# 1. Overview

Title: 1. Overview

Content:

This document outlines the design and architecture of the Microsoft Power Platform automation solution to be developed. It covers key components including:

### Case Management:

Implementation of workflows in Power Automate to manage case processes and track progress using Power Platform's case management capabilities. The use of Power Automate's automation solutions enables monitoring and management of processes through its platform, facilitating efficient case handling and real-time progress tracking.

### Business Rules:

Utilization of Power Automate and Power Apps to automate business processes and enforce rules through business logic implemented via Power Automate flows and business rules. Power Platform's low-code environment allows for efficient workflow design and logic implementation, ensuring adherence to business rules throughout the application lifecycle.

### User Interface:

Power Platform focuses on automation while providing capabilities to interact with user interfaces through Power Apps. This enables the automation of backend processes that enhance user interactions across various applications integrated within the Power Platform ecosystem, improving user experience and operational efficiency.

### Integration:

Connecting various data sources and services using Power Platform's integration capabilities, including Power Automate connectors and Common Data Service (CDS). This ensures seamless data flow across applications and services through integrations with REST and SOAP APIs, external databases, and other enterprise systems, fostering a connected ecosystem for data sharing and process automation.

### Security:

Implementation of role-based access control and data security measures within Power Platform, ensuring that only authorized users can execute automated workflows and access sensitive data. The Power Platform provides robust access management features and security protocols to safeguard automation processes and maintain compliance with regulatory requirements.

### Deployment:

Strategies for deploying automated workflows using Power Automate's deployment tools and Application Lifecycle Management (ALM) practices, along with managing the application lifecycle through best practices for maintaining and updating automation projects within the Power Platform ecosystem. This includes version control, environment management, and change management processes to ensure smooth transitions during updates and enhancements.

# 2. User Authentication

# Microsoft Power Platform Design: User Authentication

## 1. Login

### Authentication Method:

Users will authenticate using their enterprise LDAP credentials integrated with Azure Active Directory (AAD) for seamless authentication. This will be achieved by configuring Azure AD authentication in Power Apps and using Azure AD APIs for validating user credentials.

### Implementation Steps:

## 1. Authentication Configuration:

* In Power Apps, enable Azure AD Authentication by configuring the Azure Active Directory as an identity provider.
* Ensure proper claims are mapped for user identification between Power Apps and Azure AD.

## 2. Response Handling:

* Use Power Automate to create a flow that evaluates the response from Azure AD.
* Based on the result, redirect the user to the Power Apps dashboard or display an error message using a notification component.

## 2. Password Policy

### Requirements:

* Minimum length: 8 characters
* Must include:
* At least one uppercase letter
* At least one lowercase letter
* At least one digit
* At least one special character

### Implementation Steps:

## 1. Define Password Validation Logic:

* Create a Power Automate flow that uses expressions to define regex patterns for password validation during user registration.
* Use the `triggerOutputs()` function to capture registration data and validate input values.

## 2. Enforce Password Policy:

* Implement a Power Automate flow that triggers on the `When an item is created` event in the user database to check the password against the defined policy before proceeding with user creation.

## 3. Multi-Factor Authentication (MFA)

### Verification Method:

A One-Time Password (OTP) will be sent via email or SMS for additional verification.

### Implementation Steps:

## 1. Configure Azure MFA:

* Set up Azure Multi-Factor Authentication settings within Azure AD as required.

## 2. Send OTP:

* Use Power Automate to send the OTP via email using the "Send an email (V2)" action.
* For SMS notifications, integrate with a third-party service like Twilio using the "HTTP" action to send SMS.

## 3. OTP Generation Workflow:

* Create a Power Automate flow that generates the OTP and invokes the sending mechanism.

## 4. Session Timeout

### Configuration:

Implement a session timeout of 15 minutes after inactivity.

### Implementation Steps:

## 1. Monitor User Activity:

* Use Power Apps session management features to monitor user sessions.

## 2. Log User Activity:

* Implement logging of user interactions using a SharePoint list or Dataverse table to record user activity.

## 3. Session Timeout Trigger:

* Use a Power Automate scheduled flow to check for inactivity. If no activity is detected for 15 minutes, use the Power Apps API to log out the user or update their session status.

## 5. Account Lockout

### Policy:

User accounts will be locked after 5 failed login attempts.

### Implementation Steps:

## 1. Track Login Attempts:

* Create a custom field (e.g., `failedLoginAttempts`) in the user record within the Dataverse to keep track of the number of failed login attempts.

## 2. Log Failed Attempts:

* Utilize a Power Automate flow that increments the `failedLoginAttempts` counter and logs each failed attempt to the user record.

## 3. Implement Lockout Logic:

* Use a Power Automate flow that checks if the count reaches 5.
* If the threshold is met, execute an action to lock the account (e.g., update the user status field to "Locked" and send a notification to an admin using the "Send an email (V2)" action).

This Microsoft Power Platform design replicates the user authentication functionalities originally defined in the ServiceNow design document.