Feature Development Plan for client Success platform

Feature Name: Role-Based Management Implementation and Login

This task encompasses implementing role-based access control (RBAC) for Admin, Project Manager (PM), Auditor, and Client roles within the platform. Additionally, it involves integrating Microsoft authentication via Auth0, enabling stakeholders to log in using their Microsoft accounts. This integration enhances security and user convenience by leveraging existing Microsoft credentials for platform access.

1. Requirement Analysis

The feature aims to enhance the security and user experience of the Customer Success Platform by implementing role-based access control (RBAC) and integrating Microsoft authentication via Auth0 for user registration and login.

Key Requirements:

Role-Based Management:

- Define roles such as Admin, Project Manager (PM), Auditor, and Client.
- Assign permissions and access levels to each role.
- Restrict access to platform features based on user roles.

Registration/Login with Microsoft:

- Enable users to log in using their Microsoft accounts.
- Integrate Auth0 to handle authentication and user management.
- Retrieve user details such as name, email, and role from Microsoft accounts.

2. Impact Analysis

Affected APIs:

- Authentication endpoints for user login.
- Authorization endpoints for role-based access control.

Analysis:

The implementation of these features may require changes to the existing database schema and API endpoints. It may also impact the way users interact with the platform.

3. Database Schema Changes: Not applicable

The following tables will be used for the CRUD operations:

- Approved Team: This table will store information about the approved team members for each project.
- Resources: This table will store information about the resources allocated to each project.
- Client Feedback: This table will store feedback received from clients.
- Project Updates: This table will store updates about each project.
- MoMs of Client Meetings: This table will store the minutes of meetings (MoMs) of client meetings.

Additional Tables

Additional tables will be created to store user details and project information:

- User Details Table: This table will store information about each user, including their role and Microsoft account details. It will have fields such as UserID, Name, Role, Email.
- Project Table: This table will store information about each project, including the last five fields updated via the CRUD operations. It will have fields such as ProjectID, ApprovedTeam, Resources, ClientFeedback, ProjectUpdates, and MoMs

4. API Endpoints Design

Endpoint Summary:

- Frontend
  - /Login: Endpoint for user login.
- Backend (React web api)
  - /authenticateUserRole: Endpoint to check authenticated user role.

Design Details:

/Login

- HTTP Method: POST
- Request Parameters: User credentials (email, password)
- Response Body Schema: JSON object containing authentication token or error message
- Error Codes and Messages: 401 Unauthorized, 500 Internal Server Error

/authenticateUserRole:

- HTTP Method: POST
- Middleware for every request to check any modification in user’s role
- This function also calls Auth0 to get the user's role details.

Pseudo Code for Key Functionalities

Functionality: User Login

Front End

```
@Component({
  selector: 'app-login-button',
  template: '<button (click)="login()">Log in</button>'
})
export class LoginButtonComponent {
  constructor(private auth: AuthService) {}

  login() {
    this.auth.loginWithRedirect();
  }
}
```

Middleware

```
FUNCTION authenticateUserRole(user)
    CALL Auth0 API to get user detail
    IF credentials are valid THEN
            IF authentication is successful THEN
                RETURN authentication token
            ELSE
                RETURN error message
    ELSE
        RETURN validation error message
END FUNCTION
```

5 Deployment and Configuration Changes (Optional)

- **New App Settings:** List any new application settings or environment variables needed.
- **Deployment Changes:** Detail any changes required in the deployment process, including new flags, configuration files, or deployment steps.
- **Testing and Verification:** Outline the approach for testing these changes, including unit tests, integration tests, and any manual testing required.

Feature Name: Last CRUD Operations

The requirement involves implementing CRUD (Create, Read, Update, Delete) operations for the last 5 fields within the Customer Success Platform. These fields include Approved Team, Resources, Client Feedback, Project Updates, and Minutes of Meetings (MoMs) of client meetings. The goal is to provide functionalities to manage and track information related to these aspects of the projects efficiently

1. Requirement Analysis

This feature involves implementing CRUD (Create, Read, Update, Delete) operations for the last 5 fields of the Customer Success Platform: Approved Team, Resources, Client Feedback, Project Updates, and Minutes of Meetings (MoMs) of client meetings.

Key Requirements:

Approved Team:

- Create, read, update, and delete operations for managing the approved project team members.

Resources:

- CRUD operations for managing project resources, including their roles, start date, and end date.

Client Feedback:

- Implement functionalities to add, view, update, and delete client feedback for projects.

Project Updates:

- Enable users to post updates related to project progress, dates, and milestones.

Minutes of Meetings (MoMs):

- Implement CRUD operations for managing MoMs of client meetings, including date, duration, and meeting details.

2. Impact Analysis

Affected APIs:

- API endpoints for CRUD operations on each of the last 5 fields Analysis:
- Implementing CRUD operations will impact the backend API endpoints responsible for data manipulation.
- Creating database schemas may be necessary to accommodate new data structures or relationships.
- User interface components in the Angular frontend will need to be updated to interact with the new CRUD functionalities.

3. Database Schema Changes:

Current Schema Overview:

- Database tables for projects may include fields for Approved Team, Resources, Client Feedback, Project Updates, and MoMs
- Define relationships between these entities and other relevant entities in the database schema.

4. API Endpoints Design

Endpoint Summary:

- /api/approvedteam: Endpoint for CRUD operations on Approved Team.
- /api/resources: Endpoint for CRUD operations on Resources.
- /api/clientfeedback: Endpoint for CRUD operations on Client Feedback.
- /api/projectupdates: Endpoint for CRUD operations on Project Updates.
- /api/clientmeetingmom: Endpoint for CRUD operations on MoMs of client meetings.

5. Pseudo Code for curd

```
// CRUD operations for Approved Team FUNCTION CreateApprovedTeam(teamData)
// Implementation logic to create approved team END FUNCTION

FUNCTION GetApprovedTeam(teamId)
// Implementation logic to retrieve approved team by ID END FUNCTION

FUNCTION UpdateApprovedTeam(teamId, updatedData)
// Implementation logic to update approved team by ID with updated data END FUNCTION

FUNCTION DeleteApprovedTeam(teamId)
// Implementation logic to delete approved team by ID END FUNCTION

// Similar functions for Resources, Client Feedback, Project Updates, and MoMs
```