

To-Do List Project (Django)

A To-Do List web application with RESTful APIs and Bootstrap-based web interface.
Built using Django with raw SQL queries (no ORM) as per requirements.

Features

1. Add, view, update, and delete tasks
 2. RESTful API endpoints for CRUD operations
 3. Bootstrap web interface for user-friendly interaction
 4. Validations for required fields & correct date format (YYYY-MM-DD)
 5. Default task status set to Pending
 6. Separate apps for API and Web Interface
 7. SQLite database for storage
-

Setup Instructions

1.Clone Repository

```
git clone https://github.com/DILIP3524/todoapp.git  
cd todoapp
```

2.Create Virtual Environmnt

```
python -m venv venv  
source venv/bin/activate (Linux/Mac)  
venv\Scripts\activate (Windows)
```

3.Install Dependencies

```
pip install -r requirements.txt
```

Start Server

python manage.py runserver

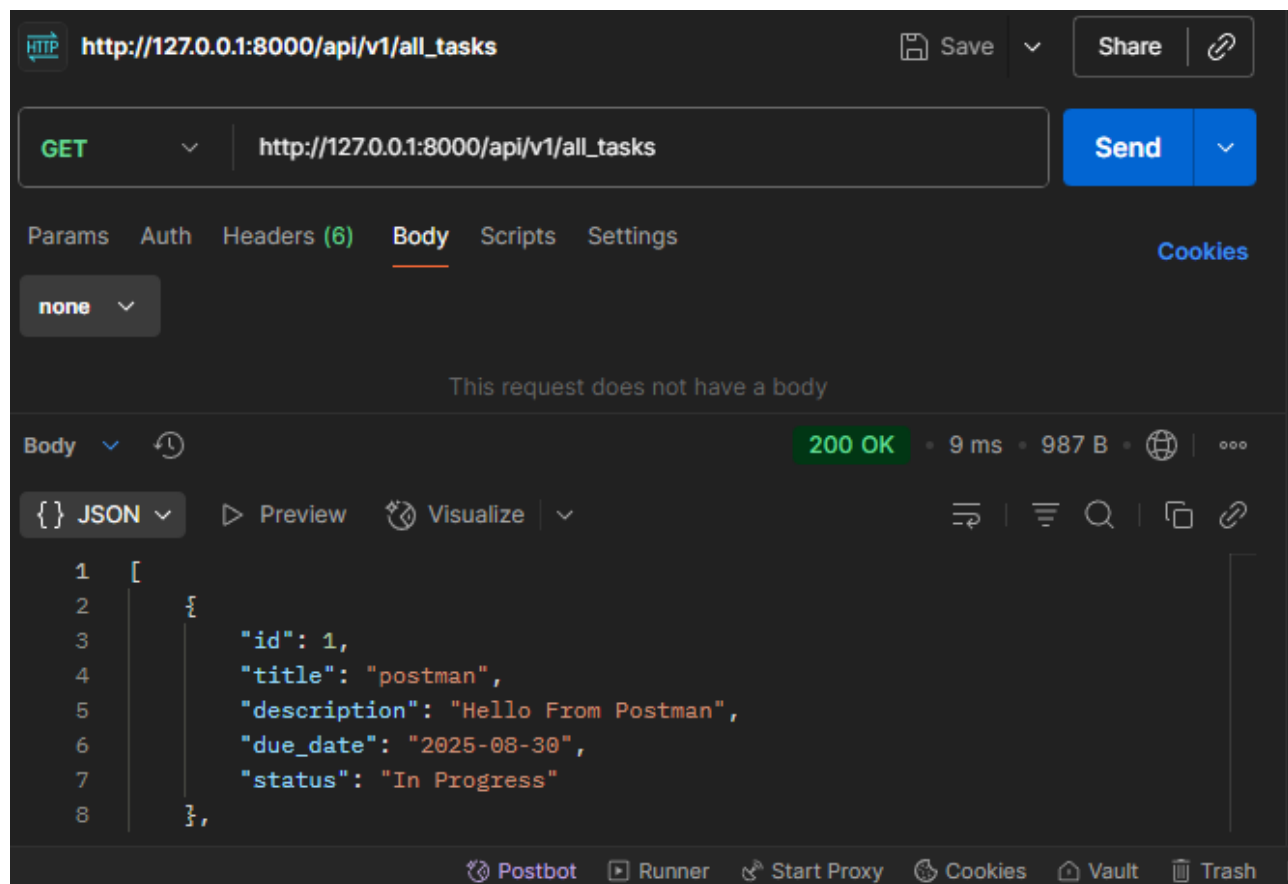
The project will be live at: <http://127.0.0.1:8000/>

API Endpoints

Get All Tasks

Endpoint :- [GET /api/v1/all_tasks/](http://127.0.0.1:8000/api/v1/all_tasks/)

Response :-



Create a Task

Endpoint :- [POST /api/v1/all_tasks/](http://127.0.0.1:8000/api/v1/all_tasks/)

```
Body: {
  "title": "Task",
  "description": "Details",
  "due_date": "2025-08-15"
}
```

Response:-

HTTP **http://127.0.0.1:8000/api/v1/all_tasks** Save Share

POST **http://127.0.0.1:8000/api/v1/all_tasks** Send

Params Auth Headers (8) **Body** Scripts Settings Cookies Beautify

raw JSON

```
1 {
2   "title": "Create Salary Report",
3   "description": "Create Salary Report For This Month",
4   "due_date": "2025-08-10"
5 }
```

Body 201 Created • 140 ms • 403 B • Preview Visualize

```
1 {
2   "message": "The Task Added Succesfully and Default Task Status Is Pending"
3 }
```

Get Single Task

Endpoint :- **GET** /api/v1/get_task/<pk>/

Response:-

HTTP **http://127.0.0.1:8000/api/v1/get_task/1** Save Share

GET **http://127.0.0.1:8000/api/v1/get_task/1** Send

Params Auth Headers (8) **Body** Scripts Settings Cookies Beautify

raw JSON

```
1 {
2   "id": 1,
3   "title": "postman",
4   "description": "Hello From Postman",
5   "due_date": "2025-08-30",
6   "status": "In Progress"
7 }
```

Body 200 OK • 3 ms • 448 B • Preview Visualize

Update a Task

Endpoint :- PUT [/api/v1/get_tasks/<pk>/](#)

Body: {

```
"description": "Create Salary Report For This Month Only for Management Team",  
"status": "In Progress"  
}
```

Response:-

The screenshot displays a REST client interface with the following details:

- URL:** `http://127.0.0.1:8000/api/v1/get_task/9`
- Method:** PUT
- Body (JSON):**

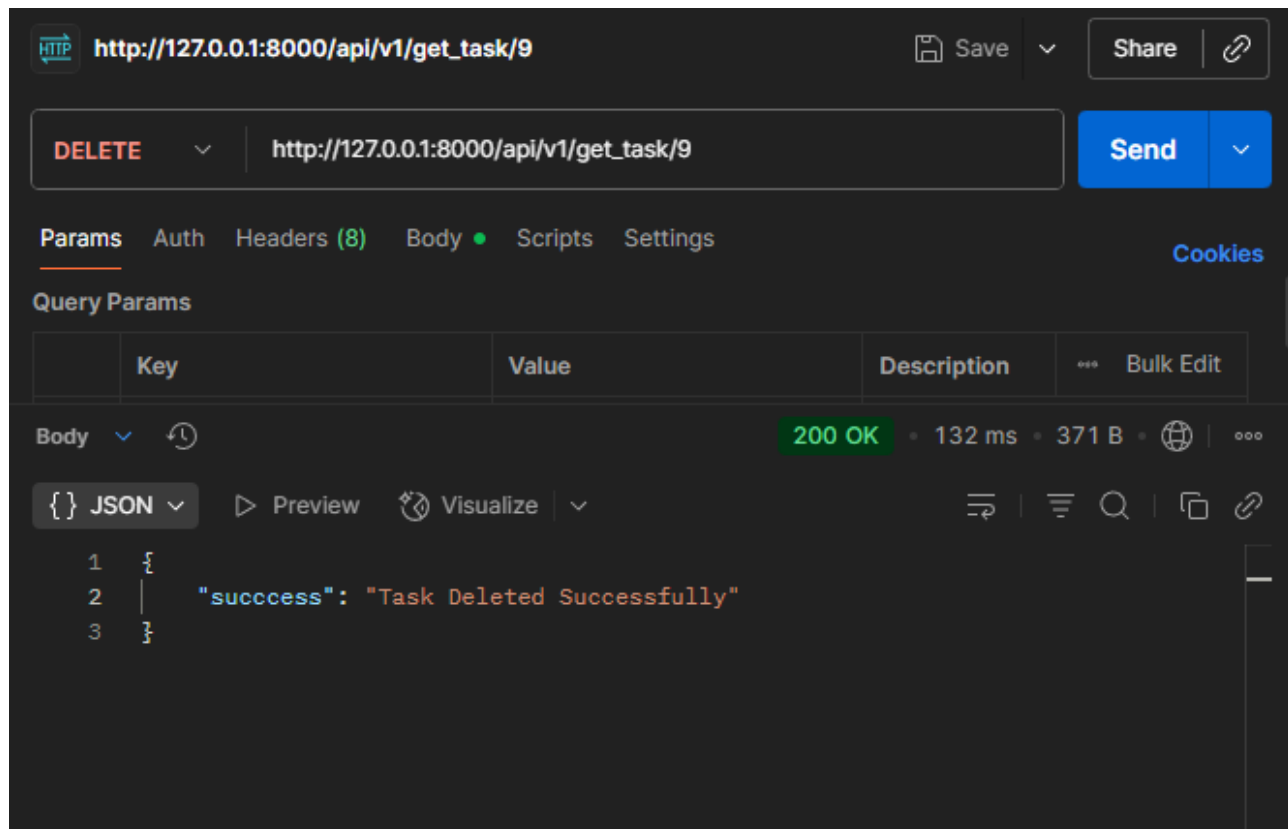
```
{  
  "description": "Create Salary Report For This Month Only for Management Team",  
  "status": "In Progress"  
}
```
- Response:** 200 OK, 124 ms, 370 B
- Response Body (JSON):**

```
{  
  "message": "Task updated successfully"  
}
```

Delete a Task

Endpoint :- **DELETE** [/api/v1/get_tasks/<pk>/](http://127.0.0.1:8000/api/v1/get_tasks/<pk>/)

Response:-



Web Interface

Navigate to:- <http://127.0.0.1:8000/>

Add tasks via form

View tasks in a table

Update or delete tasks

Tech Stack

Backend: Django 5, Python 3

Database: SQLite

Frontend: Bootstrap 5

Testing: Django Test Client / Pytest

Version Control: Git

Testing

Run All Tests:- *pytest*

Tests cover task creation, validation, retrieval, update, and deletion.