Personal Task Manager API Documentation

Base URL

The base URL for the Personal Task Manager API is `http://localhost:3000` (replace with your actual server URL).

Authentication

All endpoints (except for signup and login) require JWT (JSON Web Token) authentication. Include the generated token in the `Authorization` header for authenticated requests.

Example:

```

Authorization: your\_jwt\_token\_here

```

Endpoints

1. User Authentication

1.1 Sign Up

- \*\*Endpoint:\*\* `POST /auth/signup`

- \*\*Description:\*\* Register a new user.

- \*\*Request Body:\*\* and don’t forget ```json

{

"username": "your\_username",

"password": "your\_password",

"email": "your\_email"

}

```

- \*\*Response:\*\*

```json

{

"token": "your\_generated\_jwt\_token"

}

```

1.2 Login

- \*\*Endpoint:\*\* `POST /auth/login`

- \*\*Description:\*\* Authenticate and obtain a JWT token.

- \*\*Request Body:\*\*

```json

{

"username": "your\_username",

"password": "your\_password",

"email": "your\_email"

}

```

- \*\*Response:\*\*

```json

{

"token": "your\_generated\_jwt\_token"

}

```

2. Tasks

2.1 Create a Task

You should add in Headers in postman

Key: authorization

Value : the token you got when you created the user

- \*\*Endpoint:\*\* `POST /api/tasks`

- \*\*Description:\*\* Create a new task.

- \*\*Request Body:\*\*

```json

{

"title": "Task Title",

"description": "Task Description",

"status": "pending",

"dueDate": "2024-03-15"

}

```

- \*\*Response:\*\*

```json

{

"\_id": "task\_id",

"title": "Task Title",

"description": "Task Description",

"status": "pending",

"dueDate": "2024-03-15",

"userId": "user\_id"

}

```

2.2 Get Tasks

- \*\*Endpoint:\*\* `GET /api/tasks`

- \*\*Description:\*\* Retrieve tasks.

- \*\*Optional Query Parameters:\*\*

- `status`: Filter tasks by status (e.g., `completed`).

- `dueDate`: Filter tasks by due date (e.g., `2024-03-15`).

- \*\*Response:\*\*

```json

[

{

"\_id": "task\_id",

"title": "Task Title",

"description": "Task Description",

"status": "pending",

"dueDate": "2024-03-15",

"userId": "user\_id"

},

// ...additional tasks

]

```

2.3 Update a Task

- \*\*Endpoint:\*\* `PATCH /api/tasks/:taskId`

- \*\*Description:\*\* Update a task.

- \*\*Request Body:\*\*

```json

{

"title": "Updated Task Title",

"status": "completed"

}

```

- \*\*Response:\*\*

```json

{

"\_id": "task\_id",

"title": "Updated Task Title",

"description": "Task Description",

"status": "completed",

"dueDate": "2024-03-15",

"userId": "user\_id"

}

```

2.4 Delete a Task

- \*\*Endpoint:\*\* `DELETE /api/tasks/:taskId`

- \*\*Description:\*\* Delete a task.

- \*\*Response:\*\*

```json

{

"message": "Task deleted successfully"

}

```

3. Categories

3.1 Create a Category

- \*\*Endpoint:\*\* `POST /api/categories`

- \*\*Description:\*\* Create a new category.

- \*\*Request Body:\*\*

```json

{

"name": "Work"

}

```

- \*\*Response:\*\*

```json

{

"\_id": "category\_id",

"name": "Work",

"tasks": [],

"userId": "user\_id"

}

```

3.2 Get Categories

- \*\*Endpoint:\*\* `GET /api/categories`

- \*\*Description:\*\* Retrieve categories.

- \*\*Response:\*\*

```json

[

{

"\_id": "category\_id",

"name": "Work",

"tasks": ["task\_id1", "task\_id2"],

"userId": "user\_id"

},

// ...additional categories

]

```

3.3 Update a Category (Add or Remove Task)

- \*\*Endpoint:\*\* `PATCH /api/categories/:categoryId`

- \*\*Description:\*\* Update a category (add or remove tasks).

- \*\*Request Body:\*\*

```json

{

"taskId": "task\_id",

"action": "add" // or "remove"

}

```

- \*\*Response:\*\*

```json

{

"\_id": "category\_id",

"name": "Work",

"tasks": ["task\_id1", "task\_id2"],

"userId": "user\_id"

}

```

3.4 Delete a Category

- \*\*Endpoint:\*\* `DELETE /api/categories/:categoryId`

- \*\*Description:\*\* Delete a category.

- \*\*Response:\*\*

```json

{

"message": "Category deleted successfully"

}

```