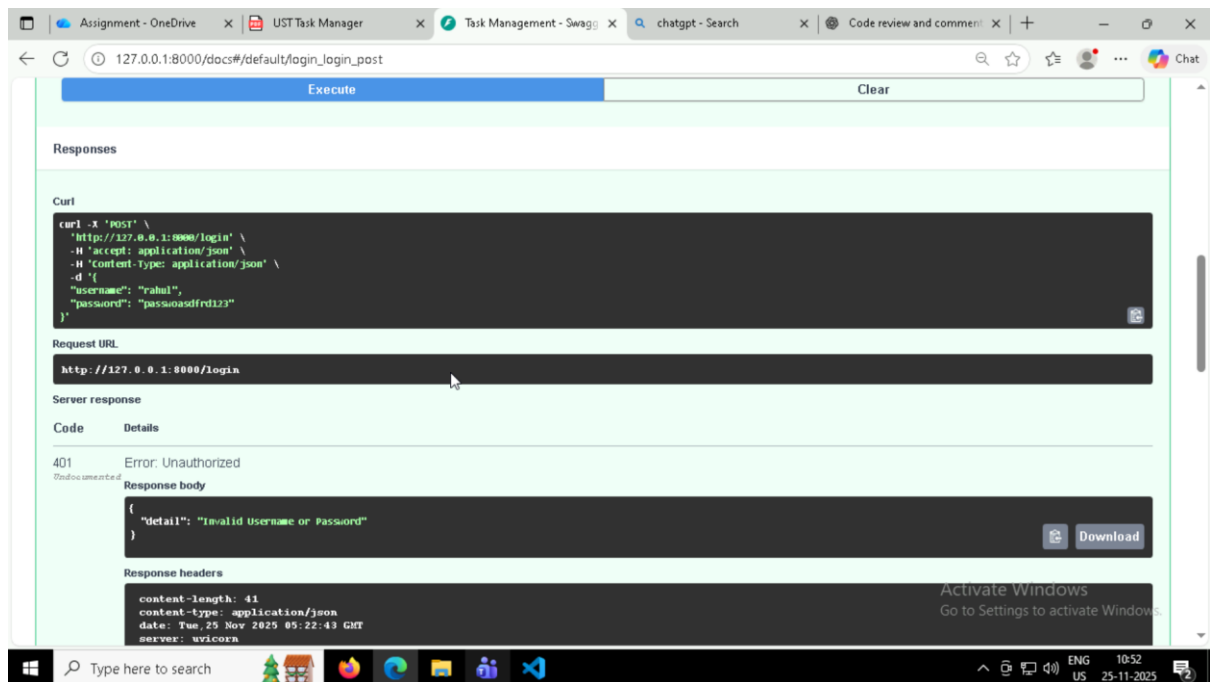


## SCENERIO 2 –Login Failure

### Invalid password



Swagger UI for the `POST /login` endpoint. The interface shows the method `POST` and the path `/login`. The `Parameters` tab is selected, showing "No parameters". The `Request body` is required and set to `application/json`. The request body is a JSON object: `{ "username": "rahul", "password": "wecanpass" }`. The `password` field is highlighted in red, indicating it is incorrect. The `Cancel` and `Reset` buttons are visible.



Swagger UI showing the execution of the login request. The `Execute` button has been clicked, and the `Responses` tab is selected. The `Request URL` is `http://127.0.0.1:8000/login`. The `Server response` is a `401` status code with the message `Error: Unauthorized`. The `Response body` is a JSON object: `{ "detail": "Invalid Username or Password" }`. The `Response headers` are: `content-length: 41`, `content-type: application/json`, `date: Tue, 25 Nov 2025 09:22:43 GMT`, and `server: uvicorn`. The `Download` button is visible next to the response body.

### SCENARIO 3- Create Task

#### Creating new task -Request

POST /task Add Task

Parameters

No parameters

Request body required

application/json

Edit Value | Schema

```
{
  "title": "Book train tickets",
  "description": "Book IRCTC tickets to Chennai"
}
```

#### Response -

Response body

```
{
  "id": 1,
  "title": "Book train tickets",
  "description": "Book IRCTC tickets to Chennai",
  "completed": false
}
```

Download

#### SCENARIO 4-Create Task with Missing Title

```
{  
  "description": "Book IRCTC tickets to Chennai"  
}
```

##### Response body

```
{  
  "detail": [  
    {  
      "type": "missing",  
      "loc": [  
        "body",  
        "title"  
      ],  
      "msg": "Field required",  
      "input": {  
        "description": "Book IRCTC tickets to Chennai"  
      }  
    }  
  ]  
}
```

##### Response headers

## SCENARIO 5 -Update Task

Request to update task using id -

**id** \* required  
integer  
(path)

1

**Request body** required

Edit Value | Schema

```
{  
  "title": "Pay Water Bill",  
  "description": "Pay BWSSB bill",  
  "completed": true  
}
```

Response –

**Response body**

```
{  
  "id": 1,  
  "title": "Pay Water Bill",  
  "description": "Pay BWSSB bill",  
  "completed": true  
}
```

Response headers

## SCENARIO 6-Get Non-Existing Task

The screenshot shows the Swagger UI interface for a REST API. The endpoint is `GET /task/{id}` with the description "Get Task By Id". The parameter `id` is required and has a value of `10`. The `Execute` button has been clicked, and the response is displayed. The response is a 404 status code with the message "Error: Not Found". The response body is a JSON object: `{ "detail": "Task doesn't exist!" }`. The Swagger UI also shows the request URL `http://127.0.0.1:8000/task/10` and the server response.

Parameters

Name	Description
<code>id</code> * required integer (path)	

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/task/10' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXNC9.ej2a8RI0I3pY9N1NC1sIm44C16ATc2NDAlNDYjM09udmF79hC-Au5knd4P0ED672q19LkjaFT7c1TK1c'
```

Request URL

`http://127.0.0.1:8000/task/10`

Server response

Code	Details
404	Error: Not Found

Response body

```
{
  "detail": "Task doesn't exist!"
}
```

## SCENARIO 7-Delete Non-Existing Task

The screenshot shows the Swagger UI interface for a REST API. The endpoint is `DELETE /task/{id}` with the description "Delete Profile". The parameter `id` is required and has a value of `10`. The `Execute` button has been clicked, and the response is displayed. The response is a 404 status code with the message "Error: Not Found". The response body is a JSON object: `{ "detail": "Task Not Found" }`. The Swagger UI also shows the request URL `http://127.0.0.1:8000/task/10` and the server response.

Parameters

Name	Description
<code>id</code> * required integer (path)	

Execute Clear

Responses

Curl

```
curl -X 'DELETE' \
  'http://127.0.0.1:8000/task/10' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXNC9.ej2a8RI0I3pY9N1NC1sIm44C16ATc2NDAlNDYjM09udmF79hC-Au5knd4P0ED672q19LkjaFT7c1TK1c'
```

Request URL

`http://127.0.0.1:8000/task/10`

Server response

Code	Details
404	Error: Not Found

Response body

```
{
  "detail": "Task Not Found"
}
```