

# Service Design - New

- Deliverables Checklist :: Demo and Review Status
    - Brillio Technical Team
    - Experian Technical Team
  - Overview
  - Business Use Cases
    - WF-01: Service Catalog
    - WF-02: Setup - CI/CD Harness & Terraform Scripting
    - WF-03: DNS Creation, Registration and Certificate Management
      - De-centralized VPC endpoint mapping w.r.t. BU account:
    - WF-04: Admin - Portal Customization
    - WF-05: B2C (Business-to-Consumer) users - Portal Customization
    - WF-06: B2B (Business-to-Business) users - Onboarding
    - WF-07: B2B (Business-to-Business) users - API Invocation
  - High Level Diagram
  - Important documentation links related

## **Deliverables Checklist :: Demo and Review Status**

9	b) Portal Customization: Publishing APIs to Developer Portal												
10	c) Portal Customization: Creating apps, adding products using Drupal Form												
11	API Security Assessment												
12	Final Demo 1 - End to end integration												
13	Final Demo 2 - end to end integration												
14	Final Demo 3 - end to end integration												

## Overview

**Goal/Purpose:** Implement the AWS API Gateway including portal customization, CI/CD, perform a full API Security review and deploy two APIs end to end. Deliver the templates to deploy the enterprise gateway and Microgateways and deploy them to sandbox or testing environment. Experian cloud team will deploy the templates on higher environment.

**Out of Scope:** Assisting the BU development team in accelerating the AWS API Gateway adoption, ie. development support as well as architecture support

### Standard Operating Protocols (SoP):

The templates will be applied on Sandbox or testing environment, Experian cloud team will be responsible for deploying the scripts to other environment. The project team requires appropriate access to the necessary systems, tools, and environments (Sandbox access, Service Catalog, GitHub, CI/CD, Okta access, Harness . Need administrative access to the Drupal content management system (CMS).

### Highlights of the suggested requirements for the Enterprise AWS API GW solution:

- API Proxies should be tech stack agnostic and aligned to Open API Specification
- Managed API Proxies instances and enterprise ingress separately
- Utilize cloud-native single or cross-role-based security key vaults
- Manage similar proxies under one App with multiple Apps under one product
- Utilize serverless computing to emulate the pre-flows and post-flows
- Utilize Account and organization-wide guardrails to manage the product
- Loosely Coupled with IdP so any major IdP vendor can be integrated with plug and play
- Secure OAuth Flow – prefer not to use passwords or implicit
- Manage similar proxies under one App with multiple Apps under one product
- Automate the API Gateway from Service Catalog
- Automate the Proxy Upload
- Automate the Proxy deployment artifacts as input to the Developer portal publishing
- Utilize the policy engine to evaluate IaC configurations before deployment

Ensure the AWS API Gateway proxies are handled with similar or the same WAF as the current solution Automate the Proxy Upload

- Cloud-native OWASP API Security Top 10 controls at the API Gateway
- Cloud-native OWASP API Security Top 10 controls at the API Gateway
- Implement the OWASP API Security Top 10 at the Gateway level (API Product integrated as part of DevOps pipeline)

#### Self-Documentation API with Swagger generation

- Create outputs from automated deployment that can be used to publish in the portal Expand the capability of Experian to integrate with new API-M solutions
- Integrate API-M solution with the portal deployment
- Automate change management with human-in-the-loop controls

Apigee will Continue to serve as Experian Enterprise API-M in parallel Experian AWS API Gateway will be available for Business Units to use on AWS. A decision tree will be published to determine when to use Apigee Vs. AWS Okta will be used as one of the Identity provider options for the AWS API Gateway

• Introduce more identity providers in the future Experian API Portal will serve as the external API Developer Portal The API AWS Gateway will be available as a service on AWS Service Catalog A CI/CD pipeline will be available for deploying the APIs on the AWS API gateway A shared global Enterprise AWS API Gateway will be introduced to enforce global security policies.

- Enterprise AWS API Gateway will be owned by API CoE
- The Micro gateways will be owned by BUs

## Business Use Cases

Following are the business use cases for the various workflows (WF) of the solution

### WF-01: Service Catalog

**Primary Actor:** APICoE Admin Team

**Maintained By:** APICoE team

**Description:** This workflow is part of the infrastructure setup processes that will act as a pre-requisite for BU users to manage API-LCM. Once the workflow completes, it will ensure that a customized portfolio of ServiceCatalog products are created for the BU Admin, that when executed (as a Harness job, refer WF-02) will launch the relevant products for the specified portfolio.

**Usage:**

- 1) When a new Service Catalog portfolio needs to be created for a specific Business Unit

**Components:**

- 1) Harness
- 2) Service Catalog
- 3) AWS Account
- 4) Okta Integration

**Pre-requisites:**

- 1) Integrate Harness with SSO
- 2) Provide access to APICoE Admin to log into Harness via SSO
- 3) The AWS account, Region, Availability Zone, IAM roles, IAM policy is provisioned by the Cloud Platform Engineering (CPE) team
- 4) Harness Job for launching service catalog products is created for APICoE Admin

**Steps:**

- 1) APICoE logs into Harness via SSO
- 2) APICoE Admin runs job 'Create ServiceCatalog Portfolio' that will internally invoke '[ServiceCatalogMain.tf](#)', for creating a portfolio, associate portfolio w.r.t. the products, and assign the portfolio as a Service Catalog entry.
- 3) The Service Catalog portfolio created, will need to be allowed to be shared with the master account.

**Result:**

- 1) The portfolio with relevant products for BU Admin are setup under Service Catalog

### WF-02: Setup - CI/CD Harness & Terraform Scripting

**Primary Actor:** Business Unit (BU) Team

**Maintained By:** APICoE team

**Description:** This workflow is part of the infrastructure setup processes that will act as a pre-requisite for BU users to manage API-LCM. Once the workflow completes, it will ensure the end-to-end infrastructure for a new Enterprise Gateway solution will be available for the respective Business unit.

**Components:**

- 1) Harness
- 2) Service Catalog
- 3) AWS Account
- 4) Okta Integration

**Usage:**

- 1) When new BU needs to create a dedicated Enterprise Gateway solution
- 2) When BU would want to onboard a new API

**Pre-requisites:**

- 1) Integrate Harness with SSO
- 2) Provide access to BU User/BU Admin and APICoE Admin to log into Harness via SSO
- 3) The AWS account, VPC, VPCe, subnet, Availability Zone, CIDR block, Region, IAM roles, IAM policy, S3 is provisioned by the Cloud Platform Engineering (CPE) team
- 4) The portfolio with relevant products for BU Admin and APICoE Admin are setup under Service Catalog
- 5) Harness Job for launching service catalog products is created for BU Admin and APICoE Admin

**Steps:**

- 1) BU Admin receives confirmation from CPE team on dedicated infrastructure provisioning
- 2) BU logs into Harness and modifies service overrides, for the job 'Launch BU Enterprise Gateway', with the infrastructure parameters information provided by CPE team

Alternatively, BU can also send serviceNow request to APICoE admin to modify file under bitbucket repository w.r.t. infrastructure info provided by CPE team and check into the repository

- 3) The Harness job at the background triggers the '[ServiceCatalogMain.tf](#)' that will allow the BU Admin to launch products related to the portfolio 'BU Admin'
- 4) The job returns the result of successfully launching products under the respective AWS account.
- 5) The following products are launched/provisioned associated with BU AWS account  
Enterprise Gateway, Lambda function, AWS Certificate Manager, REST APIs, AWS Cognito, AWS DynamoDB, CloudTrail, CloudWatch
- 5) Check in the AWS console to view the above mentioned AWS services are provisioned.

## **WF-03: DNS Creation, Registration and Certificate Management**

**Primary Actor:** APICoE Admin Team

**Maintained By:** APICoE team

**Description:** This workflow is part of the infrastructure setup processes that will allow Business Units to setup and apply certificates to Enterprise gateway services on cloud and allow Business Users to invoke APIs from their respective enterprises.

**Usage:**

- 1) This will allow the certificates created for a specific DNS to be applied to the AWS Certificate Manager and Enterprise Gateway as a part of establishing certificate based authentication allowing traffic to be routed through CSS Ingress to AWS API GW solution.

**Components:**

- 1) S3
- 2) AWS Certificate Manager
- 3) Lambda Function
- 4) AWS API GW
- 5) Microgateway

**Pre-requisites:**

- 1) Integrate Harness with SSO
- 2) Provide access to APICoE Admin to log into Harness via SSO
- 3) The AWS account, Region, Availability Zone, IAM roles, IAM policy is provisioned by the Cloud Platform Engineering (CPE) team
- 4) Harness Job for launching service catalog products is created for APICoE Admin

**Steps:**

- 1) APICoE team gets approval from marketing team on 'DNS' entry. The APICoE raises a request for a new DNS entry via Email sent to the CSS team. A serviceNow ticket is created with the CName/DNS
- 2) CSS team creates and registers the domain and updates the environment based on ServiceNow approval
- 3) Certificate created by the APICoE team. 3 Certificates are manually downloaded and the content is copied into a pem file eg. 'certwith\_chain.pem'
- 4) API CoE team applies the pem file to both, WAF and S3 bucket
- 5) APICoE logs into Harness via SSO
- 6) APICoE Admin runs job 'Create ServiceCatalog Portfolio' that will internally invoke '[ServiceCatalogMain.tf](#)', for creating a portfolio, associate portfolio w.r.t. the products, and assign the portfolio as a Service Catalog entry.
- 7) The Service Catalog portfolio created, will need to be allowed to be shared with the master account.
- 8) Lambda function updateTruststore() gets triggered & update the AWS Certificates Manager
- 9) Lambda function applyCertificate() gets triggered & updates AWS API GW with the certificate
- 10) VPC endpoint routing table will be configured to identify the businessunit and the respective VPC endpoint eg /cisisusgateway/stage, /bisusgateway/test
- 11) Rules needs to be defined as routes in Enterprise API GW eg. cisisusgateway/v1/token, bisusgateway/stage/v1/token

Refer Cloud Security Stack (CSS) Ingress for Domain Registration and Certificate Management.

Diagram Source - E-Connect (v3) Ingress - Simplified - Cloud Engineering - Confluence Global ([experian.com](https://experian.com))

<https://pages.experian.com/display/SC/E-Connect+%28v3%29+Ingress++Simplified#tab-Flow-to+Experian+Private+API+Gateway>

### Result:

- The DNS is registered on WAF and certificates are applied for Business users to be able to access the relevant APIs using the public url 'enterpriseapi.experian.com'.

### De-centralized VPC endpoint mapping w.r.t. BU account:

The excel sheet defining the example of setting up the environment w.r.t. VPC endpoints for each Business Unit account.

Environment	BU	URI for the Business user	WAF	VPC Endpoint	Custom DNS for API GW	AWS API GW Routing Table	REST API
PROD	SandBox	<a href="https://enterpriseapi.experian.com/cisapigateway/v1/token">https://enterpriseapi.experian.com/cisapigateway/v1/token</a>	enterpriseapi.experian.com waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.com waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://enterpriseapi.experian.com/cisapigateway/v1/checksum">https://enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	enterpriseapi.experian.com waf_id -> 89762 - /cisapigateway/	enterpriseapi.experian.com waf_id -> 89762 - /cisapigateway/	enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
PROD	Sandbox	<a href="https://enterpriseapi.experian.com/cisapigateway/v1/token">https://enterpriseapi.experian.com/cisapigateway/v1/token</a>	sandbox.enterpriseapi.experian.com waf_id -> 23728 - /cisapigateway/	sandbox.enterpriseapi.experian.com waf_id -> 23728 - /cisapigateway/	sandbox.enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://enterpriseapi.experian.com/cisapigateway/v1/checksum">https://enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	sandbox.enterpriseapi.experian.com waf_id -> 23726 - /cisapigateway/	sandbox.enterpriseapi.experian.com waf_id -> 23726 - /cisapigateway/	sandbox.enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
UAT	Sandbox	<a href="https://uat.enterpriseapi.experian.com/cisapigateway/v1/token">https://uat.enterpriseapi.experian.com/cisapigateway/v1/token</a>	uat.enterpriseapi.experian.com waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.com waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://uat.enterpriseapi.experian.com/cisapigateway/v1/checksum">https://uat.enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	uat.enterpriseapi.experian.com waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.com waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
NON-PROD	Stage	<a href="https://stage.enterpriseapi.experian.com/cisapigateway/v1/token">https://stage.enterpriseapi.experian.com/cisapigateway/v1/token</a>	stage.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://stage.enterpriseapi.experian.com/cisapigateway/v1/checksum">https://stage.enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	stage.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
TEST	Sandbox	<a href="https://test.enterpriseapi.experian.com/cisapigateway/v1/token">https://test.enterpriseapi.experian.com/cisapigateway/v1/token</a>	test.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://test.enterpriseapi.experian.com/cisapigateway/v1/checksum">https://test.enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	test.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.com waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
DEV	Sandbox	<a href="https://dev.enterpriseapi.experian.com/cisapigateway/v1/token">https://dev.enterpriseapi.experian.com/cisapigateway/v1/token</a>	dev.enterpriseapi.experian.com waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.com waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.com	cisapigateway/v1/token	/v1/token
		<a href="https://dev.enterpriseapi.experian.com/cisapigateway/v1/checksum">https://dev.enterpriseapi.experian.com/cisapigateway/v1/checksum</a>	dev.enterpriseapi.experian.com waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.com waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.com	cisapigateway/v1/checksum	/v1/checksum
PROD	Sandbox	<a href="https://enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	enterpriseapi.experian.co.uk waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.co.uk waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	enterpriseapi.experian.co.uk waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.co.uk waf_id -> 89764 - /cisapigateway/	enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum
PROD	Sandbox	<a href="https://sandbox.enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://sandbox.enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	sandbox.enterpriseapi.experian.co.uk waf_id -> 23728 - /cisapigateway/	sandbox.enterpriseapi.experian.co.uk waf_id -> 23728 - /cisapigateway/	sandbox.enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://sandbox.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://sandbox.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	sandbox.enterpriseapi.experian.co.uk waf_id -> 23726 - /cisapigateway/	sandbox.enterpriseapi.experian.co.uk waf_id -> 23726 - /cisapigateway/	sandbox.enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum
UAT	Sandbox	<a href="https://uat.enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://uat.enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	uat.enterpriseapi.experian.co.uk waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.co.uk waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://uat.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://uat.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	uat.enterpriseapi.experian.co.uk waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.co.uk waf_id -> 109844 - /cisapigateway/	uat.enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum
NON-PROD	STAGE	<a href="https://stage.enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://stage.enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	stage.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://stage.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://stage.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	stage.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	stage.enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum
TEST	Sandbox	<a href="https://test.enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://test.enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	test.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://test.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://test.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	test.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.co.uk waf_id -> 615781 - /cisapigateway/	test.enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum
DEV	Sandbox	<a href="https://dev.enterpriseapi.experian.co.uk/cisapigateway/v1/token">https://dev.enterpriseapi.experian.co.uk/cisapigateway/v1/token</a>	dev.enterpriseapi.experian.co.uk waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.co.uk waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.co.uk	cisapigateway/v1/token	/v1/token
		<a href="https://dev.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum">https://dev.enterpriseapi.experian.co.uk/cisapigateway/v1/checksum</a>	dev.enterpriseapi.experian.co.uk waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.co.uk waf_id -> 510844 - /cisapigateway/	dev.enterpriseapi.experian.co.uk	cisapigateway/v1/checksum	/v1/checksum

## WF-04: Admin - Portal Customization

**Primary Actor:** APICoE Admin

**Maintained By:** APICoE team

**Description:** This workflow essentially highlights the process for publishing APIs to AWS API Gateway from Experian Drupal Environment. To publish APIs admin needs to Authenticate Via OKTA. Parameters that are required to Publish an API are ,API/Product name, API region portfolio user, Business unit, Threshold limit and Swagger file.

### Components:

- Okta Integration
- REST APIs
- AWS Account

### 4) Drupal Setup

### 5) Bitbucket

### 6) AWS Enterprise Gateway solution

### Usage:

- When APICoE admin should be able to publish APIs for BU user Consumption via Drupal AWS connector

### Pre-requisites:

- Swagger file is provided by the Business unit
- The drupal environment should be up and running
- AWS Enterprise Gateway infrastructure should be up and running

### High-level Requirements

- The APICoE Admin (Internal) user login via SSO. The user clicks on <https://enterpriseapi.experian.com/admin> (Jagadeesh to confirm)
  - The Admin user should automatically get logged in via Okta SSO and get the <>form name>>
- Once SSO authenticated on OKTA Admin will be redirected to Form to publish APIs to AWS API Gateway.
- Admin to enter all the details on the form as below
  - Create an Admin form with following fields:
  - API/Product Name (/token,/checksum) **Input Box**
  - API Region (UK & US) **Dropdown**
  - Portfolio User (BU Admin, APICoE Admin) **Dropdown**
  - Business unit (BIS Gateway, CIS Gateway) **Dropdown**
  - Threshold Limit (Limit for API publishing) **Input Box**
  - Browse (to upload swagger file) **File Input**
  - Submit button and Clear **Button**
- After filling all the details when admin clicks Submit, It will invoke AWS Drupal connector and publish the API to AWS API gateway.
- Once the request is success, AWS gateway will return the success response along with the display/list of all APIs from the AWS API gateway (All the APIs which published for that particular region)

**Form Design :**

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

**Primary Actor:** APICoE Admin

**Maintained By:** APICoE team

**Description:** This workflow essentially highlights the process for allowing end users to register and log into the gdp portal to explore MyApp creation and adding products/APIs for a specific region, to it. This would also enable them to generate AWS Cognito Client\_id/client\_secret and further generate JWT access\_token creation for them to be able to invoke the relevant APIs. This feature would only be restricted for B2C users on sandbox environment. This would not be deployed or is meant to be used on Production (live) environment.

**Components:**

- 1) Okta Integration
- 2) REST APIs
- 3) AWS Account
- 4) Drupal Setup
- 5) Bitbucket
- 6) AWS Enterprise Gateway solution

**Usage:**

- 1) The B2C user should be able to register as a user, login into the portal, create a customized app, add products/APIs to the customized app, generate client\_id/client\_secret, authorize the APIs and generate JWT token

**Pre-requisites:**

- 1) The drupal environment should be up and running
- 2) Relevant APIs should already be published
- 3) AWS Enterprise Gateway infrastructure should already be running

**Steps:**

- 1) Log into Developer Portal (<https://developer.experian.com>)
- 2) Click on My Apps link in user navigation menu and choose the Region(Click on View My Apps button)
- 3) Click on the create new app button  
A new form that will take the 'App Name' as an input and checkbox of APIs that could be listed in the app. 'Add App'
- 4) The 'client\_id' and 'client\_secret' gets generated from Drupal and stored in the Drupal DB  
These will be used to make the calls from your application to the Experian API.
- 5) The 'New App' should get listed under the respective region under 'MyApps' with information from Drupal DB  
APP Name | client\_id | client\_secrets | Operations (Edit/Delete) | Status (Approved)
- 6) Get an access token from AWS Cognito  
Get your access token using the Client ID and Client Secret from the application created earlier along with your Developer Portal username and password.  
The call to get the Oauth2 token is a POST request with a Content-Type which needs to be specified as JSON; the response will also be in JSON format:

**Request example**

```
curl -X POST https://sandbox-us-api.experian.com/oauth2/v1/token \
-H 'Accept: application/json' \
-H 'Content-type: application/json' \
-H 'Grant_type: password' \
-d '{"username": "<USERNAME>","password": "<PASSWORD>","client_id": "<CLIENT_ID>","client_secret": "<CLIENT_SECRET>"'}
```

**Response example**

```
{
  "issued_at": "1478405901908",
  "expires_in": "1800",
  "token_type": "Bearer",
  "access_token": "eyJraWQiOjBSmpTMXJQQjdJODBHwgybmNsSIZPQkF3V3B3ZTVYbINKZUdSZHdpcEYxliwidHlwIjoiSlDULiwiYWxnIjoiUlMyNTYifQ.eyJzdWliOjmcvKzGllliwiRW1haWwiOjmcvKzXJpYy52YW5kZXJlHN0QGV4cGVyaWFuLmNvbSlslkZpcnN0TmFtZSI6ImZyZWRkaWU
iL
  Cjpc3MiOjFWFBFUklBTlsIkxhc3ROYW1IjoidmFuZGVyIGVscylsmV4cCl6MTUwOTAyNDkxMSwiaWF0ljoxNTA5MDE3NzExLCJqdGkiOi
  I1YTdlYzJhZS00YzdiLTQ3MzktYmU0MS1hMDdjZTBmNTc2N2YifQ.MINbv9JtA9JeBW0cp6mhRZ7xhOvu3o18WDu73xInj14w1fZRX0PaHFR9
  OsCgzGrn-7s46vS2vyScd_MWYfFRWQ8TUGpZ6Gbdh43l_B4UJxu5Uujh1bhrWA1KCsr5p7LKNi6Pxhc76oVd2EAd0l3X7um-d_flds1N4KA
  XmYtXT_oU8DkIKHYiWH6L5Yx3Ue_kYQwXikqU0nXdvab35KyFCza9XqSJEEVEubTSdTvVluzv4AJxN5X-yEtzOtNTV_Yynj4KzdYv8tpuoF2
  LGdzp4G0fOe8mLRNY3g4rCfAnpe0yc1h6Lhh0TMhu2e8jalro4dx7b3VhLIXULm1RqPw",
  "refresh_token": "3lb5SjC6AOUx5R47ffobFFi8DhGIC2GO"
}
```

7) A session\_token is created by Drupal to maintain the user session state

7) Send the access\_token to Okta

8) Send your first request, by using the access\_token as a bearer token in the authorization header and make your first API call as shown below.

**Request example**

```
curl -X GET
https://sandbox-us-api.experian.com/businessinformation/businesses/v1/search' \
-H 'accept: application/json' \
-H 'authorization: Bearer
eyJraWQiOjBSmpTMXJQQjdJODBHwgybmNsSIZPQkF3V3B3ZTVYbINKZUdSZHdpcEYxliwidHlwIjoiSlDULiwiYWxnIjoiUlMyNTYifQ.
eyJzdWliOjmcvKzGllliwiRW1haWwiOjmcvKzXJpYy52YW5kZXJlHN0QGV4cGVyaWFuLmNvbSlslkZpcnN0TmFtZSI6ImZyZWRka
WU
iLCJpc3MiOjFWFBFUklBTlsIkxhc3ROYW1IjoidmFuZGVyIGVscylsmV4cCl6MTUwOTAyNDkxMSwiaWF0ljoxNTA5MDE3NzExLCJqdG
  kiO11YTdlYzJhZS00YzdiLTQ3MzktYmU0MS1hMDdjZTBmNTc2N2YifQ.MINbv9JtA9JeBW0cp6mhRZ7xhOvu3o18WDu73xInj14w1fZRX0
  PaHFR9OsCgzGrn-7s46vS2vyScd_MWYfFRWQ8TUGpZ6Gbdh43l_B4UJxu5Uujh1bhrWA1KCsr5p7LKNi6Pxhc76oVd2EAd0l3X7um-
d_fli
  lds1N4KAXmYtXT_oU8DkIKHYiWH6L5Yx3Ue_kYQwXikqU0nXdvab35KyFCza9XqSJEEVEubTSdTvVluzv4AJxN5X-
yEtzOtNTV_Yynj4Kz
  dYv8tpuoF2LGdzp4G0fOe8mLRNY3g4rCfAnpe0yc1h6Lhh0TMhu2e8jalro4dx7b3VhLIXUL
y
```

**Result:**

1) The B2C user successfully receives the JWT access token and is able to explore creating new Apps and select relevant products

## WF-06: B2B (Business-to-Business) users - Onboarding

**Primary Actor:** Business Users, APICoE Admin

**Maintained By:** APICoE team

**Description:** This workflow represent the complete flow of Busines (B2B) user onboarding w.r.t. Okta and AWS Cognito

**Usage:**

1) This will allow the Business to Business (B2B) users authenticate with Okta, authorize with AWS Cognito, generate client\_id/client\_secrets, and generate & return JWT access\_token.

**Components:**

- 1) S3
- 2) AWS Certificate Manager
- 3) Lambda Functions
- 4) AWS API GW
- 5) Microgateway
- 6) AWS Cognito
- 7) AWS DynamoDB
- 8) Microgateway

**Pre-requisites:**

- 1) B2B user vetting
- 2) Creation of EWACS account
- 3) Creation of B2B user in OKTA based on EWACS user information.
- 4) Infrastructure provisioning for the complete AWS API GW solution

**Steps:**

- 1) Business Unit will raise a ServiceNow request to onboard a new B2B user
- 2) The GSA Support team will then go ahead and create an user profile on OKTA
- 3) Business Unit will request client\_id/client\_secret, token\_url for the developer application from APICoE team
- 4) APICoE team will Launch Cognito Client App via Service catalog
- 5) Create Client App in B2B AWS Cognito user pool for the B2B user as per the user profile details (variables) defined using drupal forms. Client\_id /client\_secret is generated for the Cognito Client app
- 6) APICoE team maps Cognito Client application with respective API GW
- 7) AWS Cognito returns 'Client\_id/client\_secret' to APICoE team
- 8) APICoE team shares client\_id/client\_secret, token\_url with Business Unit
- 9) Business user uses token\_url to invoke Access Token Gateway using 'grant\_type=password', Okta user/password
- 10) Access Token Gateway sends the 'access token request' to Enterprise Gateway
- 11) Enterprise Gateway returns the 'JWT access token' back to the Business user

**Result:**

- 1) The B2B user successfully receives the JWT access token

## **WF-07: B2B (Business-to-Business) users - API Invocation**

**Primary Actor:** Business Users, APICoE Admin

**Maintained By:** APICoE team

**Description:** This workflow represent the complete flow of Busines (B2B) user onboarding w.r.t. Okta and AWS Cognito

**Usage:**

- 1) This will allow the Business to Business (B2B) users to invoke the relevant APIs.

**Components:**

- 1) S3
- 2) AWS Certificate Manager
- 3) Lambda Functions
- 4) AWS API GW
- 5) Microgateway
- 6) AWS Cognito
- 7) AWS DynamoDB
- 8) Microgateway

**Pre-requisites:**

- 1) B2B user onboarding
- 2) APIs published
- 3) Infrastructure provisioning for the complete AWS API GW solution

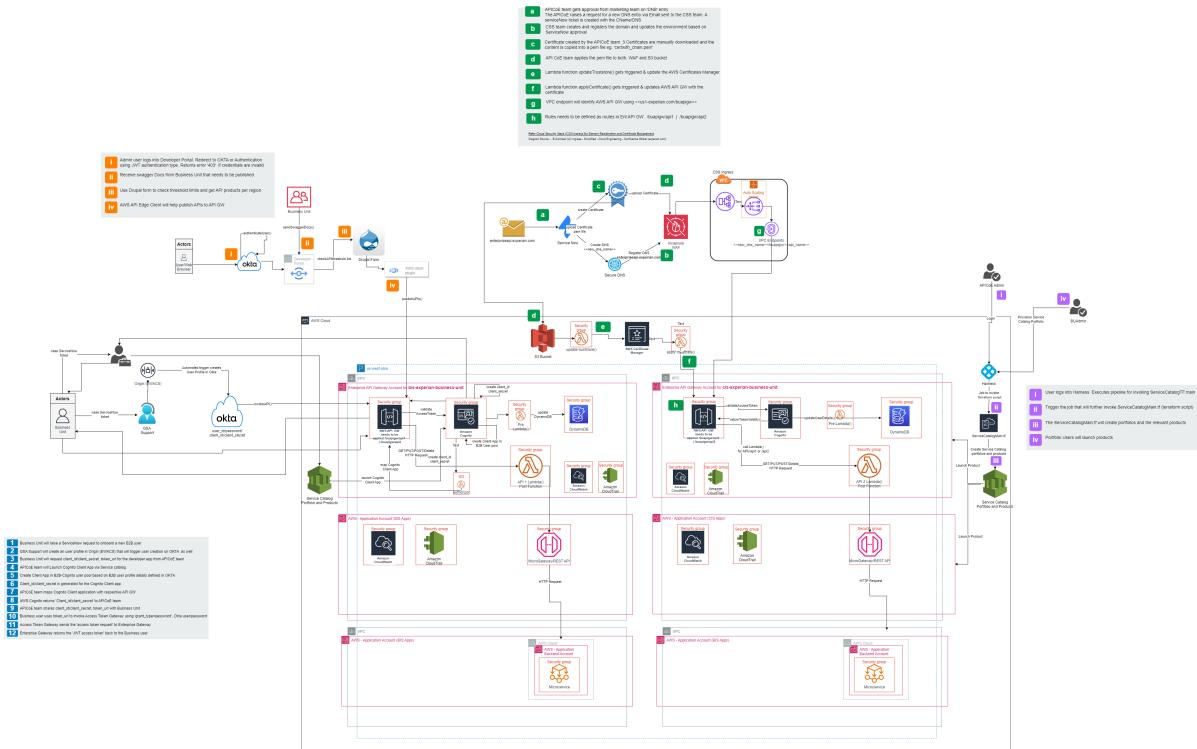
**Steps:**

- 1) Business user accesses the '[enterpriseapi.experian.com](http://enterpriseapi.experian.com)' using JWT access\_token on request header. The request reaches the Enterprise Gateway first.
- 2) Enterprise Gateway will validate the access token and the scope with AWS Cognito.
- 3) AWS Cognito returns token validity back to Enterprise Gateway.
- 4) If the token is invalid then 'Unauthorized message is sent to the user from Enterprise Gateway. If the token is valid, then the request is forwarded to MicroGateway from Enterprise Gateway.
- 5) Microgateway will forward the request to the respective backend API.
- 6) Backend API will return the data with HTTP response.

**Result:**

- 1) The B2B user is successfully able to invoke APIs

## High Level Diagram



Pre-Functions being used (Lambda function names. the names will change and version history will be able to track it)	
Name	Use case
OKTA-COGNITO-PRE-SIGNUP-DYNAMO	When user sign up for the very first time , user is created in AWS Cognito and this lambda function will capture the user details and store it in Dynamo db table
B2B-token-generator	This lambda function validates the user from OKTA , then check same in AWS Cognito and then generate JWT access token
auto-create-cert	This lambda function checks the S3 bucket for .pem and .key file and once that happens , lambda function imports a new certificate in AWS certificate manager
acm-api	This lambda function works on event bridge ,whenever a new certificate is imported in AWS certificate manager, lambda will apply that on respective API gateway
Post-Function being used (Lambda function names. the names will change and version history will be able to track it)	
Name	Use case
multi-rest-api-lambda-node	This lambda is integrated with Enterprise gateway and send request to the respective microgateway based on the routes
auto-reimport-cert	This lambda is integrated with a folder in S3 bucket that will be used for latest .pem and .key file for certification renew. This function will reimport the existing certificate present in ACM with latest files.

## Important documentation links related

- Certificates Management and DNS registration [Certificates Management and DNS Registration](#)
- Okta Integration with WAF and API Gateway [Manual Okta Integration with WAF and API Gateway](#)
- Service Catalog [delete the page](#)
- Authorization/Authentication APIs Test cases scenarios
- Developer Portal Test cases scenarios
- CI/CD
- Terraform name conventions [Terraform Best Practices#NameConvention](#)
- Terraform scripts organization [Terraform Best Practices#StandardStructure](#)
- Bitbucket project link <https://code.experian.local/projects/APDE/repos/eits-enterpriseapigateway-aws/browse>