**practice Tasks (2)**

**2. TODO List API**

A simple CRUD-based TODO list app.

**Schema:**

* **Task**: title, description, status (pending/completed), dueDate

**Tasks:**

* **Add a Task** (POST /tasks)
* **Get All Tasks** (GET /tasks)
* **Update Task Status** (PATCH /tasks/:id)
* **Delete a Task** (DELETE /tasks/:id)
* **Filter Tasks by Status** (GET /tasks?status=pending)

|  |  |  |
| --- | --- | --- |
| Operation | Method | URL Example |
| Add a Task | POST | http://localhost:5000/tasks?title=Buy Milk&description=  Get from store&status=pending&dueDate=2025-02-25 |
| Get All Tasks | GET | http://localhost:5000/tasks |
| Update Task Status | PATCH | http://localhost:5000/updatetask?id=taskID&status=completed |
| Delete a Task | DELETE | http://localhost:5000/deletetask?id=taskID |
| Filter Tasks by Status | GET | http://localhost:5000/tasks?status=pending |

**✅ TODO List API - Endpoints & URLs**

**Code**

const express = require("express");

const mongoose = require("mongoose");

const app = express();

app.use(express.json());

mongoose.connect("mongodb://127.0.0.1:27017/todoDB", { useNewUrlParser: true, useUnifiedTopology: true })

  .then(() => console.log("MongoDB Connected"));

const Task = mongoose.model("Task", new mongoose.Schema({

    title: String, description: String, status: String, dueDate: String

}));

// ✅ Add a Task

app.post("/tasks", async (req, res) => res.send(await new Task(req.query).save()));

// ✅ Get All Tasks (with optional filtering)

app.get("/tasks", async (req, res) => {

    const filter = req.query.status ? { status: req.query.status } : {};

    res.send(await Task.find(filter));

});

// ✅ Update Task Status

app.patch("/updatetask", async (req, res) => {

    const { id, status } = req.query;

    if (!id) return res.status(400).send({ message: "Task ID is required" });

    res.send(await Task.findByIdAndUpdate(id, { status }, { new: true }));

});

// ✅ Delete a Task

app.delete("/deletetask", async (req, res) => {

    if (!req.query.id) return res.status(400).send({ message: "Task ID is required" });

    res.send(await Task.findByIdAndDelete(req.query.id));

});

app.listen(5000, () => console.log("Server running on port 5000"));

Here are **10 MongoDB queries** and **5 API queries** to practice using your **todoDB** database

### ****1. Find All Tasks with "Pending" Status****

**db.tasks.find({ status: "Pending" });**

### ****2. Find Tasks Due Before a Specific Date****

(Replace *"2025-03-10"* with your desired date)

**db.tasks.find({ dueDate: { $lt: "2025-03-10" } });**

### ****3. Count the Number of "Completed" Tasks****

**db.tasks.countDocuments({ status: "Completed" });**

### ****4. Update the Description of a Task****

(Replace *{task\_id}* with an actual task’s ID)

**db.tasks.updateOne({ \_id: ObjectId("{task\_id}") }, { $set: { description: "Updated Task Description" } });**

### ****5. Change Status of All "In Progress" Tasks to "Completed"****

**db.tasks.updateMany({ status: "In Progress" }, { $set: { status: "Completed" } });**

### ****6. Delete All Tasks with "Cancelled" Status****

**db.tasks.deleteMany({ status: "Cancelled" });**

### ****7. Find the Most Recent Task (Based on Due Date)****

**db.tasks.find().sort({ dueDate: -1 }).limit(1);**

### ****8. Retrieve Tasks Sorted by Title (Ascending Order)****

**db.tasks.find().sort({ title: 1 });**

### ****9. Find Tasks Containing "Meeting" in the Title****

**db.tasks.find({ title: /Meeting/i });**

### ****10. Group Tasks by Status and Count Them****

**db.tasks.aggregate([**

**{ $group: { \_id: "$status", totalTasks: { $sum: 1 } } }**

**]);**

## **5 API Queries** (For Testing the Express API)

### ****1. Add a New Task****

**POST /tasks?title=Project Work&description=Complete the report&status=Pending&dueDate=2025-03-15**

### ****2. Get All "Completed" Tasks****

**GET /tasks?status=Completed**

### ****3. Update a Task’s Status to "Completed"****

(Replace *{task\_id}* with an actual task’s ID)

**PATCH /updatetask?id={task\_id}&status=Completed**

### ****4. Delete a Specific Task****

(Replace *{task\_id}* with an actual task’s ID)

**DELETE /deletetask?id={task\_id}**

### ****5. Get All Tasks Without Any Filters (Retrieve All Tasks)****

**GET /tasks**