Task Management API - Functional Documentation

---

1. Project Overview

This project is a full-featured **Task Management REST API** built using Node.js and PostgreSQL, with user authentication, task lifecycle tracking, and logging. The system allows registered users to manage tasks with features such as creating, updating, completing, deleting, and filtering tasks. It also provides a secure JWT-based authentication flow and generated Swagger API documentation.

---

2. Tech Stack & Tools Used

Layer	Technology
Language	Node Js (with TypeScript)
Framework	Express.js
ORM	TypeORM
Database	PostgreSQL
Authentication	JWT (jsonwebtoken)
Validation	class-validator
Documentation	Swagger (`swagger-ui-express`) and Postman
Dev Tools	ts-node-dev, pgAdmin
Deployment	Render.com

---

3. Authentication Flow

1. **Signup**: User registers using `/auth/signup`.
2. **Login**: User logs in with `/auth/login` and receives a JWT.
3. **Protected Routes**: All task-related routes are protected with `verifyToken` middleware.
4. **JWT Verification**: Token is passed in `Authorization: Bearer <token>` header.
5. **Profile Access**: Authenticated users can fetch their profile using `/users/me`.

---

4. Task Lifecycle

Action	Endpoint	Description
Create Task	`POST /tasks`	Add a new task
Get All Tasks	`GET /tasks`	List tasks with filters
Get Single Task	`GET /tasks/:id`	View a single task
Update Task	`PUT /tasks/:id`	Edit task details
Complete Task	`PATCH /tasks/:id/complete`	Mark task as completed
Delete Task	`DELETE /tasks/:id`	Soft delete a task
Task Logs	Auto-logged internally	Tracks create/update/delete/complete

All dates are returned in `DD-MM-YYYY` format, and logs are returned as an array for each task.

---

5. Route Grouping

Group	Base Route	Modules
Auth	`/auth`	Signup, Login
Users	`/users`	Get user profile

Tasks	`/tasks`	Full task CRUD + logs
Docs	`/docs`	Swagger UI

Each route is protected using middleware and follows RESTful structure.

---

6. Assumptions & Limitations

Assumptions:

- One user → many tasks
- JWT token expiry managed on frontend
- All logs are internal, not user-editable

Limitations:

- No email verification on signup
- No role-based access (admin/staff)
- No file uploads or comments

---

7. Setup Instructions

Prerequisites:

- Node.js & npm
- PostgreSQL (or use Render DB)

Local Setup:

https://github.com/Imman-A-Josh/TASK-MANAGEMENT-API.git
cd TASK-MANAGEMENT-API
npm install

Environment File (`.env`)

PORT=5000
DB_HOST=dpg-d16ie56mcj7s73c6cvqg-a.oregon-postgres.render.com
DB_PORT=5432
DB_USERNAME=dbuser
DB_PASSWORD=ZSClKDWacCZoFybwXCyKq03BcmGBI4em
DB_NAME=taskmanager_nea0
JWT_SECRET=task_secret_api@2025

Run the App:

npm run dev

Open Swagger Docs:

https://task-management-api-1riq.onrender.com/docs

Deployment

- Hosted on Render
- DB hosted via Render PostgreSQL
- All env variables securely configured in dashboard

Live Url

https://task-management-api-1riq.onrender.com/