Microsoft Dynamics Banking Accelerator-BIAN API

Add section at top on

1. Scope

Two APIs from the BIAN API Catalogue were implemented, with the goal to demonstrate how to make BIAN compliant calls to the Dynamics 365 using the Banking Accelerator and the BIAN API.:

1. ***Collateral Asset Administration (V1)*** [https://portal.bian.org/bian/api-console?id=35]
2. ***Consumer Loan (V1)*** [https://portal.bian.org/bian/api-profile?apiId=32]

The Banking Accelerator doesn't include to all the domains covered in the BIAN Service Landscape, and so the choice of the APIs to be implemented was based on two of the top use cases voices by our banking customers.

2. Setup

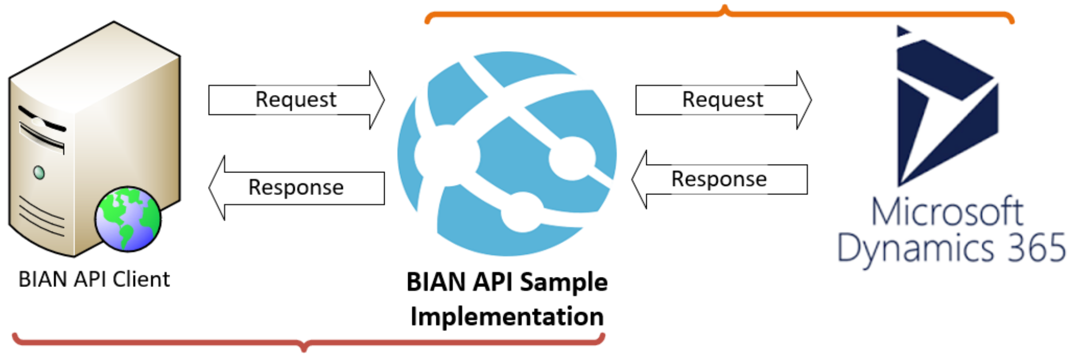
2.1 Pre-requisites

1. Obtain the sample code from GitHub
    a. https://github.com/microsoft/Industry-Accelerator-FinancialServices/tree/master/apps/samplecode.
2. Installation of the Microsoft Dynamics Banking Accelerator
3. Azure subscription.
4. Registered Sign-in for BIAN
5. A general understanding of .NET and OAuth

2.2 Connections

The key element of this setup is to configure the connection between the BIAN API Sample and the Microsoft Dynamics 365 instance that contains the Banking Accelerator.

Additionally, to that, a configuration is needed to allow clients to consume the BIN API end points.



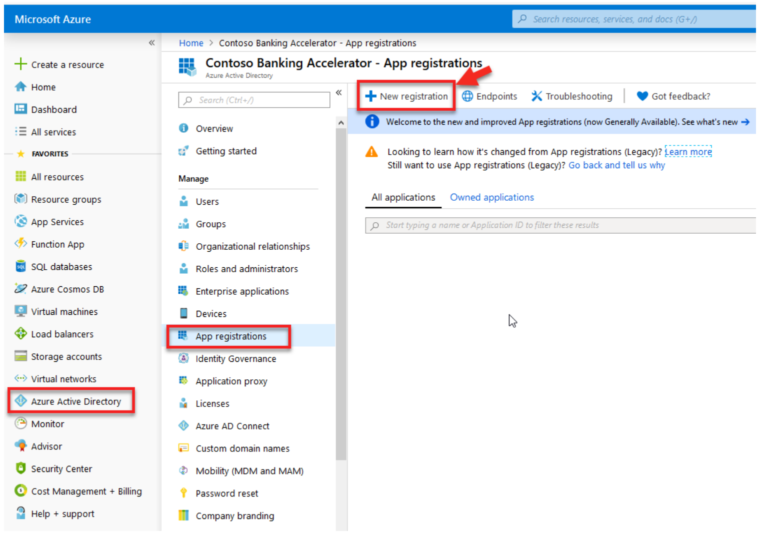
The next sections describe how to setup these connections and uses Postman to show case a client connection to the BIAN API.

2.3 Connection between the BIAN API Sample Implementation & Microsoft Dynamics Banking Accelerator

The very first step is to make sure that the BIAN API can connect to the Banking Accelerator instance. For that, it is required to register the application with the Microsoft Identify Platform.

The steps below displays a simple application registration, but additional information on how to register an application can be found here: Quickstart: Register an application with the Microsoft identity platform

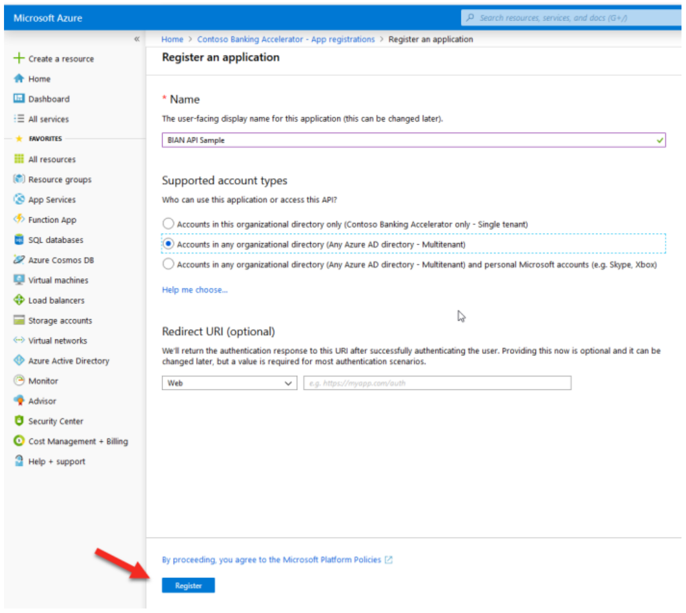
1 – Sign in to the Azure Portal (https://portal.azure.com), navigate to “Azure Active Directory / App registrations”, and click “New Registration”. Make sure you are logged on the same tenant where the Banking Accelerator is installed.



Welcome to the new and improved App registrations (now Generally Available). See what's new →

Looking to learn how it's changed from App registrations (Legacy)? Learn more
Still want to use App registrations (Legacy)? Go back and tell us why

2 – Enter the name of the application, select the types of accounts that can access the application, and click “Register”



Register an application

* Name

The user-facing display name for this application (this can be changed later).

BIAN API Sample

Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Contoso Banking Accelerator only - Single tenant)
- [✓ Accounts in any organizational directory (Any Azure AD directory - Multitenant)]
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

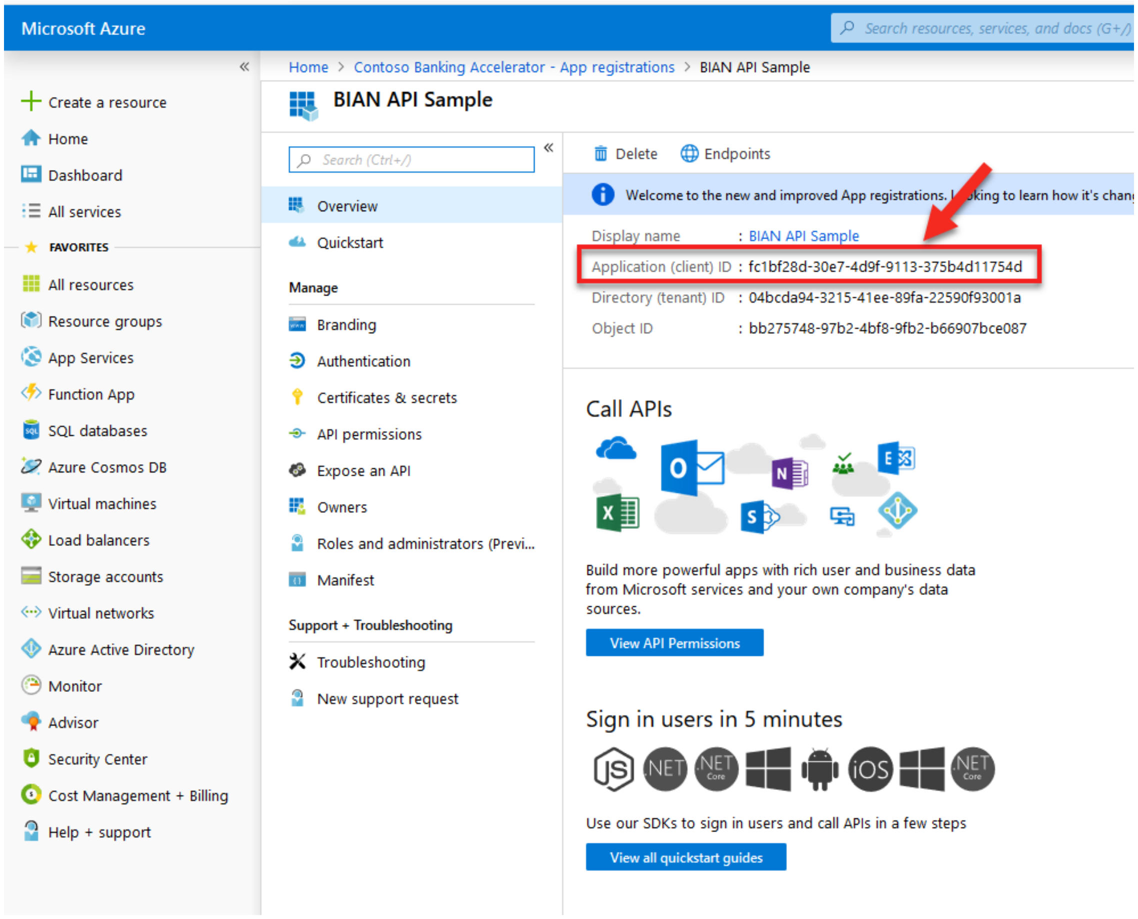
Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web | e.g. https://myapp.com/auth

By proceeding, you agree to the Microsoft Platform Policies

3 – Copy the Application/Client ID, for later use:



Welcome to the new and improved App registrations. Looking to learn how it's chan

Display name	: BIAN API Sample
Application (client) ID	: fc1bf28d-30e7-4d9f-9113-375b4d11754d
Directory (tenant) ID	: 04bcda94-3215-41ee-89fa-22590f93001a
Object ID	: bb275748-97b2-4bf8-9fb2-b66907bce087

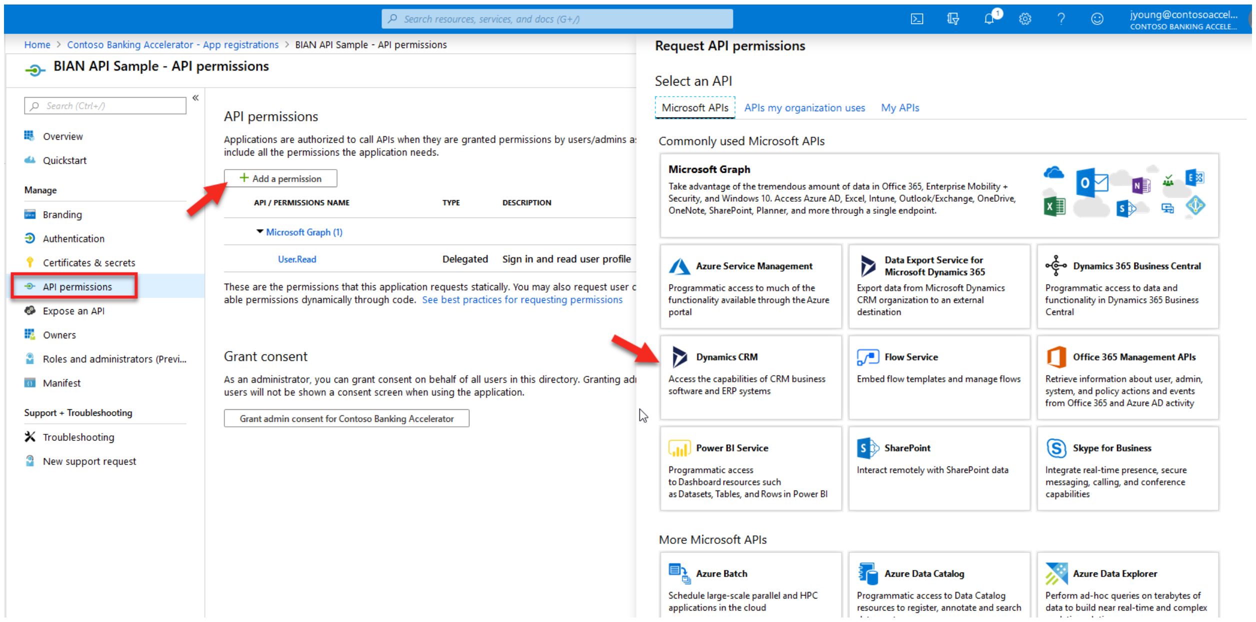
Call APIs

Build more powerful apps with rich user and business data from Microsoft services and your own company's data sources.

Sign in users in 5 minutes

Use our SDKs to sign in users and call APIs in a few steps

4 – Next, we need to add the right permission to this newly registered application. Navigate to “API permissions”, click “Add a permission”, and select “Dynamics CRM”


Search resources, services, and docs (G+/)
jyoung@contosoaccel...
CONTOSO BANKING ACCELE...
Home > Contoso Banking Accelerator - App registrations > BIAN API Sample - API permissions
BIAN API Sample - API permissions
Search (Ctrl+/)
Overview
Quickstart
Manage
Branding
Authentication
Certificates & secrets
API permissions
Expose an API
Owners
Roles and administrators (Previ...
Manifest
Support + Troubleshooting
Troubleshooting
New support request
Add a permission
Grant admin consent for Contoso Banking Accelerator
Microsoft APIs
APIs my organization uses
My APIs

API permissions

Applications are authorized to call APIs when they are granted permissions by users/admins a
include all the permissions the application needs.

API / PERMISSIONS NAME	TYPE	DESCRIPTION
Microsoft Graph (1)		
User.Read	Delegated	Sign in and read user profile

These are the permissions that this application requests statically. You may also request user c
able permissions dynamically through code. See best practices for requesting permissions

Grant consent

As an administrator, you can grant consent on behalf of all users in this directory. Granting adı
users will not be shown a consent screen when using the application.

Request API permissions

Select an API

Commonly used Microsoft APIs

Microsoft Graph
Take advantage of the tremendous amount of data in Office 365, Enterprise Mobility + Security, and Windows 10. Access Azure AD, Excel, Intune, Outlook/Exchange, OneDrive, OneNote, SharePoint, Planner, and more through a single endpoint.

Azure Service Management
Programmatic access to much of the functionality available through the Azure portal

Data Export Service for Microsoft Dynamics 365
Export data from Microsoft Dynamics CRM organization to an external destination

Dynamics 365 Business Central
Programmatic access to data and functionality in Dynamics 365 Business Central

Dynamics CRM
Access the capabilities of CRM business software and ERP systems

Flow Service
Embed flow templates and manage flows

Office 365 Management APIs
Retrieve information about user, admin, system, and policy actions and events from Office 365 and Azure AD activity

Power BI Service
Programmatic access to Dashboard resources such as Datasets, Tables, and Rows in Power BI

SharePoint
Interact remotely with SharePoint data

Skype for Business
Integrate real-time presence, secure messaging, calling, and conference capabilities

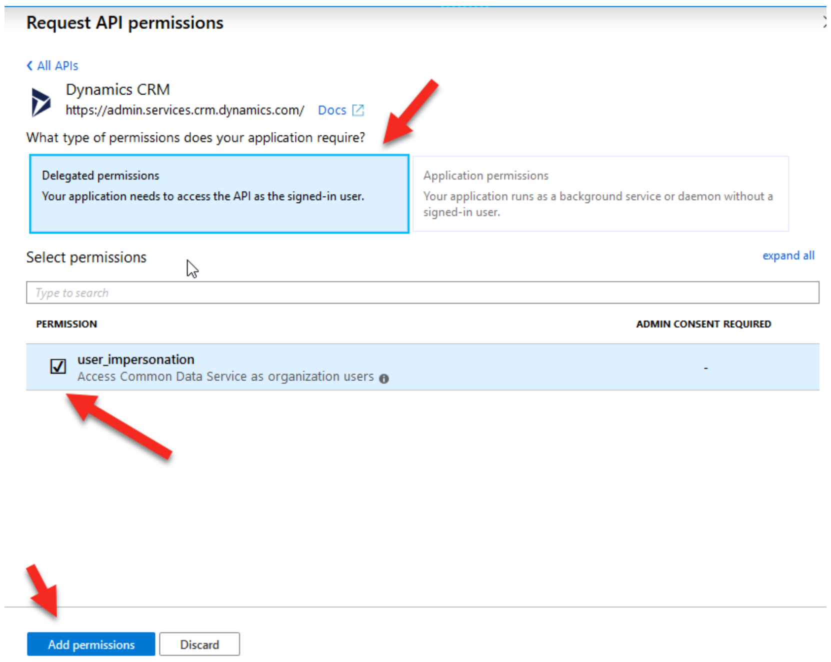
More Microsoft APIs

Azure Batch
Schedule large-scale parallel and HPC applications in the cloud

Azure Data Catalog
Programmatic access to Data Catalog resources to register, annotate and search

Azure Data Explorer
Perform ad-hoc queries on terabytes of data to build near real-time and complex

5 - Select “Delegated permissions”, check “user_impersonation”, and click “Add permissions”



Request API permissions

All APIs

Dynamics CRM
https://admin.services.crm.dynamics.com/ Docs

What type of permissions does your application require?

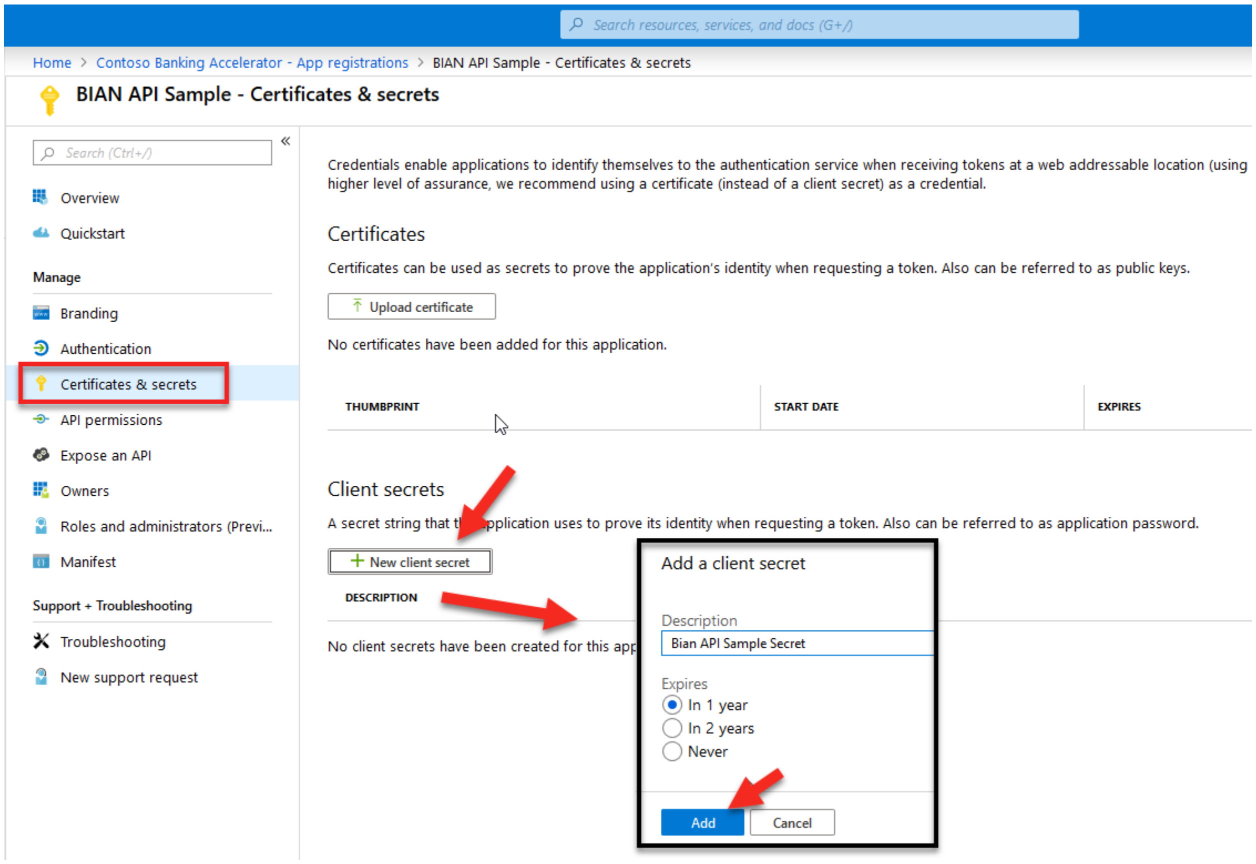
Delegated permissions
Your application needs to access the API as the signed-in user.

Application permissions
Your application runs as a background service or daemon without a signed-in user.

Select permissions

PERMISSION	ADMIN CONSENT REQUIRED
☑ user_impersonation Access Common Data Service as organization users	-

6 – Next, navigate to “Certificates & secrets”, click “New client secret”, enter a description, select expiration time, and click “Add”



Home > Contoso Banking Accelerator - App registrations > BIAN API Sample - Certificates & secrets

BIAN API Sample - Certificates & secrets

Credentials enable applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Certificates

Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

No certificates have been added for this application.

THUMBPRINT	START DATE	EXPIRES

Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

DESCRIPTION

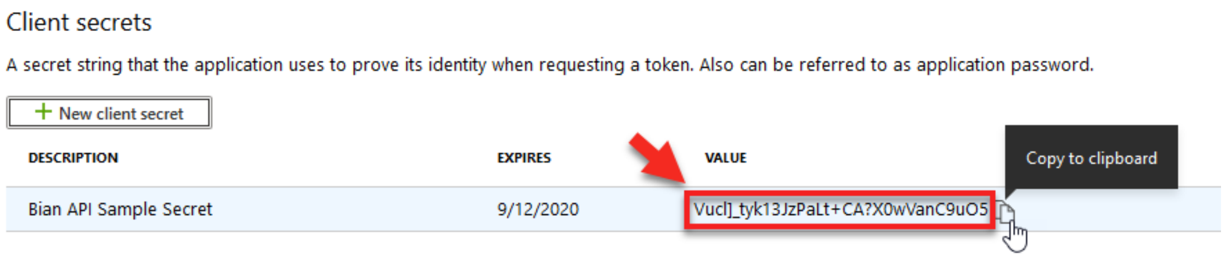
No client secrets have been created for this app

Add a client secret

Description: Bian API Sample Secret

Expires: [✓ In 1 year] In 2 years Never

7 – Copy the newly generated client secret, for later use

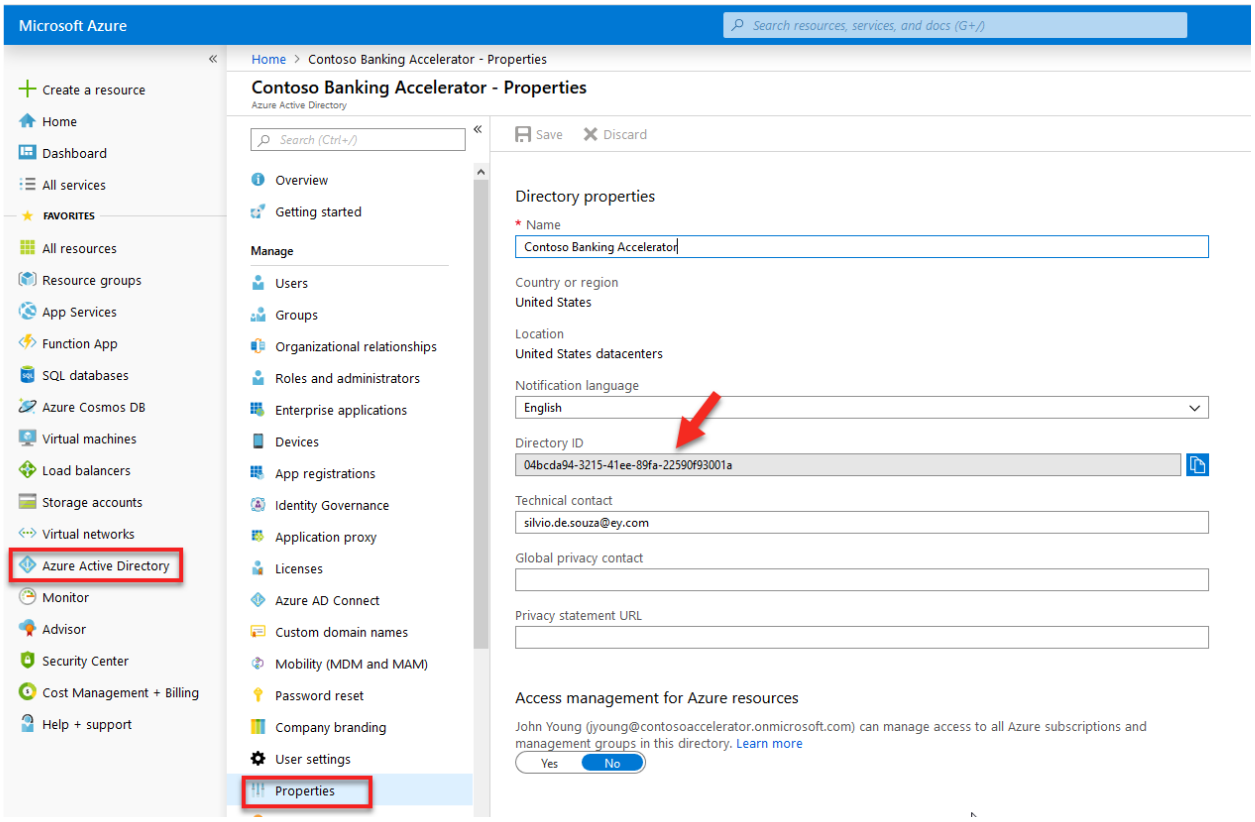


Client secrets

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

DESCRIPTION	EXPIRES	VALUE
Bian API Sample Secret	9/12/2020	Vucl]_tyk13JzPaLt+CA?X0wVanC9uO5

8 – Navigate to "Azure Active Directory/Properties" and copy the "Directory ID" (aka "Tenant ID"), for later use.



Contoso Banking Accelerator - Properties

Azure Active Directory

Directory properties

* Name: Contoso Banking Accelerator

Country or region: United States

Location: United States datacenters

Notification language: English

Directory ID: 04bcda94-3215-41ee-89fa-22590f93001a

Technical contact: silvio.de.souza@ey.com

Global privacy contact:

Privacy statement URL:

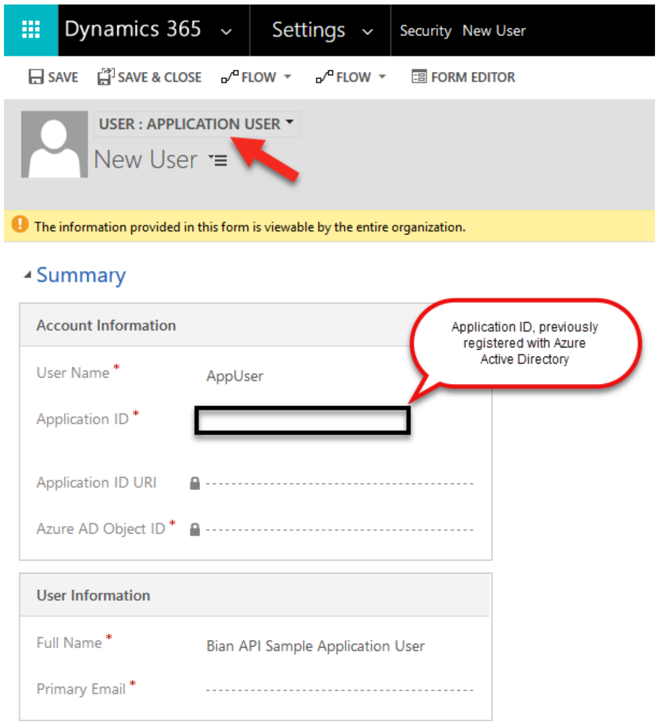
Access management for Azure resources

John Young (jyoung@contosoaccelerator.onmicrosoft.com) can manage access to all Azure subscriptions and management groups in this directory. Learn more

Yes [✓ No]

Additionally, to registering the application with Azure, an application user must be added to Dynamics.

9 – Log on to Dynamics and navigate to “Settings / Security / Users” to add a new user. Make sure that the user type has been set to “Application User”. Enter the required fields, and for “Application ID” enter the value copied in step #3.



The information provided in this form is viewable by the entire organization.

Summary

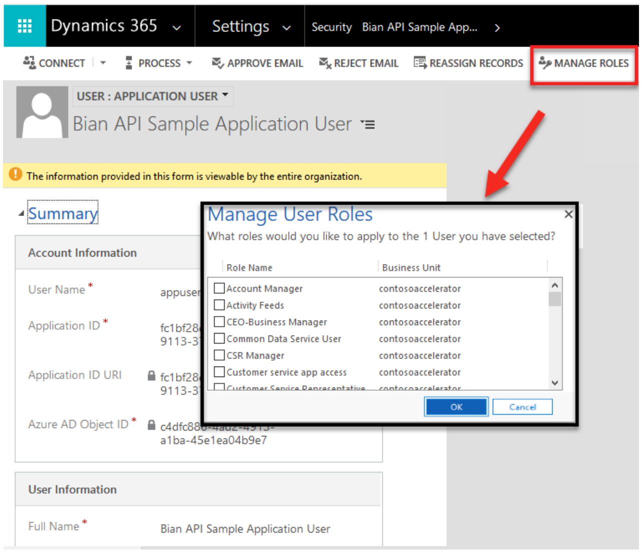
Account Information

User Name *	AppUser
Application ID *	
Application ID URI	
Azure AD Object ID *	

User Information

Full Name *	Bian API Sample Application User
Primary Email *	

10 – Next, this user must be assigned a security role. The suggested approach is the creation of a custom security role.



The information provided in this form is viewable by the entire organization.

Summary

Account Information

User Name *	appuser
Application ID *	fc1bf28 9113-3
Application ID URI	fc1bf28 9113-3
Azure AD Object ID *	c4dfc88 a1ba-45e1ea04b9e7

User Information

Full Name *	Bian API Sample Application User

Manage User Roles

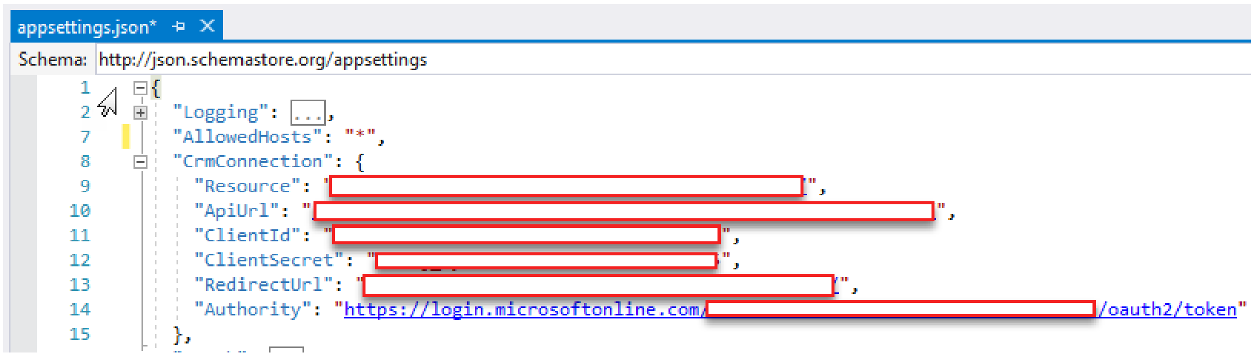
What roles would you like to apply to the 1 User you have selected?

Role Name	Business Unit
☐ Account Manager	contosoaccelerator
☐ Activity Feeds	contosoaccelerator
☐ CEO-Business Manager	contosoaccelerator
☐ Common Data Service User	contosoaccelerator
☐ CSR Manager	contosoaccelerator
☐ Customer service app access	contosoaccelerator
☐ Customer Service Representative	contosoaccelerator

For more details on how to create user & security roles, visit [Use Multi-Tenant Server-to-server authentication]()

11 – Last step in setting up the connection between the BIAN Sample API & Microsoft Dynamics Banking Accelerator, is to update the section “CrmConnection”, in the “appsettings.json” file, in the Visual Studio solution.

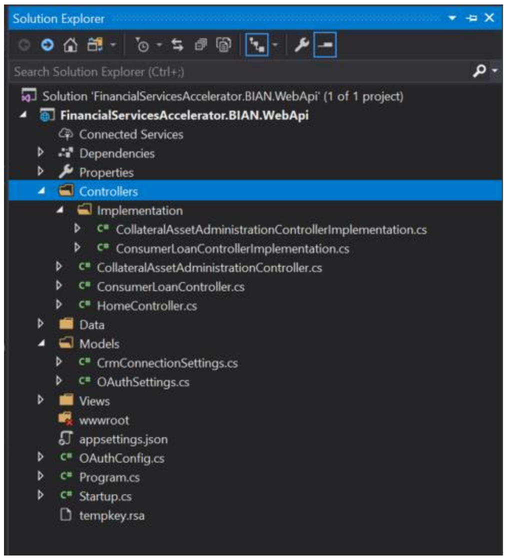
a. **Resource**: Root URL of your Dynamics instance (ex. https://demo.crm.dynamics.com/)
b. **ApiUrl**: URL to the WebAPI endpoint of your Dynamics instance (ex. https://demo.api.crm.dynamics.com/api/data/v9.0/)
c. **ClientId**: The application ID registered with Azure (see step #3).
d. **ClientSecret**: The application secret, generated during the application registration with Azure (see step #7)
e. **RedirectUrl**: Redirect URL from your Azure AD app
f. **Authority**: https://login.microsoftonline.com/**{Your Azure Tenant ID}**/oauth2/token (see step #8)



```
{
  "Logging": {...},
  "AllowedHosts": "*",
  "CrmConnection": {
    "Resource": "[redacted]",
    "ApiUrl": "[redacted]",
    "ClientId": "[redacted]",
    "ClientSecret": "[redacted]",
    "RedirectUrl": "[redacted]",
    "Authority": "https://login.microsoftonline.com/[redacted]/oauth2/token"
  },
```

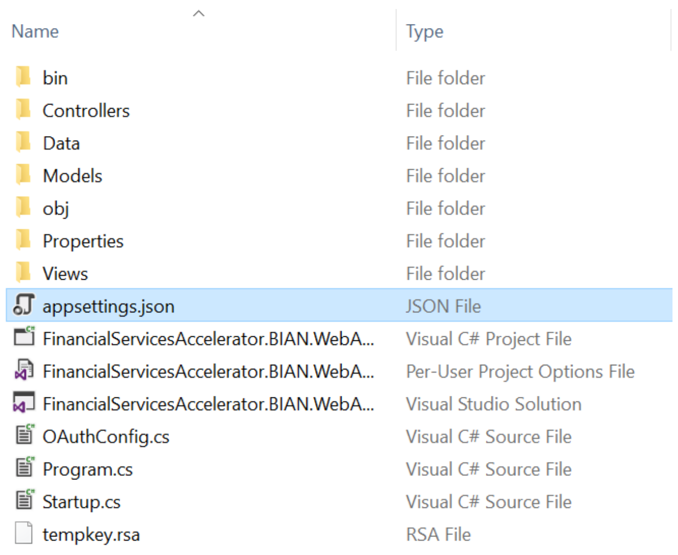
2.4 Connection to the BIAN API Sample Implementation

In GitHub navigate to the BIAN sample package in the apps/samplecode folder and open the FinancialServicesAccelerator.BIAN.WebApi.csproj visual studio Project File. You’ll find the Visual Studio project you’ll find the two controllers of Collateral Asset Administration and Consumer Loan under Controllers → Implementation.



```
Solution 'FinancialServicesAccelerator.BIAN.WebApi' (1 of 1 project)
  FinancialServicesAccelerator.BIAN.WebApi
    Connected Services
    Dependencies
    Properties
    Controllers
      Implementation
        CollateralAssetAdministrationControllerImplementation.cs
        ConsumerLoanControllerImplementation.cs
      CollateralAssetAdministrationController.cs
      ConsumerLoanController.cs
      HomeController.cs
    Data
    Models
      CrmConnectionSettings.cs
      OAuthSettings.cs
    Views
    wwwroot
    appsettings.json
    OAuthConfig.cs
    Program.cs
    Startup.cs
    tempkey.rsa
```

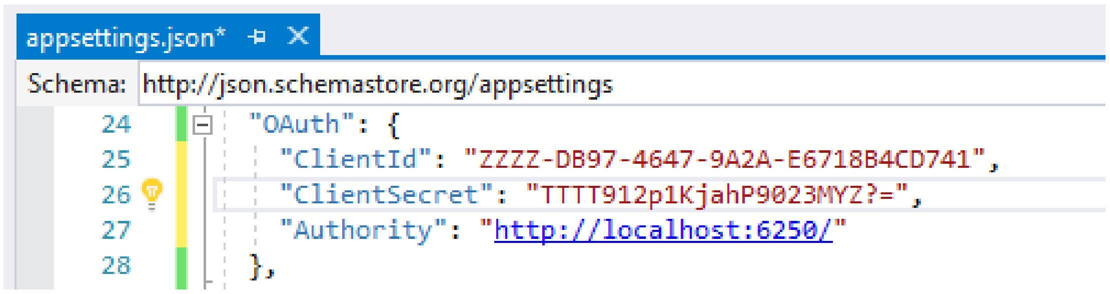
The API implementation makes use of the IdentityServer framework for authentication, and the next steps showcase how to consume a local instance of the API (under Visual Studio IIS express), using Postman as the client.



Name	Type
bin	File folder
Controllers	File folder
Data	File folder
Models	File folder
obj	File folder
Properties	File folder
Views	File folder
appsettings.json	JSON File
FinancialServicesAccelerator.BIAN.WebA...	Visual C# Project File
FinancialServicesAccelerator.BIAN.WebA...	Per-User Project Options File
FinancialServicesAccelerator.BIAN.WebA...	Visual Studio Solution
OAuthConfig.cs	Visual C# Source File
Program.cs	Visual C# Source File
Startup.cs	Visual C# Source File
tempkey.rsa	RSA File

1 – First step is to edit the section "OAuth" in the "appsettings.json" file, in the Visual Studio solution.

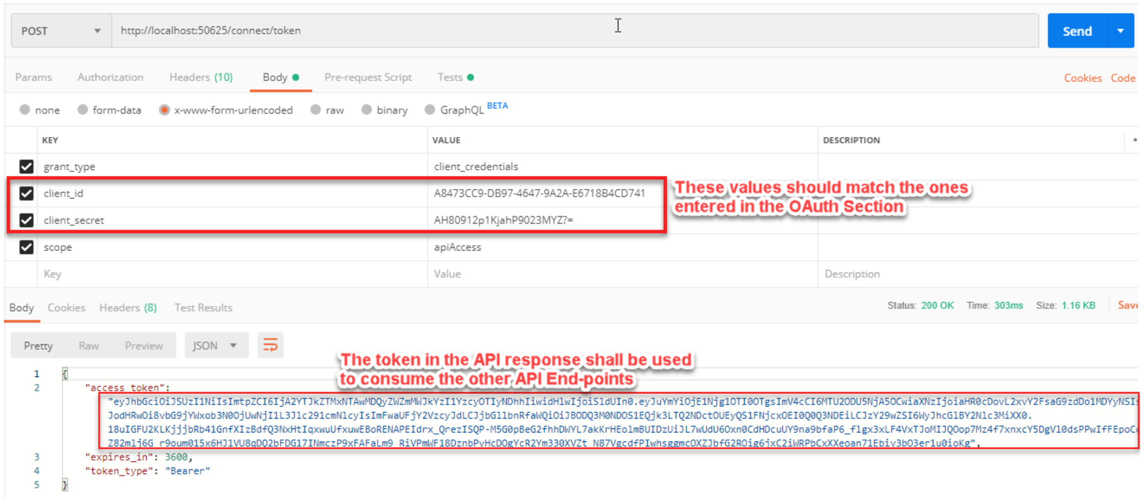
a. **ClientId:** ID of your choosing for connecting to the hosted API
b. **ClientSecret:** Key/secret of your choosing for connecting to the hosted API
c. **Authority:** URL of server that authenticates the client request.



```
"OAuth": {
  "ClientId": "ZZZZ-DB97-4647-9A2A-E6718B4CD741",
  "ClientSecret": "TTTT912p1KjahP9023MYZ?=",
  "Authority": "http://localhost:6250/"
},
```

2 – Start an instance of the BIAN API

3 – Using Postman, the first step before consuming any of the API methods, is the authentication request. If successful, it will return a token, that shall be included in API requests.



http://localhost:50625/connect/token

	KEY	VALUE	DESCRIPTION
✓	grant_type	client_credentials	
✓	client_id	A8473CC9-DB97-4647-9A2A-E6718B4CD741	
✓	client_secret	AH80912p1KjahP9023MYZ?=	
✓	scope	apiAccess	
	Key	Value	Description

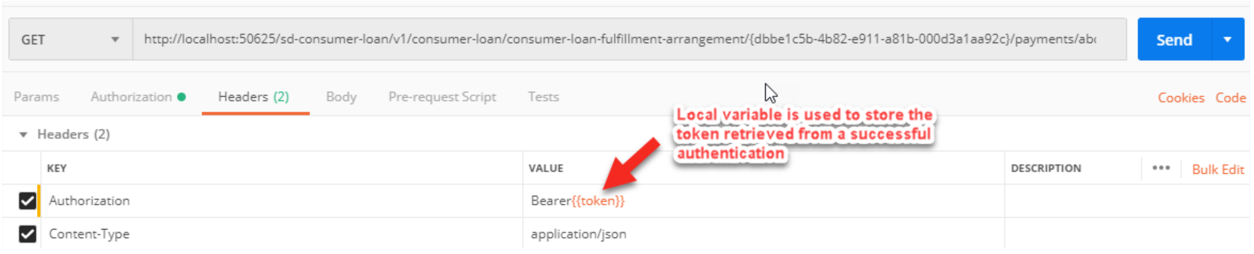
These values should match the ones entered in the OAuth Section

Status: 200 OK Time: 303ms Size: 1.16 KB

The token in the API response shall be used to consume the other API End-points

```
{
    "access_token":
      "eyJhbGciOiJSUzI1NiIsImtpZCI6IjA2YTJkZTMxNTAwMDQyZWZmMWJkYzI1YzcyOTIyNDhhIiwidHlwIjoiSldUIn0.eyJuYmYiOjE1Njg1OTI0OTgsImV4cCI6MTU2ODU5NjA5OCwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdDo1MDYyNSI
      JodHRwOi8vbG9jYWxob3N0OjUwNjI1L3Jlc291cmNlcyIsImFwaUFjY2VzcyJdLCJjbGllbnRfaWQiOiJBODQ3M0NDOS1EQjk3LTQ2NDctOUEyQS1FNjcxOEI0Q0Q3NDEiLCJzY29wZSI6WyJhcGlBY2Nlc3MiXX0.
      18uIGFU2KLKjjjbRb41GnfXIzBdfQ3NxHtIqxwuUfxuwEBoRENAPEIdrx_QrezISQP-M5G0pBeG2fhhDWYL7akKrHEolmBUIDzUiJL7wUdU6Oxn0CdHDcuUY9na9bfaP6_flgx3xLF4VxTJoMIJQOop7Mz4f7xnxcY5DgV10dsPPwIfFEpoC
      Z82mlj6G_r9oum015x6HJ1VU8qDQ2bFDG17INmczP9xFAFaLm9_RiVPmWF18DznbPvHcDOgYcR2Ym330XVZt_N87VgcdfPIwhsggmcOXZJbfG2ROig6jxC2iWRPbCxXXeoan71Ebiv3bO3er1u0ioKg",
    "expires_in": 3600,
    "token_type": "Bearer"
}
```

4. With a valid token, Postman is ready to consume the API, by including it in the API calls. Using Postman variables, the token can be stored in a local variable and referenced in the API requests.



http://localhost:50625/sd-consumer-loan/v1/consumer-loan/consumer-loan-fulfillment-arrangement/{dbbe1c5b-4b82-e911-a81b-000d3a1aa92c}/payments/ab

Local variable is used to store the token retrieved from a successful authentication

	KEY	VALUE	DESCRIPTION
✓	Authorization	Bearer{{token}}	
✓	Content-Type	application/json	

3. Deployment

The suggested approach to deploy the BIAN API is to publish it to Azure as a Web App. For that, create a new .NET CORE AppService in Azure, then copy the site URL generated by Azure and populated it in step 1C of section 2.4 .

For additional information on how to deploy the solution to Azure, visit [Publish a Web app to Azure App Service using Visual Studio]

4. Understand the code

4.1 Introduction

This web application is based on an ASP.NET Core Web project template.

4.2 Authorization

The "Startup.cs" file configures the application to properly authenticate the calls to Dynamics, by using the middleware "OAuthConfig.cs", and the settings retrieved from appsettings.json.

4.3 CDM

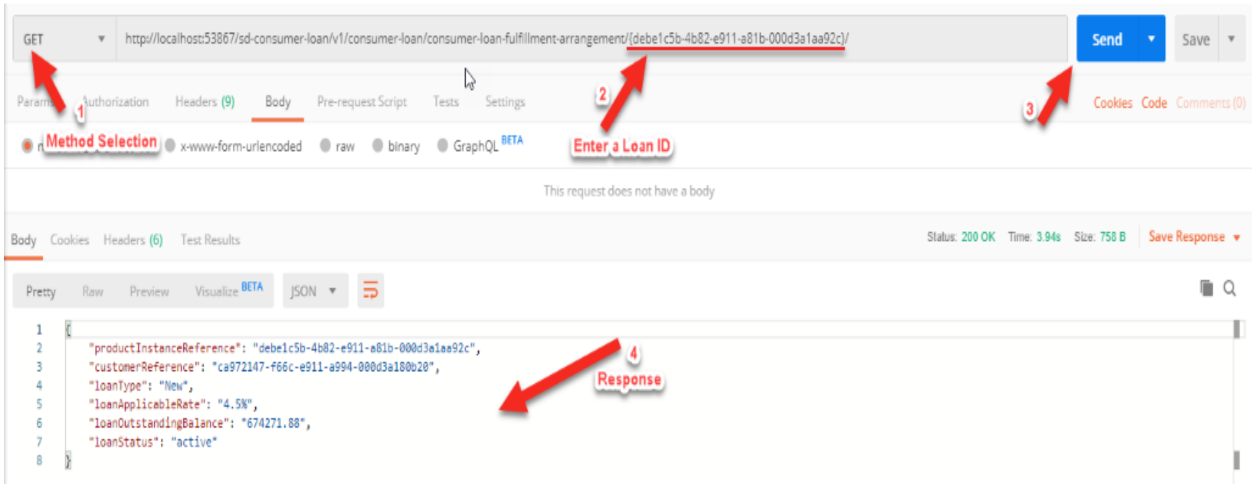
The class "Data\CdsWebApi.cs" is the middleware that connects to the Dynamics Web API and implements the operations with the Banking Accelerator CDM.

4.4 Result/how to use the sample implementation

With the loaded Visual Studio project from step 2.4 you can find an example BIAN compliant call in the ConsumerLoanControllerImplementation.cs file. One of the ***Consumer Loan (V1)*** API endpoint is *consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id},* which is used to retrieve a loan.

```
public Task<ConsumerLoanFulfillmentArrangementResponse>
RetrieveConsumerLoanFulfillmentArrangementWithBQsAsync(string cr_reference_id)
        {
            return Task.Run((Func<ConsumerLoanFulfillmentArrangementResponse>)(() =>
            {
                var loanId = ParseGuid(cr_reference_id, "cr_reference_id");

                var loan = _cdsWebApi.Retrieve("msfsi_financialproducts", loanId,
"msfsi_loantype",
                    "_msfsi_customerid_value", "msfsi_outstandingtotalamount",
"msfsi_loanmaturitydate",
                    "msfsi_loanstartdate", "msfsi_interestrate", "statecode");

                var response = new ConsumerLoanFulfillmentArrangementResponse()
                {
                    LoanType =
loan.Attributes.ContainsKey("msfsi_loantype@odata.community.display.v1.formattedvalue"
)
                               ?
loan.Attributes["msfsi_loantype@odata.community.display.v1.formattedvalue"]?.ToString(
)
                               : "",
                    CustomerReference =
loan.Attributes["_msfsi_customerid_value"]?.ToString(),
                    LoanOutstandingBalance =
loan.Attributes["msfsi_outstandingtotalamount"]?.ToString(),
                    LoanMaturityDate =
FormatDateString(loan.Attributes["msfsi_loanmaturitydate"]?.ToString()),
                    LoanOriginationDate =
FormatDateString(loan.Attributes["msfsi_loanstartdate"]?.ToString()),
                    LoanStatus = loan.Attributes["statecode"]?.ToString() == "0"
                                ? "active"
                                : "inactive",
                    ProductInstanceReference = loanId.ToString(),
                };

                var unformattedInterestRate =
loan.Attributes["msfsi_interestrate"]?.ToString();
                if (string.IsNullOrEmpty(unformattedInterestRate) == false)
                {
                    response.LoanApplicableRate = $"{unformattedInterestRate}%";
                }

                return response;
            }));
           }
```

In the Banking Accelerator CDM, a BIAN loan is mapped to a financial product (entity "msfsi_financialproducts"). To retrieve a specific loan, just replace *{cr_reference_id}* with a valid finance product ID.



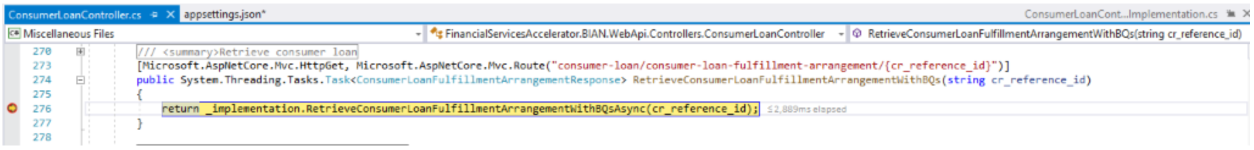
```
http://localhost:53867/sd-consumer-loan/v1/consumer-loan/consumer-loan-fulfillment-arrangement/{debe1c5b-4b82-e911-a81b-000d3a1aa92c}/
```

This request does not have a body

Status: 200 OK Time: 3.94s Size: 758 B

```
{
    "productInstanceReference": "debe1c5b-4b82-e911-a81b-000d3a1aa92c",
    "customerReference": "ca972147-f66c-e911-a994-000d3a180b20",
    "loanType": "New",
    "loanApplicableRate": "4.5%",
    "loanOutstandingBalance": "674271.88",
    "loanStatus": "active"
}
```

Optionally, in the running instance of Visual Studio – if in debug mode – a breakpoint may be set at the implementation of this endpoint. This may be helpful to better understand some of the mapping implementations.


ConsumerLoanController.cs
appsettings.json*
ConsumerLoanCont...Implementation.cs
Miscellaneous Files
FinancialServicesAccelerator.BIAN.WebApi.Controllers.ConsumerLoanController
RetrieveConsumerLoanFulfillmentArrangementWithBQs(string cr_reference_id)
≤2,889ms elapsed

```
/// <summary>Retrieve consumer loan
[Microsoft.AspNetCore.Mvc.HttpGet, Microsoft.AspNetCore.Mvc.Route("consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}")]
public System.Threading.Tasks.Task<ConsumerLoanFulfillmentArrangementResponse> RetrieveConsumerLoanFulfillmentArrangementWithBQs(string cr_reference_id)
{
    return _implementation.RetrieveConsumerLoanFulfillmentArrangementWithBQsAsync(cr_reference_id);
}
```

4.5 BIAN API Endpoints Implementation

Collateral Asset Administration

The class *CollateralAssetAdministrationController* (Controllers\CollateralAssetAdministrationController.cs) contains the methods associated with the API endpoints (mappings below), while the class *CollateralAssetAdministrationControllerImplementation* (Controllers\Implementation\CollateralAssetAdministrationControllerImplementation.cs) contains the concrete implementation of these methods.

Action	BIAN Endpoint	Controller Method
Put	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/updation	UpdateCollateralAssetAdministrativePlan
Post	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/recording	RecordCollateralAssetAdministrativePlan
Post	collateral-asset-administration/collateral-asset-administrative-plan/requisition	RequestCollateralAssetAdministrativePlanCreate

Put	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/requisition	RequestCollateralAssetAdministrativePlanValuationCreate
Put	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/valuations/{bq_reference_id}/requisition	RequestCollateralAssetAdministrativePlanValuationUpdate
Put	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/captures/requisition	RequestCollateralAssetAdministrativePlanCaptureCreate
Post	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/captures/{bq_reference_id}/requisition	RequestCollateralAssetAdministrativePlanCaptureUpdate
Put	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/captures/{bq_reference_id}/requisition	RequestCollateralAssetAdministrativePlanCaptureUpdate
Get	collateral-asset-administration/collateral-asset-administrative-plan/	RetrieveCollateralAssetAdministrationReferenceIds
Get	collateral-asset-administration/collateral-asset-administrative-plan/behavior-qualifiers/	RetrieveCollateralAssetAdministrationBehaviorQualifiers
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/{behavior-qualifier}/	RetrieveBehaviorQualifierReferenceIds
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/	RetrieveCollateralAssetAdministrativePlan
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/captures/{bq_reference_id}	RetrieveCollateralAssetAdministrativePlanCapture
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/valuations/{bq_reference_id}	RetrieveCollateralAssetAdministrativePlanValuation
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/maintenances/{bq_reference_id}	RetrieveCollateralAssetAdministrativePlanMaintenance
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/updates/{bq_reference_id}	RetrieveCollateralAssetAdministrativePlanUpdate
Get	collateral-asset-administration/collateral-asset-administrative-plan/{cr_reference_id}/reportings/{bq_reference_id}	RetrieveCollateralAssetAdministrativePlanReporting

Consumer Loan

The class *ConsumerLoanController* (Controllers\ConsumerLoanController.cs) contains the methods associated with the API endpoints (mappings below), while the class *ConsumerLoanControllerImplementation* (Controllers\Implementation\ConsumerLoanControllerImplementation.cs) contains the concrete implementation of these methods.

Action	BIAN End-Point	Controller
Post	consumer-loan/consumer-loan-fulfillment-arrangement/initiation	InitiateConsumerLoanFulfillmentArrangement
Post	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/recording	RecordConsumerLoanFulfillmentArrangement
Put	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/disbursements/{bq_reference_id}/execution"	ExecuteConsumerLoanFulfillmentArrangementDisbursementUpdate
Post	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/disbursements/execution	ExecuteConsumerLoanFulfillmentArrangementDisbursementCreate
Put	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/payments/{bq_reference_id}/execution	ExecuteConsumerLoanFulfillmentArrangementRepaymentUpdate
Post	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/payments/execution	ExecuteConsumerLoanFulfillmentArrangementRepaymentCreate
Put	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/withdrawals/{bq_reference_id}/execution	ExecuteConsumerLoanFulfillmentArrangementWithdrawalUpdate
Post	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/withdrawals/execution	ExecuteConsumerLoanFulfillmentArrangementWithdrawalCreate
Put	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/restructurings/{bq_reference_id}/requisition	RequestConsumerLoanFulfillmentArrangementRestructuringUpdate
Post	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/restructurings/requisition	RequestConsumerLoanFulfillmentArrangementRestructuringCreate
Get	consumer-loan/consumer-loan-fulfillment-arrangement/	RetrieveConsumerLoanReferenceIds
Get	consumer-loan/consumer-loan-fulfillment-arrangement/behavior-qualifiers/	RetrieveConsumerLoanBehaviorQualifiers
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/{behavior_qualifier}/	RetrieveBehaviorQualifierReferenceIds
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}	RetrieveConsumerLoanFulfillmentArrangementWithBQs
Delete	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}	TerminateConsumerLoanFulfillmentArrangement
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/disbursements/{bq_reference_id}	Retrieve_Disbursement
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/maintenances/{bq_reference_id}	Retrieve_Maintenance
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/withdrawals/{bq_reference_id}	Retrieve_Withdrawal

Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/payments/{bq_reference_id}	Retrieve_Payment
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/restructurings/{bq_reference_id}	Retrieve_Restructuring
Get	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/statements/{bq_reference_id}	Retrieve_Statements
Put	consumer-loan/consumer-loan-fulfillment-arrangement/{cr_reference_id}/updation	UpdateConsumerLoanFulfillmentArrangement