Soal Homework

Buatlah project API menggunakan expressjs dan sequelize (postgresql) menggunakan table todo yang berisi field title

```
TERMINAL  PORTS  DEBUG CONSOLE  PROBLEMS  OUTPUT                                                    node

PS E:\RAKAMIN\WEEK 11\11. Homework -Unit Testing & Deployment> node app.js
Server is running on port 3000
Executing (default): SELECT table_name FROM information_schema.tables WHERE table_schema = 'public' AND table_name = 'Todos'
Executing (default): SELECT i.relname AS name, ix.indisprimary AS primary, ix.indisunique AS unique, ix.indkey AS indkey, array_agg(a.
attnum) as column_indexes, array_agg(a.attname) AS column_names, pg_get_indexdef(ix.indexrelid) AS definition FROM pg_class t, pg_clas
s i, pg_index ix, pg_attribute a WHERE t.oid = ix.indrelid AND i.oid = ix.indexrelid AND a.attrelid = t.oid AND t.relkind = 'r' and t.
relname = 'Todos' GROUP BY i.relname, ix.indexrelid, ix.indisprimary, ix.indisunique, ix.indkey ORDER BY i.relname;
Database and tables created!
```

```
PS E:\RAKAMIN\WEEK 11\11. Homework -Unit Testing & Deployment> node app.js
Server is running on port 3000
Executing (default): SELECT table_name FROM information_schema.tables WHERE table_schema = 'public' AND table_name = 'Todos'
Executing (default): SELECT i.relname AS name, ix.indisprimary AS primary, ix.indisunique AS unique, ix.indkey AS indkey, array_agg(a.
attnum) as column_indexes, array_agg(a.attname) AS column_names, pg_get_indexdef(ix.indexrelid) AS definition FROM pg_class t, pg_clas
s i, pg_index ix, pg_attribute a WHERE t.oid = ix.indrelid AND i.oid = ix.indexrelid AND a.attrelid = t.oid AND t.relkind = 'r' and t.
relname = 'Todos' GROUP BY i.relname, ix.indexrelid, ix.indisprimary, ix.indisunique, ix.indkey ORDER BY i.relname;
Database and tables created!
Executing (default): SELECT "id", "title", "deleted", "createdAt", "updatedAt" FROM "Todos" AS "Todo";
```

Kemudian pastikan terdapat fitur dibawah ini

1. List All Todo

```
// List All Todo
router.get('/todos', async (req, res) => {
  try {
    const todos = await Todo.findAll();
    res.json(todos);
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'Internal Server Error' });
  }
})
```

2. Detail Todo

```
// Detail Todo
router.get('/todos/:id', async (req, res) => {
  const todoId = req.params.id;
  try {
    const todo = await Todo.findByPk(todoId);
    if (todo) {
      res.json(todo);
    } else {
      res.status(404).json({ error: 'Todo not found' });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'Internal Server Error' });
  }
});
```

3. Create Todo

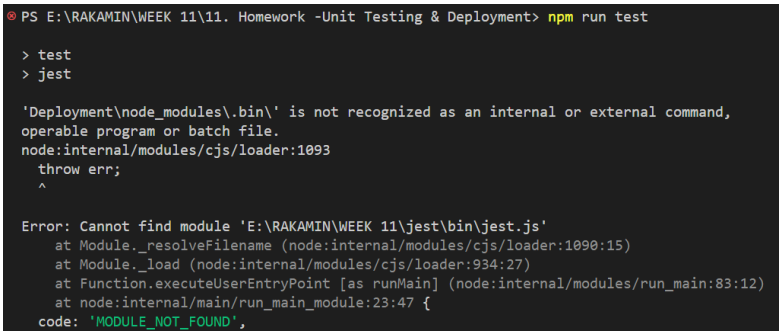
```
// Create Todo
router.post('/todos', async (req, res) => {
  const { title } = req.body;
  try {
    const newTodo = await Todo.create({ title });
    res.status(201).json(newTodo);
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'Internal Server Error' });
  }
});
```

4. Delete Todo (Soft Delete)

```
// Delete Todo (Soft Delete)
router.delete('/todos/:id', async (req, res) => {
  const todoId = req.params.id;
  try {
    const todo = await Todo.findByPk(todoId);
    if (todo) {
      await todo.update({ deleted: true });
      res.json({ message: 'Todo deleted' });
    } else {
      res.status(404).json({ error: 'Todo not found' });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'Internal Server Error' });
  }
});
```

Buatlah semua unit testing untuk masing-masing fitur api (crud unit testing),

```
// crud unit testing

const request = require('supertest');
const app = require('../app');

describe('CRUD API Tests', () => {
  let todoId;

  it('should create a new todo', async () => {
    const response = await request(app)
      .post('/api/todos')
      .send({ title: 'New Todo' });
    expect(response.statusCode).toBe(201);
    expect(response.body.title).toBe('New Todo');
    todoId = response.body.id;
  });

  it('should get all todos', async () => {
    const response = await request(app)
      .get('/api/todos');
    expect(response.statusCode).toBe(200);
    expect(Array.isArray(response.body)).toBe(true);
  });

  it('should get todo details by ID', async () => {
    const response = await request(app)
      .get(`/api/todos/${todoId}`);
    expect(response.statusCode).toBe(200);
    expect(response.body.title).toBe('New Todo');
  });

  it('should soft delete a todo by ID', async () => {
    const response = await request(app)
      .delete(`/api/todos/${todoId}`);
    expect(response.statusCode).toBe(200);
    expect(response.body.message).toBe('Todo deleted');

    const deletedTodoResponse = await request(app)
      .get(`/api/todos/${todoId}`);
    expect(deletedTodoResponse.statusCode).toBe(404);
    expect(deletedTodoResponse.body.error).toBe('Todo not found');
  });
});
```



Ngestak disini kak 😟

jalankan api tersebut menggunakan Dockerfile serta Docker Compose dan jalankan CI/CD menggunakan GitLab