Project 1 Task: Develop an Expense Tracking Application

Directors are furious and would like to know where all company money is going. Manifest wants you to create a sophisticated expense tracking application that will enable departments manage and keep a record of all company expenses spent by their employees.

The expense tracker application will follow this process.

a. The company has an Employee
b. An employee will have a Role and Department field.
c. An Employee of the company creates an Expense.
d. The expense must have date and time, details, amount field and type.
e. You can include other fields you feel is nice to have.
f. The expense must have a Type.
g. The Type of expense can be categorized into Internal & External.
h. This must be selected as a drop down field when creating expenses.
i. The expense can also be grouped into different Category such as Fuel, Internet, Transportation, Printing e.t.c
j. Again, this must be selected as a drop down field when creating expenses in the system.
k. When an expense is created, the default status is **pending**.
l. If an expense is less than N1,000 - the default status will automatically change to **approve** instead of pending.
m. If the role of an employee is a manager, they can review an expense that is assigned to their department.
n. We can have different managers by department.
o. When the department manager reviews the expense, the status will change to **approved** or **declined** based on selection by the manager.
p. All employees with role **manager within the department** can see expenses of the department.
q. IMPORTANT: When expenses are created, the expense should be automatically assigned to a manager and department based on the employee department.
r. For instance, if an employee working in the engineering department creates an expense, only the manager of engineering should be able to review the expense, not all the managers in the company.
s. So you should try to include an **approver** field in the **Expense** model. This field should be automatically set based on the employee that is creating this expense.

2. Start the application by developing the model diagram.
3. Create the CRUD for Employee, Expense, Type & Category.
4. Develop the association between these models.

5. Develop the basic working CRUD for the Model has a starting point for applications.
6. Develop the controller logic for these associations based on the requirements. Take your time with this but make sure your controller logic is clean and easy to read.
7. You can also choose to develop a model for **Role, Department & Status** to ease the CRUD process or put these fields in Employee and Expense models respectively - this is up to you.
8. I want to see all API endpoints tested successfully.
9. I want to see 5 major pages - **Employees, Expense, Type and Category** and I also want to see a **Dashboard** that provides a summary of what is going on in the company. In the dashboard, I want to see the total amount spent till date, the top five expenses, expenses listed by category, the most recent expenses e.t.c
10. Then I want to see the models within the models. For instance, Expenses listed by employees or within Employees page, to see the expenses created by them e.t.c
11. Follow the milestones strategy. Work in Git branches.
12. Create a unique repository, share within yourself and myself.
13. I have provided you with the Lessons needed to test your application. You have two weeks to deliver the MVP so start coding now.

Project 2 Task: **Develop an Information Logging Application called DocuTrack**

When developers create applications, some employees find it very hard to use. Manifest wants you to create a simple information logging application called **documentation** - this will enable developers to document how to use certain applications they develop. Employees can go on this application to read and maybe make a comment if they have any issues they encountered when using a certain company application. This application is very similar to my BlogApp project but must be branded in such a way that it reflects Documenting as opposed to Blogging - see the requirements.

The documentation application will follow this process.

a. The documentation app will have an Employee model.
b. An employee will have a **Role and Department** field.
c. An Employee creates a Documentation for an application called Document
d. The Document must have a field called subject, description, **status application, type and category.**
e. The Type of document can be categorized into Internal & External.
f. Selection of type should be a radio select button.
g. If a document is created as Internal, only employees with Role called manager or the employees that belong to that department (that created the document) should be able to read/update or delete the document.
h. The document can also be grouped into different Application such as Timesheet App, Expense Tracker App, Career App, Todo App e.t.c
i. The application field should be a select drop down. So employees can select an Application they are writing the document for.
j. The document can also be grouped into different Category such as
k. 'How To', 'Common Issues', 'Database Schema', 'Model', 'Routes' e.t.c
l. Again, the selection for Category should be a drop down option.
m. The fields that need to be filled should be dynamic based on the Category selected. For instance, if the Category selected is Model, the form fields that need to be filled should be different from when a Category called route is selected e.t.c you can also make certain fields required based on the category selected.
n. When a document is created, the status to be changed to **Draft**.
o. An employee can change the status of the document they created.
p. If the role of an employee is a manager, they can read, edit, delete or publish any document created by them or any employee.
q. If the role of an employee is an editor, they can read, edit or **re-publish** the document. I.e. they cannot publish a document that is in draft by another employee, only admin can do that. They can only publish their own document or re-publish an existing document if it has already been published by the creator.

r. When a document is created, a Comment box should appear at the bottom of the document details page.
s. Employees can make comments on each document published.

2. Start the application by developing the model diagram.
3. Create the CRUD for Employee, Document, Comment, Type & Category.
4. Develop the association between these models
5. Develop the logic for these associations based on the requirements.
6. Develop the controller logic for these associations based on the requirements. Take your time with this but make sure your controller logic is clean and easy to read.
7. You can also choose to develop a model for **Role, Department & Status** to ease the CRUD process or put these fields in Employee and Document models respectively - this is up to you.
8. I want to see all API endpoints tested successfully.
9. I want to see 5 major pages - **Employees, Document, Application, Type and Category** and I also want to see a **Dashboard** that provides a summary of what is going on in the company. I.e. Total documented create, documented by category, document by application, types e.t.c I also want to see the last five documents created in each of the categories.
10. Follow the milestones strategy.
11. I have provided you with the Lessons needed to test your application without depending on anyone. You have two weeks to deliver the MVP so