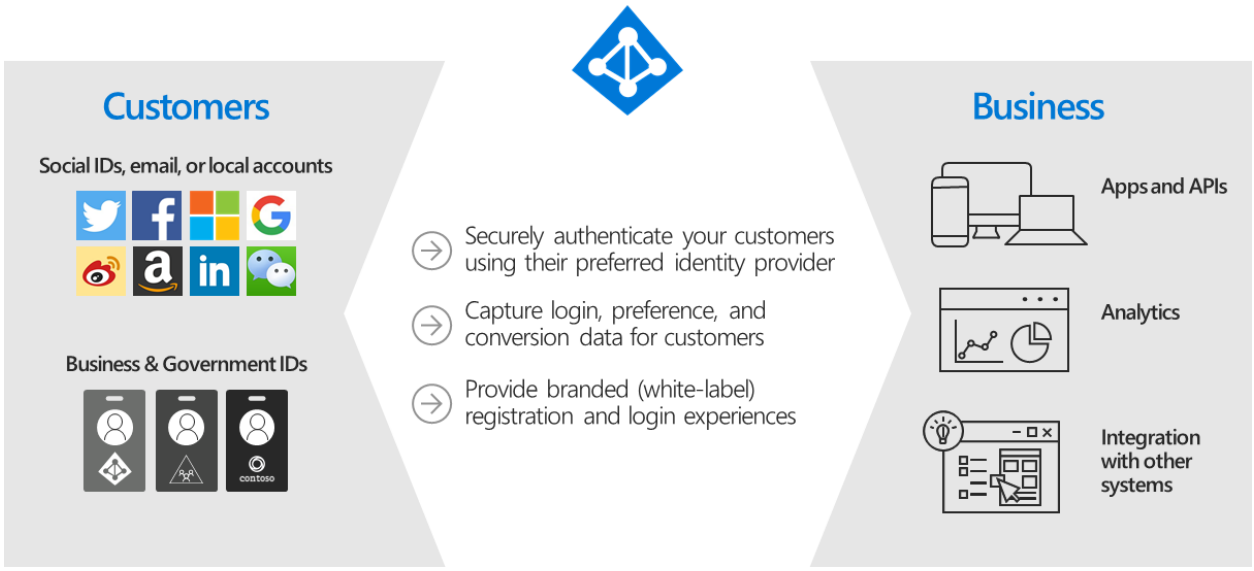
Authentication and Authorization using Azure B2C

Aadhaar Integration with Azure B2C for Secure Financial Transactions

---

Introduction:

In the digital age, securing financial transactions is paramount. This blog post delves into implementing Aadhaar authentication services within Azure AD B2C. Azure AD B2C is a customer identity access management (CIAM) solution capable of supporting millions of users and billions of authentications per day. It takes care of the scaling and safety of the authentication platform, monitoring, and automatically handling threats.



Problem Statement:

In the landscape of digital finance, traditional authentication methods often fall short in terms of security and user convenience. The need for a secure and user-friendly authentication system for financial transactions is crucial to mitigate risks associated with identity theft and unauthorized access.

Solution/Architecture:

The proposed solution integrates Azure B2C, a comprehensive identity management service, with Aadhaar authentication services. This comprehensive guide explores the step-by-step process of integrating Aadhaar authentication services into Azure B2C, creating a robust and secure framework for identity verification in financial transactions.

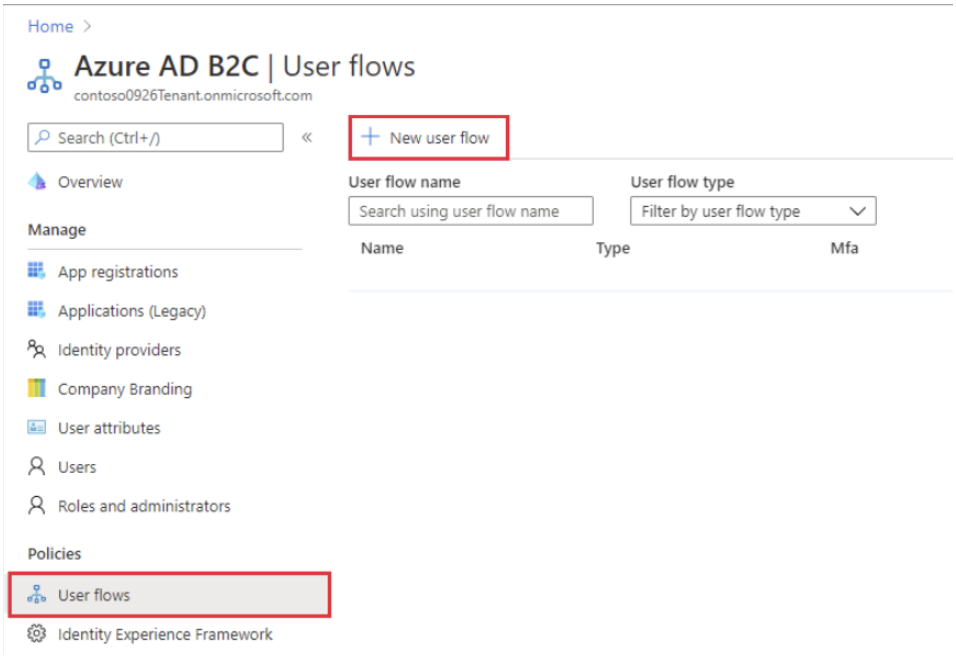
Prerequisites:

1. Azure B2C Tenant: Set up an Azure B2C tenant with the necessary user flows and policies.
2. Aadhaar API Access: Obtain access to Aadhaar authentication APIs from the Unique Identification Authority of India (UIDAI).

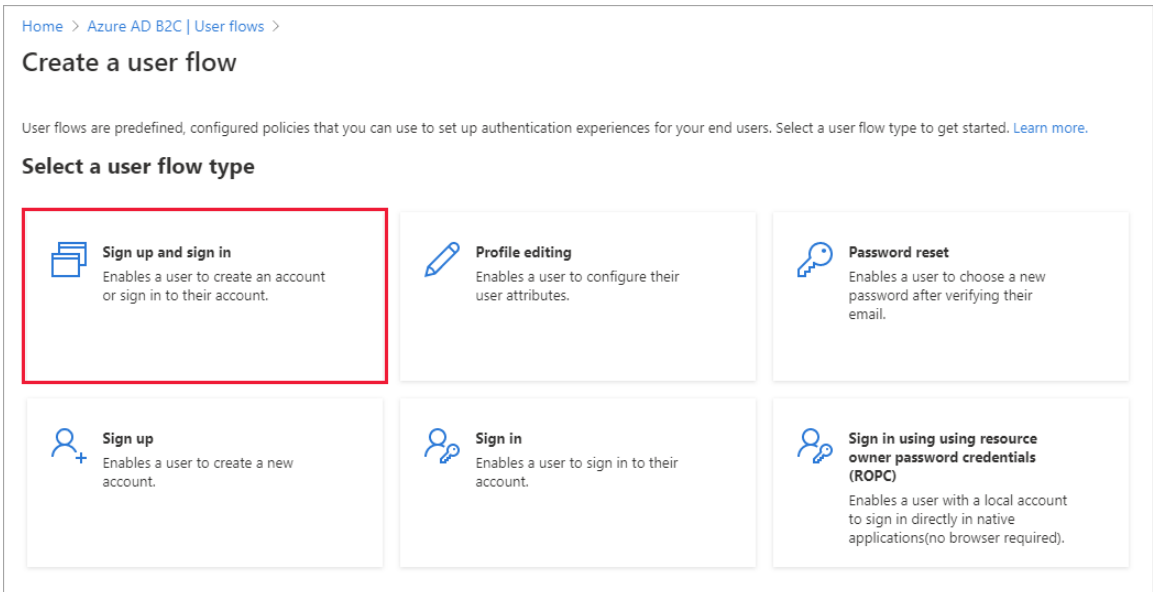
Step 1: Setting Up Azure B2C Tenant:

Create a New User Flow:

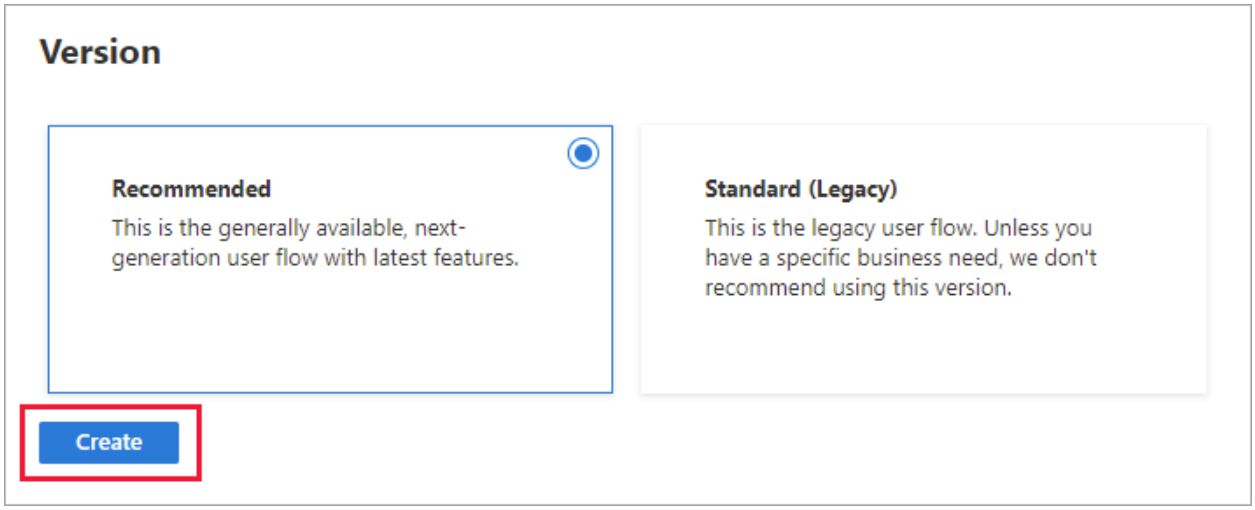
- Log in to the Azure portal, navigate to Azure B2C, and create a new user flow for Aadhaar authentication.
- Specify the user attributes required for verification.



- On the Create a User flow page, select Signup and sign in the user flow.

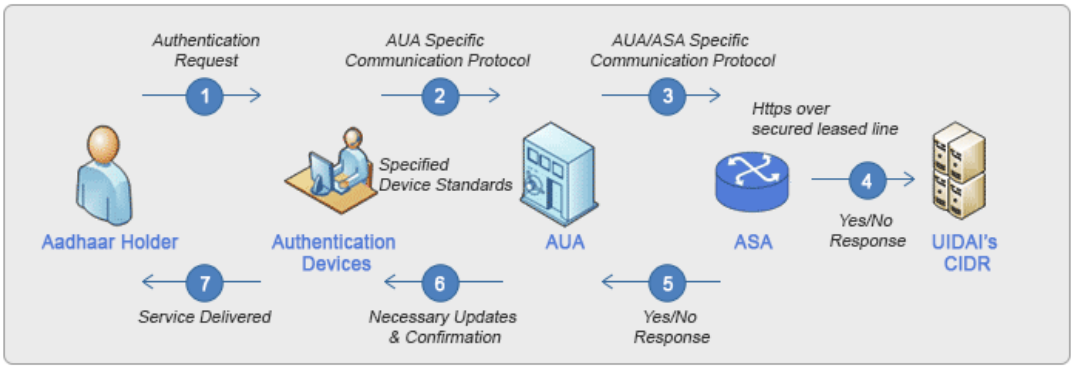


- Under Select a version, select Recommended, and then select Create.



Configure Identity Providers:

- Add Aadhaar as an identity provider in the Azure B2C settings.
- Configure the identity provider settings with the necessary metadata obtained from UIDAI.
- On the sign-up or sign-in page, Azure AD B2C presents a list of external identity providers the user can choose for sign-in. Once a user selects an external identity provider, they're redirected to the selected provider's website to complete their sign-in. After they successfully sign in, they're returned to Azure AD B2C for authentication with your application.



- Use Aadhaar OTP Request API: https://uidai.gov.in/ecosystem/authentication-devices-documents/authentication-documents.html

- Refer to Aadhaar Docs for more info https://uidai.gov.in/images/resource/Aadhaar_Authentication_API-2.5_Revision-1_of_January_2022.pdf

Step 2: Setting Up Azure B2C Tenant:

1. Obtain Aadhaar API Credentials:

    - Obtain the necessary credentials (Client ID, Client Secret) from UIDAI for accessing Aadhaar authentication APIs.

2. Integrate Aadhaar APIs in Azure Functions (Node.js):

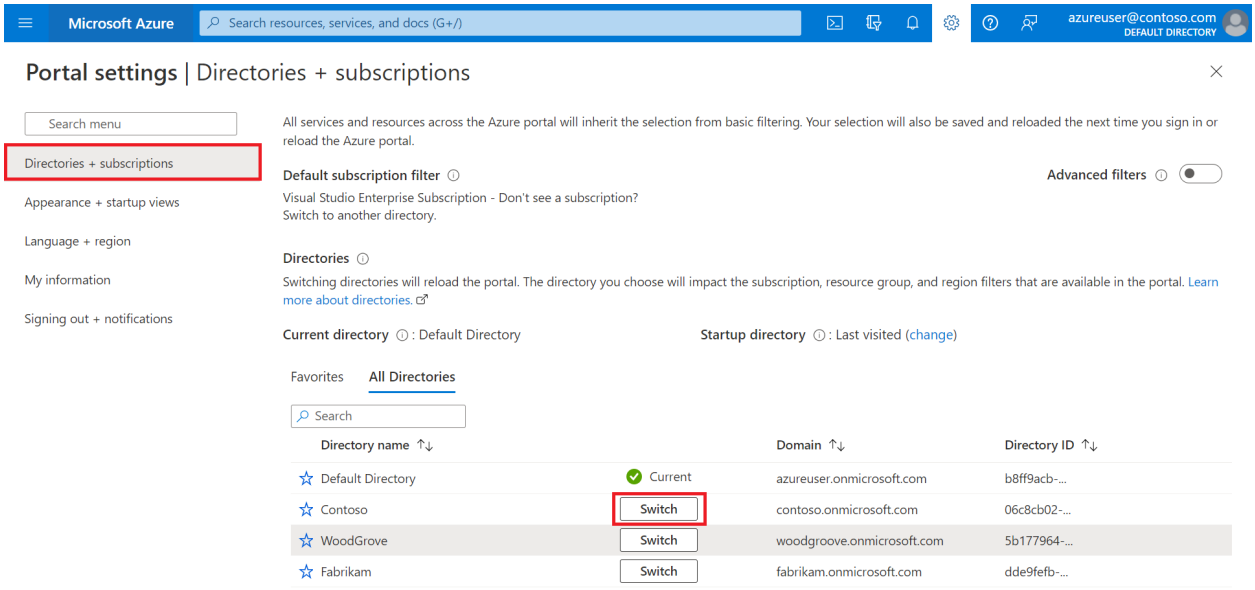
    - Create an Azure Function to handle Aadhaar API calls. (https://uidai.gov.in/images/resource/aadhaar_otp_request_api_2_5.pdf)
    - Use the Axios library to make secure API calls to Aadhaar services.

```
module.exports = async function (context, req) {
    const aadhaarApiUrl = 'https://aadhaarapi.example.com/verify';
    const { aadhaarNumber, biometricData } = req.body;

    try {
        const response = await axios.post(aadhaarApiUrl, {
            aadhaarNumber,
            biometricData,
        });

        // Process Aadhaar verification response
        context.res = {
            status: 200,
            body: response.data,
        };
    } catch (error) {
        // Handle errors
        context.res = {
            status: 500,
            body: error.message,
        };
    }
};
```

Step 3: Setting Up Azure B2C Tenant:

1. In the Azure portal toolbar, select the Directories + Subscriptions icon.

2. On the Portal settings | Directories + subscriptions page, find your Microsoft Entra directory that contains your subscription in the Directory name list and then select the Switch button next to it.



3. Add Microsoft.AzureActiveDirectory as a resource provider for the Azure subscription you're using (learn more):

- On the Azure portal, search for and select Subscriptions.
- Select your subscription, and then in the left menu, select Resource Provider. If you don't see the left menu, select the Show the menu for < name of your subscription > icon at the top left part of the page to expand it.
- Make sure the Microsoft.AzureActiveDirectory row shows a status of Registered. If it doesn't, select the row, and then select Register.

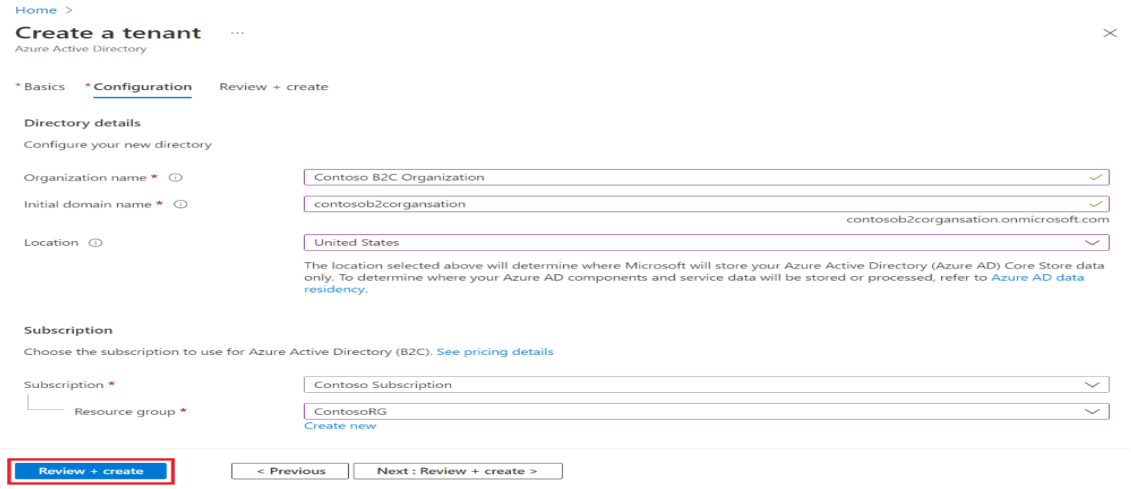
4. On the Azure portal menu or from the Home page, select Create a resource Search for Azure Active Directory B2C, and then select Create.

5. For the Organization name, enter a name for your Azure AD B2C tenant.

6. For the Initial domain name, enter a domain name for your Azure AD B2C tenant.

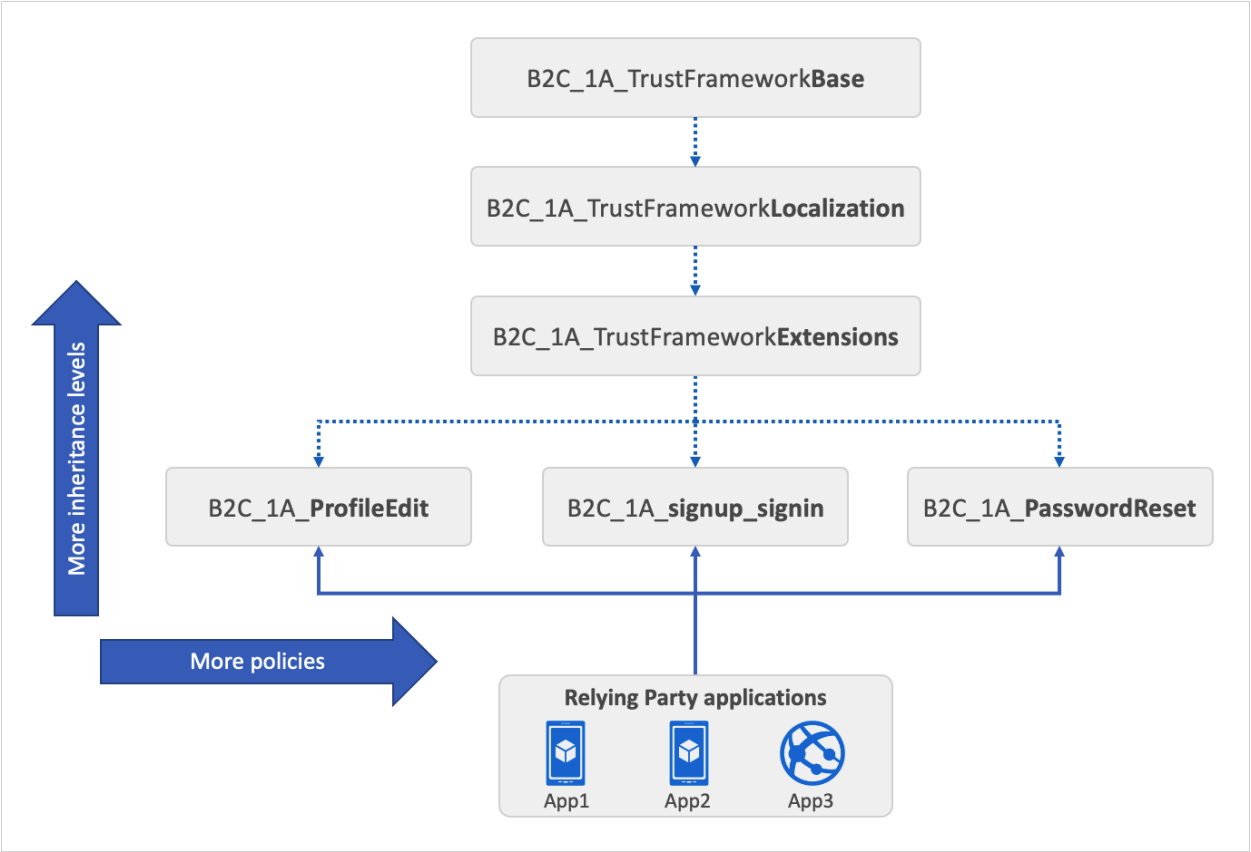
7. For Subscription, select your subscription from the list.

8. For the Resource group, select or search for the resource group that will contain the tenant.



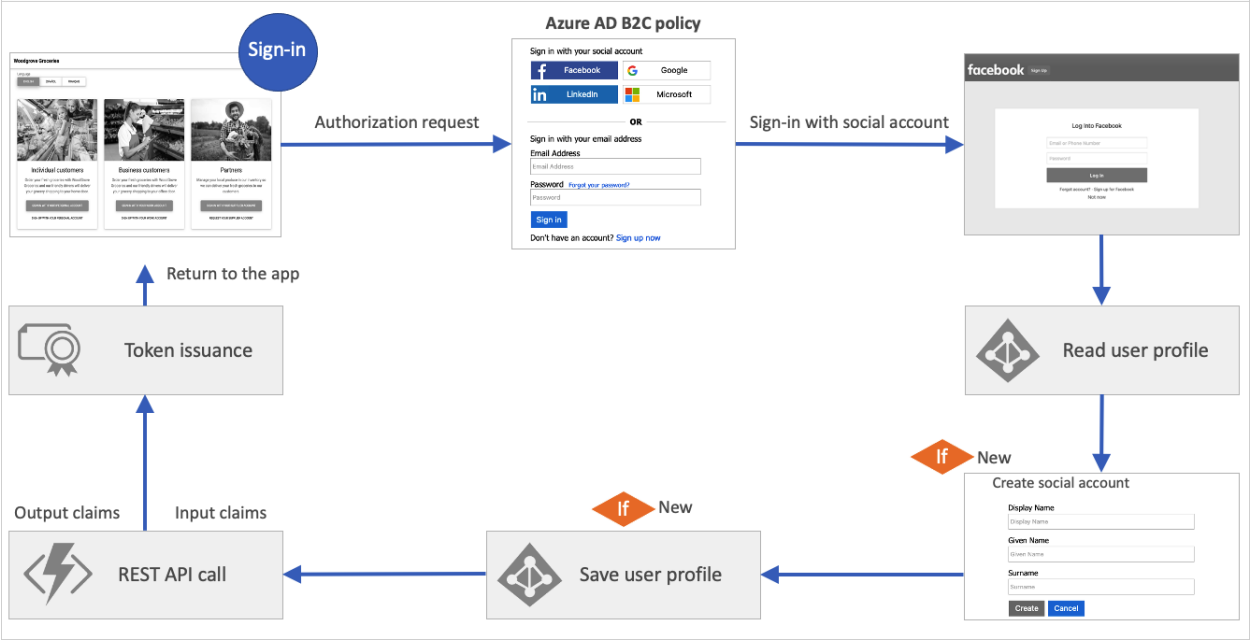
9. Custom Policies:

- Create custom policies in Azure B2C to integrate the Aadhaar API verification process.
- Specify the necessary claims and transformations in the custom policies.



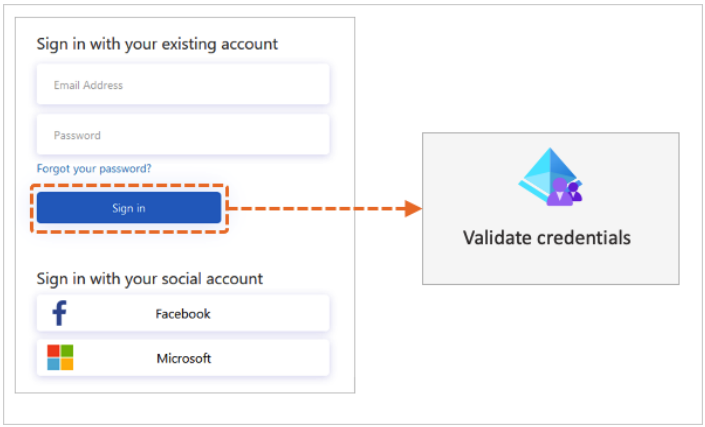
10. User Journey Configuration:

- Define user journeys that incorporate Aadhaar authentication in the Azure B2C custom policies.
- Map the claims received from Aadhaar to Azure B2C user attributes.



11. Validation technical profile
    When a user interacts with the user interface, you may want to validate the data that is collected. To interact with the user, a self-asserted technical profile must be used.

    To validate the user input, a validation technical profile is called from the self-asserted technical profile. A validation technical profile is a method to call any non-interactive technical profile. In this case, the technical profile can return output claims or an error message. The error message is rendered to the user on-screen, allowing the user to retry.



Step 4: Testing and Debugging:

1. Test User Flows:

    a. Create custom policies in Azure B2C to integrate the Aadhaar API verification process.
    b. Ensure that user attributes are correctly mapped and populated.

2. User Journey Configuration:

    a. Integrate Azure Application Insights for logging and monitoring Aadhaar authentication events.
    b. Use Application Insights to identify and resolve any issues during the authentication process.

Step 5: Addressing Challenges:

1. Reliability Enhancement:

    - Implement retry mechanisms in case of Aadhaar API failures.
    - Monitor and analyze API response times for optimisation.

2. Regulatory Compliance:

    - Regularly check for updates to UID regulations and update the integration accordingly.
    - Ensure secure handling of Aadhaar data to comply with privacy laws.

Challenges in implementing the solution:

While Azure B2C offers a powerful solution, there are some challenges to consider:

1. Learning Curve:

Getting familiar with Azure B2C's features and configuration options may take time for developers who are new to the platform.

2. Customization Complexity:

While customization is a strength, complex customization scenarios may require expertise in policy configuration.

3. Cost:

Azure B2C is a paid service, and the cost can vary depending on the number of users and transactions.

Business Benefit:

The integration of Aadhaar authentication with Azure B2C for secure financial transactions offers several significant business benefits, contributing to the overall growth and trustworthiness of digital financial services in India. Here are the key business benefits highlighted in the blog:

Enhanced Security with Azure B2C Authentication:

The integration of Aadhaar authentication with Azure B2C establishes a robust and secure identity verification process. Azure B2C's authentication capabilities, combined with Aadhaar's unique features, create a multi-layered security framework, significantly reducing the risk of unauthorized access and identity theft.

Improved Trust through Govt-Backed Authentication:

Leveraging Aadhaar, a government-backed identity system, enhances user trust in the authentication process. Businesses

utilizing Azure B2C with Aadhaar authentication demonstrate a commitment to incorporating trusted and reliable identity verification methods in financial transactions.

Cost-Efficient Identity Verification:

Aadhaar integration, coupled with Azure B2C, offers a cost-efficient solution for identity verification. Businesses can leverage the existing Aadhaar infrastructure, reducing the need for extensive investments in separate identity verification systems and operational costs.

Conclusion:

Integrating Aadhaar authentication services with Azure B2C provides a secure and reliable framework for financial transactions. By following this detailed guide, businesses can establish a robust identity verification process, enhancing the security and trustworthiness of digital financial services in India.

Author: Vasanth Korada
Email: vasanthkorada999@gmail.com