Basic Task Management REST API

REST Concepts Implemented

This API follows RESTful design principles:

- **Statelessness**: Each request includes all required information.
- **Resource-based URLs**: /tasks represents the task collection.
- **HTTP Methods**: GET (retrieve), POST (create), PUT (update), DELETE (remove).
- **JSON Format**: Consistent data exchange in JSON via jsonify.
- **Status Codes**: Proper use of HTTP response codes (200, 201, 400, 404, 500).

API Design Highlights

- **Unique Identifiers**: Tasks are identified using UUIDs.
- **Timestamps**: Each task tracks created_at and updated_at.
- **Input Validation**: Enforces required fields and valid status values.
- **Error Handling**: Provides structured error responses with success flags.
- **Filtering**: Query parameters enable status-based filtering.
- **Documentation**: Self-documenting endpoints with API info.

Quick Start

```
pip install flask
python restapi_IMT2023540.py
```

Access API at: http://localhost:5000

API Demonstration & Testing

API Server Status Flask server is running on http://localhost:5000

Available Endpoints

1) Check API Root Info

curl -X GET http://127.0.0.1:5000/

Purpose: Get API documentation and available endpoints

2) Health Check
curl -X GET http://127.0.0.1:5000/status

Purpose: Check if the API server is running properly

3) Fetch All Tasks

curl -X GET http://127.0.0.1:5000/tasks

Purpose: Retrieve all tasks in the system

4) Fetch Task by ID

curl -X GET http://127.0.0.1:5000/tasks/<task_id>

Purpose: Retrieve a specific task by its unique ID

5) Create New Task

curl -X POST http://127.0.0.1:5000/tasks \

-H "Content-Type: application/json" \

-d '{"name": "Read REST API paper", "details": "Understand RESTful constraints", "state": "pending"}'

Purpose: Create a new task (title is required, description and status are optional)

6) Update Task

curl -X PUT http://127.0.0.1:5000/tasks/<task_id> \

-H "Content-Type: application/json" \

-d '{"name": "Updated Task Name", "state": "completed"}'

Purpose: Update an existing task (all fields are optional)

7) Delete a Task

curl -X DELETE http://127.0.0.1:5000/tasks/<task_id>

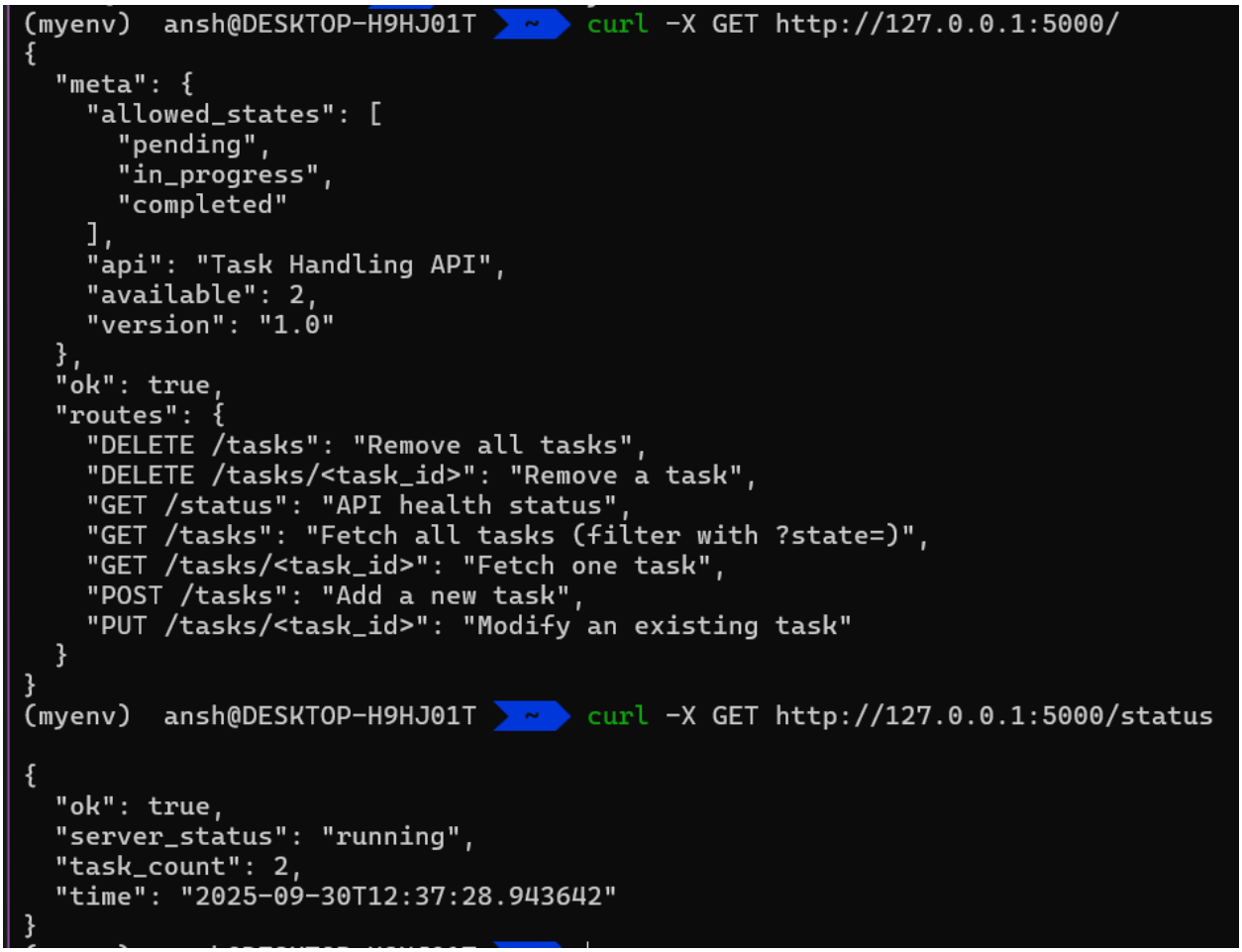
Purpose: Delete a specific task by its ID

8) Delete all Tasks

curl -X DELETE http://127.0.0.1:5000/tasks

Purpose: Delete all tasks (bonus endpoint)

Get and health:



All tasks and task by id:

```
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X GET http://127.0.0.1:5000/tasks
{
  "items": [
    {
      "created": "2025-09-30T10:00:00",
      "details": "Write Flask REST API with CRUD endpoints",
      "modified": "2025-09-30T10:00:00",
      "name": "Finish API homework",
      "state": "in_progress",
      "task_id": "5785781a-3372-43ab-a1c2-1a6d38c66d18"
    },
    {
      "created": "2025-09-30T09:00:00",
      "details": "Revise Flask, REST fundamentals",
      "modified": "2025-09-30T09:00:00",
      "name": "Prepare for SE exam",
      "state": "pending",
      "task_id": "5b214fb6-f481-4cd1-8808-5dc546d7c403"
    }
  ],
  "ok": true,
  "total": 2
}
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X GET http://127.0.0.1:5000/tasks/
\5785781a-3372-43ab-a1c2-1a6d38c66d18
{
  "ok": true,
  "task": {
    "created": "2025-09-30T10:00:00",
    "details": "Write Flask REST API with CRUD endpoints",
    "modified": "2025-09-30T10:00:00",
    "name": "Finish API homework",
    "state": "in_progress",
    "task_id": "5785781a-3372-43ab-a1c2-1a6d38c66d18"
  }
}
```

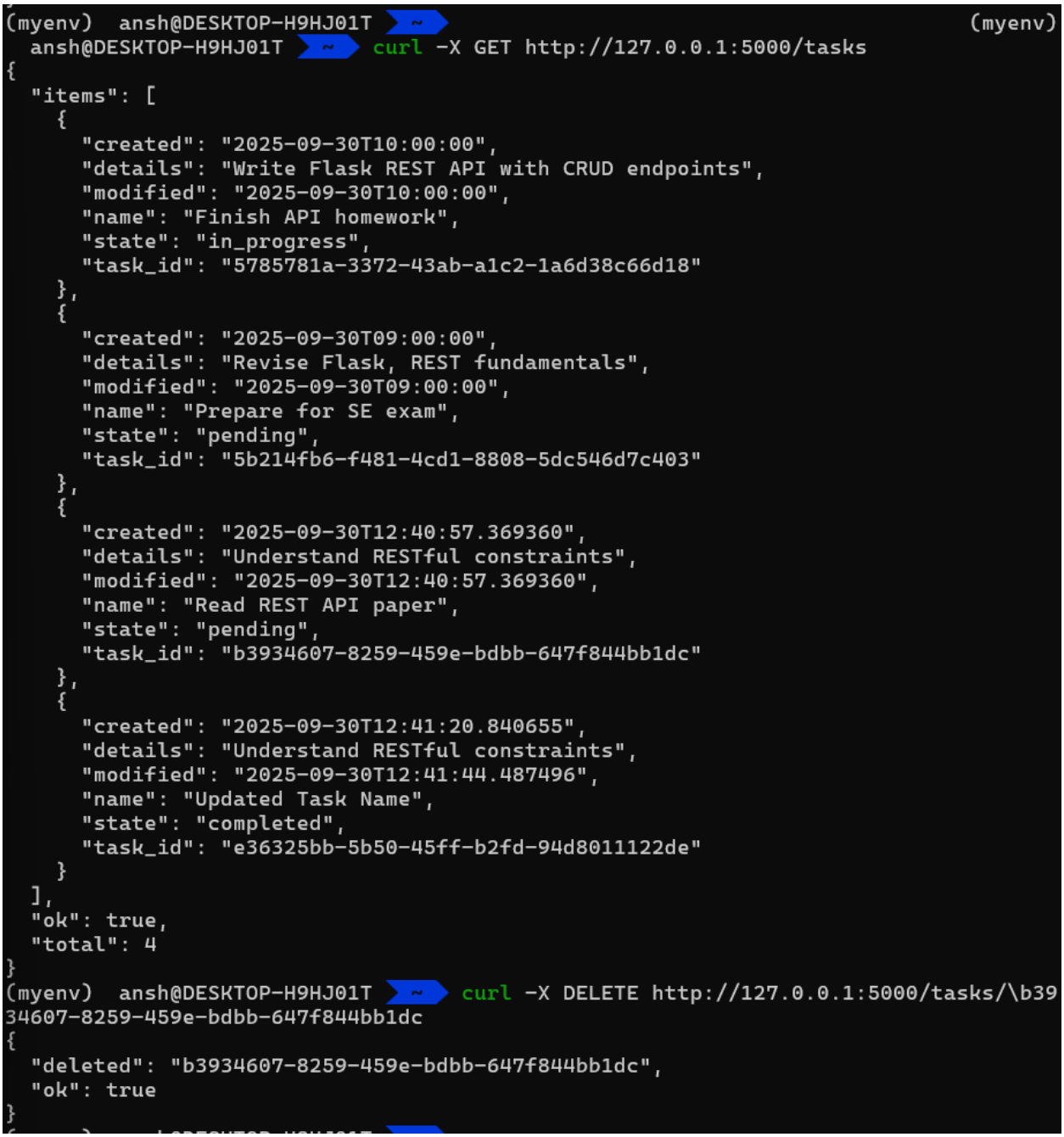
Create and update task:

```
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X POST http://127.0.0.1:5000/tasks
 \
-H "Content-Type: application/json" \
-d '{"name": "Read REST API paper", "details": "Understand RESTful constrain
ts", "state": "pending"}'

{
  "ok": true,
  "task": {
    "created": "2025-09-30T12:41:20.840655",
    "details": "Understand RESTful constraints",
    "modified": "2025-09-30T12:41:20.840655",
    "name": "Read REST API paper",
    "state": "pending",
    "task_id": "e36325bb-5b50-45ff-b2fd-94d8011122de"
  }
}
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X PUT http://127.0.0.1:5000/tasks/
\e36325bb-5b50-45ff-b2fd-94d8011122de \
-H "Content-Type: application/json" \
-d '{"name": "Updated Task Name", "state": "completed"}'

{
  "ok": true,
  "task": {
    "created": "2025-09-30T12:41:20.840655",
    "details": "Understand RESTful constraints",
    "modified": "2025-09-30T12:41:44.487496",
    "name": "Updated Task Name",
    "state": "completed",
    "task_id": "e36325bb-5b50-45ff-b2fd-94d8011122de"
  }
}
```

Check if the task is actually added and then test delete:



Check if the task is deleted and then delete all:

```
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X GET http://127.0.0.1:5000/tasks
{
  "items": [
    {
      "created": "2025-09-30T10:00:00",
      "details": "Write Flask REST API with CRUD endpoints",
      "modified": "2025-09-30T10:00:00",
      "name": "Finish API homework",
      "state": "in_progress",
      "task_id": "5785781a-3372-43ab-a1c2-1a6d38c66d18"
    },
    {
      "created": "2025-09-30T09:00:00",
      "details": "Revise Flask, REST fundamentals",
      "modified": "2025-09-30T09:00:00",
      "name": "Prepare for SE exam",
      "state": "pending",
      "task_id": "5b214fb6-f481-4cd1-8808-5dc546d7c403"
    },
    {
      "created": "2025-09-30T12:41:20.840655",
      "details": "Understand RESTful constraints",
      "modified": "2025-09-30T12:41:44.487496",
      "name": "Updated Task Name",
      "state": "completed",
      "task_id": "e36325bb-5b50-45ff-b2fd-94d8011122de"
    }
  ],
  "ok": true,
  "total": 3
}
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X DELETE http://127.0.0.1:5000/tasks
{
  "ok": true,
  "removed_count": 3
}
(myenv)  ansh@DESKTOP-H9HJ01T  ~  curl -X GET http://127.0.0.1:5000/tasks
{
  "items": [],
  "ok": true,
  "total": 0
}
```