API Testing Documentation: Flask To-Do List CRUD API
Prepared by: Amisha Halarnkar

Introduction

This project involves developing and testing a simple To-Do List REST API using Python Flask framework and testing it manually using Postman. The API supports basic CRUD operations: creating, reading, updating, and deleting to-do tasks.

Test Methodology

Manual testing is done with Postman, where each API endpoint is tested with valid, invalid, boundary, and negative cases. Automated tests cover repetitive validation of CRUD operations using Python testing libraries.

Manual Testing

Objective

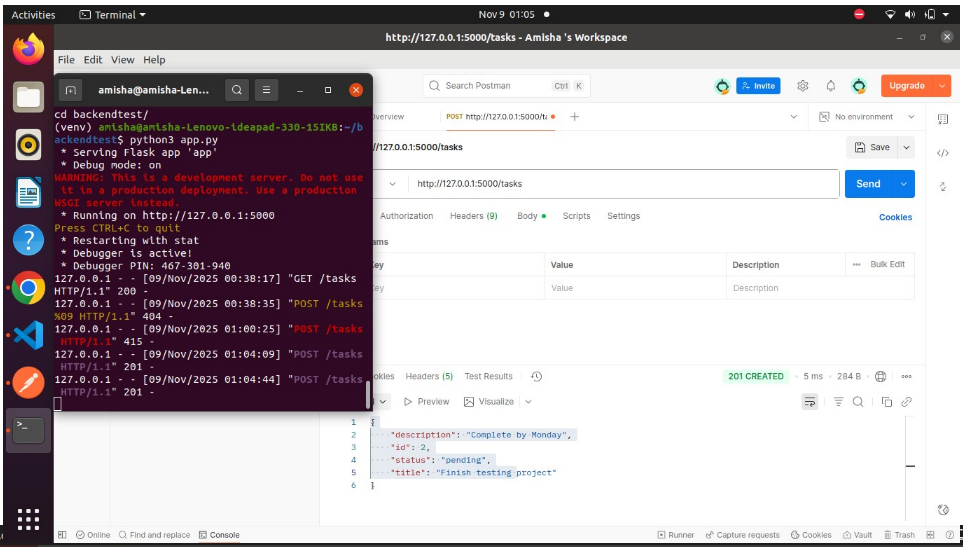
To verify the functionality and robustness of the API endpoints for managing to-do tasks

Test Environment

- Operating System: Windows 10 or Ubuntu Linux
- Development: Python 3., Flask framework
- Testing Tool: Postman
- Automation: pytest

Test Case ID	Endpoint	HTTP Method	Description	Request Body (JSON)	Expected Status Code	Expected Response	Actual Status Code	Actual Response	Pass/ Fail	
TC_001	/tasks	GET	Get all tasks	None	200	Empty JSON array or list of task objects	200	[]	Pass	
TC_002	/tasks	POST	Create new task with valid data	{"title": "Finish project", "description": "Complete by Monday"}	201	Newly created task object with id and status	201	Created task JSON	Pass	
TC_003	/tasks	POST	Create new task with missing title	{"description": "No title"}	400	Error message: "Title is required"	400	Error JSON	Pass	
TC_004	/tasks/1	GET	Get task by valid ID	None	200	Task object with id 1	200	Task JSON	Pass	
TC_005	/tasks/999	GET	Get task by invalid ID	None	404	Error message: "Task not found"	404	Error JSON	Pass	
TC_006	/tasks/1	PUT	Update task status	{"status": "completed"}	200	Updated task object	200	Updated task JSON	Pass	
TC_007	/tasks/999	PUT	Update nonexistent task	{"status": "pending"}	404	Error message: "Task not found"	404	Error JSON	Pass	
TC_008	/tasks/1	DELETE	Delete task by valid ID	None	200	Message: "Task deleted"	200	Deletion confirmation	Pass	
TC_009	/tasks/999	DELETE	Delete nonexistent task	None	200	Message: "Task deleted" (idempotent behavior)	200	Deletion confirmation	Pass	

- During the testing phase, no significant bugs were identified indicating a stable and well functioning API within the covered test scenarios. This reflects the robustness of the API implementation. However, testing should bee ongoing process including load testing, and exploratory testing.

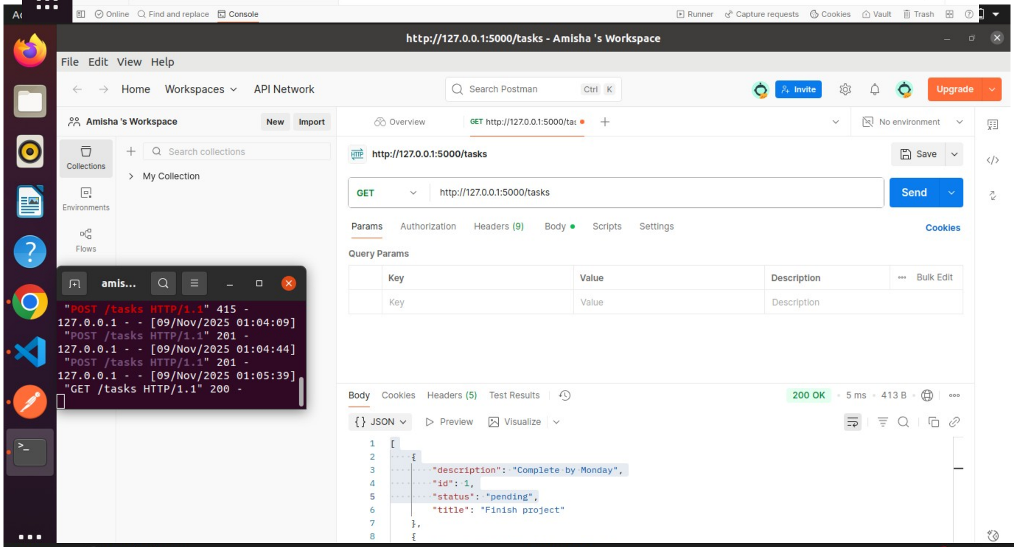


http://127.0.0.1:5000/tasks - Amisha 's Workspace

```
cd backendtest/
(venv) amisha@amisha-Lenovo-ideapad-330-15IKB:~/b
ackendtest$ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use
 it in a production deployment. Use a production
WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 467-301-940
127.0.0.1 - - [09/Nov/2025 00:38:17] "GET /tasks
HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 00:38:35] "POST /tasks
%09 HTTP/1.1" 404 -
127.0.0.1 - - [09/Nov/2025 01:00:25] "POST /tasks
 HTTP/1.1" 415 -
127.0.0.1 - - [09/Nov/2025 01:04:09] "POST /tasks
 HTTP/1.1" 201 -
127.0.0.1 - - [09/Nov/2025 01:04:44] "POST /tasks
 HTTP/1.1" 201 -
```

POST http://127.0.0.1:5000/tasks — 201 CREATED · 5 ms · 284 B

```
{
    "description": "Complete by Monday",
    "id": 2,
    "status": "pending",
    "title": "Finish testing project"
}
```

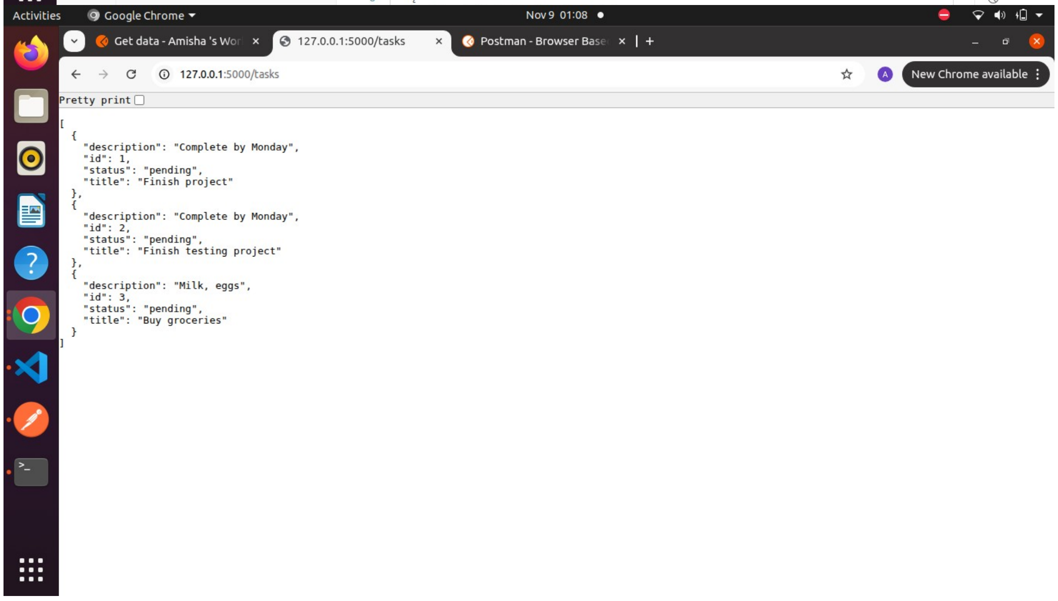


http://127.0.0.1:5000/tasks - Amisha 's Workspace

```
 "POST /tasks HTTP/1.1" 415 -
127.0.0.1 - - [09/Nov/2025 01:04:09]
 "POST /tasks HTTP/1.1" 201 -
127.0.0.1 - - [09/Nov/2025 01:04:44]
 "POST /tasks HTTP/1.1" 201 -
127.0.0.1 - - [09/Nov/2025 01:05:39]
 "GET /tasks HTTP/1.1" 200 -
```

GET http://127.0.0.1:5000/tasks — 200 OK · 5 ms · 413 B

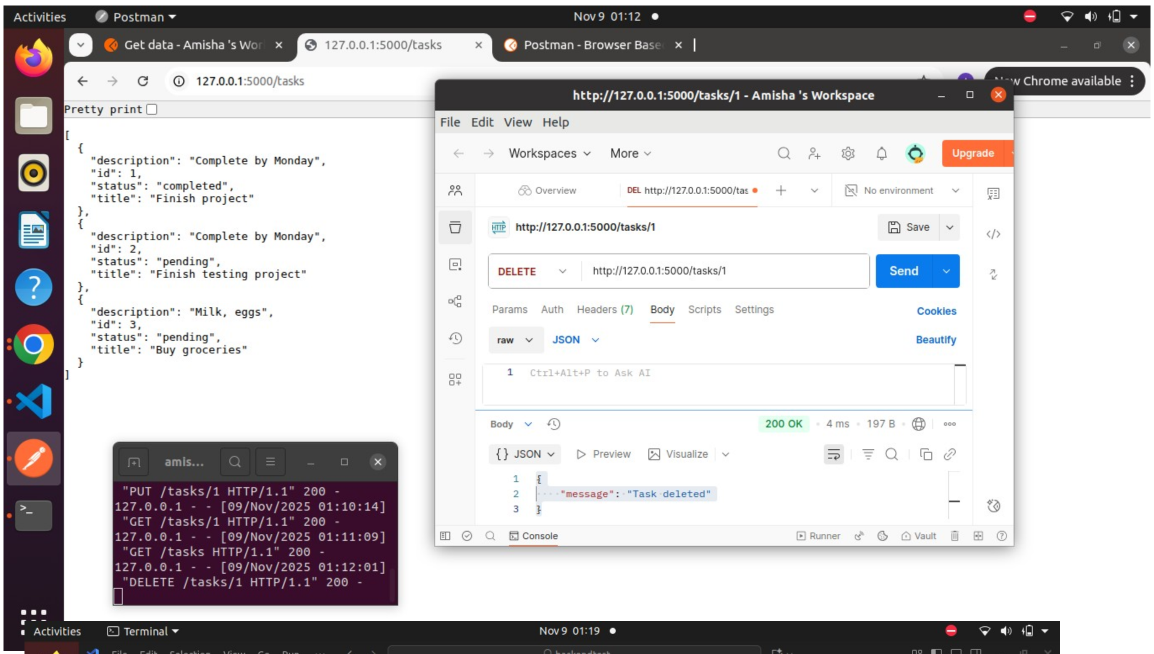
```
[
    {
        "description": "Complete by Monday",
        "id": 1,
        "status": "pending",
        "title": "Finish project"
    },
    {
```



127.0.0.1:5000/tasks

Pretty print ☐

```
[
  {
    "description": "Complete by Monday",
    "id": 1,
    "status": "pending",
    "title": "Finish project"
  },
  {
    "description": "Complete by Monday",
    "id": 2,
    "status": "pending",
    "title": "Finish testing project"
  },
  {
    "description": "Milk, eggs",
    "id": 3,
    "status": "pending",
    "title": "Buy groceries"
  }
]
```



127.0.0.1:5000/tasks

```
[
  {
    "description": "Complete by Monday",
    "id": 1,
    "status": "completed",
    "title": "Finish project"
  },
  {
    "description": "Complete by Monday",
    "id": 2,
    "status": "pending",
    "title": "Finish testing project"
  },
  {
    "description": "Milk, eggs",
    "id": 3,
    "status": "pending",
    "title": "Buy groceries"
  }
]
```

http://127.0.0.1:5000/tasks/1 - Amisha 's Workspace

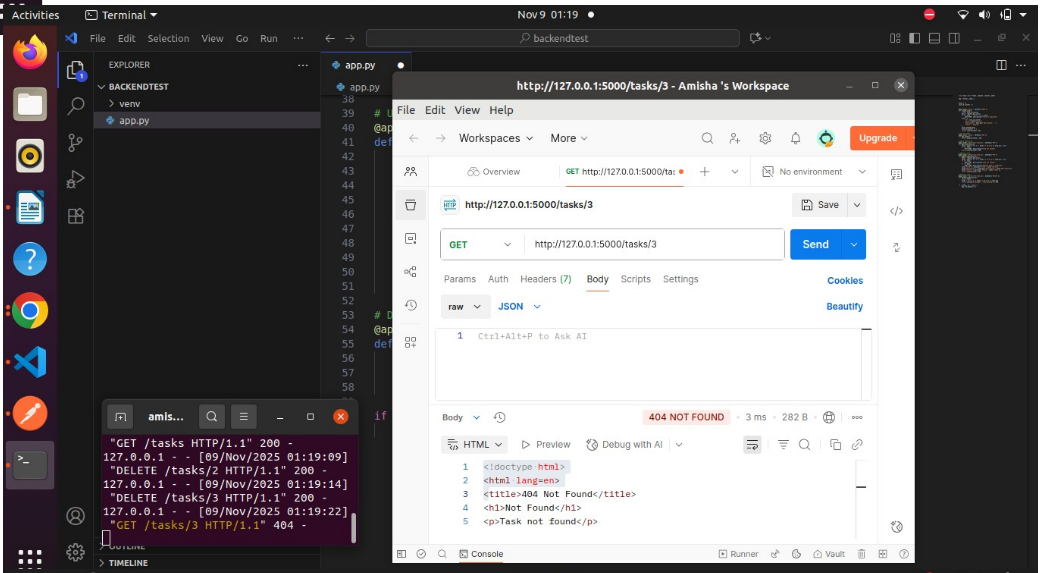
DELETE http://127.0.0.1:5000/tasks/1

Body: raw JSON

200 OK · 4 ms · 197 B

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

```
 "PUT /tasks/1 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:10:14]
 "GET /tasks/1 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:11:09]
 "GET /tasks HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:12:01]
 "DELETE /tasks/1 HTTP/1.1" 200 -
```



http://127.0.0.1:5000/tasks/3 - Amisha 's Workspace

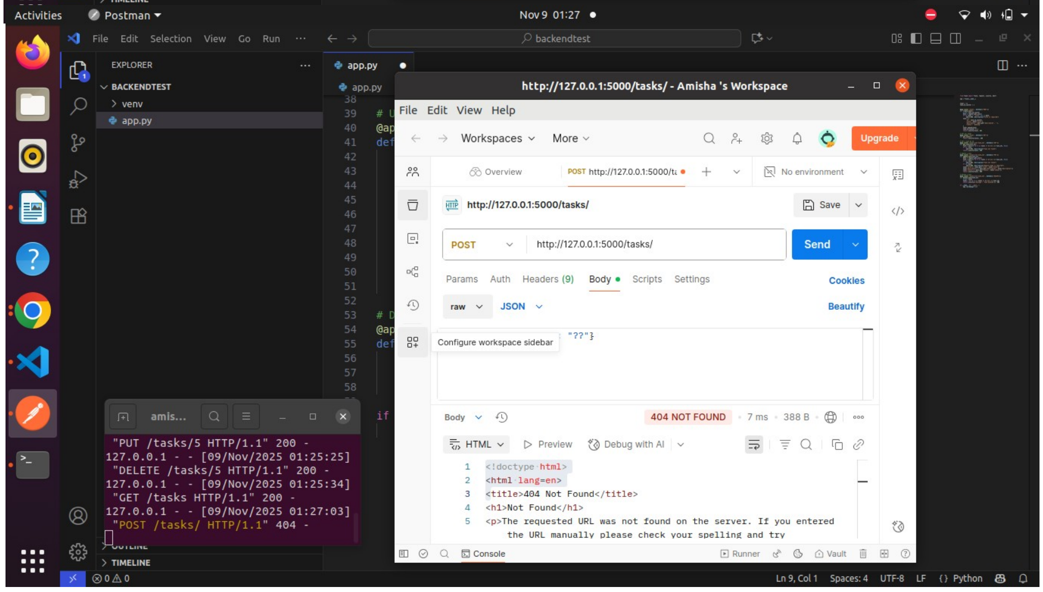
GET http://127.0.0.1:5000/tasks/3

Body: raw JSON

404 NOT FOUND · 3 ms · 282 B

```
<!doctype html>
<html lang=en>
<title>404 Not Found</title>
<h1>Not Found</h1>
<p>Task not found</p>
```

```
 "GET /tasks HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:19:09]
 "DELETE /tasks/2 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:19:14]
 "DELETE /tasks/3 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:19:22]
 "GET /tasks/3 HTTP/1.1" 404 -
```


Activities
Postman
Nov 9 01:27
File Edit Selection View Go Run
backendtest
EXPLORER
BACKENDTEST
venv
app.py
OUTLINE
TIMELINE
Configure workspace sidebar
Ln 9, Col 1 Spaces: 4 UTF-8 LF Python

http://127.0.0.1:5000/tasks/ - Amisha 's Workspace

POST http://127.0.0.1:5000/tasks/

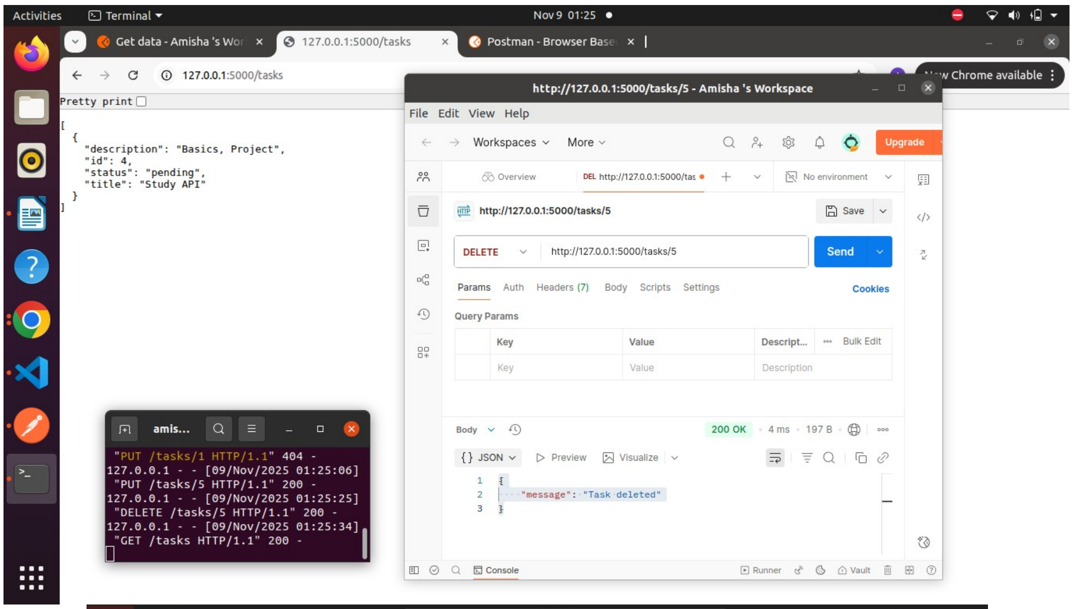
Body: raw JSON

```
"??"}
```

404 NOT FOUND · 7 ms · 388 B

```
<!doctype html>
<html lang=en>
<title>404 Not Found</title>
<h1>Not Found</h1>
<p>The requested URL was not found on the server. If you entered
   the URL manually please check your spelling and try
```

```
 "PUT /tasks/5 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:25:25]
 "DELETE /tasks/5 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:25:34]
 "GET /tasks HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:27:03]
 "POST /tasks/ HTTP/1.1" 404 -
```



127.0.0.1:5000/tasks

Pretty print

```
[
  {
    "description": "Basics, Project",
    "id": 4,
    "status": "pending",
    "title": "Study API"
  }
]
```

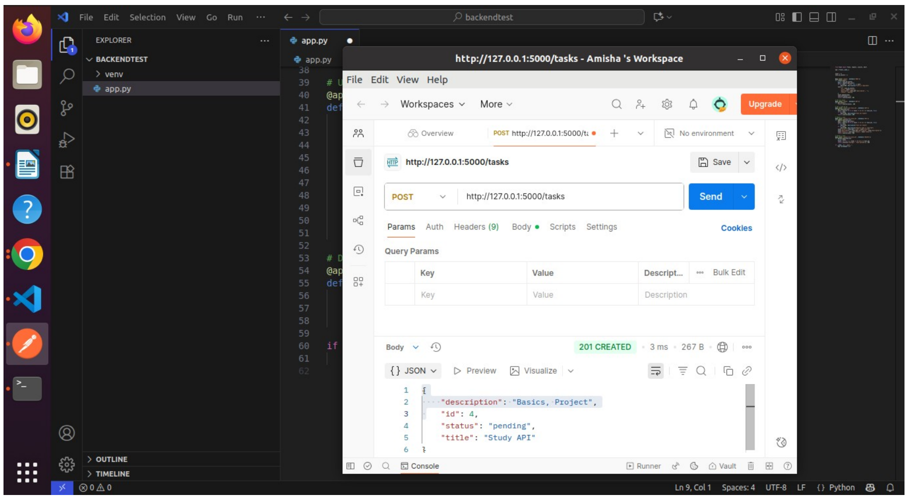
```
 "PUT /tasks/1 HTTP/1.1" 404 -
127.0.0.1 - - [09/Nov/2025 01:25:06]
 "PUT /tasks/5 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:25:25]
 "DELETE /tasks/5 HTTP/1.1" 200 -
127.0.0.1 - - [09/Nov/2025 01:25:34]
 "GET /tasks HTTP/1.1" 200 -
```

http://127.0.0.1:5000/tasks/5 - Amisha 's Workspace

DELETE http://127.0.0.1:5000/tasks/5

200 OK · 4 ms · 197 B

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

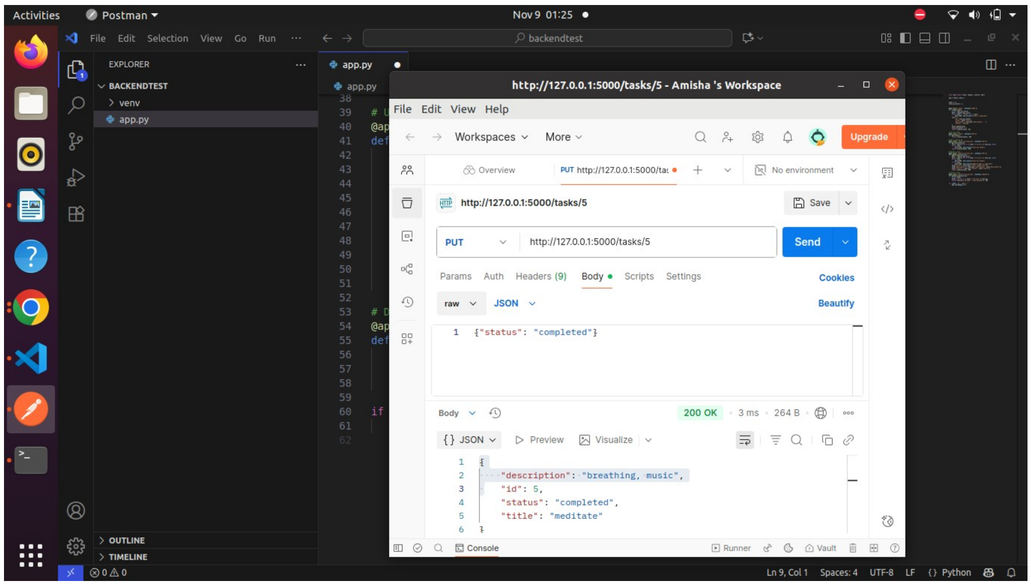


http://127.0.0.1:5000/tasks - Amisha 's Workspace

POST http://127.0.0.1:5000/tasks

201 CREATED · 3 ms · 267 B

```
{
    "description": "Basics, Project",
    "id": 4,
    "status": "pending",
    "title": "Study API"
}
```



http://127.0.0.1:5000/tasks/5 - Amisha 's Workspace

PUT http://127.0.0.1:5000/tasks/5

```
{"status": "completed"}
```

200 OK · 3 ms · 264 B

```
{
    "description": "breathing, music",
    "id": 5,
    "status": "completed",
    "title": "meditate"
}
```

Automation testing

Test cases:

test_get_tasks_empty(): Verify initial task list is empty.

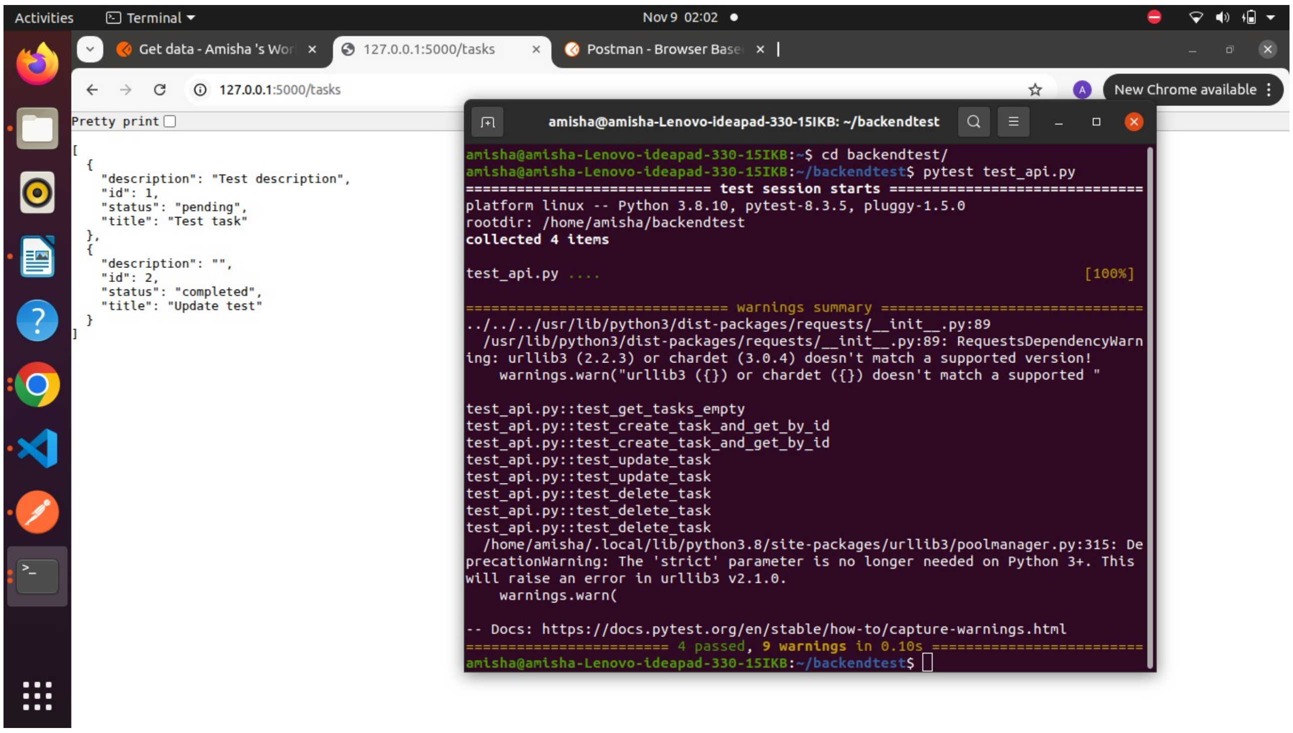
test_create_task_and_get_by_id(): Create a task and retrieve it by ID.

test_update_task(): Create task, update status, verify update.

test_delete_task(): Create task, delete it, and ensure deletion.

Sample Script

```
import requests

BASE_URL = "http://127.0.0.1:5000"

def test_get_tasks_empty():
resp = requests.get(f"{BASE_URL}/tasks")
assert resp.status_code == 200
assert resp.json() == []

def test_create_task_and_get_by_id():
data = {"title": "Test task", "description": "Test description"}
resp = requests.post(f"{BASE_URL}/tasks", json=data)
assert resp.status_code == 201
task = resp.json()
assert task["title"] == data["title"]
task_id = task["id"]

resp2 = requests.get(f"{BASE_URL}/tasks/{task_id}")
assert resp2.status_code == 200
assert resp2.json()["id"] == task_id

def test_update_task():
data = {"title": "Update test"}
resp = requests.post(f"{BASE_URL}/tasks", json=data)
task_id = resp.json()["id"]

update_data = {"status": "completed"}
resp2 = requests.put(f"{BASE_URL}/tasks/{task_id}", json=update_data)
assert resp2.status_code == 200
assert resp2.json()["status"] == update_data["status"]

def test_delete_task():
data = {"title": "Delete test"}
resp = requests.post(f"{BASE_URL}/tasks", json=data)
task_id = resp.json()["id"]

resp2 = requests.delete(f"{BASE_URL}/tasks/{task_id}")
assert resp2.status_code == 200
assert resp2.json()["message"] == "Task deleted"

resp3 = requests.get(f"{BASE_URL}/tasks/{task_id}")
assert resp3.status_code == 404
```



```
[
  {
    "description": "Test description",
    "id": 1,
    "status": "pending",
    "title": "Test task"
  },
  {
    "description": "",
    "id": 2,
    "status": "completed",
    "title": "Update test"
  }
]
```

```
amisha@amisha-Lenovo-ideapad-330-15IKB:~$ cd backendtest/
amisha@amisha-Lenovo-ideapad-330-15IKB:~/backendtest$ pytest test_api.py
============================ test session starts ============================
platform linux -- Python 3.8.10, pytest-8.3.5, pluggy-1.5.0
rootdir: /home/amisha/backendtest
collected 4 items

test_api.py ....                                                     [100%]

============================== warnings summary ==============================
../../../usr/lib/python3/dist-packages/requests/__init__.py:89
  /usr/lib/python3/dist-packages/requests/__init__.py:89: RequestsDependencyWarn
ing: urllib3 (2.2.3) or chardet (3.0.4) doesn't match a supported version!
    warnings.warn("urllib3 ({}) or chardet ({}) doesn't match a supported "

test_api.py::test_get_tasks_empty
test_api.py::test_create_task_and_get_by_id
test_api.py::test_create_task_and_get_by_id
test_api.py::test_update_task
test_api.py::test_update_task
test_api.py::test_delete_task
test_api.py::test_delete_task
test_api.py::test_delete_task
  /home/amisha/.local/lib/python3.8/site-packages/urllib3/poolmanager.py:315: De
precationWarning: The 'strict' parameter is no longer needed on Python 3+. This
will raise an error in urllib3 v2.1.0.
    warnings.warn(

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
======================== 4 passed, 9 warnings in 0.10s ========================
amisha@amisha-Lenovo-ideapad-330-15IKB:~/backendtest$
```

Summary of Findings

The API functions as intended for basic CRUD operations. Most endpoints respond with correct status codes and data formats. Error handling can be enhanced for non-existent resource IDs. Postman manual tests confirm usability and quick feedback loops for API testing.