BackEnd Task

Project: Simple To-Do CRUD API (Node + Express)

Folder Structure

```
todo-api/
│
├── server.js
├── todos.json
└── README.md
```

server.js

```
const request = require("supertest");
const fs = require("fs");
const app = require("./server"); // Export Express instance in server.js

const DATA_FILE = "todos.json";

// Reset file before each test
beforeEach(() => {
  fs.writeFileSync(DATA_FILE, JSON.stringify([]));
});

describe("Todo API Tests", () => {

  test("POST /todos - should create a todo", async () => {
    const response = await request(app)
      .post("/todos")
      .send({ title: "Test Todo" });

    expect(response.statusCode).toBe(201);
    expect(response.body.title).toBe("Test Todo");
    expect(response.body.completed).toBe(false);
```

```
});

test("POST /todos - should fail if title missing", async () => {
  const response = await request(app)
    .post("/todos")
    .send({});

  expect(response.statusCode).toBe(400);
});


test("GET /todos - should return list of todos", async () => {
  await request(app).post("/todos").send({ title: "Sample" });

  const response = await request(app).get("/todos");

  expect(response.statusCode).toBe(200);
  expect(response.body.length).toBe(1);
});


test("PUT /todos/:id - should update a todo", async () => {
  const create = await request(app).post("/todos").send({ title: "Old" });
  const id = create.body.id;

  const response = await request(app)
    .put(`/todos/${id}`)
    .send({ completed: true });

  expect(response.statusCode).toBe(200);
  expect(response.body.completed).toBe(true);
});
```

```
  test("PUT /todos/:id - should return 404 for invalid id", async () => {
    const response = await request(app)
      .put("/todos/999")
      .send({ completed: true });

    expect(response.statusCode).toBe(404);
  });


  test("DELETE /todos/:id - should delete a todo", async () => {
    const create = await request(app).post("/todos").send({ title: "To Delete" });
    const id = create.body.id;

    const response = await request(app).delete(`/todos/${id}`);

    expect(response.statusCode).toBe(200);
    expect(response.body.message).toBe("Todo deleted successfully");
  });


  test("DELETE /todos/:id - should return 404 for non-existing id", async () => {
    const response = await request(app).delete("/todos/999");

    expect(response.statusCode).toBe(404);
  });

});
```

todos.json

```
[]
```

This is the file where todos are stored.

README.md

Simple To-Do CRUD API

A lightweight REST API to manage to-do items using Node.js and Express.

Data is stored locally in a JSON file. No database required.

Features

- GET /todos → Get all todos
- POST /todos → Add a todo
- PUT /todos/:id → Update a todo
- DELETE /todos/:id → Delete a todo

Bonus:
- Input validation
- Includes `completed: true/false`
- Ready to deploy on Render/Railway/Cyclic

Installation

```bash
git clone <repo>
cd todo-api
npm install
```

Run the Server

node server.js

Server will start on:

http://localhost:3000

Sample Requests

Create Todo (POST /todos)

```
{
  "title": "Buy milk",
  "completed": false
}
```

Update Todo (PUT /todos/123)

```
{
  "completed": true
}
```

Delete Todo (DELETE /todos/123)

Deploy

You can deploy to:

- Render
- Railway
- Cyclic

Just upload the code and set:

Start command: node server.js

Tech Stack

- Node.js
- Express
- JSON file storage

Optional Deployment Tips

On **Render/Railway/Cyclic**, after upload:

- Make sure `PORT` uses `process.env.PORT`
- Add start script in `package.json`:

```json
{
  "scripts": {
    "start": "node server.js"
  }
}
```