TODO-LIST-APP

API DOCUMENTATION

INDEX

GET /backend-api/todos/

Retrieve a list of all todo items, optionally filtered by title or completion status.

Authentication Requirements

- **No authentication required**
- Accessible to all users

Request Parameters

Parameter	Type	Description	Required
title	String	Filter todos by title (case-insensitive)	No
completed	Boolean	Filter todos by completion status (true/false)	No

Example Request

text

```
GET /backend-api/todos/?title=example&completed=true
Host: your-api-domain.com
Accept: application/json
```

Response Structure

Successful response (200 OK):

json

```
[
  {
    "id": 1,
    "title": "Example Todo",
    "completed": true,
    "created_at": "2025-03-14T06:40:47.307Z",
    "completed_at": "2025-03-14T06:40:47.307Z"
  }
```

]

Error Responses:

- **400 Bad Request:** Invalid filter parameters.
- **500 Internal Server Error:** Server-side failure.

Filtering Logic

The filtering is powered by the TodoFilter class:

- **Title:** Partial matches are supported using a case-insensitive search (icontains).
- **Completed:** Filters todos based on the completed field (True for completed, False for incomplete).

Best Practices for Filtering

1. Use the title filter for searching specific tasks or narrowing down results based on partial matches.
2. Combine filters (title, completed) to retrieve more targeted results.
3. Handle empty responses gracefully when no matching items are found.
4. Validate query parameters client-side before making requests to avoid unnecessary errors (e.g., invalid values for completed).

GET /backend-api/todos/{id}/

Retrieve details of a single todo item by its ID.

Authentication Requirements

- **No authentication required**
- Accessible to all users

Request Parameters

No additional parameters required. The id is passed in the URL.

Example Request

text

GET /backend-api/todos/1/

Host: your-api-domain.com

Accept: application/json

Response Structure

Successful response (200 OK):

json

```
{
  "id": 1,
  "title": "Example Todo",
  "completed": true,
  "created_at": "2025-03-14T06:40:47.307Z",
  "completed_at": "2025-03-14T06:40:47.307Z"
}
```

Error Responses:

- **404 Not Found:** Todo item does not exist.
- **500 Internal Server Error:** Server-side failure.

POST /backend-api/todos/

Create a new todo item.

Authentication Requirements

- **No authentication required**

Request Body Parameters

Field	Type	Description	Required
title	String	Title of the todo item	Yes
completed	Boolean	Initial completion status	No

Example Request

text

```
POST /backend-api/todos/
Host: your-api-domain.com
Content-Type: application/json


{
  "title": "New Todo",
  "completed": false
}
```

Response Structure

Successful response (201 Created):

json

```
{
  "id": 2,
  "title": "New Todo",
  "completed": false,
  "created_at": "2025-03-14T06:50:00.000Z",
  "completed_at": null
}
```

Error Responses:

- **400 Bad Request:** Invalid request data.
- **500 Internal Server Error:** Server-side failure.

PATCH /backend-api/todos/{id}/

Partially update an existing todo item by its ID.

Authentication Requirements

- **No authentication required**

Request Body Parameters

Field	Type	Description	Required
title	String	Updated title	No
completed	Boolean	Updated completion status	No

Example Request

text

```
PATCH /backend-api/todos/1/
Host: your-api-domain.com
Content-Type: application/json

{
  "completed": true
}
```

Response Structure

Successful response (200 OK):

json

```
{
  "id": 1,
  "title": "Example Todo",
  "completed": true,
  "created_at": "2025-03-14T06:40:47.307Z",
  "completed_at": "2025-03-14T07:00:00.000Z"
}
```

Error Responses:

- **400 Bad Request:** Invalid request data.

- **404 Not Found:** Todo item does not exist.
- **500 Internal Server Error:** Server-side failure.

DELETE /backend-api/todos/{id}/

Delete a todo item by its ID.

Authentication Requirements

- **No authentication required**

Example Request

```
text

DELETE /backend-api/todos/1/

Host: your-api-domain.com

Accept: application/json
```

Response Structure

Successful response (204 No Content): No body returned.

Error Responses:

- **404 Not Found:** Todo item does not exist.
- **500 Internal Server Error:** Server-side failure.

POST /backend-api/todos/{id}/complete/

Mark a todo item as completed, setting the completed_at timestamp to the current time.

Authentication Requirements

- **No authentication required**

Example Request

```
text

POST /backend-api/todos/1/complete/

Host: your-api-domain.com
```

Accept: application/json

Response Structure

Successful response (200 OK):

```
json
{
  "id": 1,
  "title": "Example Todo",
  "completed": true,
  "created_at": "2025-03-14T06:40:47.307Z",
  "completed_at": "2025-03-14T07:00:00.000Z"
}
```

Error Responses:

- **404 Not Found:** Todo item does not exist.
- **500 Internal Server Error:** Server-side failure.

POST /backend-api/todos/{id}/incomplete/

Mark a todo item as incomplete, clearing the completed_at timestamp.

Authentication Requirements

- **No authentication required**

Example Request

```
text
POST /backend-api/todos/1/incomplete/
Host: your-api-domain.com
Accept: application/json
```

Response Structure

Successful response (200 OK):

```
json
{
  "id": 1,
  "title": "Example Todo",
  "completed": false,
  "created_at": "2025-03-14T06:40:47.307Z",
  "completed_at": null
}
```

Error Responses:

- **404 Not Found:** Todo item does not exist.
- **500 Internal Server Error:** Server-side failure.

Best Practices for Using the API

1. Use appropriate filters (title, completed) to reduce unnecessary data retrieval for /todos/.
2. Handle error responses gracefully by implementing fallback mechanisms or user notifications.
3. Use the /complete/ and /incomplete/ endpoints to manage task statuses instead of directly updating the completed field via PATCH for better semantic clarity.
4. Monitor server responses and follow HTTP best practices for retries on transient errors (e.g., 500 Internal Server Error).

This implementation follows RESTful standards and ensures robust functionality for managing Todo items efficiently.