Support GraphQL Federation for GraphQL APIs in WSO2 API-M

May 14th, 2024

Introduction

What is Federation

GraphQL Federation is an architecture model that allows multiple GraphQL services, known as subgraphs or federated services, to be combined into a single schema or API.

This unified data graph enables clients to query and receive responses from multiple services using a single request.

Benefits of GraphQL Federation

- Increased Developer Productivity
- Increased flexibility
- Scalability
- Better separation of concerns

The Significance of GraphQL Federation in the API Management Gateway Layer

- This will enhance developer experience by abstracting away the complexities associated with setting up and managing federated GraphQL services
- Developers gain centralized control over their entire API ecosystem.
- Providing support for federation in the API management layer ensures consistent QoS.
- Distinguishing WSO2 API-M from competitors that lack GraphQL federation capabilities, strengthening the position of the platform

Competitor Analysis

API Management Platform	Federation Support	Notes
Tyk	Yes , Mentioned in Documentation	Subgraph Creation using GUI (should include federation directive)
Hasura	Yes , Mentioned in Documentation	supports the Apollo Federation v1 spec
Apigee	Federation support is not mentioned in the documentation	Article that integrates Apigee with StepZen (a GraphQL server) is available to enable federation
IBM	Federation support is not mentioned in the documentation	Article that integrates IBM with StepZen (a GraphQL server) is available to enable federation

Proposed Solution and Implementation

Proof of Concept

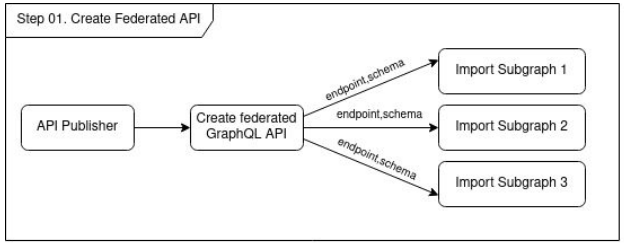
- To Implement GraphQL federation using Java technologies
- graphql-orchestrator-java library Github :
  - https://github.com/graph-quilt/graphql-orchestrator-java
  - Documentation : https://graph-quilt.github.io/graphql-orchestrator-java/

- **Scenarios Implemented**

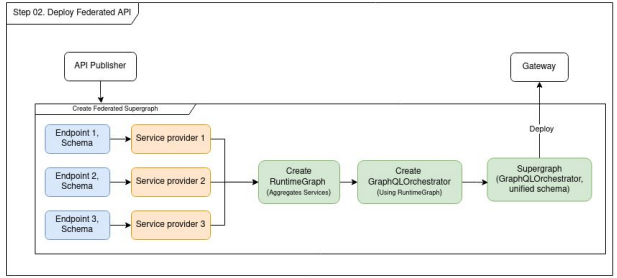
1. Creation of federated graph by federating Subgraph endpoints
2. Resolving a Simple Query
3. Resolving a Complex Query - Nested
4. Resolving a Complex Query - multiple queries in the same level

Proposed Solution

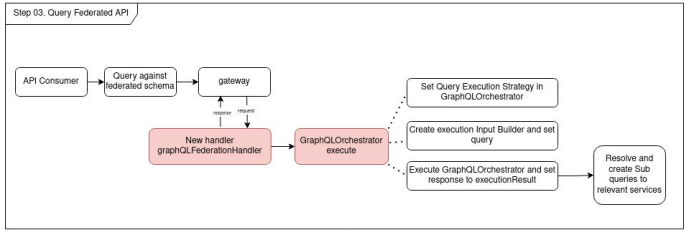
- Let API developers import multiple subgraphs providing SDL and endpoint



- During deploying create service providers , combine to create runtime graph which create graphqlOrchestrator object that will be stored along with the unified SDL .



- Implement a new handler to handle the query execution with multiple backends.



Continuation

- Mutations

```
// Add Pet
ExecutionInput addPetEI = ExecutionInput
        .newExecutionInput()
        .query('''
            mutation AddNewPetAndUser($newpet: InputPet!, $newuser: NewUserInput!) {
                addPet(pet: $newpet)
                @merge (if: true) {
                    id name
                }
                addUser(newUser: $newuser) {
                    id firstName
                }
            }
        ''')
        .variables(ImmutableMap.of("newpet", newpetMap, "newuser", newUserMap))
        .build()
```

Continuation

- Subscriptions - Not supported yet

  https://github.com/graph-quilt/graphql-orchestrator-java/issues/167

- Query Execution Flow
  - Parallel network calls to Data Providers
  - Batching queries

  https://graph-quilt.github.io/graphql-orchestrator-java/key-concepts/graphql-query-execution/

- QoS for the federated API
  - Provided for Supergraphs - traffic shaping,security

Orbit bundle creation

- Added Maven Dependencies and created an orbit Bundle
  https://github.com/wso2/orbit/pull/1109

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">

    <groupId>org.wso2.orbit.graphQLFederation</groupId>
    <artifactId>graphQL-federation</artifactId>
    <version>5.0.26.wso2v1</version>

    <modelVersion>4.0.0</modelVersion>
    <packaging>bundle</packaging>
    <name>WSO2 Carbon Orbit - GraphQL Federation</name>
    <description>
        This bundle will export packages from graphQL java Orchestrator
    </description>
    <url>http://wso2.org</url>

    <distributionManagement>
        <repository>
            <id>wso2.releases</id>
            <name>WSO2 internal Repository</name>
            <url>https://maven.wso2.org/nexus/content/repositories/releases/</url>
        </repository>
    </distributionManagement>

    <dependencies>
        <dependency>
            <groupId>com.intuit.graphql</groupId>
            <artifactId>graphql-orchestrator-java</artifactId>
            <version>5.0.26</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/org.eclipse.emf/org.eclipse.emf.ecore -->
        <dependency>
            <groupId>org.eclipse.emf</groupId>
            <artifactId>org.eclipse.emf.ecore</artifactId>
            <version>2.20.0</version>
        </dependency>
```

New Handler Creation and Engaging

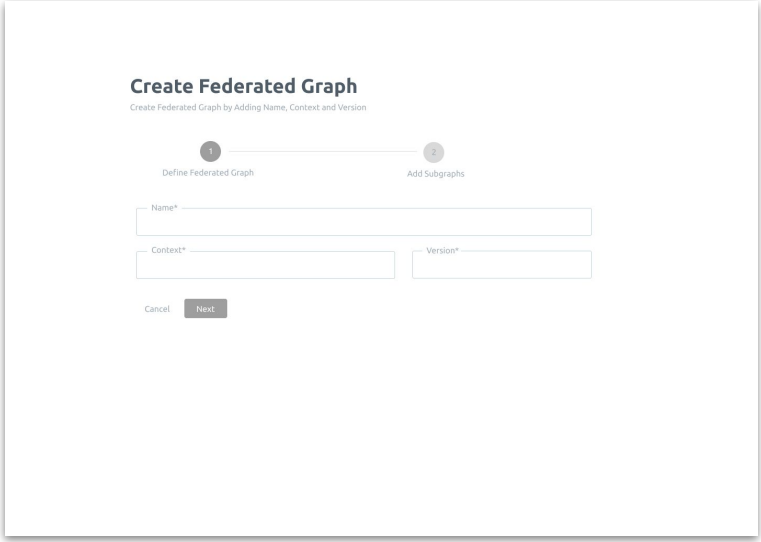
- Created New Handler for Federation and engaged it

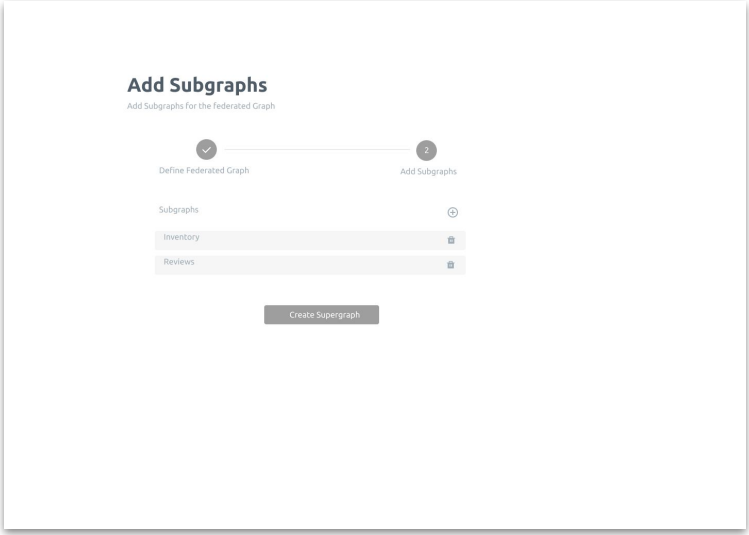
```java
package org.wso2.carbon.test;
import ...

public class customAuthenticationHandler extends AbstractHandler {

    private static final HttpClient httpClient = HttpClients.createDefault();
    private static final ObjectMapper MAPPER = new ObjectMapper();
    public static final String GRAPHQL_PAYLOAD = "GRAPHQL_PAYLOAD";
    public static final String HTTP_SC = "HTTP_SC";
    public static final String NO_ENTITY_BODY = "NO_ENTITY_BODY";
    public static final String APPLICATION_JSON_MEDIA_TYPE = "APPLICATION_JSON_MEDIA_TYPE";

    public boolean handleRequest(MessageContext messageContext) {
        try {
            //We have to keep this RuntimeGraph in gateway internal data holder
            RuntimeGraph runtimeGraph = getRuntimeGraph();
            ExecutionStrategy queryExecutionStrategy = new AsyncExecutionStrategy();
            GraphQLOrchestrator.Builder builder = GraphQLOrchestrator.newOrchestrator();
            builder.runtimeGraph(runtimeGraph);
            builder.queryExecutionStrategy(queryExecutionStrategy);
            GraphQLOrchestrator graphQLOrchestrator = builder.build();

            //Build ExecutionInput
            ExecutionInput.Builder eiBuilder = ExecutionInput.newExecutionInput();
            String payload = messageContext.getProperty(GRAPHQL_PAYLOAD).toString();
            eiBuilder.query(payload);
            ExecutionInput executionInput = eiBuilder.build();

            //Execute GraphQL Query
            Map<String, Object> executionResult = graphQLOrchestrator.execute(executionInput).get().toSpecification();
            constructResponse(messageContext, toJson(executionResult));
        } catch (Exception e) {
            System.out.println("Error handling request: " + e.getMessage());
        }
        return false;
    }
```

UI Frames for Publisher Portal

- Created UI Frames for the publisher Portal

  https://www.figma.com/design/4FuCF06X16LuJ7ArTG2O0I/GraphQL-Federation---UI?node-id=0%3A1&t=HuJlcC4ev2Kxo00I-1





Thanks!

wso2.com