Task Management API Documentation

Welcome to Task Management Rest Framework API in Python Flask. This API provides you the functionality of to perform CRUD (Create, Read, Update, and Delete) operations on tasks. Each task has properties such as ID, Title, Description, Due Date, and Status.

Libraries and Utilities

Task Management API is build using Python Flask.

Following libraries are used to create this Application:

- Flask (version=2.3.2): **Flask** is a lightweight web framework for Python that allows you to build web applications quickly and easily.
- jsonify(version= 0.5): **jsonify** is a function in the Flask library that converts a Python object into a JSON response, making it easy to return JSON data from a Flask route.
- request(version= 2.31.0): The **request** library in Flask provides access to incoming request data, allowing you to retrieve and manipulate data sent by the client during an HTTP request.

API Endpoints

1. Create a new task

   Endpoint: `POST /todo`

   Create a new task with the specified properties. The request body should be in JSON.

2. Retrieve a single task by its ID

   Endpoint: `GET /todo/ {id}`

   Retrieve a task by its unique ID. Replace `{id}` in the URL with the ID of the task you want to retrieve.

3. Update an existing task

   Endpoint: `PUT /todo/ {id}`

   Update an existing task with the specified ID. Replace `{id}` in the URL with the ID of the task you want to update. The request body should be in JSON format.

4. Delete a task

   Endpoint: `DELETE /todo/ {id}`

   Delete a task with the specified ID. Replace `{id}` in the URL with the ID of the task you want to delete.

5. List all tasks

   Endpoint: `GET /todo`

   Retrieve a list of all tasks available in the system.

6. Pagination

   Endpoint: `GET /todo?page={number}`

   It help in Pagination of tasks.

Explanation

1. **Create a new task**

   Create a new task with the specified properties. The request body should be in JSON format and include the following fields:

   title (string, required): The title of the task.

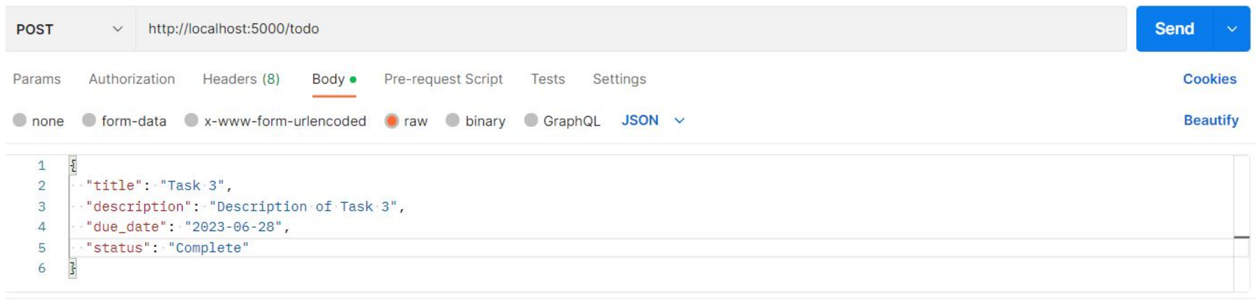
   description (string, optional): The description of the task.

   due_date (string, optional): The due date of the task (format: "YYYY-MM-DD").

   status (string, optional): The status of the task. Possible values are "Incomplete", "In Progress", or "Completed". Default value is "Incomplete".

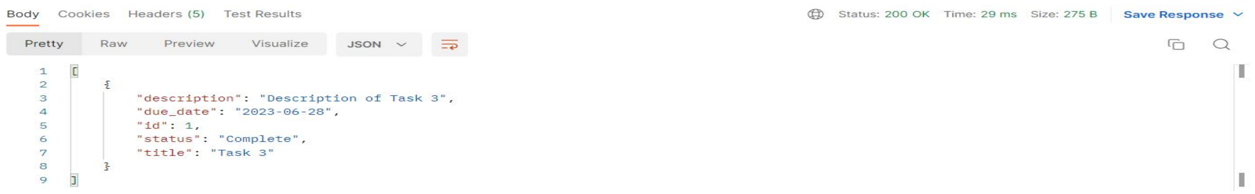
   Request = **POST**, **/todo**

   Example:


POST
http://localhost:5000/todo
Send
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Beautify

```
{
  "title": "Task 3",
  "description": "Description of Task 3",
  "due_date": "2023-06-28",
  "status": "Complete"
}
```

   Output:

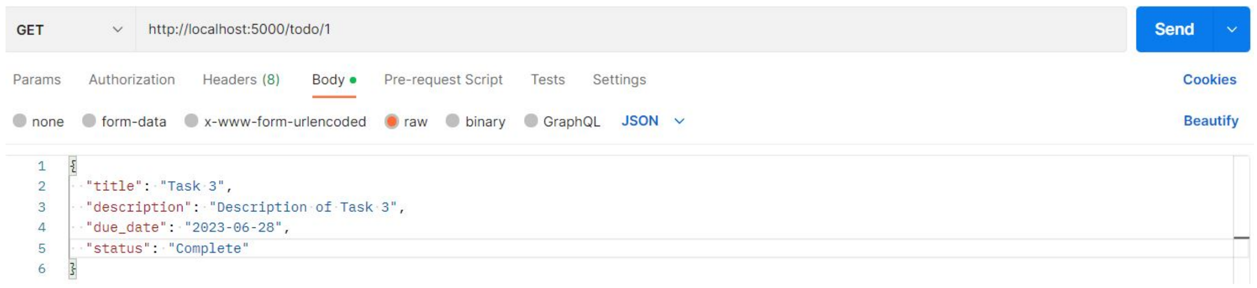

Body Cookies Headers (5) Test Results
Status: 200 OK Time: 29 ms Size: 275 B
Save Response
Pretty Raw Preview Visualize JSON

```
[
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 1,
        "status": "Complete",
        "title": "Task 3"
    }
]
```

2. **Retrieve a Single request by id:**
   Request= **GET, /todo/ {id}**

   Retrieve a task by its ID. Replace `{id}` in the URL with the ID of the task you want to retrieve.

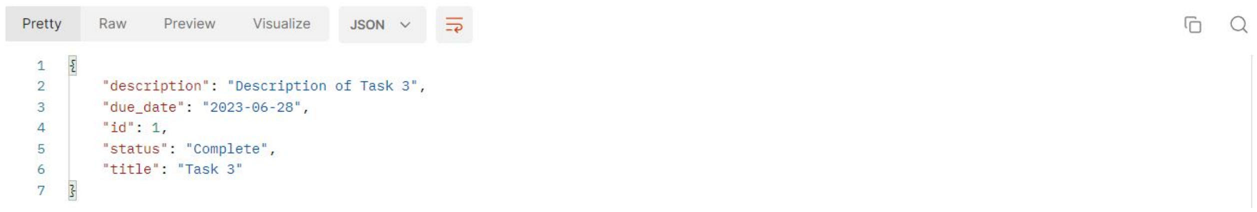
   Example Request:



```
http://localhost:5000/todo/1
```

```
{
  "title": "Task 3",
  "description": "Description of Task 3",
  "due_date": "2023-06-28",
  "status": "Complete"
}
```

   Output:



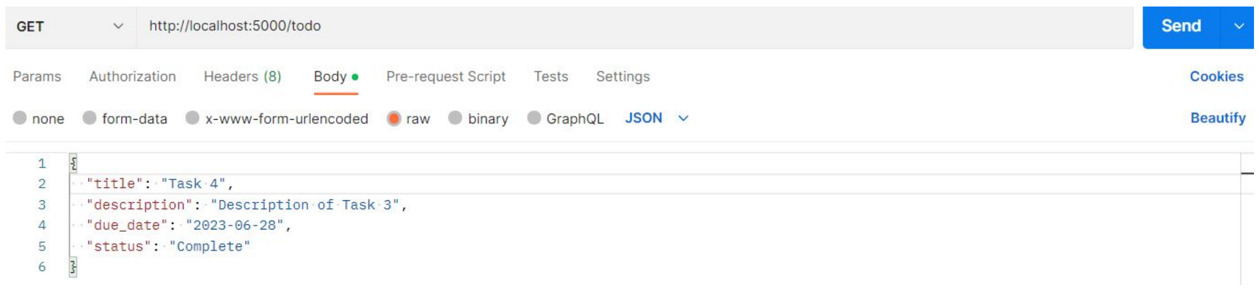
```
{
    "description": "Description of Task 3",
    "due_date": "2023-06-28",
    "id": 1,
    "status": "Complete",
    "title": "Task 3"
}
```

3. **Get all the request**

   Request= **GET, /todo**

   It is used to retrieve all the Tasks.

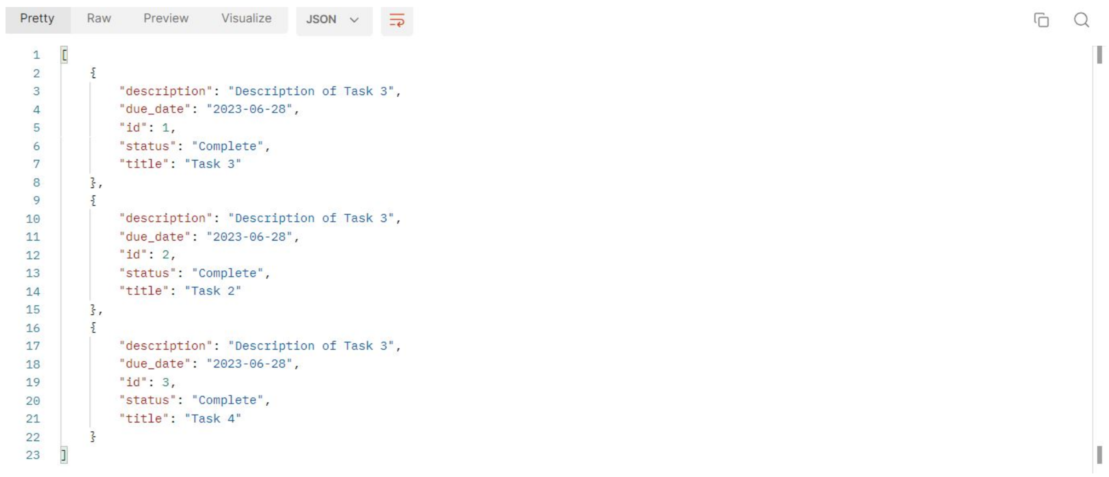
   Example:



```
http://localhost:5000/todo
```

```
{
  "title": "Task 4",
  "description": "Description of Task 3",
  "due_date": "2023-06-28",
  "status": "Complete"
}
```

   Output:



```
[
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 1,
        "status": "Complete",
        "title": "Task 3"
    },
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 2,
        "status": "Complete",
        "title": "Task 2"
    },
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 3,
        "status": "Complete",
        "title": "Task 4"
    }
]
```

4. Update task by id

Request: **PUT, /todo/{id}**

Update an existing task with the specified ID. Replace `{id}` in the URL with the ID of the task you want to update. The request body should be in JSON format and can include the following fields:

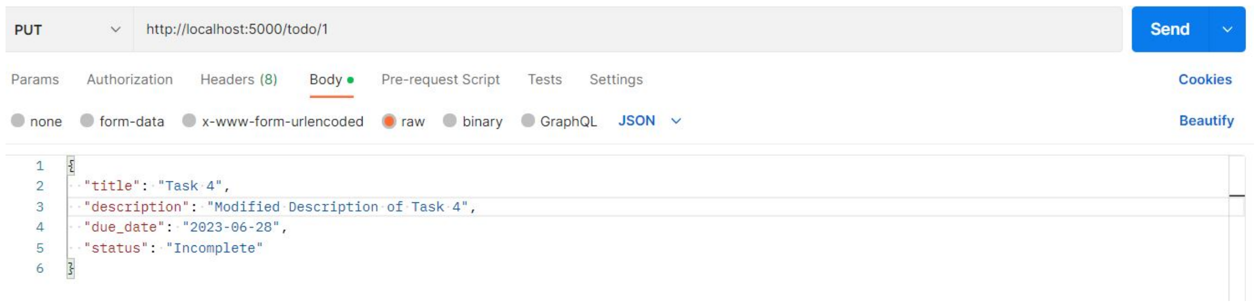
`title` (string, optional): The updated title of the task.

`description` (string, optional): The updated description of the task.

`due_date` (string, optional): The updated due date of the task (format: "YYYY-MM-DD").

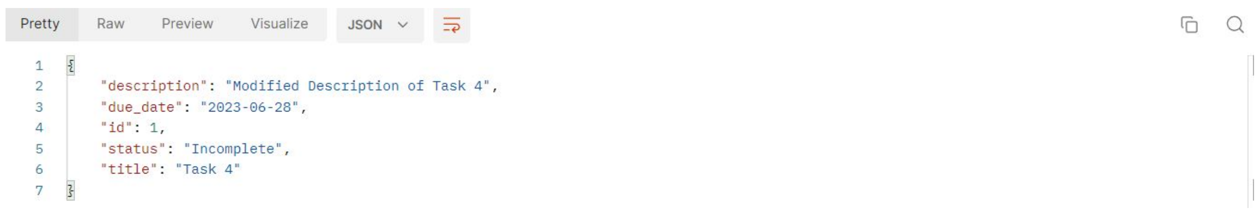
`status` (string, optional): The updated status of the task. Possible values are "Incomplete", "In Progress", or "Completed".

Example:



```
{
  "title": "Task 4",
  "description": "Modified Description of Task 4",
  "due_date": "2023-06-28",
  "status": "Incomplete"
}
```

Output:



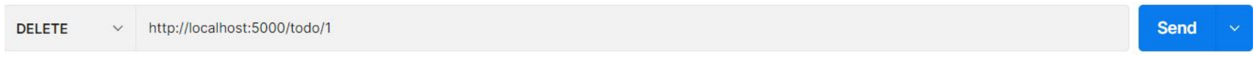
```
{
    "description": "Modified Description of Task 4",
    "due_date": "2023-06-28",
    "id": 1,
    "status": "Incomplete",
    "title": "Task 4"
}
```

5. **Delete a Task**

   Request=**DELETE, /todo/{id}**

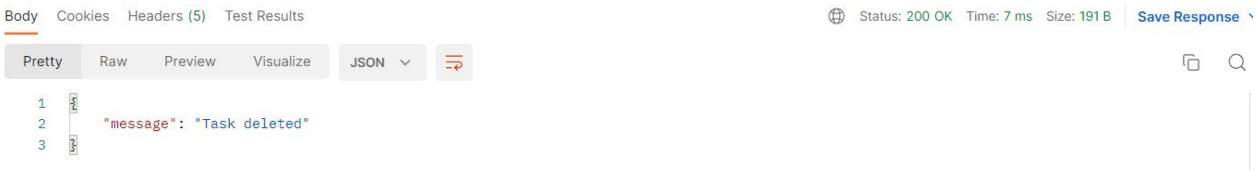
   Delete a task with the specified ID. Replace `{id}` in the URL with the ID of the task you want to delete.

   Example:



```
http://localhost:5000/todo/1
```

   Output:



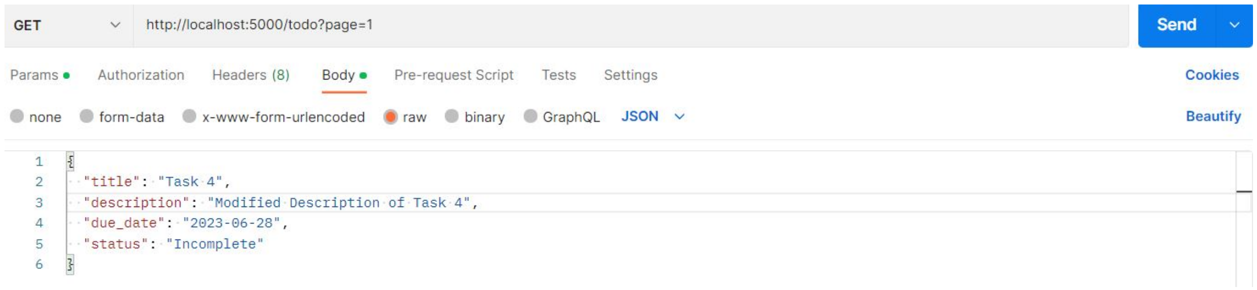
Status: 200 OK Time: 7 ms Size: 191 B

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

6. **Pagination:**

   Request: **GET, /todo? page=1**

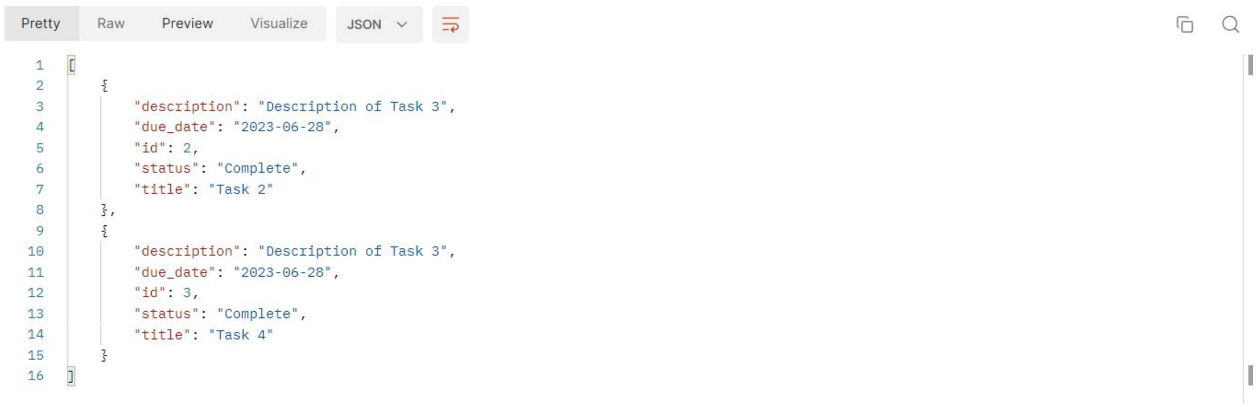
   Example:



```
http://localhost:5000/todo?page=1
```

```
{
  "title": "Task 4",
  "description": "Modified Description of Task 4",
  "due_date": "2023-06-28",
  "status": "Incomplete"
}
```

   Output:



```
[
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 2,
        "status": "Complete",
        "title": "Task 2"
    },
    {
        "description": "Description of Task 3",
        "due_date": "2023-06-28",
        "id": 3,
        "status": "Complete",
        "title": "Task 4"
    }
]
```