Task Manager API – Documentation

Setup: How to Run the Application Locally

1. Clone the Repository

```
git clone prashanttnegi/Task-API

cd task-manager-api
```

2. Install Dependencies

```
npm install
```

3. Configure Environment Variables

Create a `.env` file in the root directory and add:

```
PORT=5000

MONGO_URI=mongodb://localhost:27017/taskmanager

JWT_SECRET=your_secret_key
```

Make sure MongoDB is running locally (`mongod` command).

4. Start the Application

```
node app.js
```

You should see:

```
MongoDB connected

Server running on port 5000
```

API Overview

The API is organized under these main endpoints:

User Authentication & Profile

Method	Endpoint	Description
POST	`/api/auth/register`	Register a new user
POST	`/api/auth/login`	Login and receive a JWT
POST	`/api/auth/logout`	Invalidate JWT via blacklist
GET	`/api/users/me`	Get authenticated user profile

Admin: User Role Management

Method	Endpoint	Description
GET	`/api/users`	Admin-only: get all users

Task Management (RBAC enforced)

Method	Endpoint	Description	Access
GET	`/api/tasks`	View assigned or team tasks	All
POST	`/api/tasks`	Create a new task	Manager/Admin
PUT	`/api/tasks/:id`	Update a task	All
DELETE	`/api/tasks/:id`	Delete a task	Manager/Admin

Method	Endpoint	Description	Access
POST	`/api/tasks/:id/assign`	Assign task to user	Manager/Admin

Analytics

Method	Endpoint	Description
GET	`/api/tasks/analytics`	Get task stats (completed, pending, etc.)

Swagger API Docs

Available at: http://localhost:5000/api-docs

Provides:

- All available routes
- Parameters, request & response schemas
- Security requirements

Usage Examples

Register

```
POST /api/auth/register
Content-Type: application/json
{
  "username": "alice",
  "email": "alice@example.com",
  "password": "StrongPass123"
}
```

Login

```
POST /api/auth/login
{
  "emailOrUsername": "alice",
  "password": "StrongPass123"
}
```

Create Task (Manager/Admin only)

```
POST /api/tasks
Authorization: Bearer <token>
{
  "title": "Review PR",
  "description": "Review the backend pull request",
  "dueDate": "2025-04-15",
  "priority": "high"
}
```

Assumptions and Design Decisions

Authentication

- JWTs are used for stateless authentication.
- Logout works by blacklisting JWTs using **Redis** to ensure secure invalidation.

Role-Based Access Control

- `admin` has full control over users and tasks.

- `manager` can assign and manage tasks within their team.
- `user` can only **view tasks assigned to them**.

Task Visibility

- Managers see all tasks from users within their team.
- Admins see all tasks system-wide.
- Users see only their assigned tasks.

Validation

- Registration validates:
  - **Email format**
  - **Strong passwords** (min 8 characters, uppercase, lowercase, digit)
- All routes handle errors and return meaningful messages.

Optional Features

- Swagger UI enabled via `swagger.yaml`
- Redis used for token invalidation
- Rate limiting & Helmet used for security