

# **Study Definition Repository (SDR) Reference Implementation**

## **Infra Migration Guide (Release 0.5 to Release 2.0) Version 1.1**

### **Disclaimer**

These materials and information, as well as the underlying code/application they relate to, are provided by TransCelerate Biopharma Inc. AS IS. Any party using or relying on this information, these materials, and/or the underlying code/application do so entirely at their own risk. Neither TransCelerate nor its members will bear any responsibility or liability for any harm, including indirect or consequential harm, that a user may incur from use or misuse of this information, materials, or underlying code/application.

TransCelerate does not endorse any particular software, system, or service. And the use of specific brands of products or services by TransCelerate and its collaboration partners in developing the SDR Reference Implementation should not be viewed as any endorsement of such products or services. To the extent that the SDR Reference Implementation incorporates or relies on any specific branded products or services, this resulted out of the practical necessities associated with making a reference implementation available to demonstrate the SDR's capabilities.

## Document History

Version No.	Date	Author	Revision Description
V1.0	20-Mar-2023	ACN	Initial Draft
V1.1	27-Apr-2023	ACN	Removed the configuration of Auth token endpoint in SDR RI (endpoint deprecated) – section 3.2.2

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>6</b>
1.1	Overview.....	6
1.2	Scope of Document .....	6
1.3	Intended Audience.....	6
1.4	Definitions and Acronyms.....	6
1.5	Out of Scope .....	7
<b>2</b>	<b>Infrastructure changes for Release V2.0 .....</b>	<b>7</b>
2.1	Summary .....	7
2.2	References .....	8
<b>3</b>	<b>Steps to Migrate .....</b>	<b>8</b>
3.1	Resources Configurations for Change Audit Feature .....	8
3.1.1	Create Delegated Subnet.....	8
3.1.2	Create Azure Service Bus .....	9
3.1.3	Create App Service Plan for Function App .....	11
3.1.4	Create Function App.....	11
3.1.5	Changes to Cosmos DB Collections .....	14
3.1.6	Key Vault access for function app.....	16
3.2	Resources Configuration for USDM Versioning .....	18
3.2.1	Disable Client Certificate in APIM Custom Domain .....	18
3.2.2	APIM Route Configuration .....	19
3.2.3	Enable Client Certificate on API Group Inbound Policies .....	23
3.2.4	Cosmos DB Updates .....	28
3.2.5	Data Migration .....	29
3.3	Additional Cloud Configurations .....	30
3.3.1	Key Vault Configurations .....	30
3.4	Build and Deployment Updates .....	31
3.4.1	GitHub Secrets updates for deploying Function App .....	31
3.4.2	Code Deployment .....	34
<b>4</b>	<b>Containerized Deployment .....</b>	<b>35</b>
4.1	Recreate App Service Plans and App Services .....	35
4.1.1	Destroy existing UI app service Plan and App Service .....	35
4.1.2	Recreate UI app Service Plan.....	35
4.1.3	Recreate UI App Service.....	36
4.1.4	SDR UI App Service Configuration .....	36

---

4.1.5	Destroy existing API app service Plan and App Service.....	38
4.1.6	Recreate API App Service Plan.....	39
4.1.7	Create API App Service .....	39
4.1.8	SDR API App Service Configuration.....	39
4.1.9	Create Azure Container Registry .....	43
4.1.10	Container Registry Configuration.....	43
4.2	GitHub Actions/YAML/Build & Deploy Script Updates.....	44
4.2.1	GitHub Actions to deploy container image to UI App Service.....	45
4.2.2	GitHub Actions to deploy container image to API App Service .....	46
4.2.3	GitHub Secrets used in Workflow.....	47

## List of figures

Figure 1 Delegated Subnet .....	9
Figure 2 Service Bus Namespace Queue Configurations.....	10
Figure 3 Service Bus Namespace Queue Shared access policies Configurations.....	10
Figure 4 Service Bus Namespace Queue Shared access policies .....	11
Figure 5 Function App Inbound Traffic Configurations.....	12
Figure 6 Function App Outbound Traffic Configurations.....	12
Figure 7 Function App Diagnostic Configurations.....	13
<i>Figure 8 Function App Identity Configurations.....</i>	<i>13</i>
Figure 9 Function App new Application settings .....	14
Figure 10 Network Configurations for Azure Cosmos DB for MongoDB .....	15
Figure 11 Adding ChangeAudit Data Collections.....	15
Figure 12 Adding Indexing Policy for ChangeAudit Data Collection.....	16
Figure 13 Providing Access Policy for Function App .....	17
Figure 14 Custom domains Configuration .....	18
Figure 15 Custom domains Configuration – Uncheck Negotiate Client Certificate.....	19
Figure 16 SDR API Route Configuration .....	21
Figure 17 SDR UI Route Configuration .....	22
Figure 18 APIM Inbound Processing.....	24
Figure 19 APIM (API Management) - Inbound Policy .....	26
Figure 20 Adding StudyDefinitions Data Collections .....	29
Figure 21 Adding Indexing Policy for StudyDefinitions Collection .....	29
Figure 22 Create Secret.....	31
Figure 23 GitHub Actions Secrets .....	32
Figure 24 Add new GitHub Action Secret.....	32
Figure 25 Function App GitHub Secret.....	33
Figure 26 Adding a new task in yml file to Publish Function App Build Artifact .....	33
Figure 27 Adding new key vault value in yml file .....	34
Figure 28 Destroy existing UI app service Plan and App Service .....	35
Figure 29 SDR UI Network Configuration.....	37
Figure 30 SDR UI Diagnostic Configuration .....	38
Figure 31 SDR API App Service Details.....	39
Figure 32 SDR API Access Restriction Configuration.....	40
Figure 33 SDR API Network Configuration.....	41
Figure 34 SDR API Diagnostic Configuration .....	41
Figure 35 SDR API Identity Configuration .....	42
Figure 36 SDR API Diagnostic Configuration .....	42
Figure 37 Container Registry Basics Details .....	43
Figure 38 Enabling Admin User Settings.....	44
Figure 39 GitHub Action for Containerized Build .....	45
Figure 40 Container Registry GitHub Secrets for API.....	48
Figure 41 Container Registry GitHub Secrets for UI App Service .....	48

# 1 Introduction

## 1.1 Overview

This document details the steps required to migrate SDR Reference Implementation infrastructure from release Version 0.5 to Version 2.0 for users who have set up their own SDR instance for the Study Definition Repository – Reference Implementation on Azure Cloud Platform<sup>1</sup>. It provides details for deploying the new resources using azure portal. Additionally, it provides details of containerized deployment for the SDR RI API and UI applications.

## 1.2 Scope of Document

Scope of the document includes steps how to migrate SDR RI infrastructure from release 0.5 to release 2.0 on Azure Cloud.

## 1.3 Intended Audience

This document assumes a good understanding of Azure concepts and services. The audience for this document is users who have set up their own SDR instance (on Azure Cloud) and running version of SDR Release V0.5.

## 1.4 Definitions and Acronyms

Term / Abbreviation	Definition
API	Application Programming Interface
DB	Database
DDF	Digital Data Flow
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
LLD	Low Level Design
REST	Representational State Transfer
SDR	Study Definition Repository
UI	User Interface
URL	Uniform Resource Locator
VNet	Virtual Network
CDISC USDM	Unified Study Definitions Model (USDM versioning is supporting major versions of CDISC USDM versions)
USDM Versioning	Hosting studies conformant to multiple versions of CDISC USDM in a single instance of SDR
Change Audit	Capturing the audit information including user, date, time and action of study data CRUD activities

<sup>1</sup> To be clear, TransCelerate does not endorse any particular software, system, or service. And the use of specific brands of products or services by TransCelerate and its collaboration partners in developing the SDR Reference Implementation should not be viewed as any endorsement of such products or services. To the extent that the SDR Reference Implementation incorporates or relies on any specific branded products or services, this resulted out of the practical necessities associated with making a reference implementation available to demonstrate the SDR's capabilities.

## 1.5 Out of Scope

This document does not include any instructions to upgrade SDR on Cloud platforms other than Azure. Nonetheless, the SDR Platform Agnostic Recommendations can be referred to for corresponding resource configurations to Infrastructure changes for Release V2.0.

## 2 Infrastructure changes for Release V2.0

### 2.1 Summary

As part of SDR Azure infrastructure migration, the below changes have been implemented and configured to upgrade from Release V0.5 to V2.0.

- For Change Audit feature, below list of new resources have been added. Additionally, a new collection in CosmosDB has been created to hold change audit information.
  - Azure Service Bus
  - App Service Plan for Function App
  - Function App
  - Storage Account
- For USDM Versioning below changes have been done to APIM and Cosmos DB
  - New APIs have been implemented and APIM routes configured.
  - New common collection has been created to hold studies conformant to all USDM versions supported by SDR V2.0.
  - Additionally, Data updates have been done on USDM version 1.0 studies which will be covered in the data migration section.
- The certificate authentication has been removed for API endpoints supporting SDR UI and now is enabled only for SDR API endpoints.
- These are the resources that have been created to support Containerized Deployment. Existing App Service plans and App Services have been deleted.
  - Azure Container Registry
  - App Service Plan and App Service for UI App Service
  - App Service Plan and App Service for API App Service
- GitHub actions have been introduced to support containerized deployment. Containerized deployment is optional, and users can also use the existing standard way of deployment of build to the existing App services.

The users who want to install a new version of SDR from scratch can export data or redo their SDR setup from scratch, they can refer to the platform document guide and setup. Before that, users should take a backup of existing data. For data migration of existing data into new SDR setup user can use one of the recommended tools to export data collections and then import back to new environment.

## 2.2 References

Please refer to the documents below for information on latest SDR Architecture and Azure Resource Configurations.

Document Name	Document Link
SDR Solution Architecture	<a href="#">ddf-sdr-ri-solution-architecture-v3.0</a>
Low Level Design	<a href="#">ddf-sdr-ri-platform-low-level-design-v3.0</a>
Platform agnostic recommendations	<a href="#">ddf-sdr-ri-platform-agnostic-recommendations-v2.0</a>
SDR Platform Setup and Deployment Guide	<a href="#">ddf-sdr-ri-platform-setup-and-deployment-guide-v4.0</a>

## 3 Steps to Migrate

### PRE-REQUISITES:

- SDR Running instance on Azure (SDR Release V0.5).
- Minimum Contributor level of access at Subscription Level.
- Optionally, basic understanding of containerized deployments.

### 3.1 Resources Configurations for Change Audit Feature

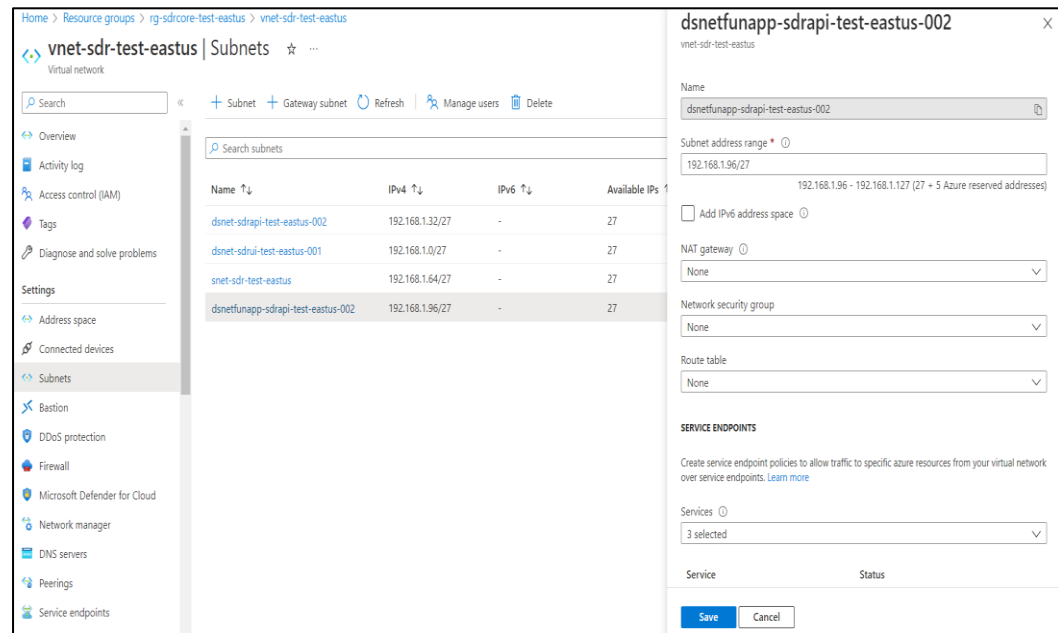
#### 3.1.1 Create Delegated Subnet

##### STEPS

1. Login to Azure Portal
2. Click on the resource groups tab.
3. Select the **sdrcore** resource group.
4. Refer the LLD and create a new delegated subnet.
  - Name
  - Subnet address range
  - Add IPv6 address space.
  - NAT gateway
  - Network Security Group
  - Route table
  - Services
  - Delegate subnet to a service



Figure 1 Delegated Subnet



### 3.1.2 Create Azure Service Bus STEPS

1. Login to Azure Portal
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create a new azure service bus.

#### SERVICE BUS CONFIGURATION

##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the Service Bus Namespace and navigate to queues tab.
5. Create a Queue with name (**changeauditqueue**) then click on create.
6. Navigate to **changeauditqueue**, click on Shared Access Policies.
7. From Shared Access Policies add SAS Policy with Manage entity and capture the connection string values.
8. Go to Monitoring Tab and select Diagnostic Setting.
9. Click on add Diagnostic Settings and give the Diagnostic Setting Name.
10. Select Logs and Metrics and check "Send to Log Analytics Workspace".
11. Save the settings.

**Note:** Please make a note of Queue name and connection string value required to be added in **Key Vaults Configurations Section 3.3.1** and **Function App Application Settings Section 3.1.4**.

Figure 2 Service Bus Namespace Queue Configurations

The screenshot shows the 'Create queue' configuration page in the Azure portal. The left sidebar displays the navigation menu with 'Queues' selected under 'Entities'. The main area shows a table with columns 'Name', 'Status', and 'Max size', currently displaying 'No results'. The right-hand configuration panel includes the following settings:

- Name:** changeauditsbqueue
- Max queue size:** 1 GB
- Max delivery count:** 10
- Message time to live:** 14 Days, 0 Hours, 0 Minutes, 0 Seconds
- Lock duration:** 0 Days, 0 Hours, 0 Minutes, 30 Seconds
- ☐ Enable dead lettering on message expiration
- ☐ Enable partitioning

A 'Create' button is located at the bottom right of the configuration panel.

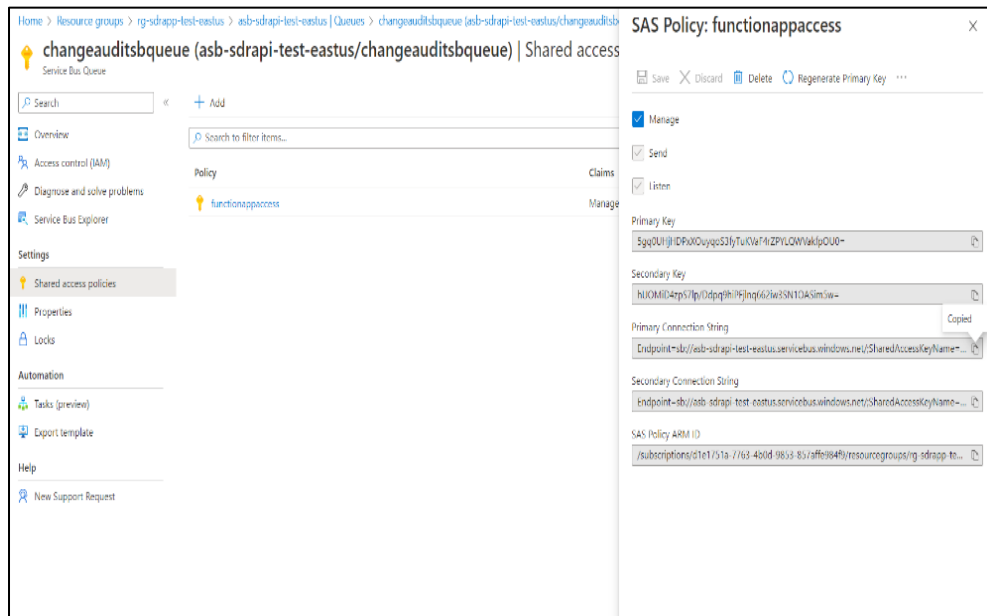
Figure 3 Service Bus Namespace Queue Shared access policies Configurations

The screenshot shows the 'Add SAS Policy' configuration page in the Azure portal. The left sidebar displays the navigation menu with 'Shared access policies' selected under 'Settings'. The main area shows a table with columns 'Policy' and 'Claims', currently displaying 'no policies have been set up yet'. The right-hand configuration panel includes the following settings:

- Policy name:** functionappaccess
- ☒ Manage
- ☒ Send
- ☒ Listen

A 'Create' button is located at the bottom right of the configuration panel.

Figure 4 Service Bus Namespace Queue Shared access policies



### 3.1.3 Create App Service Plan for Function App

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create a new app service plan for function app.

### 3.1.4 Create Function App

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create a function app.

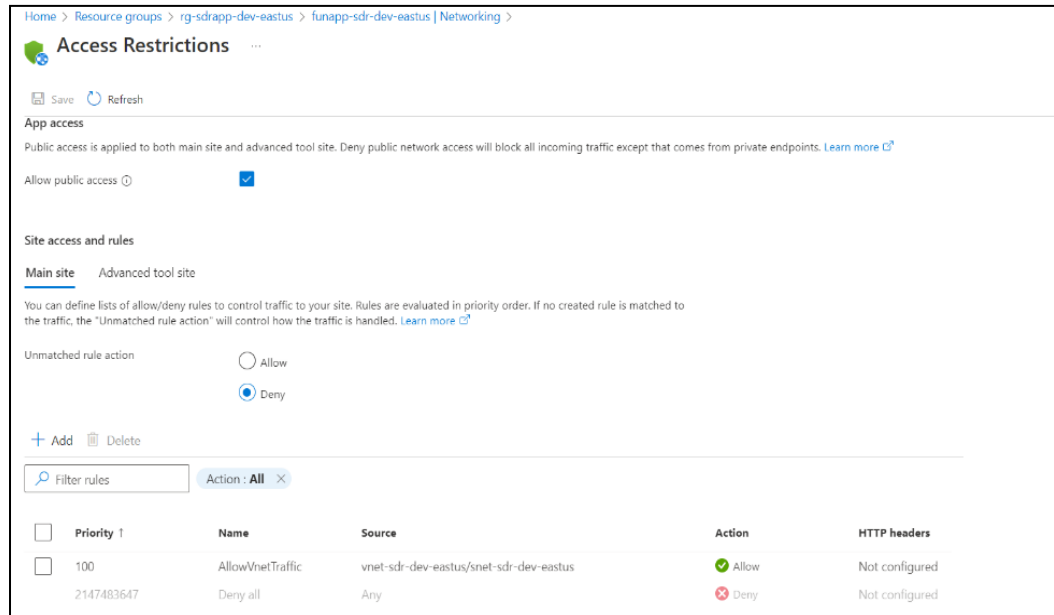
### FUNCTION APP CONFIGURATION

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the Function App and navigate to networking tab.
5. Click on Access Restrictions and Add Rule
6. Click on VNet Integration and Add VNet and Subnet then click on Ok.
7. Go to Monitoring Tab and select Diagnostic Setting
8. Click on add Diagnostic Settings and give the Diagnostic Setting Name
9. Select Logs and Metrics and send to Log Analytics Workspace

10. Save the settings.
11. Go to Identity tab and enable System assigned managed identity and save the settings.
12. Navigate to Settings tab and click on Configuration tab.
13. Click on new application setting and add new Application Settings by using below mentioned table.
14. Save the settings.

Figure 5 Function App Inbound Traffic Configurations



Home > Resource groups > rg-sdrapp-dev-eastus > funapp-sdr-dev-eastus | Networking >

**Access Restrictions**

Save Refresh

**App access**

Public access is applied to both main site and advanced tool site. Deny public network access will block all incoming traffic except that comes from private endpoints. [Learn more](#)

Allow public access ☒

**Site access and rules**

Main site Advanced tool site

You can define lists of allow/deny rules to control traffic to your site. Rules are evaluated in priority order. If no created rule is matched to the traffic, the "Unmatched rule action" will control how the traffic is handled. [Learn more](#)

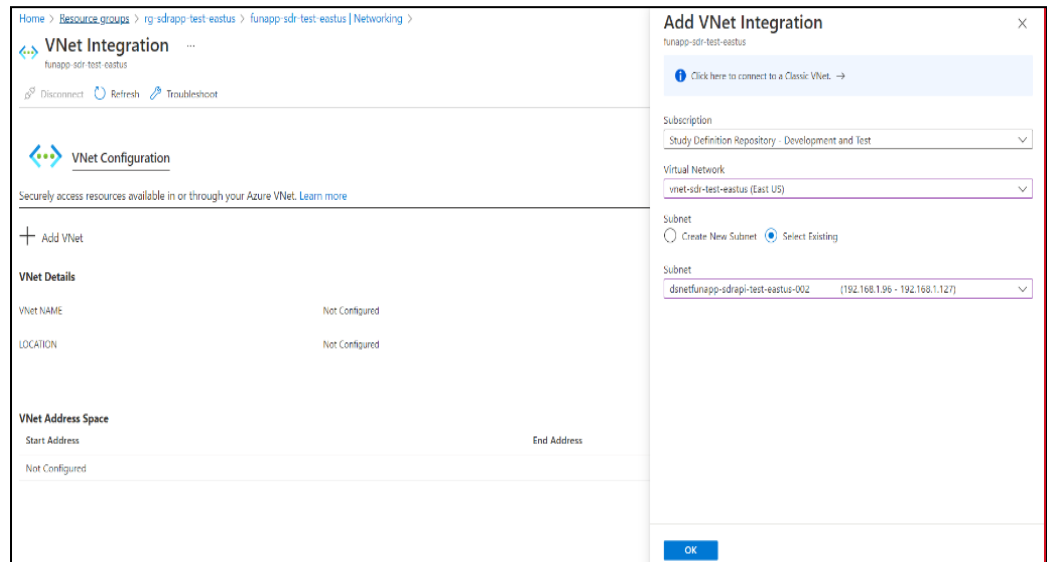
Unmatched rule action ☐ Allow ☒ Deny

+ Add Delete

Filter rules Action: All

Priority	Name	Source	Action	HTTP headers
100	AllowVnetTraffic	vnet-sdr-dev-eastus/snet-sdr-dev-eastus	Allow	Not configured
2147483647	Deny all	Any	Deny	Not configured

Figure 6 Function App Outbound Traffic Configurations



Home > Resource groups > rg-sdrapp-test-eastus > funapp-sdr-test-eastus | Networking >

**VNet Integration**

funapp-sdr-test-eastus

Disconnect Refresh Troubleshoot

**VNet Configuration**

Securely access resources available in or through your VNet. [Learn more](#)

+ Add VNet

**VNet Details**

VNet NAME Not Configured

LOCATION Not Configured

**VNet Address Space**

Start Address End Address

Not Configured

**Add VNet Integration**

funapp-sdr-test-eastus

Click here to connect to a Classic VNet. ->

Subscription Study Definition Repository - Development and Test

Virtual Network vnet-sdr-test-eastus (East US)

Subnet Create New Subnet Select Existing

Subnet dsnetfunapp-sdrapi-test-eastus-002 (192.168.1.96 - 192.168.1.127)

OK

Figure 7 Function App Diagnostic Configurations

Home > Resource groups > rg-sdrapp-test-eastus > funapp-sdr-test-eastus | Diagnostic settings >

### Diagnostic setting

Save Discard Delete Feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name \*

#### Logs

Categories

- ☒ Function Application Logs

#### Metrics

- ☒ AllMetrics

#### Destination details

- ☒ Send to Log Analytics workspace
- Subscription:
- Log Analytics workspace:
- ☐ Archive to a storage account
- ☐ Stream to an event hub
- ☐ Send to partner solution

Figure 8 Function App Identity Configurations

Home > Resource groups > rg-sdrapp-test-eastus > funapp-sdr-test-eastus

### funapp-sdr-test-eastus | Identity

Function App

Search

Events (preview)

Functions

- Functions
- App keys
- App files
- Proxies

Deployment

- Deployment slots
- Deployment Center

Settings

- Configuration
- Authentication
- Application Insights
- Identity**
- Backups
- Custom domains
- Custom domains (classic)

System assigned User assigned

A system assigned managed identity is restricted to one per resource and is tied to the lifecycle of this resource. You can grant permissions to the managed identity by using Azure role-based access control (Azure RBAC). The managed identity is authenticated with Azure AD, so you don't have to store any credentials in code. [Learn more about Managed identities.](#)

Save Discard Refresh Got feedback?

Status ☒ On ☐ Off

Object (principal) ID

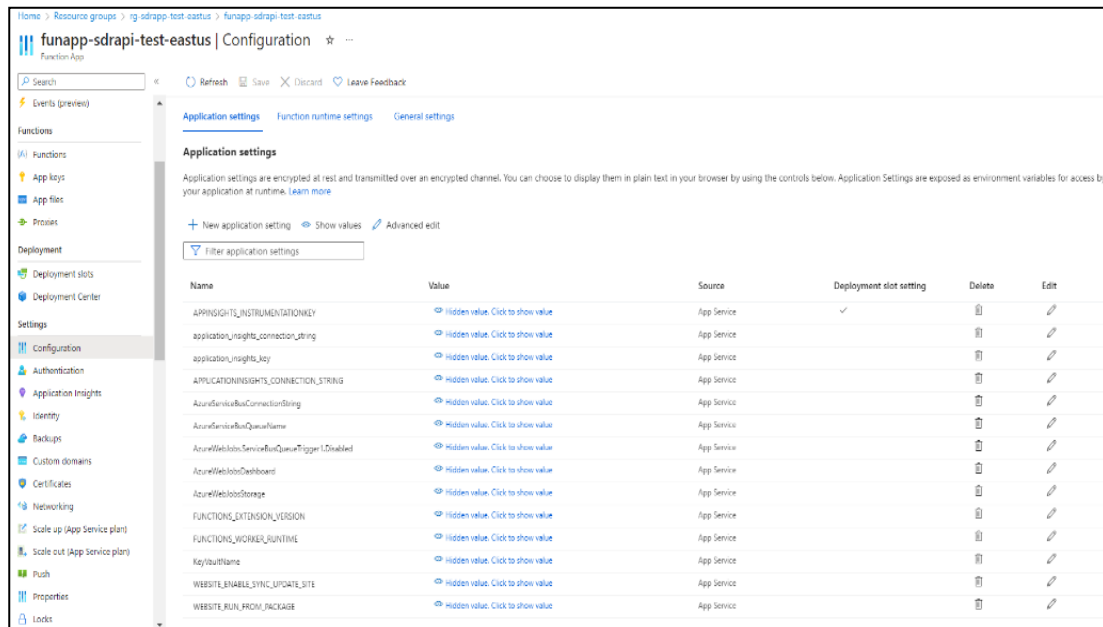
Permissions

**Info** This resource is registered with Azure Active Directory. The managed identity can be configured to allow access to other resources. Be careful when making changes to the access settings for the managed identity because it can result in failures. [Learn more](#)

[Function App Application Settings]

Name	Value
AzureServiceBusConnectionString	Refer to <a href="#">Service Bus Configuration</a> Section
AzureServiceBusQueueName	Provide Azure Service Bus Queue Name
KeyVaultName	Provide Key Vault URL form Key Vault

Figure 9 Function App new Application settings



### 3.1.5 Changes to Cosmos DB Collections

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select Azure Cosmos DB for MongoDB and navigate to networking tab.
5. Go to public access and select Selected networks, under existing VNet, add a new subnet with name as per the naming convention (E.g., **dsnetfunapp-sdrapi-qa-eastus-002**) then save the settings.
6. Go to Data Explorer and click on New Collection and give the existing database i.e., SDR and give the Collection id (**ChangeAudit**) then click on Ok.
7. Navigate to **ChangeAudit** data Collection and click on settings.
8. Click on add indexing Policy and add index as per below table.

Index Policy table

Definition	Type
<code>_id</code>	Single field

Figure 10 Network Configurations for Azure Cosmos DB for MongoDB

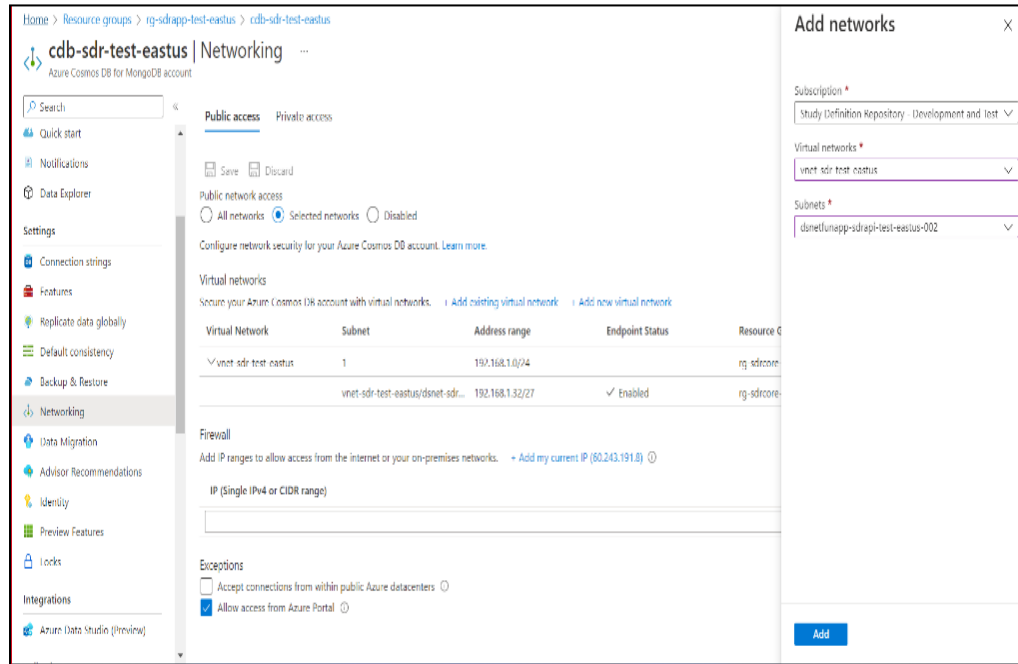


Figure 11 Adding ChangeAudit Data Collections

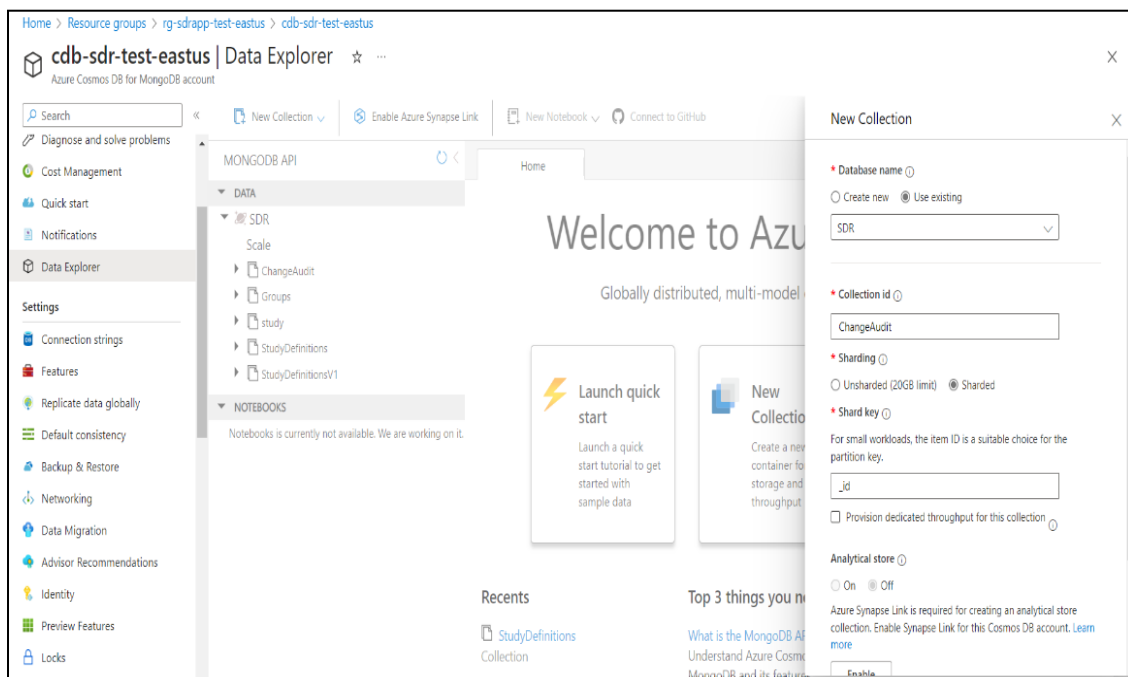
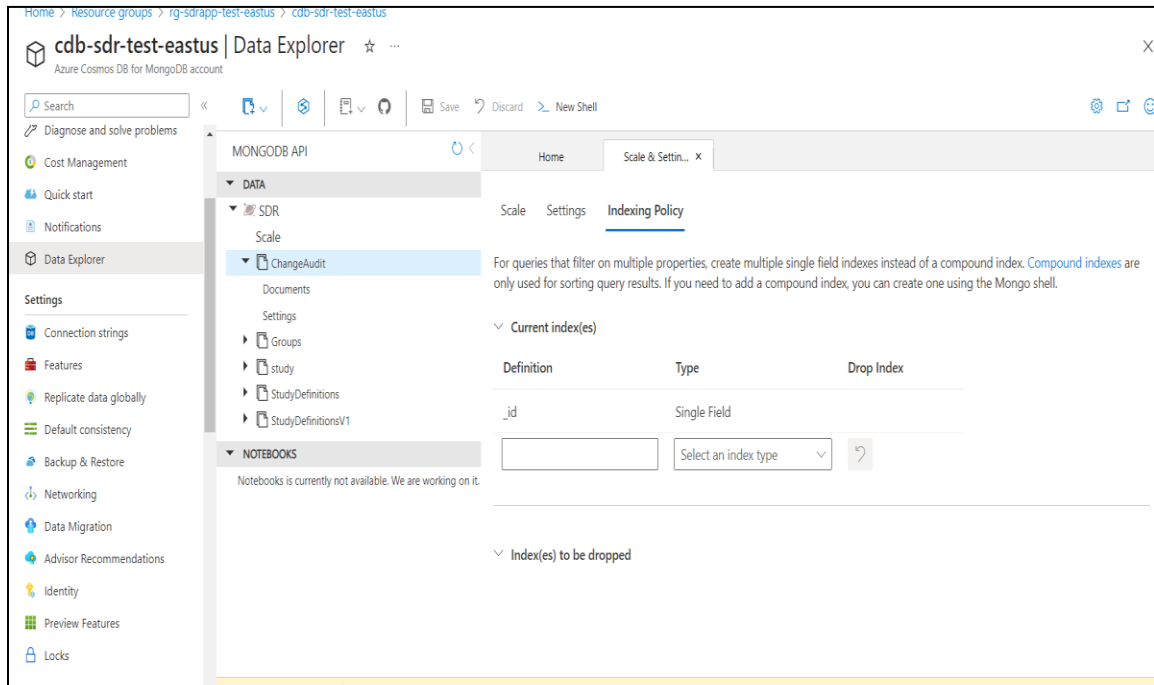


Figure 12 Adding Indexing Policy for ChangeAudit Data Collection



### 3.1.6 Key Vault access for function app

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrcore** resource group.
4. Select Azure Key Vault and go to Access policies tab.
5. Click on Create and select Secret Management Configure from a template.
6. Click on Next and search for function app name and select Function App (E.g., funapp-sdrapi-qa-eastus).
7. Click on Next and click on Create.



Figure 13 Providing Access Policy for Function App

Home > Key vaults > kv-sdr-test-eastus | Access policies >

## Create an access policy ...

kv-sdr-test-eastus

1 Permissions 2 Principal 3 Application (optional) 4 Review + create

Configure from a template

Secret Management

Key permissions	Secret permissions	Certificate permissions
Key Management Operations	Secret Management Operations	Certificate Management Operations
<input type="checkbox"/> Select all	<input checked="" type="checkbox"/> Select all	<input type="checkbox"/> Select all
<input type="checkbox"/> Get	<input checked="" type="checkbox"/> Get	<input type="checkbox"/> Get
<input type="checkbox"/> List	<input checked="" type="checkbox"/> List	<input type="checkbox"/> List
<input type="checkbox"/> Update	<input checked="" type="checkbox"/> Set	<input type="checkbox"/> Update
<input type="checkbox"/> Create	<input checked="" type="checkbox"/> Delete	<input type="checkbox"/> Create
<input type="checkbox"/> Import	<input checked="" type="checkbox"/> Recover	<input type="checkbox"/> Import
<input type="checkbox"/> Delete	<input checked="" type="checkbox"/> Backup	<input type="checkbox"/> Delete
<input type="checkbox"/> Recover	<input checked="" type="checkbox"/> Restore	<input type="checkbox"/> Recover
<input type="checkbox"/> Backup	Privileged Secret Operations	<input type="checkbox"/> Backup
<input type="checkbox"/> Restore	<input type="checkbox"/> Select all	<input type="checkbox"/> Restore
		<input type="checkbox"/> Manage Contacts

Previous Next

Home > Key vaults > kv-sdr-test-eastus | Access policies >

## Create an access policy ...

kv-sdr-test-eastus

1 Permissions 2 Principal 3 Application (optional) 4 Review + create

Only 1 principal can be assigned per access policy.  
Use the new embedded experience to select a principal. The previous popup experience can be accessed here. [Select a principal](#)

funapp-sdrapi-test-eastus

funapp-sdrapi-test-eastus  
804b8a90-153e-4f57-972c-b4d87d4db310

Selected item

funapp-sdrapi-test-eastus  
804b8a90-153e-4f57-972c-b4d87d4db310

Previous Next

## 3.2 Resources Configuration for USDM Versioning

### 3.2.1 Disable Client Certificate in APIM Custom Domain

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the API Management service and search for Custom domains.
5. Go to Custom domains and unselect Negotiate client certificate.
6. Update the settings.

Figure 14 Custom domains Configuration

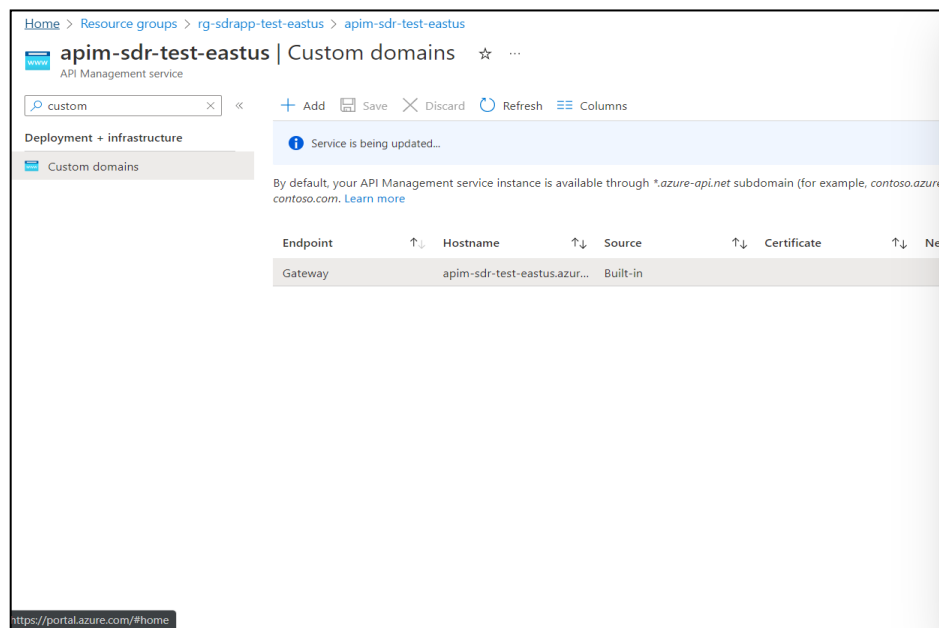
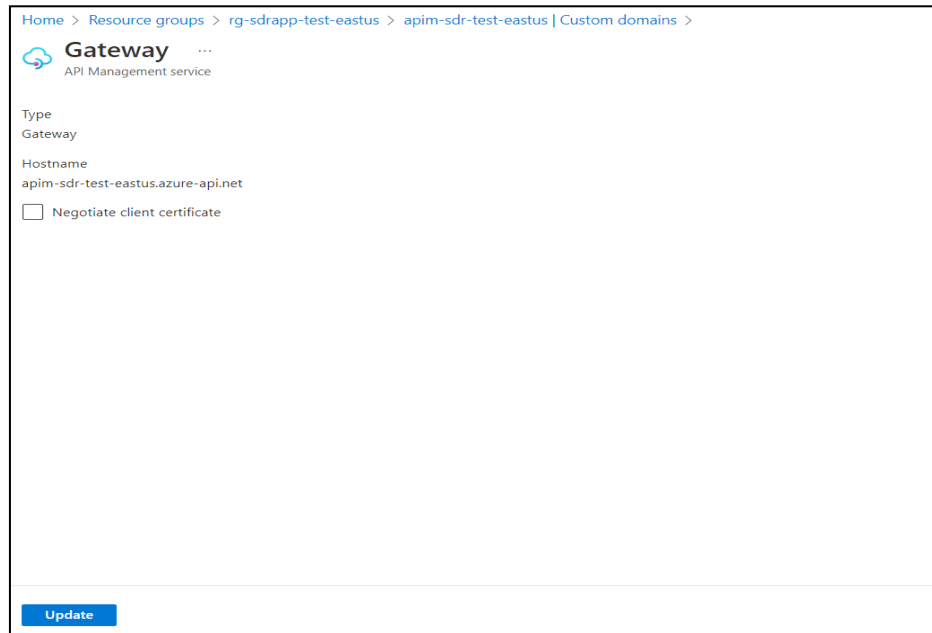


Figure 15 Custom domains Configuration – Uncheck Negotiate Client Certificate



### 3.2.2 APIM Route Configuration

First, user must clear all existing routes and reconfigure the routes as per the table below. One example is given below to configure route and same steps should be followed for all the routes.

Method	API Name	Name	APIM Route	URL	Responses
GET	SDR API	Common - Get Study History	/api	/studydefinitions/history	200 OK
GET	SDR API	Common - Get Audit Trail	/api	/studydefinitions/{studyId}/auditTrail	200 OK
GET	SDR API	Common - Get Change Audit	/api	/studydefinitions/{studyId}/changeaudit	200 OK
GET	SDR API	Common - Get API Versions	/api	/versions	200 OK
GET	SDR API	Common - Get Study Raw Data	/api	/studydefinitions/{studyId}/raw data	200 OK
GET	SDR API	Common - Get eCPT	/api	/studyDefinitions/{studyId}/studydesigns/eCPT	200 OK
GET	SDR UI Admin	Check Group Name	/api/ui/admin	/usergroups/checkgroupname	200 OK
POST	SDR UI Admin	Get Group List	/api/ui/admin	/usergroups/getgroups	200 OK

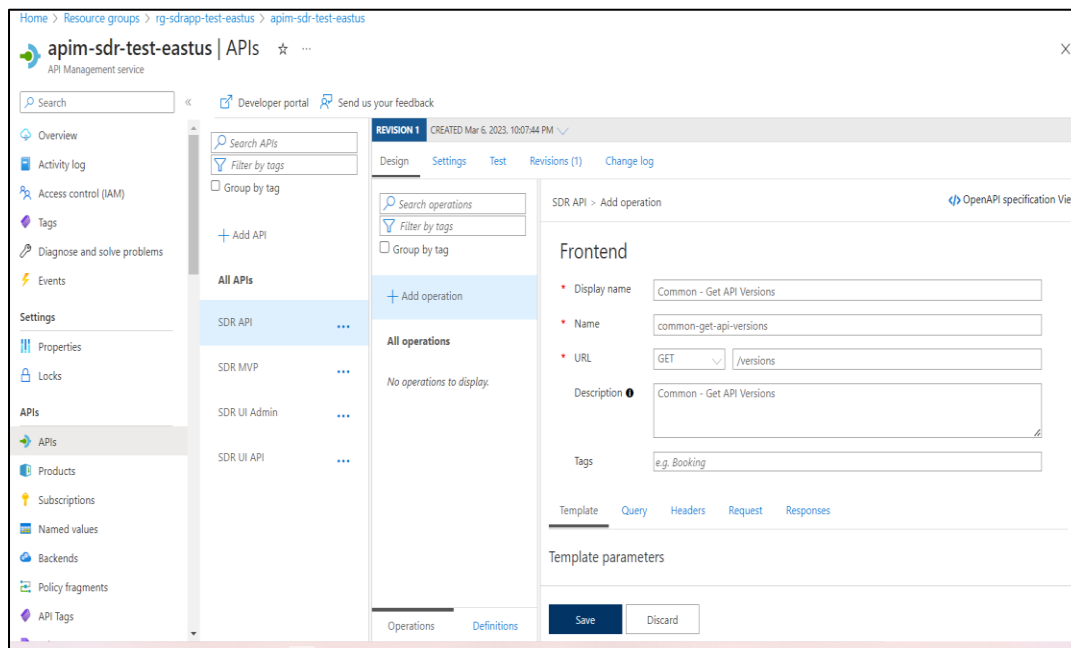
<b>GET</b>	SDR UI Admin	Get Groups	/api/ui/admin	/usergroups/getgrouplist	200 OK
<b>POST</b>	SDR UI Admin	Get Users	/api/ui/admin	/usergroups/getusers	200 OK
<b>GET</b>	SDR UI Admin	List Users	/api/ui/admin	/usergroups/listusers	200 OK
<b>POST</b>	SDR UI Admin	Post Group	/api/ui/admin	/usergroups/postgroup	200 OK
<b>POST</b>	SDR UI Admin	Post User	/api/ui/admin	/usergroups/postuser	200 OK
<b>POST</b>	SDR UI	Usage Reports	/api/ui	/reports/usage	200 OK
<b>POST</b>	SDR UI	Search API	/api/ui	/studydefinitions/search	200 OK
<b>POST</b>	SDR UI	Search Study Title	/api/ui	/studydefinitions/searchstudytitle	200 OK
<b>GET</b>	SDR UI	Common - Get Version History	/api/ui	/studydefinitions/{studyId}/auditTrail	200 OK
<b>GET</b>	SDR UI	V1 Get Study Definition	/api/ui	/v1/studydefinitions/{studyId}	200 OK
<b>GET</b>	SDR UI	V2 Get Study Definition	/api/ui	/v2/studydefinitions/{studyId}	200 OK
<b>GET</b>	SDR UI	Get Study Links	/api/ui	/studydefinitions/{studyId}/links	200 OK
<b>GET</b>	SDR UI	V2 Get Study Design SOA	/api/ui	/v2/studydefinitions/{studyId}/studydesigns/soa	200 OK
<b>GET</b>	SDR API	V1 Get Study Definition	/api	/v1/studydefinitions/{studyId}	200 OK
<b>POST</b>	SDR API	V1 Post Study Definition	/api	/v1/studydefinitions	201 Created
<b>GET</b>	SDR API	V1 Get Study Design	/api	/v1/studydesigns	200 OK
<b>GET</b>	SDR API	V2 Get Study Definition	/api	/v2/studydefinitions/{studyId}	200 OK
<b>POST</b>	SDR API	V2 Put Study Definition	/api	/v2/studydefinitions	201 Created
<b>PUT</b>	SDR API	V2 Post Study Definition	/api	/v2/studydefinitions	201 Created
<b>GET</b>	SDR API	V2 Get Study Design	/api	/v2/studydesigns	200 OK
<b>GET</b>	SDR API	V2 Get Study Design SOA	/api	/v2/studydefinitions/{studyId}/studydesigns/soa	200 OK

## • SDR API Route Configuration

### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group ration.
4. Select the API Management service and navigate to APIs Tab.
5. Select SDR API and click on Add operation.
6. Provide the details as per table above.
7. Save the settings.
  - i. Display Name
  - ii. Name
  - iii. URL
  - iv. Responses
8. Follow the same steps for all the routes.

Figure 16 SDR API Route Configuration



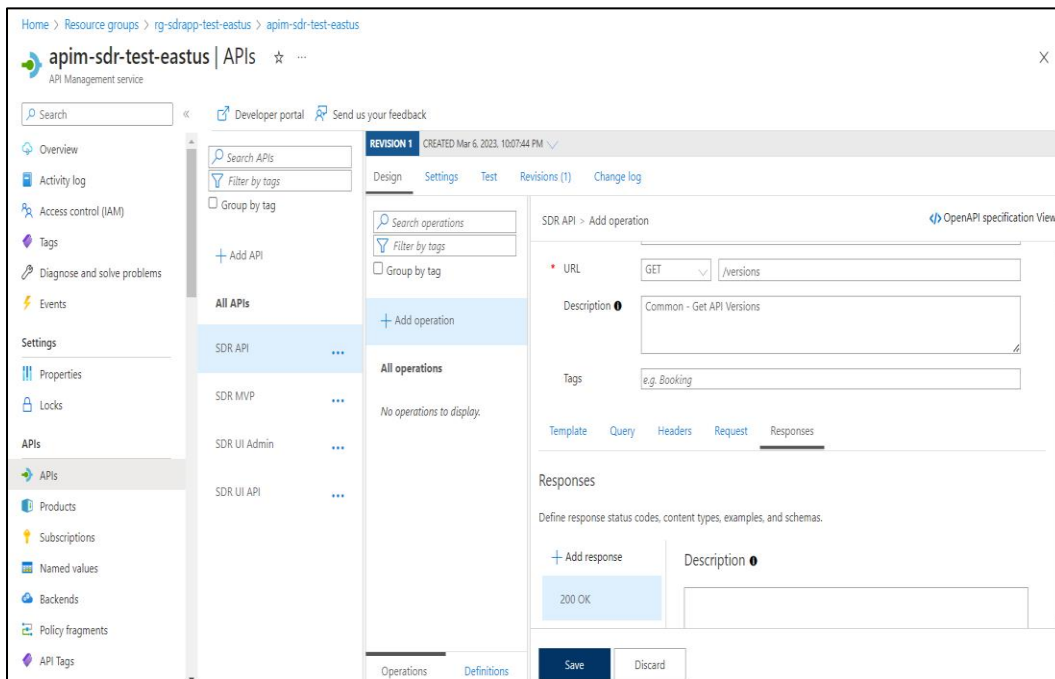
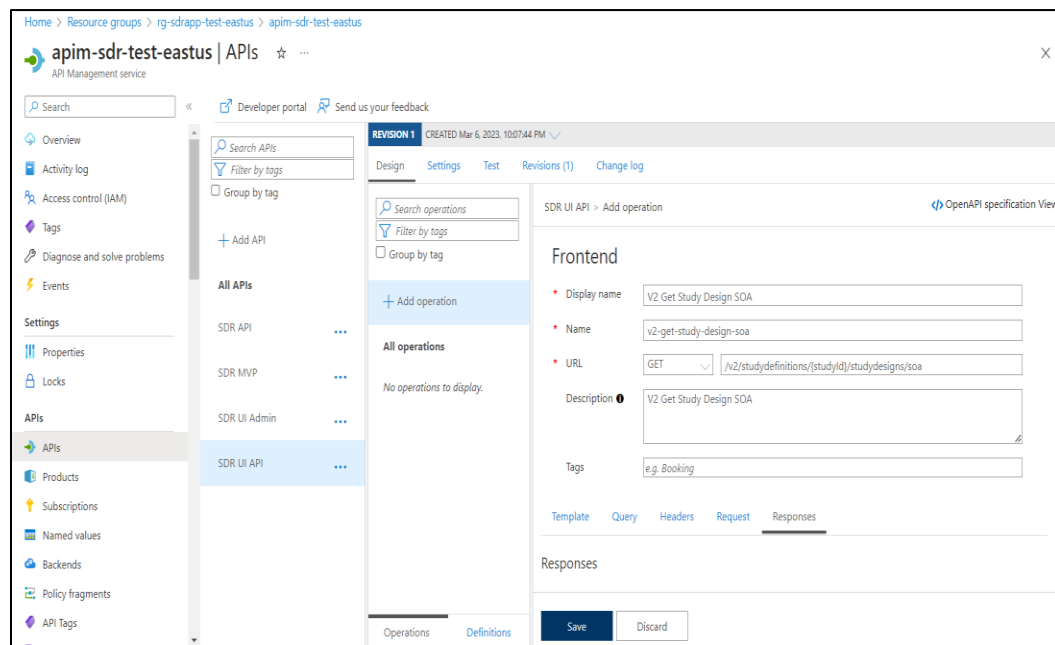
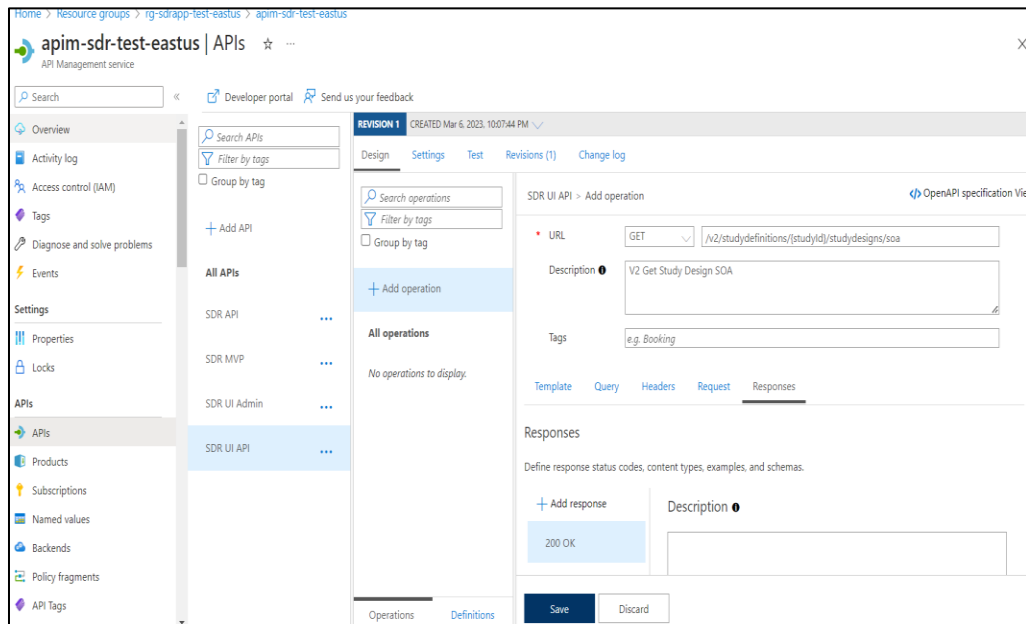


Figure 17 SDR UI Route Configuration



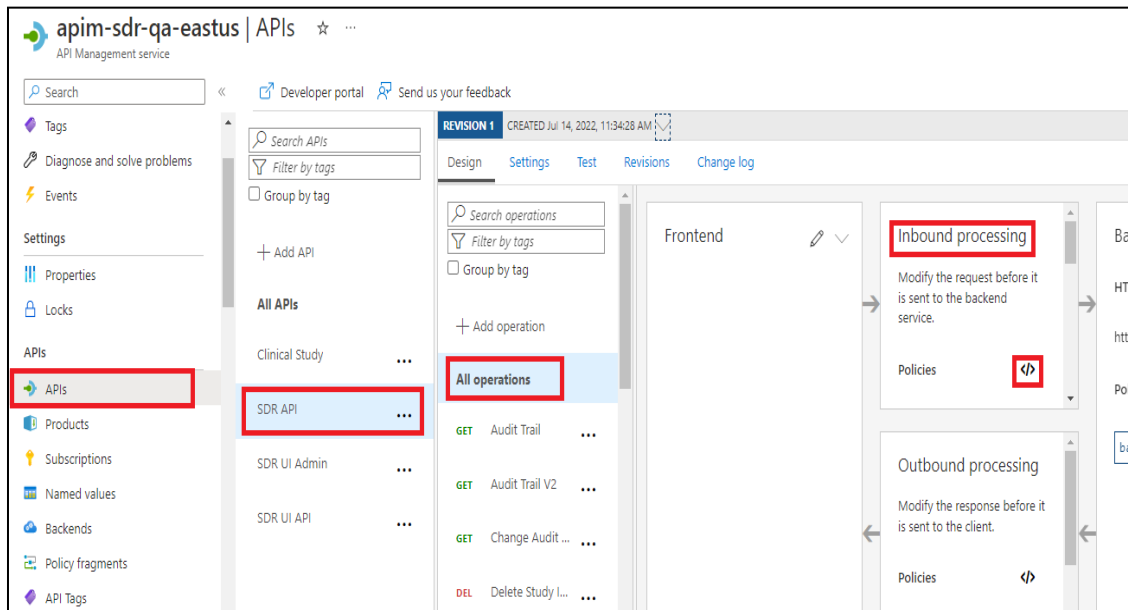


### 3.2.3 Enable Client Certificate on API Group Inbound Policies

**Configure Inbound policy on SDR API endpoint for client certificate validation**

- Configure the policy on SDR API to validate one or more attributes of a client certificate used to access APIs hosted in API Management instance.
- Go to APIs -> Select the SDR API -> Select "All Operations" -> Inbound processing -> Select Policies

Figure 18 APIM Inbound Processing



- Add the policy code below to check the thumbprint of a client certificate against certificates uploaded to API Management

```
<policies>
  <inbound>
    <set-variable name="EmailAddress" value="@{
      string name = "EmptyAuthToken"
      var="" authHeader = context.Request.Headers.GetValueOrDefault
      return="" authHeader.AsJwt()?.Claims.GetValueOrDefault="("email",
"EmptyAuthToken");
    }" />
    <set-variable name="UserName" value="@{
      string name = "EmptyAuthToken"
      var="" authHeader = context.Request.Headers.GetValueOrDefault
      return="" authHeader.AsJwt()?.Claims.GetValueOrDefault="("name",
"EmptyAuthToken");
    }" />
    <choose>
      <when condition="@((context.Variables["EmailAddress"]=="") != null)">
        <trace source="My Global APIM Policy" severity="information">
          <message>@(String.Format("{0} | {1}", context.Api.Name,
context.Operation.Name))</message>
          <metadata name="EmailAddress"
value="@((string)context.Variables["EmailAddress"]=="")" />
          <metadata name="UserName"
value="@((string)context.Variables["UserName"]=="")" />
        </trace>
      </when>
      <otherwise>
        <trace source="My Global APIM Policy" severity="information">
```

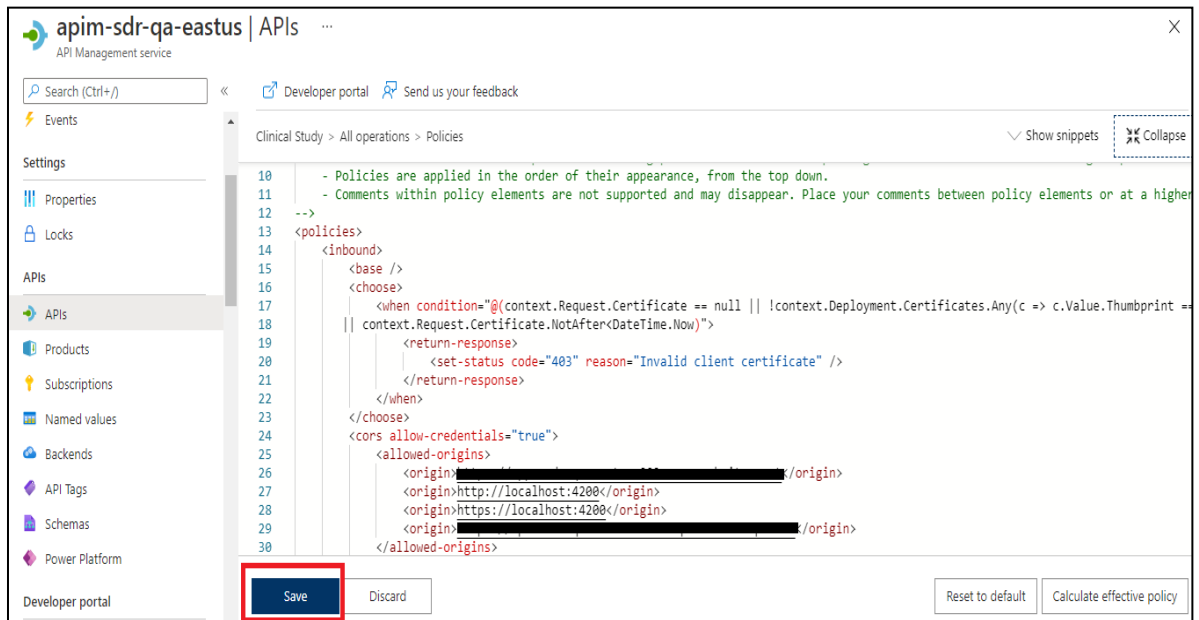


```

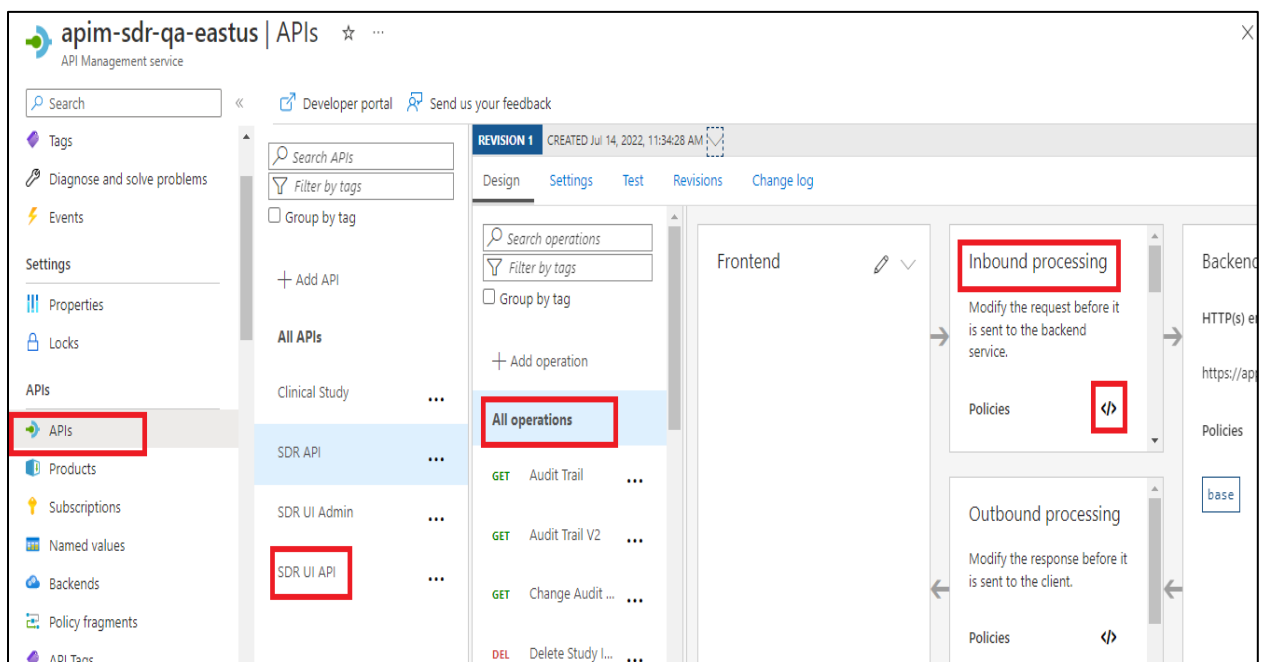
        <message>@(String.Format("{0} | {1}", context.Api.Name,
context.Operation.Name))</message>
        <metadata name="EmailAddress" value="Not Available" />
        <metadata name="UserName" value="Not Available" />
    </trace>
</otherwise>
</choose>
<base />
<choose>
    <when condition="@context.Request.Certificate == null ||
!context.Deployment.Certificates.Any(c => c.Value.Thumbprint ==
context.Request.Certificate.Thumbprint)
|| context.Request.Certificate.NotAfter"
    <DateTime.Now)">
        <return-response>
            <set-status code="403" reason="Invalid client certificate" />
        </return-response>
    </when>
</choose>
<cors allow-credentials="true">
    <allowed-origins>
        <origin>Add Backend APP Service URL</origin>
        <origin>http://localhost:4200</origin>
        <origin>https://localhost:4200</origin>
        <origin>Add API Management URL</origin>
        <origin>Add Frontend (UI) URL</origin>
    </allowed-origins>
    <allowed-methods preflight-result-max-age="300">
        <method>GET</method>
        <method>POST</method>
        <method>PATCH</method>
        <method>DELETE</method>
    </allowed-methods>
    <allowed-headers>
        <header>*</header>
    </allowed-headers>
    <expose-headers>
        <header>*</header>
    </expose-headers>
</cors>
</inbound>
<backend>
    <base />
</backend>
<outbound>
    <base />
</outbound>
<on-error>
    <base />
</on-error>
</policies>

```

Figure 19 APIM (API Management) - Inbound Policy



- **Remove Certificate Inbound policy on SDR UI API & SDR UI ADMIN endpoints**
  - Configure the policy on SDR UI API and SDR UI Admin to validate the inbound requests to access APIs hosted in API Management instance.
  - Go to APIs -> Select the SDR UI API -> Select "All Operations" -> Inbound processing -> Select Policies



- Add the policy code and make sure the inbound policy looks like below.

```
<policies>
  <inbound>
    <set-variable name="EmailAddress" value="@{
      string name = "EmptyAuthToken"
      var="" authHeader = context.Request.Headers.GetValueOrDefault
      return="" authHeader.AsJwt()?.Claims.GetValueOrDefault=""("email",
"EmptyAuthToken");
    }" />
    <set-variable name="UserName" value="@{
      string name = "EmptyAuthToken"
      var="" authHeader = context.Request.Headers.GetValueOrDefault
      return="" authHeader.AsJwt()?.Claims.GetValueOrDefault=""("name",
"EmptyAuthToken");
    }" />
    <choose>
      <when condition="@((context.Variables["EmailAddress"]=="") != null)">
        <trace source="My Global APIM Policy" severity="information">
          <message>@(String.Format("{0} | {1}", context.Api.Name,
context.Operation.Name))</message>
          <metadata name="EmailAddress"
value="@((string)context.Variables["EmailAddress"]=="")" />
          <metadata name="UserName"
value="@((string)context.Variables["UserName"]=="")" />
        </trace>
      </when>
      <otherwise>
        <trace source="My Global APIM Policy" severity="information">
          <message>@(String.Format("{0} | {1}", context.Api.Name,
context.Operation.Name))</message>
          <metadata name="EmailAddress" value="Not Available" />
          <metadata name="UserName" value="Not Available" />
        </trace>
      </otherwise>
    </choose>
  </inbound>
  <base />
  <cors allow-credentials="true">
    <allowed-origins>
      <origin>Add Backend APP Service URL</origin>
      <origin>http://localhost:4200</origin>
      <origin>https://localhost:4200</origin>
      <origin>Add API Management URL</origin>
      <origin>Add Frontend (UI) URL</origin>
    </allowed-origins>
    <allowed-methods preflight-result-max-age="300">
      <method>GET</method>
      <method>POST</method>
      <method>PATCH</method>
      <method>DELETE</method>
    </allowed-methods>
    <allowed-headers>
      <header>*</header>
    </allowed-headers>
    <expose-headers>
```

```

    <header>*</header>
  </expose-headers>
</cors>
</inbound>
<backend>
  <base />
</backend>
<outbound>
  <base />
</outbound>
<on-error>
  <base />
</on-error>
</policies>

```

- Repeat the above steps to configure the inbound policy on SDR UI Admin endpoint.
- Now for SDR UI API and SDR UI Admin the certificate Authentication is removed.

### 3.2.4 Cosmos DB Updates

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Go to Data Explorer and click on New Collection tab and use existing database i.e., SDR and give the Collection id (**StudyDefinitions**) then click on Ok.
5. Navigate to **StudyDefinitions** and click on settings.
6. Click on add indexing Policy and add indexes as per below table.

*Index policy table for StudyDefinitions*

Definition	Type
<b>_id</b>	Single Field
<b>clinicalStudy.studyId</b>	Single Field
<b>auditTrail.entryDateTime</b>	Single Field
<b>clinicalStudy.studyTitle</b>	Single Field
<b>auditTrail.usdmVersion</b>	Single Field

Figure 20 Adding StudyDefinitions Data Collections

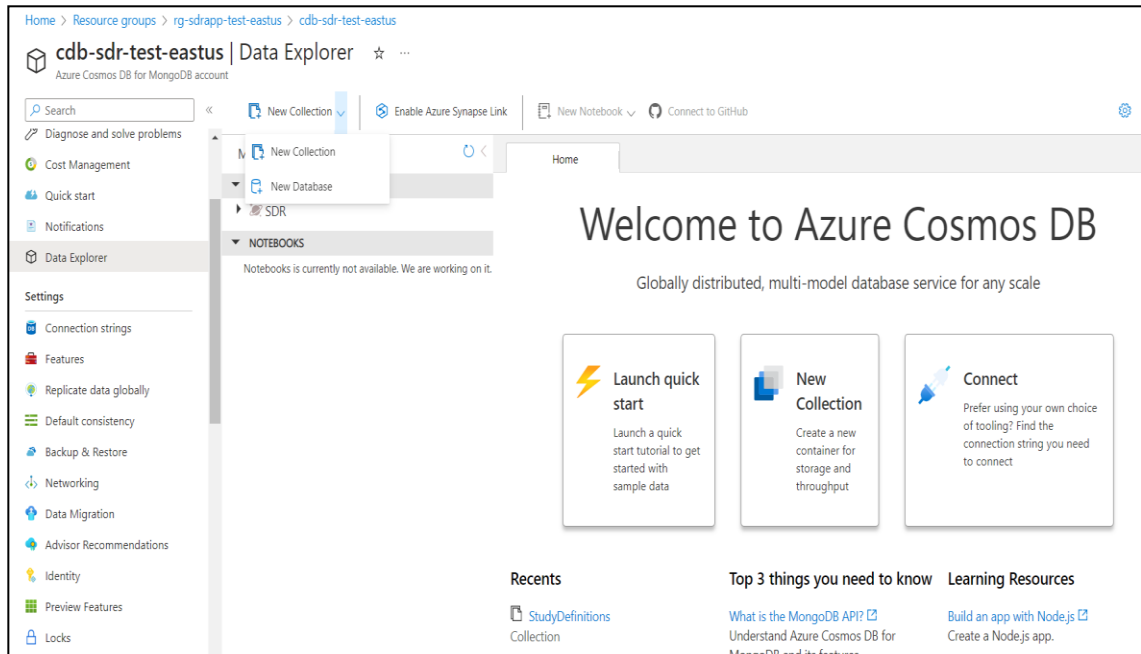
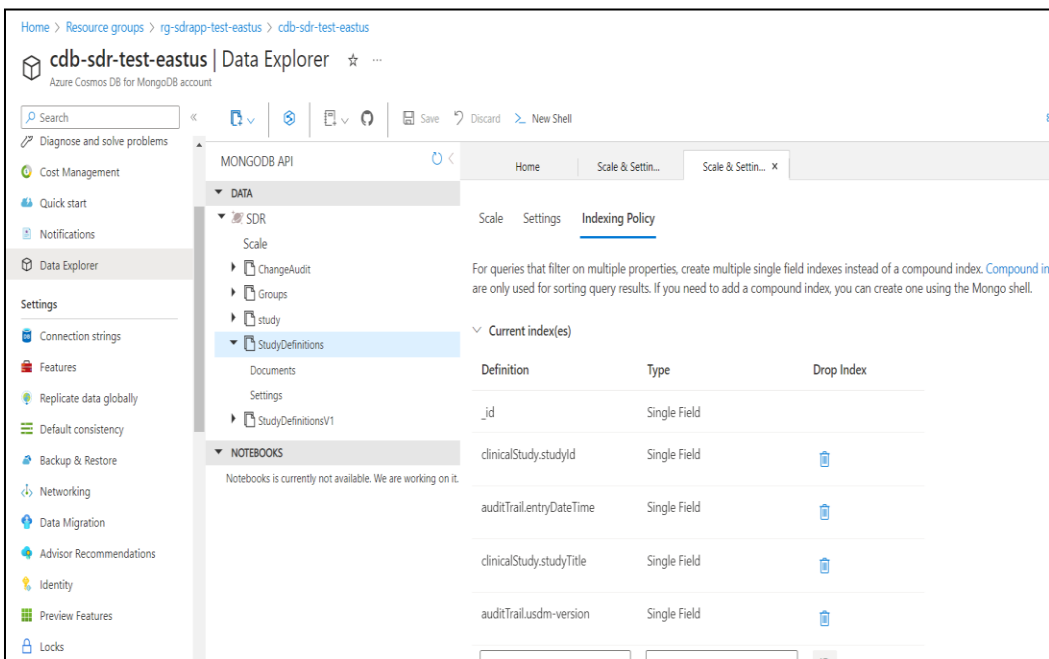


Figure 21 Adding Indexing Policy for StudyDefinitions Collection



### 3.2.5 Data Migration

#### STEPS

1. Launch Mongo shell and connect to the database using the connection string.

2. The old collection "StudyDefinitionsV1" must be updated to add "usdmVersion" parameter under AuditTrail section.
3. To add usdmVersion attribute on all documents in "StudyDefinitionsV1" collection run below command.

```
db.getCollection("StudyDefinitionsV1").updateMany({}, {$set:{"auditTrail.usdmVersion": "1.0"}}, false, true)
```

4. Run below commands to move the documents from "StudyDefinitionsV1" collection to the new common "StudyDefinitions" collection.

```
db.getCollection("StudyDefinitionsV1").aggregate([ { $merge : "StudyDefinitions" } ])
```

5. The merge operation mentioned in previous step is supported only for MongoDB version 4.4 and above. For other lower Mongo DB versions, this activity can be done by exporting the documents from the source collection and importing into the destination collection by using any third-party IDE tools for MongoDB.
6. Run below command to update the attribute name from uuid to studyId for all USDM V1.0 documents in the new common "StudyDefinitions" collection.

```
db.getCollection("StudyDefinitions").updateMany({"auditTrail.usdmVersion": "1.0"}, {$rename:{"clinicalStudy.uuid": "clinicalStudy.studyId"}}, false, true)
```

### 3.3 Additional Cloud Configurations

#### 3.3.1 Key Vault Configurations

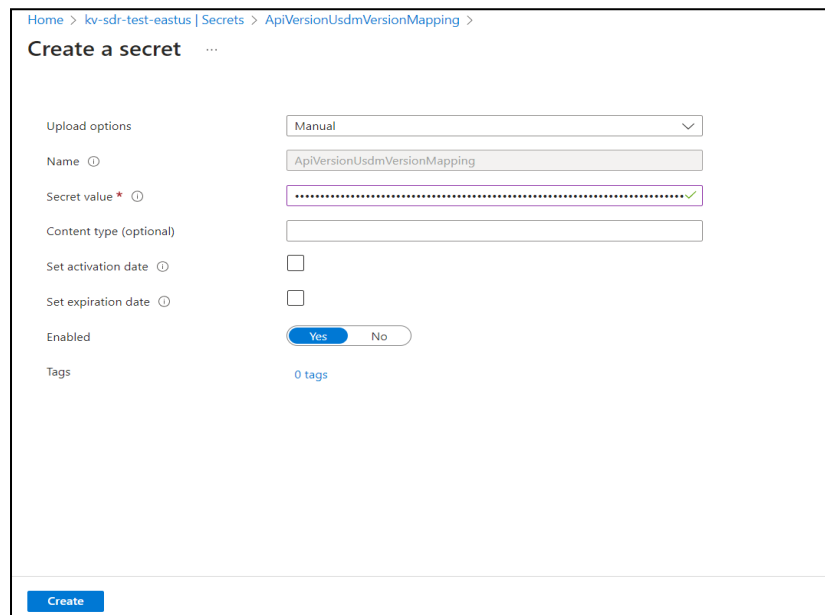
##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrcore** resource group.
4. Go to Objects tab and Click on Secret.
5. Click on Generate/Import tab and create secrets by using below table.

Name	
Env-Name	Provide a key word for the deployed environment (dev,qa,stage etc.)
ApiVersionUsdmVersionMapping	Refer to <a href="#">sdr-api-usdm-version-mapping.json</a>
SdrCptMasterDataMapping	Refer to <a href="#">sdr-cpt-master-data-mapping.json</a>
ConformanceRules	Refer to <a href="#">usdm-conformance-rules-configuration.json</a>
AzureServiceBusConnectionString	Refer to <a href="#">Service_Bus_Configuration</a> Section

<b>AzureServiceBusQueueName</b>	Refer to <a href="#">Service Bus Configuration</a> Section
---------------------------------	------------------------------------------------------------

Figure 22 Create Secret



## 3.4 Build and Deployment Updates

### 3.4.1 GitHub Secrets updates for deploying Function App

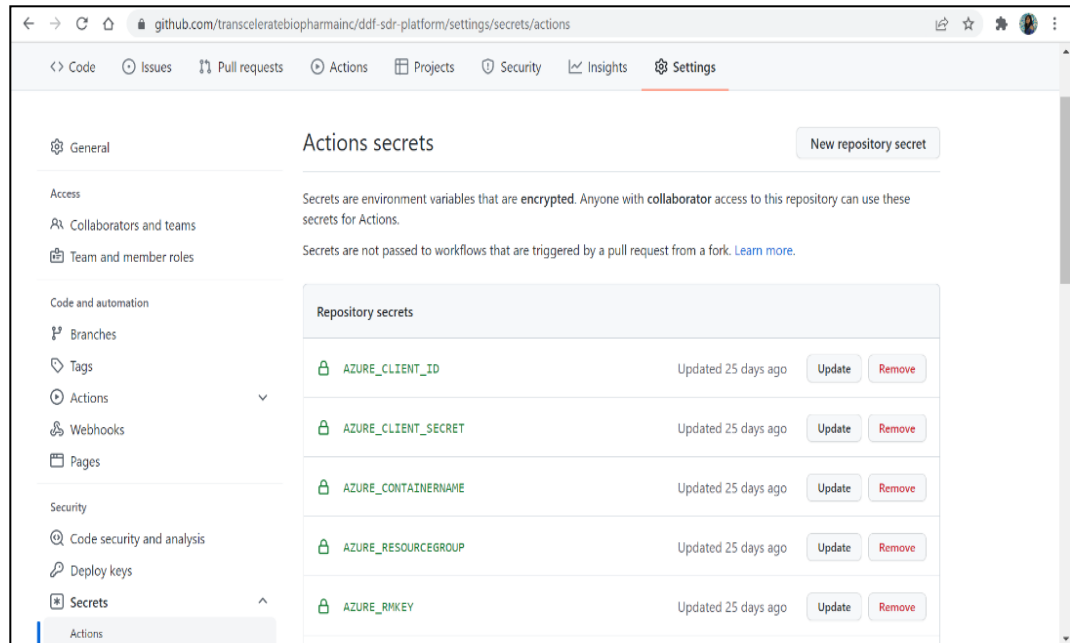
Create GitHub Secret in **ddf-sdr-api** repo required for Function App deployment.

#### Adding Secret in GitHub

##### STEPS:

1. Go to repo settings.
2. On the lower left-hand side of the screen, click on Secrets.
3. Under that click on Actions.
4. Then click on New Repository Secret.

Figure 23 GitHub Actions Secrets



5. After clicking New Repository Secret, fill the details of the secret (name and value) in the boxes. Name – AZURE\_FUNCTIONAPP\_NAME and Value – name of function app resource created in Azure e.g.: funapp-sdr-dev-eastus.

Figure 24 Add new GitHub Action Secret

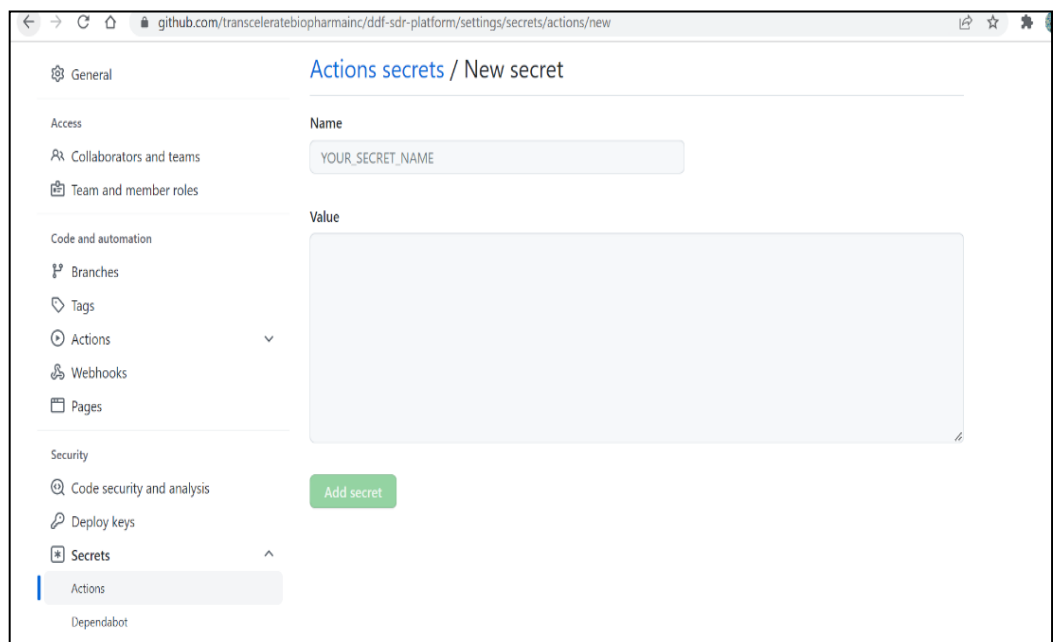
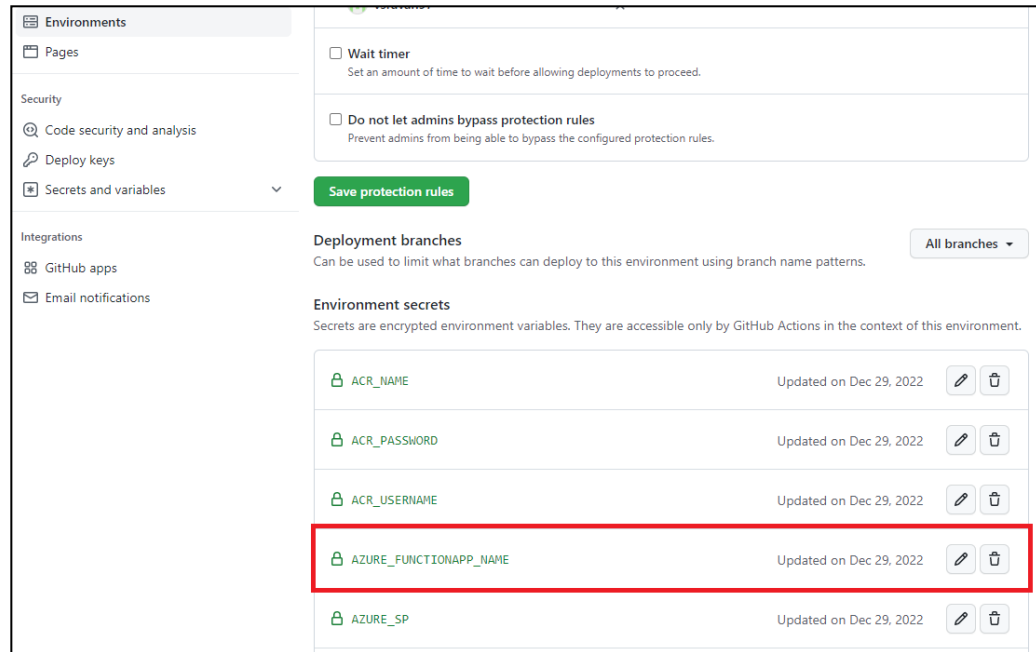




Figure 25 Function App GitHub Secret



6. Update GitHub Action yml file in ddf-sdr-api repo. A new task has been created to publish function App Artifacts. Please refer the below screen shot.

Figure 26 Adding a new task in yml file to Publish Function App Build Artifact

```
- name: Zip publish files
  shell: pwsh
  # Zip the Function App build artifact publish folder
  run: |
    Compress-Archive -Path "${{ github.workspace }}/src/TransCelerate.SDR.AzureFunctions/bin/Release/net6.0/publish/**" -DestinationPath "${{ github.workspace }}\FunAppPublish.zip"
```

```
- name: 'Publish Function App Artifact: Artifact'
  uses: actions/upload-artifact@v2
  # Publish the Function App Build artifact in github
  with:
    path: '${{ github.workspace }}/FunAppPublish.zip'
    name: FunctionAppBuild-Artifact
```

```
- name: Azure Functions Action
  uses: Azure/functions-action@v1.4.7
  with:
    # Name of the Azure Function App
    app-name: ${ secrets.AZURE_FUNCTIONAPP_NAME } # Replace with Function app name
    # Path to package or folder. *.zip or a folder to deploy
    package: '${ github.workspace }}/FunAppPublish.zip'
```

### • ddf-sdr-ui repo YAML script Changes

1. A new Key vault variable is added in yml file to display the Environment in the System Usage report.
2. Additionally, the node.js version from 12.x to 14.x. Please refer the below screen shots

Figure 27 Adding new key vault value in yml file

```
- name: Fetch Keyvault values
  uses: Azure/get-keyvault-secrets@v1
  with:
    # Name of the azure key vault
    keyvault: ${ secrets.KEYVAULT_NAME } # name of key vault in Azure portal
    # Name of the secret to be fetched
    secrets: 'AzureAd-TenantId,AzureAd-Authority,AzureAd-ClientId,AzureAd-Audience,AzureAd-RedirectUrl,AzureAd-LoginUrl,Apim-BaseUr ,Env-Name' # comma separated list of s
    id: keyvaultSecretAction # ID for secrets that you will reference

- name: Find and Replace User Name - Using Build Variable
  # Variables in environment.ts is replaced using the find script while fetching it from keyvault through secrets
  run: |
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-TenantId#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-T
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-Authority#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-ClientId#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-C
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-Audience#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-A
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-RedirectUrl#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#AzureAd-LoginUrl#}@${{ steps.keyvaultSecretAction.outputs.AzureAd-L
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#Apim-BaseUr#@${{ steps.keyvaultSecretAction.outputs.Apim-BaseUr
    find ${ github.workspace }/SDR-WebApp/src/environments/environment.ts -type f -exec sed -i ''s@{#Env-Name#}@${{ steps.keyvaultSecretAction.outputs.Env-Name }}@g''

- name: Use Node 14.x
  uses: actions/setup-node@v1
  with:
    # The node.js version to use
    node-version: 14.x
```

### 3.4.2 Code Deployment

To enable containerized deployment of SDR application, please skip this step and refer to section 4. Otherwise, the UI and API code deployment to respective App services can be done using the below updated scripts.

[ddf-sdr-api/release.yml at develop · transcelerate/ddf-sdr-api \(github.com\)](#)

[ddf-sdr-ui/release.yml at develop · transcelerate/ddf-sdr-ui \(github.com\)](#)

## 4 Containerized Deployment

The UI and API builds have been containerized as a part of SDR Release V2.0. The same has impacted the App Services on which the SDR Application is hosted. This section details the steps to enable containerized build and deployment.

This is an optional step. To continue with existing way of deployment of SDR use current App Service and Service Plan configuration and B&D scripts for UI and API for deployment as mentioned in section 3.4.2.

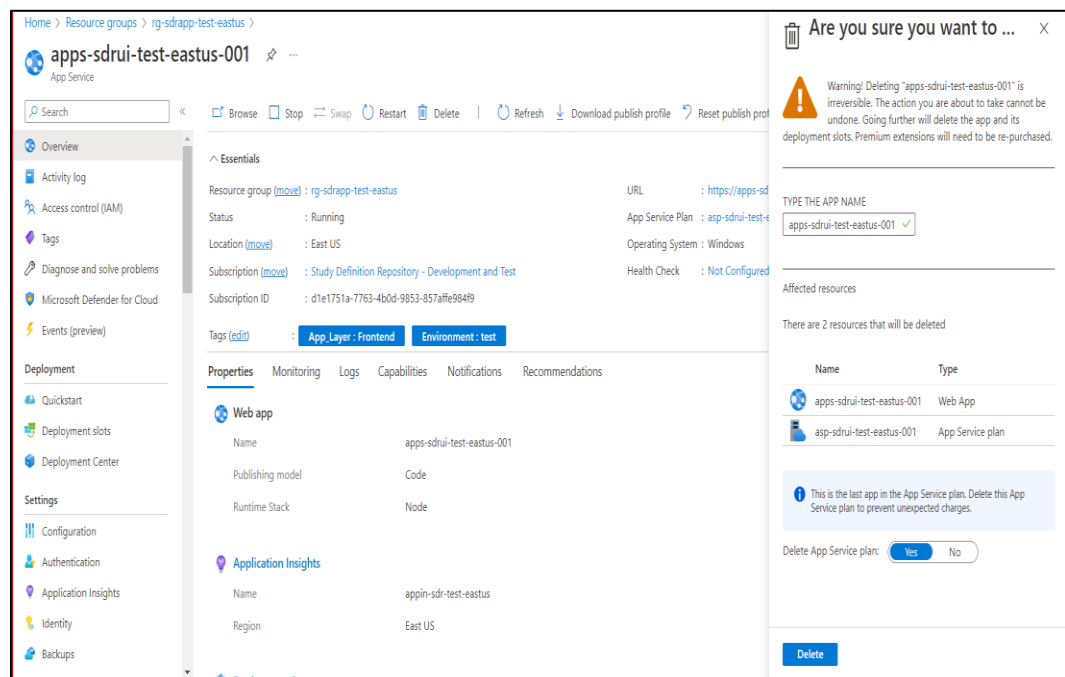
### 4.1 Recreate App Service Plans and App Services

#### 4.1.1 Destroy existing UI app service Plan and App Service

##### STEPS

1. Login to Azure Portal
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the UI App Service and Click on Delete from the Overview Page and delete both App service and App Service Plans

Figure 28 Destroy existing UI app service Plan and App Service



#### 4.1.2 Recreate UI app Service Plan

##### STEPS

1. Login to Azure Portal
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.

4. Refer the LLD document and create an App Service plan for SDR UI

#### 4.1.3 Recreate UI App Service

##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create an App Service for SDR UI.

#### 4.1.4 SDR UI App Service Configuration

##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the UI App Service and navigate to networking tab.
5. Click on VNet Integration and add VNet and subnet then click on Ok.
6. Go to Monitoring Tab and select Diagnostic Setting.
7. Click on add Diagnostic Settings and give the Diagnostic Setting Name.
8. Select Logs and Metrics and check "Send to Log Analytics Workspace"
9. Save the settings.

Figure 29 SDR UI Network Configuration

Home > Resource groups > rg-sdrapp-test-eastus > apps-sdrui-test-eastus-001 | Networking >

## VNet Integration

apps-sdrui-test-eastus-001

Disconnect Refresh

### VNet Configuration

Securely access resources available in or through your Azure VNet. [Learn more](#)

+ Add VNet

VNet Details	
VNet NAME	Not Configured
LOCATION	Not Configured

VNet Address Space	
Start Address	End Address
Not Configured	

### Add VNet Integration

apps-sdrui-test-eastus-001

Subscription  
Study Definition Repository - Development and Test

Virtual Network  
vnet-sdr-test-eastus (East US)

Subnet  
☐ Create New Subnet ☒ Select existing

Subnet  
dsnet-sdrui-test-eastus-001 (192.168.1.0 - 192.168.1.31)

OK

Home > Resource groups > rg-sdrapp-test-eastus > apps-sdrui-test-eastus-001 | Networking >

## VNet Integration

apps-sdrui-test-eastus-001

Disconnect Refresh

**Route All** ☒ Enabled  
When Route All is enabled, all outbound traffic from the app will be affected by networking configurations. [Learn more](#)

VNet Details	
VNet NAME	vnet-sdr-test-eastus
LOCATION	East US

VNet Address Space	
Start Address	End Address
192.168.1.0	192.168.1.255

Subnet Details	
Subnet NAME	dsnet-sdrui-test-eastus-001

Subnet Address Space	
Start Address	End Address
192.168.1.0	192.168.1.31

Figure 30 SDR UI Diagnostic Configuration

Home > Resource groups > rg-sdrapp-test-eastus > apps-sdrui-test-eastus-001 | Diagnostic settings >

## Diagnostic setting

Save Discard Delete Feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name \* UIAppService ✓

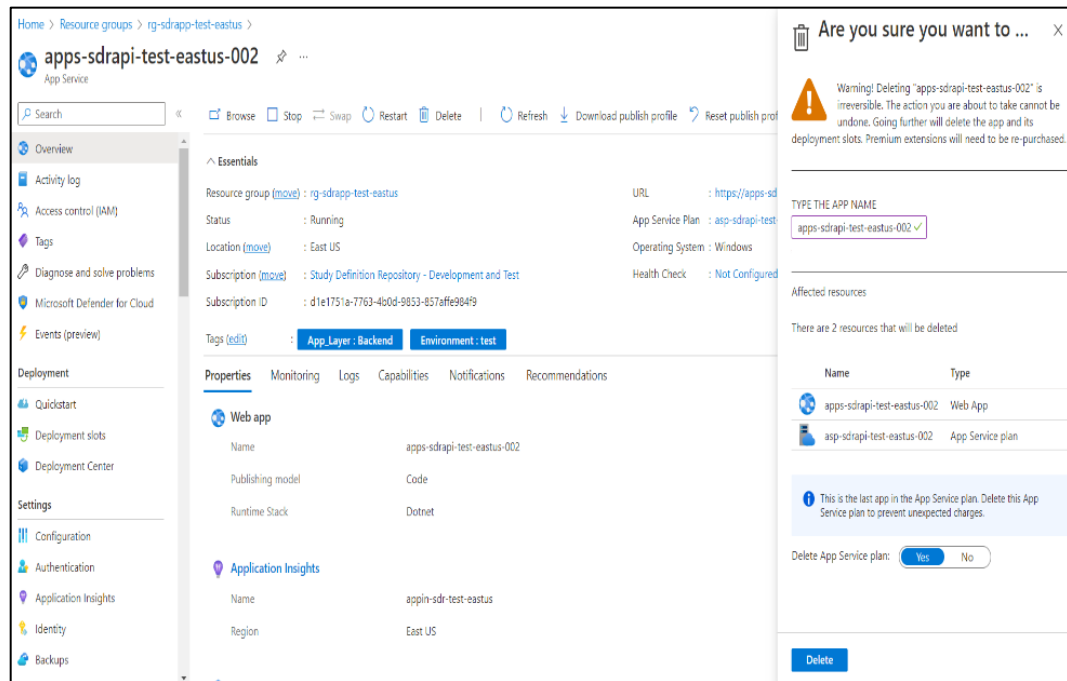
Logs	Destination details
<p>Categories</p> <ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> HTTP logs</li> <li><input checked="" type="checkbox"/> App Service Console Logs</li> <li><input checked="" type="checkbox"/> App Service Application Logs</li> <li><input checked="" type="checkbox"/> Access Audit Logs</li> <li><input checked="" type="checkbox"/> IPSecurity Audit logs</li> <li><input checked="" type="checkbox"/> App Service Platform logs</li> </ul>	<ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> Send to Log Analytics workspace</li> <li>Subscription: Study Definition Repository - Development and Test</li> <li>Log Analytics workspace: law-sdr-test-eastus ( eastus )</li> <li><input type="checkbox"/> Archive to a storage account</li> <li><input type="checkbox"/> Stream to an event hub</li> <li><input type="checkbox"/> Send to partner solution</li> </ul>
<p>Metrics</p> <ul style="list-style-type: none"> <li><input checked="" type="checkbox"/> AllMetrics</li> </ul>	

### 4.1.5 Destroy existing API app service Plan and App Service

#### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the API App Service and Click on Delete from the Overview page and delete both App Service and App Service Plan

Figure 31 SDR API App Service Details



#### 4.1.6 Recreate API App Service Plan

##### STEPS

1. Login to Azure Portal
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create an App Service plan for SDR API

#### 4.1.7 Create API App Service

##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create an App Service for SDR API.

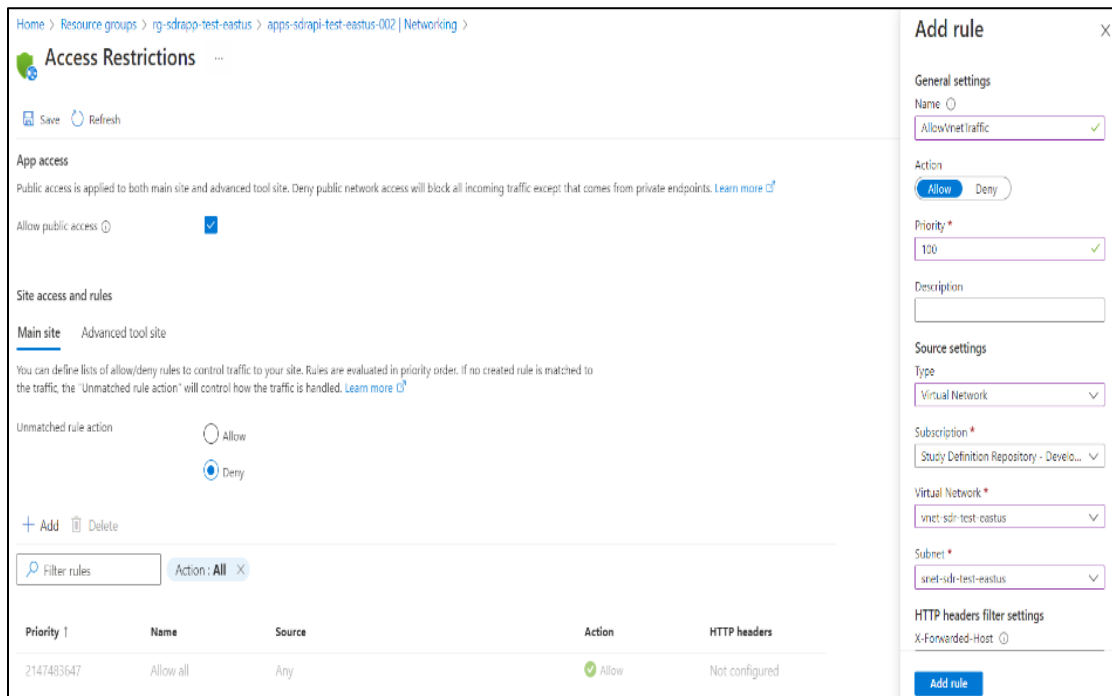
#### 4.1.8 SDR API App Service Configuration

##### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select the API App Service and navigate to networking tab.
5. Click on Access Restrictions and Add Rule.

6. Click on VNet Integration and add VNet and subnet. Then and Disable **Route All** option and click on Ok.
7. Go to Monitoring Tab and select Diagnostic Setting.
8. Click on add Diagnostic Settings and give the Diagnostic Setting Name.
9. Select Logs and Metrics and send to Log Analytics Workspace
10. Save the settings.
11. Go to Identity tab and enable System assigned managed identity and save the settings.
12. To enable KeyVault access for API app service, Navigate to **sdrcore** resource group.
13. Select Azure Key Vault and navigate to Access policies tab.
14. Click on Create and select **Secret Management** option from the **Configure a template** dropdown.
15. Select all applicable **secret management operations**
16. Click on Next and search for API app name and select API App Service (E.g., apps-sdrapi-qa-estus-002).
17. Click on Next and click on Create.

Figure 32 SDR API Access Restriction Configuration



Home > Resource groups > rg-sdrapi-test-eastus > apps-sdrapi-test-eastus-002 | Networking >

## Access Restrictions

Save Refresh

### App access

Public access is applied to both main site and advanced tool site. Deny public network access will block all incoming traffic except that comes from private endpoints. [Learn more](#)

Allow public access ☒

### Site access and rules

Main site Advanced tool site

You can define lists of allow/deny rules to control traffic to your site. Rules are evaluated in priority order. If no created rule is matched to the traffic, the 'Unmatched rule action' will control how the traffic is handled. [Learn more](#)

Unmatched rule action ☐ Allow ☒ Deny

+ Add - Delete

Filter rules Action: All

Priority	Name	Source	Action	HTTP headers
2147483647	Allow all	Any	Allow	Not configured

### Add rule

General settings

Name

Action ☒ Allow ☐ Deny

Priority

Description

Source settings

Type

Subscription

Virtual Network

Subnet

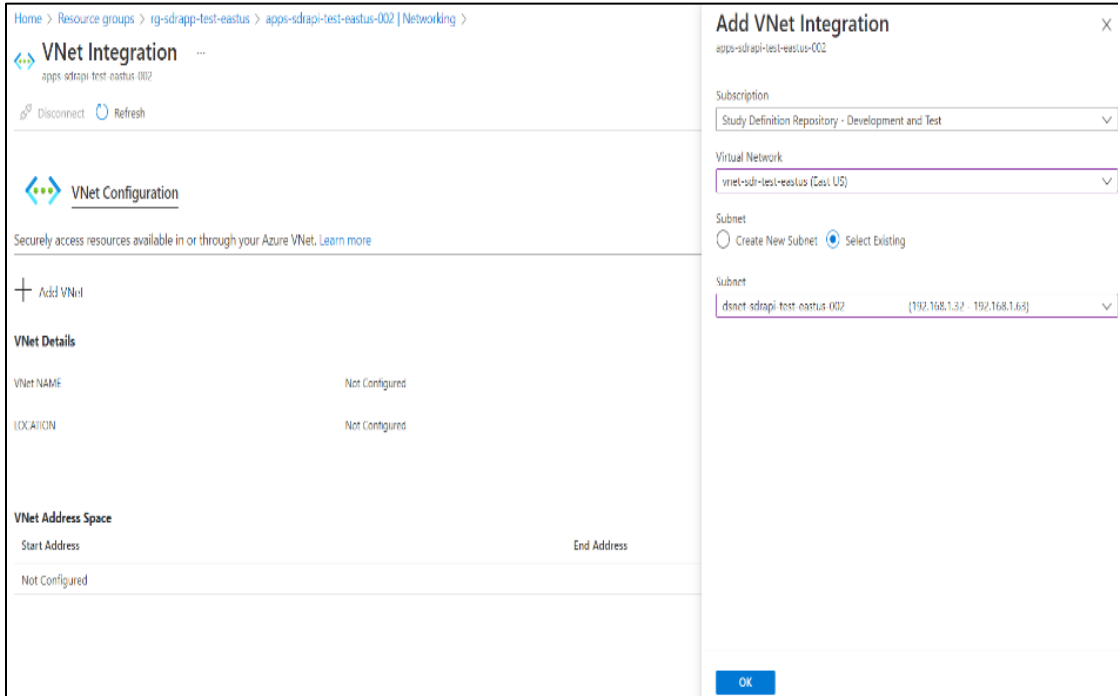
HTTP headers filter settings

X-Forwarded-Host

Add rule



Figure 33 SDR API Network Configuration



Home > Resource groups > rg-sdrapp-test-eastus > apps-sdrapi-test-eastus-002 | Networking >

### VNet Integration

apps-sdrapi-test-eastus-002

Disconnect Refresh

### VNet Configuration

Securely access resources available in or through your Azure VNet. [Learn more](#)

+ Add VNet

#### VNet Details

VNet NAME	Not Configured
LOCATION	Not Configured

#### VNet Address Space

Start Address	End Address
Not Configured	

#### Add VNet Integration

apps-sdrapi-test-eastus-002

Subscription: Study Definition Repository - Development and Test

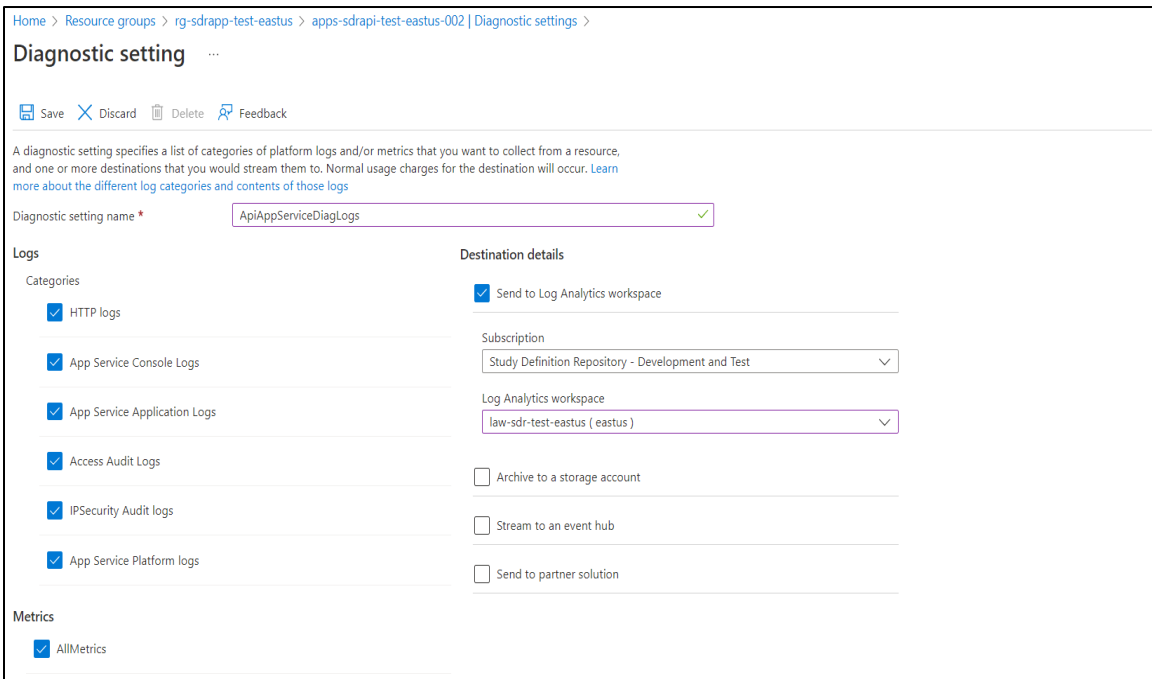
Virtual Network: vnet-sdr-test-eastus (East US)

Subnet: ☐ Create New Subnet ☒ Select Existing

Subnet: dsnet-sdrapi-test-eastus-002 (192.168.1.32 - 192.168.1.63)

OK

Figure 34 SDR API Diagnostic Configuration



Home > Resource groups > rg-sdrapp-test-eastus > apps-sdrapi-test-eastus-002 | Diagnostic settings >

### Diagnostic setting

Save Discard Delete Feedback

A diagnostic setting specifies a list of categories of platform logs and/or metrics that you want to collect from a resource, and one or more destinations that you would stream them to. Normal usage charges for the destination will occur. [Learn more about the different log categories and contents of those logs](#)

Diagnostic setting name \* ApiAppServiceDiagLogs ✓

#### Logs

Categories

- ☒ HTTP logs
- ☒ App Service Console Logs
- ☒ App Service Application Logs
- ☒ Access Audit Logs
- ☒ IP Security Audit logs
- ☒ App Service Platform logs

#### Destination details

- ☒ Send to Log Analytics workspace
- Subscription: Study Definition Repository - Development and Test
- Log Analytics workspace: law-sdr-test-eastus ( eastus )
- ☐ Archive to a storage account
- ☐ Stream to an event hub
- ☐ Send to partner solution

#### Metrics

- ☒ AllMetrics

Figure 35 SDR API Identity Configuration

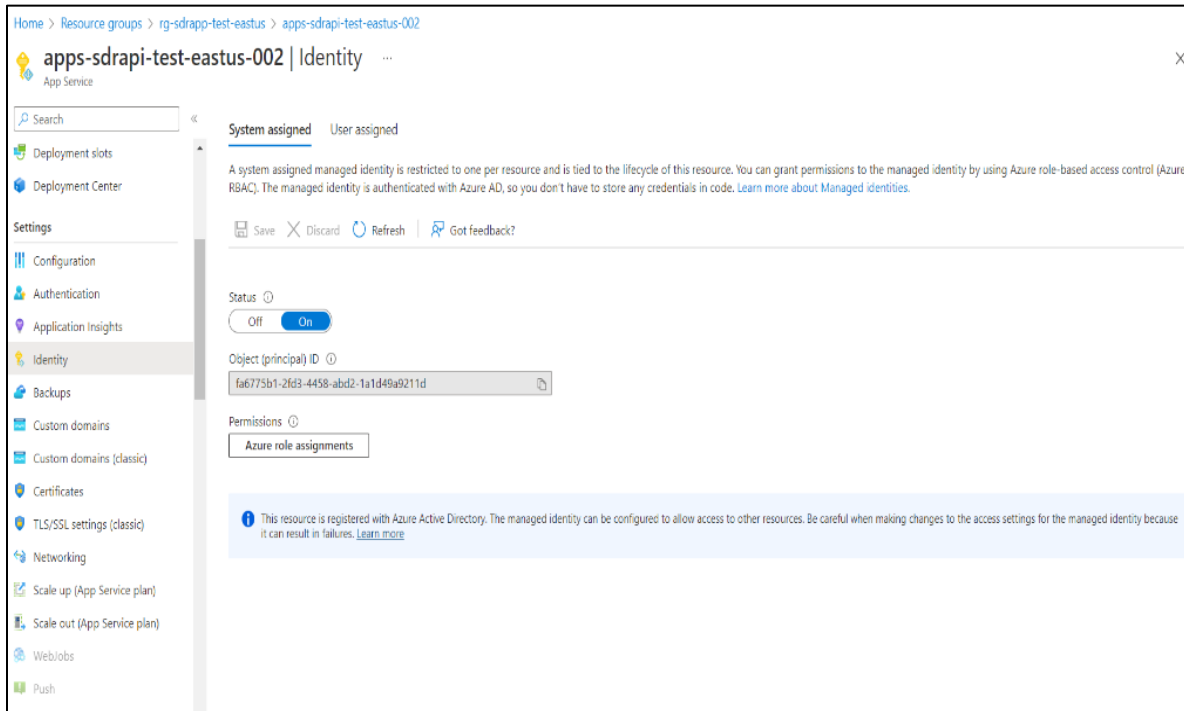
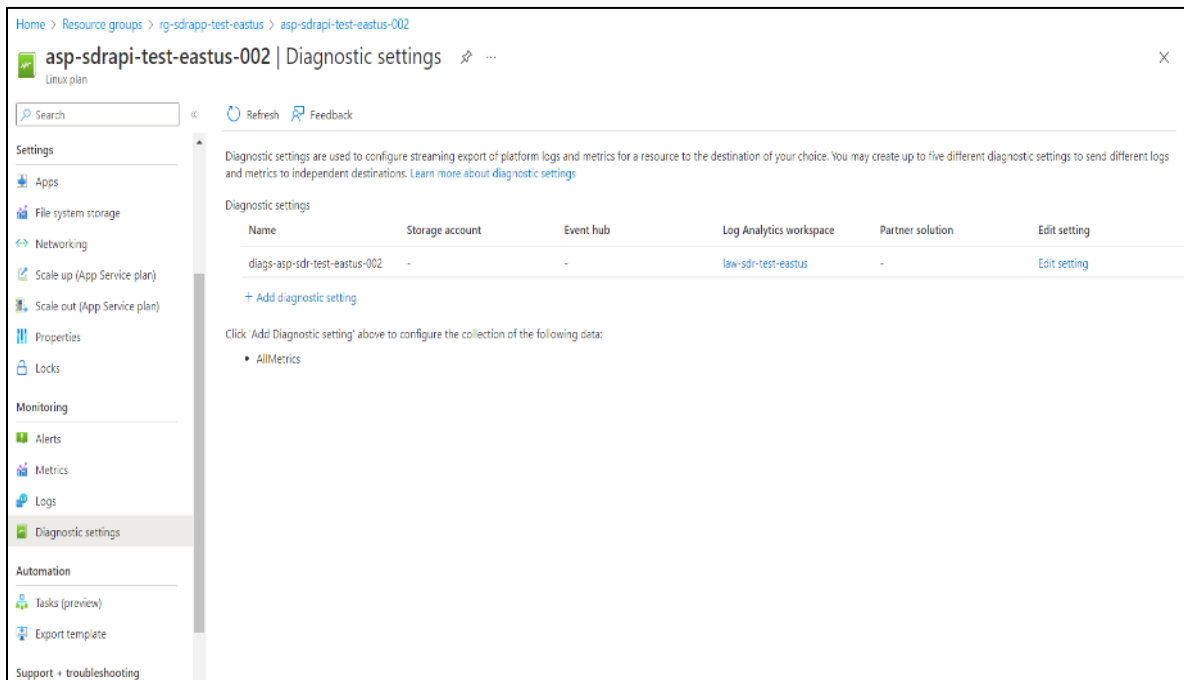


Figure 36 SDR API Diagnostic Configuration

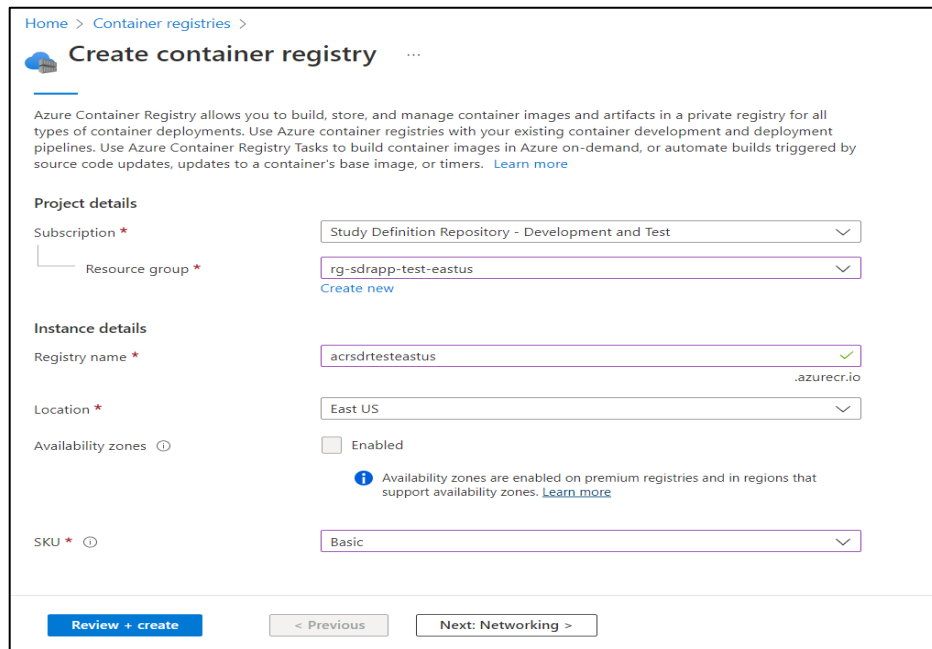


## 4.1.9 Create Azure Container Registry

### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Refer the LLD document and create an Azure Container Registry
  - Basics Details (Subscription, Resource Group)
  - Instance Details (Registry name, Location, Availability Zones, SKU)
  - Networking
  - Encryption
  - Tags

Figure 37 Container Registry Basics Details



The screenshot shows the 'Create container registry' page in the Azure Portal. The breadcrumb navigation is 'Home > Container registries >'. The page title is 'Create container registry'. Below the title is a description of Azure Container Registry and a 'Learn more' link. The form is divided into two sections: 'Project details' and 'Instance details'.

**Project details**

- Subscription \***: Study Definition Repository - Development and Test
- Resource group \***: rg-sdrapp-test-eastus (with a 'Create new' link below it)

**Instance details**

- Registry name \***: acrsdrtesteastus (with a green checkmark and '.azurecr.io' suffix)
- Location \***: East US
- Availability zones**: ☐ Enabled (with an information icon and text: 'Availability zones are enabled on premium registries and in regions that support availability zones. Learn more')
- SKU \***: Basic

At the bottom of the form are three buttons: 'Review + create' (in blue), '< Previous' (disabled), and 'Next: Networking >' (disabled).

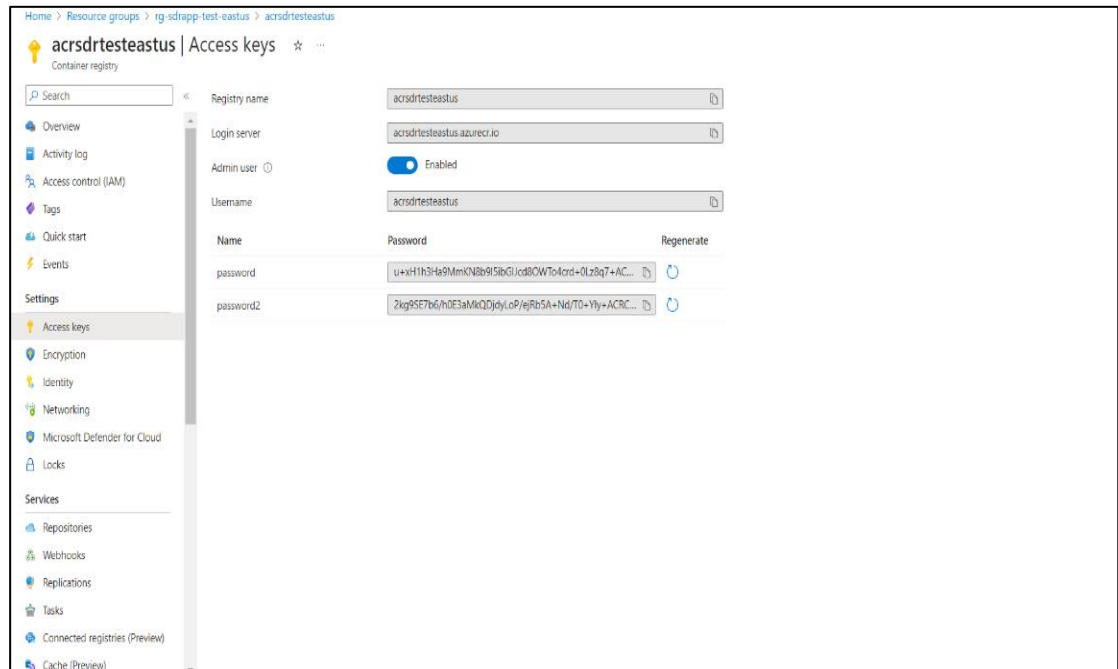
## 4.1.10 Container Registry Configuration

### STEPS

1. Login to Azure Portal.
2. Click on the Resource Groups tab.
3. Select the **sdrapp** resource group.
4. Select Container Registry and navigate to Access key and enable Admin user setting.
5. Capture Username, Password and Login Server URL.

**Note:** Please make a note of Username, Password and Login Server URL values required to be added in UI & API Application Settings.

Figure 38 Enabling Admin User Settings



## 4.2 GitHub Actions/YAML/Build & Deploy Script Updates

### PRE-REQUISITES:

- Read access to fetch Key Vault secrets in Azure Portal
- User Should have access policies set to read/retrieve secrets from Azure Key Vault in Azure Portal
- User should have Repo Admin level of Access to add/replace the GitHub secrets in GitHub.

Below are the steps to Containerize an application and deploy via ci/cd pipeline.

- Create a docker file and add it in the root folder of the GitHub repo of source code.
- Create an Azure Container Registry to store container images.
- Create GitHub Actions workflow to build the container and deploy it to App service.

The "**Dockerfile**" file is used by the docker build command to create a container image.

By selecting Create a new project and Docker is enabled, the "Dockerfile" and ".dockerignore" files are automatically generated for an ASP.NET Core Web app in Visual Studio.

### 4.2.1 GitHub Actions to deploy container image to UI App Service

The Azure Web Deploy action automates the workflow to deploy custom container image to app service.

#### PRE-REQUISITES

- Dockerfile in the project folder
- Azure Container Registry login credentials
- App Service for containers.

Below are the secrets and GitHub Actions that need to be added to GitHub workflow.

#### Secrets:

- ACR\_NAME: Azure Container Registry URL
- ACR\_USERNAME: ACR Username
- ACR\_PASSWORD: ACR Password
- AZURE\_WEBAPP\_NAME: UI App Service Name

#### Workflow Actions:

- Build and push Docker image to ACR

The actions below will be used in deployment Job to build containers using an Azure Container Registry.

Figure 39 GitHub Action for Containerized Build

```
- name: Azure Container Registry Login
  uses: Azure/docker-login@v1
  with:
    # Container registry username
    username: ${ secrets.ACR_USERNAME } # default is
    # Container registry password
    password: ${ secrets.ACR_PASSWORD } # default is
    # Container registry server url
    login-server: ${ secrets.ACR_NAME } # default is https://index.docker.io/v1/
- run: |
  cp '${ github.workspace }/Dockerfile' '${ github.workspace }/SDR-WebApp/dist/SDR-WebApp'
  cp '${ github.workspace }/nginx.conf' '${ github.workspace }/SDR-WebApp/dist/SDR-WebApp'
  cd ${ github.workspace }/SDR-WebApp/dist/SDR-WebApp
  ls
  docker build . -t ${ secrets.ACR_NAME }/sdrui:latest
  docker push ${ secrets.ACR_NAME }/sdrui:latest
```

- Deploy to App Service

The below GitHub action will be used in deployment Job to deploy container to App Service.

```
- uses: azure/webapps-deploy@v2
  with:
    app-name: ${ secrets.AZURE_WEBAPP_NAME }
    images: '${ secrets.ACR_NAME }/sdruibuild:latest'
```

#### 4.2.2 GitHub Actions to deploy container image to API App Service

The Azure Web Deploy action automates the workflow to deploy custom container image to app service.

##### PRE-REQUISITES

- Dockerfile in the project folder
- Azure Container Registry login credentials
- App Service for containers.

Below are the secrets and GitHub Actions that need to be added to GitHub workflow.

##### Secrets:

- ACR\_NAME: Azure Container Registry URL
- ACR\_USERNAME: ACR Username
- ACR\_PASSWORD: ACR Password
- AZURE\_WEBAPP\_NAME: API Web App Name

##### Workflow Actions:

- Build and push Docker image to ACR.

The actions below will be used in deployment Job to build containers using an Azure Container Registry.

```
- name: Azure Container Registry Login
  uses: Azure/docker-login@v1
  with:
    # Container registry username
    username: ${ secrets.ACR_USERNAME } # default is
    # Container registry password
    password: ${ secrets.ACR_PASSWORD } # default is
    # Container registry server url
    login-server: ${ secrets.ACR_NAME } # default is https://index.docker.io/v1/
- run: |
  cp '${ github.workspace }/Dockerfile' '${ github.workspace }/src/TransCelerate.SDR.WebApi/bin/Release/net6.0/publish/'
  cd ${ github.workspace }/src/TransCelerate.SDR.WebApi/bin/Release/net6.0/publish/
  ls
  docker build . -t ${ secrets.ACR_NAME }/sdrapibuild:latest
  docker push ${ secrets.ACR_NAME }/sdrapibuild:latest
```

- Deploy to App Service

The below GitHub action will be used in deployment Job to deploy container to App Service.

```
- uses: azure/webapps-deploy@v2
  with:
    app-name: ${ secrets.AZURE_WEBAPP_NAME }
    images: '${ secrets.ACR_NAME }/sdrapibuild:latest'
```

The below GitHub action will be used in build and deployment Job to Publish Function App folder.

```
- name: Zip publish files
  shell: pwsh
  # Zip the Function App build artifact publish folder
  run: |
    Compress-Archive -Path "${ github.workspace }/src/TransCelerate.SDR.AzureFunctions/bin/Release/net6.0/publish/**" -DestinationPath "${ github.workspace }/FunAppPublish.zip"
```

```
- name: 'Publish Function App Artifact: Artifact'
  uses: actions/upload-artifact@v2
  # Publish the Function App Build artifact in github
  with:
    path: '${ github.workspace }/FunAppPublish.zip'
    name: FunctionAppBuild-Artifact
```

```
- name: Azure Functions Action
  uses: Azure/functions-action@v1.4.7
  with:
    # Name of the Azure Function App
    app-name: ${ secrets.AZURE_FUNCTIONAPP_NAME } # Replace with Function app name
    # Path to package or folder. *.zip or a folder to deploy
    package: '${ github.workspace }/FunAppPublish.zip'
```

### 4.2.3 GitHub Secrets used in Workflow

The Below mentioned additional GitHub Secrets need to be added in UI and API Repo Secrets.

#### STEPS:

1. Login to azure portal, select resource group section.
2. Navigate to the deployed Azure Container Registry and copy the Login Server name from Overview blade. This will be the ACR\_NAME for both ddf-sdr-ui & ddf-sdr-api repo secrets.

3. Navigate to the deployed Azure Container Registry and copy the Username & Password from Access Keys blade. This will be the ACR\_USERNAME & ACR\_PASSWORD for both ddf-sdr-ui & ddf-sdr-api repo secrets.
4. Navigate to the deployed Function App Service for SDR API and copy the name from Overview blade. This will be the AZURE\_FUNCTIONAPP\_NAME for ddf-sdr-api repo secrets.
5. Login to GitHub and navigate to ddf-sdr-api -> Settings-> Secrets -> Actions and add/replace values for ACR\_NAME, ACR\_USERNAME, ACR\_PASSWORD, AZURE\_FUNCTIONAPP\_NAME by clicking on update.
6. Login to GitHub and navigate to ddf-sdr-ui -> Settings -> Secrets -> Actions and add/replace values for ACR\_NAME, ACR\_USERNAME, ACR\_PASSWORD by clicking on Update.
7. Update the Deployment yml script with Azure Container Registry in both ddf-sdr-api & ddf-sdr-ui repos.

Figure 40 Container Registry GitHub Secrets for API




















 ACR_NAME	Updated on Dec 29, 2022		
 ACR_PASSWORD	Updated on Dec 29, 2022		
 ACR_USERNAME	Updated on Dec 29, 2022		
 AZURE_FUNCTIONAPP_NAME	Updated on Dec 29, 2022		

Figure 41 Container Registry GitHub Secrets for UI App Service

 ACR_NAME	Updated on Dec 29, 2022		
 ACR_PASSWORD	Updated on Dec 29, 2022		
 ACR_USERNAME	Updated on Dec 29, 2022	