WF-05: B2C (Business-to-Consumer) users - Portal Customization

Updates

	Last Update	Jira Ticket	Author	Details
1	17 Nov 2023	EUAP-XXXX		
2				

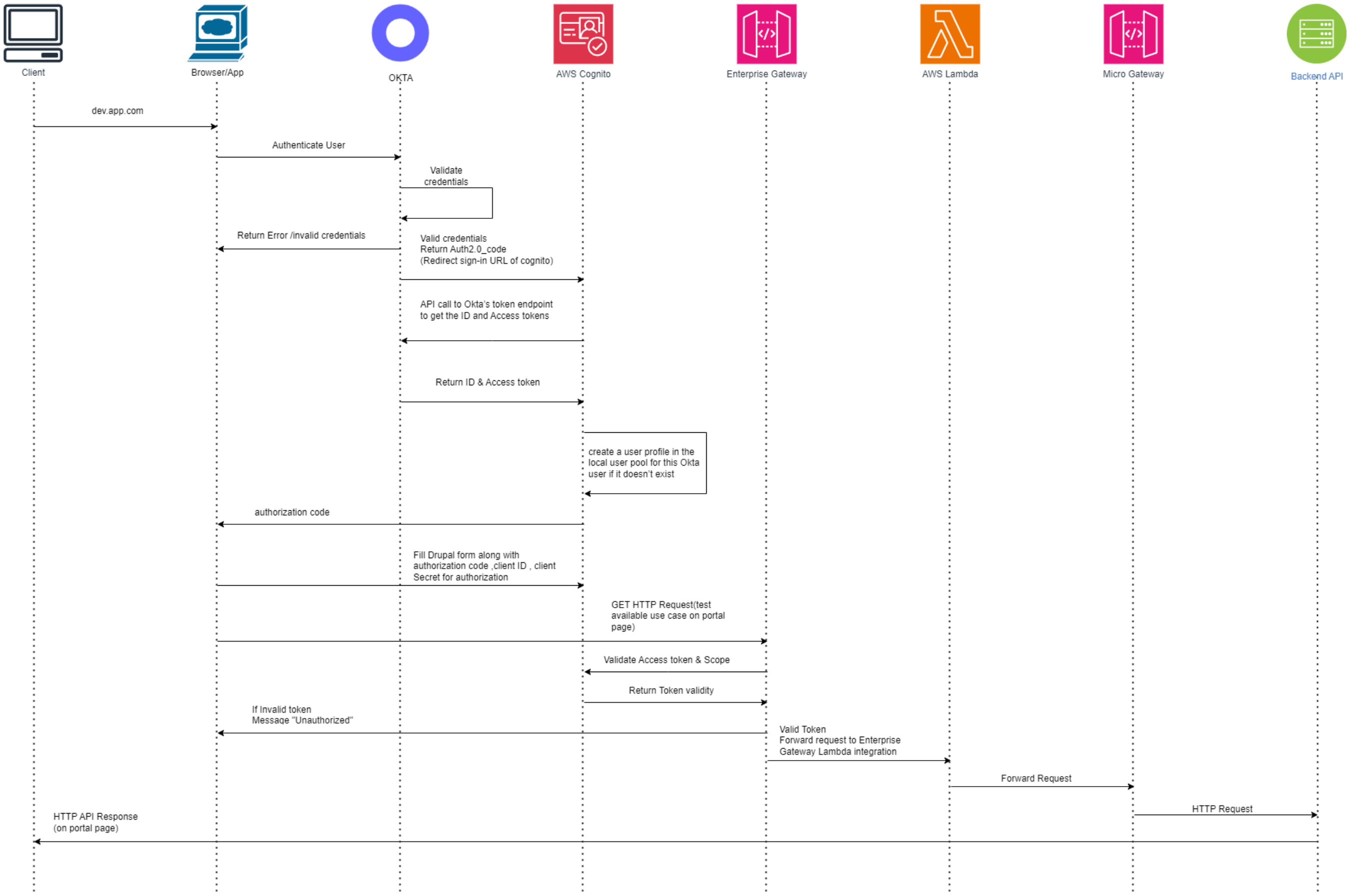
Overview

This is the page to shows how works the end-to-end process between Okta IDP, the AWS Cognito and AWS Dynamo DB.

Pre-requisites

- OIDC is used for token base authentication
- Create OKTA Identity provider in AWS and map it with the respective AWS Cognito user pool
- Add OKTA as identity provider in AWS Cognito user pool
- OKTA Application Client ID and Secrets required in AWS Cognito for OpenID connect
- Required Issuer URL from OKTA
- Map attributes between OKTA OpenID connect (OIDC) and AWS Cognito user pool. OIDC is used for token base authentication
- Configure a AWS Cognito pre-sign-up trigger
- Create a Dynamo DB table which can store the user details.
- Create a lambda function that have dynamo DB table as target for storing the user details
- Integrate AWS Cognito user pool with AWS API gateway as a authorizer

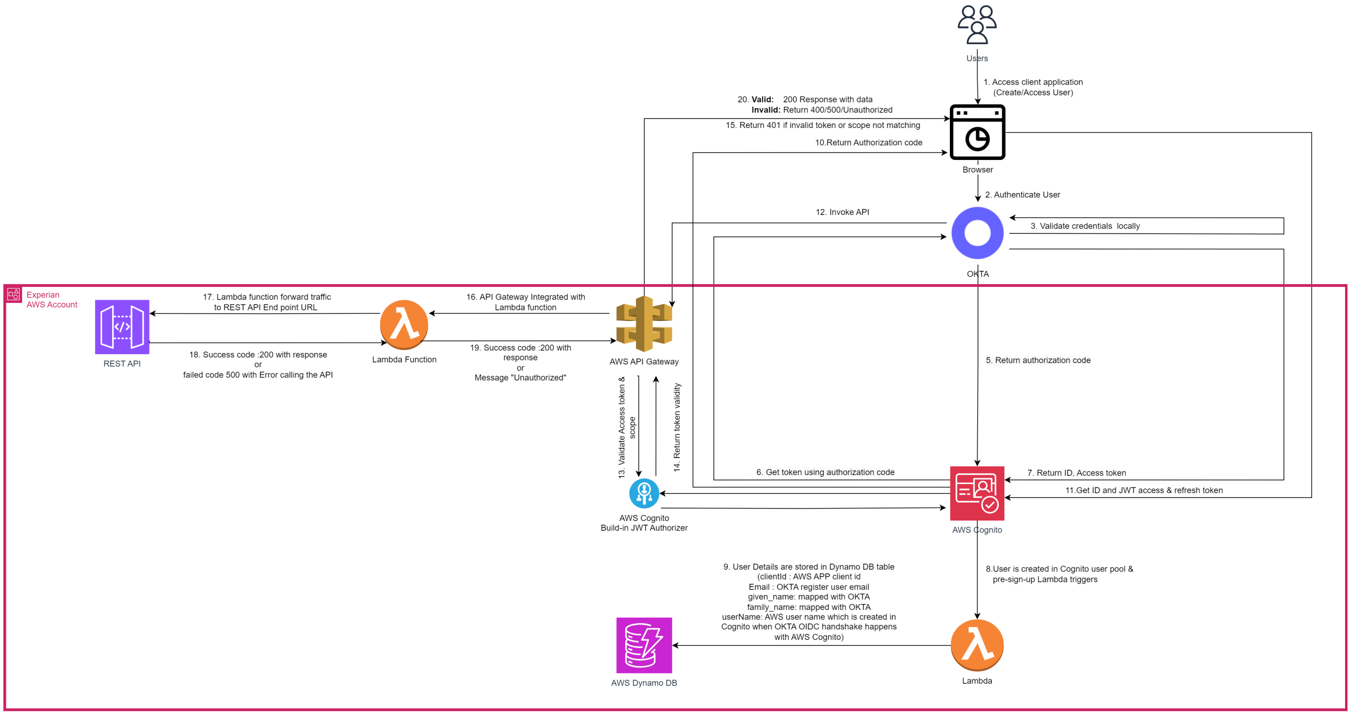
Sequence Diagram



Sequence Diagram Details

1. Client
    a. Access the URL dev.app.com
    b. ...
2. Browser/App
    a. ...
3. ...

Flow Diagram


Users
1. Access client application
(Create/Access User)
20. Valid: 200 Response with data
Invalid: Return 400/500/Unauthorized
15. Return 401 if invalid token or scope not matching
10.Return Authorization code
Browser
2. Authenticate User
12. Invoke API
3. Validate credentials locally
OKTA
Experian
AWS Account
17. Lambda function forward traffic
to REST API End point URL
16. API Gateway Integrated with
Lambda function
REST API
Lambda Function
AWS API Gateway
18. Success code :200 with response
or
failed code 500 with Error calling the API
19. Success code :200 with
response
or
Message "Unauthorized"
5. Return authorization code
13. Validate Access token &
scope
14. Return token validity
6. Get token using authorization code
7. Return ID, Access token
11.Get ID and JWT access & refresh token
AWS Cognito
Build-in JWT Authorizer
AWS Cognito
9. User Details are stored in Dynamo DB table
(clientId : AWS APP client id
Email : OKTA register user email
given_name: mapped with OKTA
family_name: mapped with OKTA
userName: AWS user name which is created in
Cognito when OKTA OIDC handshake happens
with AWS Cognito)
8.User is created in Cognito user pool &
pre-sign-up Lambda triggers
AWS Dynamo DB
Lambda

Flow Diagram Details

1. REST API
    a. ...
2. Lambda Function
3. AWS API Gateway
4. ...

Steps

a. User opens a browser and types the client app URL eg. “experian.buname.sb.com”
b. Automatically initiate an Authentication/OIDC flow with AWS Cognito to authorize url. OR Allow user to enter login details for Authentication. AWS Cognito receives the request from client app and either displays the login page with a button to initiate the OIDC federation or automatically initiate the OIDC federation with Okta by redirecting to /authorize URL.
c. Okta will return an authorization code back to AWS Cognito.
d. AWS Cognito will make a backend API call to Okta’s token endpoint to get the ID and Access tokens. Okta will return an ID and Access tokens
e. AWS Cognito will create a user profile in the local user pool for this Okta user if it doesn’t exist and redirect back to Client app with an authorization code.
f. AWS Cognito will return and ID and Access tokens
g. Client app will validate the ID token, check if it is a valid user and return the web page to the browser
h. Client app will invoke API with Access token
i. AWS API gateway will validate Access Token and scope with AWS Cognito
j. AWS Cognito will return the token validity
k. For valid token return 200 Response with Data or return 403 for invalid token on the browser
l. When OKTA return ID and Access token to Cognito. The Cognito create a user profile in the Cognito User pool .
m. As pre configure AWS lambda is attached with user pool that will send the user details in AWS Dynamo DB.
n. As the user creation happens, lambda triggers and can store below fields in Dynamo DB tables

ClientId : AWS APP client id
Email : OKTA register user email
given_name: mapped with OKTA
family_name: mapped with OKTA
userName: AWS user name which is created in Cognito when OKTA OIDC handshake happens with AWS Cognito