Feature Development Plan

Feature Names:

1. Requirement Analysis

Summary:

The customer success platform aims to automate project management and information sharing processes within the company. Key stakeholders, including clients, auditors, project managers, and admins, each have specific roles and permissions tailored to their responsibilities. The platform seeks to address the manual process of sharing project-related information via word documents by providing automated features for data gathering, auditing, and sharing.

Key Requirements:

1) **User Roles and Permissions:**

- Four roles: Client, Auditor, Project Manager, Admin.
- Each role has specific permissions:
- Auditor: Can add/delete projects, assign project managers, view project details in read-only mode, edit/add audit history, and invite clients.
- Project Manager: Can edit/add project-related details and metrics, view audit history in read-only mode.
- Admin: Has complete power over Auditor and platform administration.
- Client: Has view-only access to all project-related information.

2) **Export PDF:**

- All users should have the ability to export project details as PDF documents.

3) **Automated Email Notifications:**

- After adding audit history, an email notification should be sent to all stakeholders regarding the current status of the project.

→Objectives:

- To ensure better monitoring and transparency of project-related information.
- To automate data gathering, auditing, and sharing processes.
- To eliminate the manual effort involved in sharing project information via word documents.
- These requirements outline the core functionalities and objectives of the customer

Success platform, catering to the needs of various stakeholders and aiming for seamless automation and efficiency in project management.

2.API Endpoints Design

1.Project Budget:

Endpoint Summary: It refers to the details related to type like Fixed or Monthly along with duration and Budgeted Hours**.**

Design Details: *Endpoint URL*

→Endpoint URL: /add/project-budget

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages: 404 (Not Found)

→Endpoint URL: /add/project-budget

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added Record
- Response Body Schema: None
- Error Codes and Messages: 500 (Some Error Occurred)

2.Version History:

Endpoint Summary: It refers to the details related to version of Project including details like version, changes in version, approved date, approved by etc.

Design Details: *Endpoint URL*

→*Endpoint URL: /add/version-history*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None

- Error Codes and Messages: 404 (Not Found)

→*Endpoint URL: /add/version-history*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added Record
- Response Body Schema: None
- Error Codes and Messages: 500 (Some Error Occurred)

3.Audit History:

Endpoint Summary: It refers to the details related to version of Project including details like version, changes in version, approved date, approved by etc.

→*Endpoint URL: /add/audit-history*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages: 404 (Not Found)

→*Endpoint URL: /add/audit-history*

- *HTTP Method: POST*
- *Request Parameters: Project ID*
- *Request Body Schema: Updated/Added Record*
- *Response Body Schema: None*
- *Error Codes and Messages:500 (Some Error Occurred)*

4. Project Description:

Endpoint Summary: It refers to the details related to Project like Scope, Project

→*Endpoint URL: /add/project-description*

- *HTTP Method: GET*
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→*Endpoint URL: /add/project-description*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Updated/Added Description
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

5. Project Stack:

→*Endpoint URL: /add/project-stack*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→Endpoint URL: /add/project-stack

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added/Updated project stack
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

6. Escalation Metrics:

Endpoint Summary: It refers to the different metrics like Operational, Financial, Technical.

→*Endpoint URL: /add/escalation-metrics*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→*Endpoint URL: /add/escalation-metrics*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Type of Escalation Metrics (Financial, Operational, Technical) and respective change in Record
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

7. Stakeholders:

Endpoint Summary: It refers to the details of stakeholders including their title, name, and contact

→*Endpoint URL: /add/stakeholders*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→*Endpoint URL: /add/stakeholders*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added/Updated Record of stakeholders

- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

8. Risk Profiling:

Endpoint Summary It refers to the different types of risk involved the project like Financial, Operational, Technical, HR, External along with their severity and impact.

→*Endpoint URL: /add/risk-profiling*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→*Endpoint URL: /add/risk-profiling*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added/Updated Record of Risk
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

9. Phases:

Endpoint Summary: It refers to the details of phases with detail of each phase like start date, status, completion date.

→*Endpoint URL: /add/phases*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

→*Endpoint URL: /add/phases*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added/Updated Phase details
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

10. Sprint:

Endpoint Summary: It refers to the details of sprint with detail of each sprint like start date, status, end date and comments.

→*Endpoint URL: /add/sprint*

- HTTP Method: GET
- Request Parameters: Project ID
- Request Body Schema: None
- Response Body Schema: None
- Error Codes and Messages:404 (Not Found)

➔*Endpoint URL: /add/sprint*

- HTTP Method: POST
- Request Parameters: Project ID
- Request Body Schema: Added/Updated Phase details
- Response Body Schema: None
- Error Codes and Messages:500 (Some Error Occurred)

2. Pseudo Code for Key Functionalities

Function Overview:

The Email Notification class encapsulates functionality for sending email notifications and handling updates. It provides methods to send emails to specified recipients and handle field updates by composing and sending email notifications with updated content. This class serves as a communication module within the customer success platform, ensuring stakeholders are informed of important updates and changes.

→Pseudo code is a way to describe the functionality of a program or algorithm in a formatthat's more readable than actual code, but not as detailed as plain language. It helps in planning and understanding the logic before diving into the actual coding.

→Let's say the feature in question involves adding a new functionality to a user profile system, where the backend needs to handle a request to update a user's email address. Here's an example of how the pseudo code for this functionality might look:

Pseudo Code for Sending Emails and Handling Email Update Function:

Function Name: Send Email

Parameters:

- Recipient Email: Integer
- Subject: String (The new email address to be updated)
- Body: Email Body

Pseudo Code:
Class: Email Notification

Method: Send Email (recipient Email, subject, body)

Input:
Recipient Email: String (Recipient's email address)
Subject: String (Email subject)
Body: String (Email body)

Steps:
1. Set SMTP host, username, password, and sender email address.
2. Set email session properties including authentication and TLS settings.
3. Create a new email session instance using the provided properties and authentication.
4. Create a new Mime Message object.
5. Set the sender, recipient, subject, and body of the email message.
6. Send the email message using the Transport class.
7. Handle any Messaging Exception that occurs during the email sending process.

Method: handleFieldUpdate (recipient Email, updated Content)

Input:
Recipient Email: String (Recipient's email address)
Updated Content: String (Content to be included in the email body)

Steps:
1. Set the subject of the email as "Audit Summary".
2. Construct the email body with a greeting, the updated content, and a closing message.
3. Call the send Email method with the recipient email, subject, and constructed body to send the email notification.

This pseudo code outlines the basic logic for updating a user's email. This pseudo-code outlines the structure and logic of the Email Notification class and its methods. You can translate this pseudo-code into actual Java code by implementing the described steps using Java syntax and APIs.

Pseudo Code for Export Document:

Function: ExportToPDF

Parameters:

- data: Object (Data to be exported to PDF)

Steps:

1. Create a new PDF document instance.
2. Add a new page to the PDF document.
3. Set up the document properties such as title, author, and metadata.
4. Define the layout and formatting for the PDF content.
5. Iterate through the data to be exported.
   a. For each data entry:
      i. Format the data appropriately for the PDF.
      ii. Add the formatted data to the PDF page.
6. Save the PDF document to a file or stream.
7. Close the PDF document.0

End Function