Code Challenge: Laravel Task Manager API - Version 1

Objective:

Create a simple Task Manager API where users can manage their tasks. The API should allow users to perform CRUD operations (Create, Read, Update, Delete) on tasks. The tasks should have categories, and the API should allow filtering tasks based on their categories and due dates.

Requirements:

1. Task Model:

- Fields:
  - `title` (string): The title of the task. Example: "Complete project documentation".
  - `description` (text): A detailed description of the task. Example: "Write and review the full documentation for the new Laravel project.".
  - `due_date` (date): The due date for the task. Example: "2024-09-15".
  - `status` (enum): The status of the task, e.g., `pending`, `in_progress`, `completed`. Example: "pending".
  - `category_id` (foreign key): The category to which the task belongs. Example: 1 (Work-related tasks).

2. Category Model:

- Fields:
  - `name` (string): The name of the category. Example: "Work".
  - `description` (text): A description of the category (optional). Example: "Tasks related to professional work".

3. Endpoints:

- `GET /tasks`: List all tasks. Example response:

  ```json
  [
    {
      "id": 1,
      "title": "Complete project documentation",
      "description": "Write and review the full documentation for the new Laravel project.",
      "due_date": "2024-09-15",
      "status": "pending",
      "category_id": 1
    },
    {
      "id": 2,
      "title": "Prepare for team meeting",
      "description": "Gather all necessary reports and presentations for the quarterly team meeting.",
      "due_date": "2024-09-20",
      "status": "in_progress",
      "category_id": 2
    }
  ]
  ```

- `POST /tasks`: Create a new task. Example request body:

  ```json
  ```

```json
  {
    "title": "Plan project timeline",
    "description": "Develop a timeline for the upcoming project, including all milestones and deadlines.",
    "due_date": "2024-10-01",
    "status": "pending",
    "category_id": 1
  }
```

- `GET /tasks/{id}`: Retrieve details of a specific task. Example response:
  ```json
  {
    "id": 1,
    "title": "Complete project documentation",
    "description": "Write and review the full documentation for the new Laravel project.",
    "due_date": "2024-09-15",
    "status": "pending",
    "category_id": 1
  }
  ```

- `PUT /tasks/{id}`: Update a specific task. Example request body:
  ```json
  {
    "title": "Complete project documentation",
  ```

    "description": "Update the documentation with the latest changes.",
    "due_date": "2024-09-20",
    "status": "in_progress",
    "category_id": 1
  }

- `DELETE /tasks/{id}`: Delete a specific task. Example response:
  ```json
  {
    "message": "Task deleted successfully."
  }
  ```

- `GET /tasks?category={category_id}`: Filter tasks by category. Example response:
  ```json
  [
    {
      "id": 1,
      "title": "Complete project documentation",
      "description": "Write and review the full documentation for the new Laravel project.",
      "due_date": "2024-09-15",
      "status": "pending",
      "category_id": 1
    }
  ]
  ```

- `GET /tasks?due_date={date}`: Filter tasks by due date. Example response:

  ```json
  [
    {
      "id": 2,
      "title": "Prepare for team meeting",
      "description": "Gather all necessary reports and presentations for the quarterly team meeting.",
      "due_date": "2024-09-20",
      "status": "in_progress",
      "category_id": 2
    }
  ]
  ```

Additional Features (Optional):

- Implement user authentication and authorization.
- Allow users to set task priorities (low, medium, high).
- Add pagination to task listing endpoints.
- Enable task sorting by due date or status.
- Create a notification system to alert users as the due date approaches.