**Expense Management System**

(**Sprint-2**)

**Table of Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **Introduction** | | **3** |
|  | 1.1 | Setup Checklist for Sprint-2 Project | 3 |
| **2** | **Problem Statement** | | **4** |
|  | 2.1 | Objective | 4 |
|  | 2.2 | Abstract of Project | 4 |
|  | 2.3 | Functional Component of Project | 4 |

1. Introduction

This document outlines a case study for Sprint 2 project. The project is to develop an Expense Management System as integration of all independent microservices. This document contains the work flow of the system and gives guidelines on how to build the functionality gradually in each of the course modules.

## Setup Checklist for SPRINT-2 Project

1. **Minimum System Requirement**

* Intel Pentium 90 or higher (P166 recommended)
* Microsoft Windows 95, 98, or NT 4.0, 2k, XP, Windows 7 , Windows 10
* Memory: 8GB of RAM (8GB or more recommended)
* Internet Explorer 11.0 or higher or Chrome 45 or above

1. **Software/Tool Requirement**

* JDK 8
* Nodejs
* IDE-STS(Spring Tool Suite)
* MAVEN
* PostMan Master
* GitHub- Version Control System
* MySql or H2 database
* IDE-Visual Studio Code

2. Problem Statement

## Objective

Develop an Expense Management System that will replace old xls based claim system.

## 2.2 Abstract of the project

This project is aimed at developing an online expense claim system. This is a web based application that can be accessed over the web.

## Functional components of the project

In this project, the independent functions are divided into microservices and description of those microservices is as follows:

**Employee Code Module**: Following is a list of functionalities of the system. There is a user who can add, delete, update, search and view the employee details.

There are two microservices named Employee Command Service and Employee Query Service which contains all the CRUD operations which are to be perform in expense code module and one Database Provider Service to store the data.

1. **Employee Command Service**
2. Add Employee Code details:
   * Employee ID has to be auto generated.
   * Employee Name is a combination of uppercase, lowercase alphabets and whitespaces (cannot be empty).
   * Employee PAN is a combination of uppercase alphabets and digits (cannot be empty).
   * Employee Designation is a combination of uppercase, lowercase alphabets and whitespaces (cannot be empty).
   * Employee Domain is a combination of uppercase, lowercase alphabets and whitespaces (cannot be empty).
   * Employee DOJ needs to undergo regular date validation (DD/MM/YYYY) (cannot be empty).
   * Employee DOB needs to undergo regular date validation (DD/MM/YYYY) (cannot be empty).
   * Employee Salary needs to undergo regular number validation (cannot be empty).
   * Employee Mail ID needs to undergo regular mail ID validation (cannot be empty).
3. Modify Employee details:

* Based on existing Employee ID, modify all the fields of Employee Entity and the changes will be reflected in the database.

1. Delete an employee based on his/her Employee ID
2. **Employee Query Service**
3. Display Employee details:

* Display Employee details based on the entered Employee ID.
* If ID is not found, error message will be displayed.

1. Search details of an employee based on his/her employee ID

* The results will be filtered and displayed based on the Employee ID.

**Project Code Module**: Following is a list of functionalities of the system. There is a user who can add, delete, update and view the project details.

There are two microservices named Project Command Service and Project Query Service which contains all the CRUD operations which are to be perform in project code module and one Database Provider Service to store the data.

1. **Project Command Service**
2. Add Project Code details as listed below and inject the values into database table if data are valid else display appropriate error messages

* Project Code has to be auto generated.
* Project Description can be combination of alphabets, digits and underscores (cannot be empty).
* Start date and End date need to undergo regular date validation (DD/MM/YYYY) (cannot be empty).
* Project end date has to be greater than project start date (cannot be empty).

b. Modify Project details:

* + Based on existing project ID, display the following fields, Project Description, Start date, End date and Business Unit (validation should be taken care of).

1. Delete an project based on his/her project ID
2. **Product Query Service**
3. Display Employee details:

* Display Project details based on the entered project ID.
* If ID is not found, error message will be displayed.

1. Search details of an project based on project ID
   * The results will be filtered and displayed based on the Project ID.

**Expense Code Module**: Following is a list of functionalities of the system. There is a user who can add, delete, update, search and view the expense details.

There are two microservices named Expense Command Service and Expense Query Service which contains all the CRUD operations which are to be perform in expense code module and one Database Provider Service to store the data.

**1. Expense Command Service**

a. Add Expense Code details:

* Insert the values into database table if data is valid, else display appropriate error messages.
* Expense Code has to be auto generated.
* Expense Type must start with capital character, should only contain alphabet and it should contain minimum 3 and maximum 15 characters (cannot be empty).
* Expense Description can be combination of alphabets, digits and underscores and it should contain minimum 15 and maximum 100 characters (cannot be empty).

b. Modify Expense details:

* Based on existing expense code, update the fields Expense Code, Expense Type and Expense Description (validation should be taken care of).

1. Delete Expense details:

* Based on existing expense code, delete the fields, Expense Code, Expense Type and Expense Description.

**2. Expense Query Service**

1. Display All Expense details:

* Display the fields Expense Code, Expense Type and Expense Description for all expense codes in the database.

1. Search Expense details:

* Based on existing expense code, display the fields Expense Code, Expense Type and Expense Description.

**Expense Claims Details Module**: The functionality of this module is to integrate the micro services, i.e. Employee Code Module, Project Code Module and Expense Code Module. The portal is created through which a user can claim, view, update and delete the expense. Following is a list of detailed functionalities under this module.

**1. Expense Claim Command Service**

a. Claim Expense details:

* Enter the Employee ID. If valid, the corresponding details will be fetched from Employee Code Module and displayed. Else, the error message will be displayed and the user will not be able to proceed further.
* Select the Project Code from drop down list. The corresponding details will be fetched from Project Code Module and displayed.
* Select Expense Code from drop down list. The details will be fetched from Project Code Module and displayed.
* Enter some additional details, i.e. Start Date, End Date, Expense Amount, etc. These details, along with some required details of other micro services (Employee ID, Project Code and Expense Code) will be stored into the database.
* The Expense Code ID will be auto generated and displayed on the screen.

b. Update Claim details:

* Enter the Expense Code ID. If valid, the details will be displayed and user will be able to modify the details wherever required. Else if the ID is wrong, the error message will be displayed.

c. Delete an Expense Claim:

* Enter the Expense Code ID. If valid, all the corresponding details will be deleted from the database. Else if the Id is wrong, the error message will be displayed.

**2. Expense Claim Query Service:**

a. View Expense Claim Details by ID:

* Enter the Expense Code ID. If valid, the corresponding details will be displayed. Else if the ID is wrong, the error message will be displayed.

**User Login Module**: The functionality of this module is to check whether the logged in user is an admin or a employee and redirect them to their respective portals where they will be able to perform the operations mentioned in the above microservices. It consists of two microservice User Login Command Service and User Login Query Service which will do all operations mentioned below and one User Data Provider Service.

**1. User Login Command Service**

a. Add User details:

* Enter the username, password and role values if valid proceed else give error.
* The User ID will be auto generated and displayed on the screen.

b. Update User details:

* Enter the User ID. If valid, the details will be displayed and user will be able to modify the details wherever required. Else if the ID is wrong, the error message will be displayed.

c. Delete User:

* Enter the User ID. If valid, all the corresponding details will be deleted from the database. Else if the Id is wrong, the error message will be displayed.

**2. User Login Query Service:**

a. View User by ID:

* Enter the User ID. If valid, the corresponding details will be displayed. Else if the ID is wrong, the error message will be displayed.

b. View all:

* Based on user login details, all fields will be displayed.