

**Publisher APIs** are built over **PROVIDER APIs** so that the **JSON** structure and field values of **PROVIDER API** request and response can be modified as per the end user requirement at **IRIS layer**.

1. From the provider APIs **api-docs**, take the **swagger JSON**, **modify as per your requirement of JSON structure** and **save it**.

```

1 {
2   "swagger" : "2.0",
3   "info" : {
4     "description" : "AcBalance",
5     "version" : "v1.0.0",
6     "title" : "AcBal"
7   },
8   "host" : "localhost:9089",
9   "basePath" : "/api/v1.0.0/",
10  "tags" : [ ],
11  "schemes" : [ "http", "https" ],
12  "security" : [ {
13    "basicAuth" : [ ]
14  } ], {
15    "apiKey" : [ ]
16  } ],
17  "paths" : {
18    "/party/accounts/{accountId}" : {
19      "get" : {
20        "tags" : [ ],
21        "operationId" : "getAcBal",
22        "produces" : [ "application/json" ],
23        "parameters" : [ {
24          "name" : "accountId",
25          "in" : "path",
26          "description" : "Identifier of the account. Often ref",
27          "required" : true,
28          "type" : "string"

```

Here, the **path** is changed only.



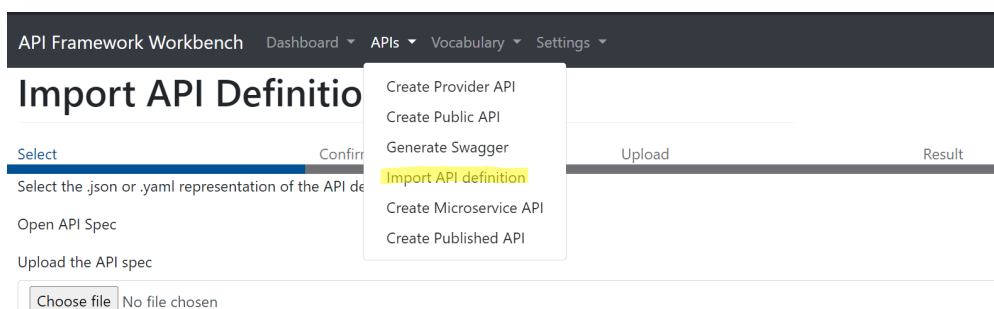
mb-api-publish-v1.0.0-swagger.json

### Modified JSON File:

2. Check the correctness of that **swagger JSON** via the open-source swagger editor, the **URL** of which is given below. If any structural errors are thrown, resolve it.

<https://editor-next.swagger.io/>

3. Import this swagger definition in **workbench** using the **"Import API definition"** option and download the publisher zip file.



## Import API Definition

[Previous](#)

Select	Confirm	Upload	Result
File	Type	Size	Last Modified
mb-api-publish-v1.0.0-swagger.json	application/json	10.2 KB	17/02/2024

[Upload](#)

Select	Confirm	Upload	Result
API Title	AcBal		
API Version	v1.0.0		
Service Id			

[Upload & Generate!](#)

## Generating Service XML

### Process complete

Click the link to download the generated artefacts

[Download](#)

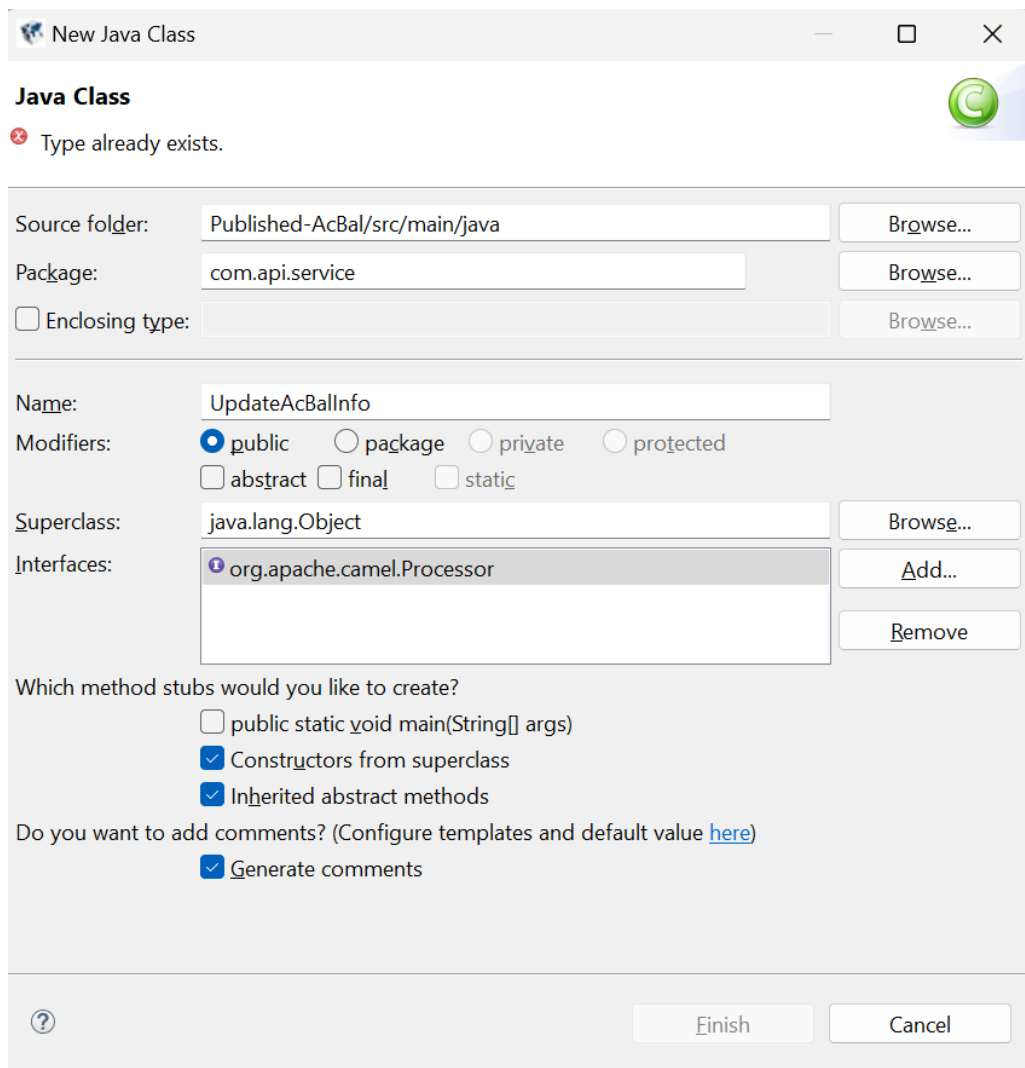
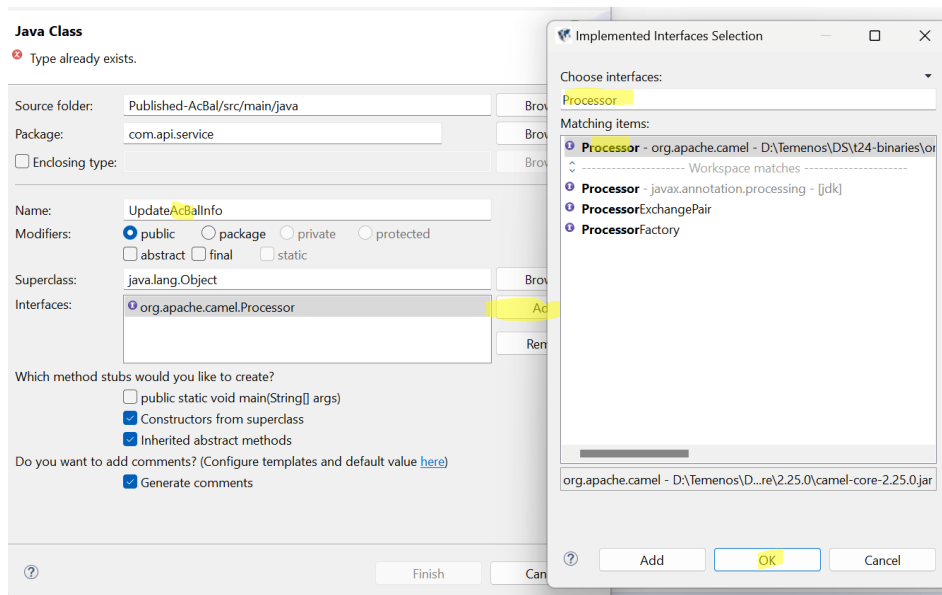
Download the zip file.

4. Create a maven project with **archetype-service catalog** and import the zip file downloaded in the previous step.

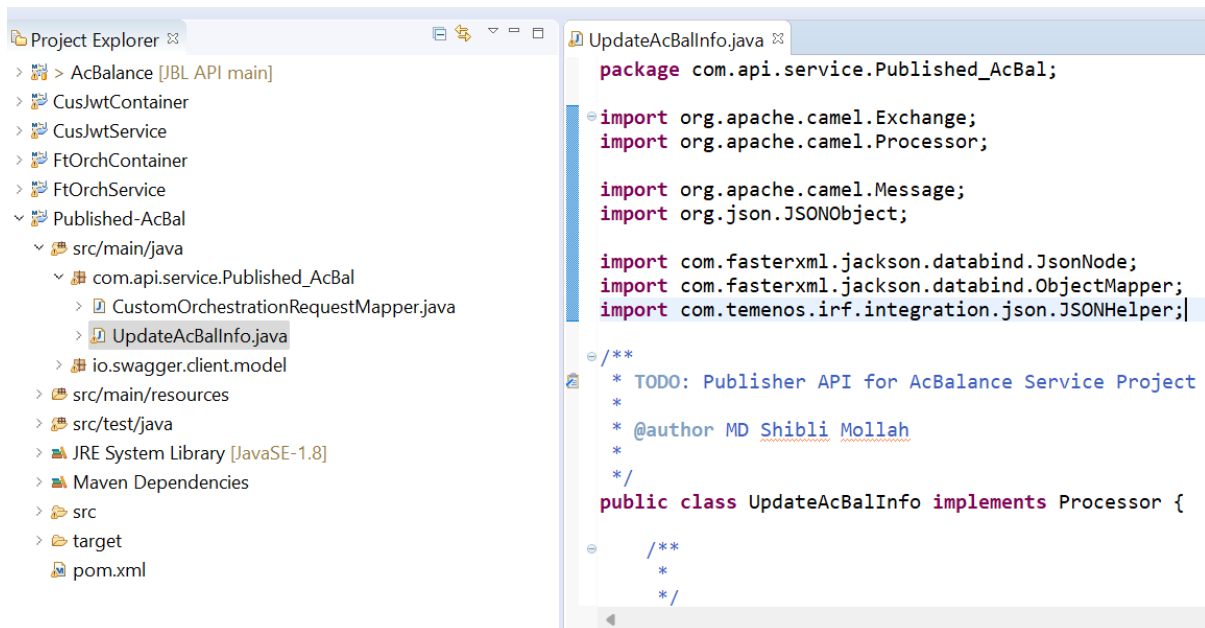
- published-myAcBalance
  - src/main/java
  - src/main/resources
    - api-docs
    - services
    - src
  - src/test/java
  - JRE System Library [JavaSE-1.8]
  - Maven Dependencies
  - src
  - target
  - pom.xml

### 8. Navigate to **publisherProject - Published-AcBal**

<<**publisherProject**>>\src\main\java\com.api.service  
package directory, right click and select **New -> Java -> Class** and enter the class name as <<AnyRelevantName>>Mapper and add the interface named "Processor" under Interfaces.



11. The mapper class will be generated as shown below.



The **process()** method in the mapper must have the code logic to transform the provider API to publisher API and vice versa.

The code logic must be in such a way that the fields present in the **JSON response** body of the **provider** API are mapped to the fields defined in the swagger of **published** API using the **published** API swagger classes generated.

### Mapper class for ENQUIRY based API:

For **ENQUIRY** API, we will require only a **response** mapper since we **do not send** a request body.

Below are the steps to create a Response Mapper for an ENQUIRY API based on ACCOUNT application.

**a.** As mentioned in **step 10**, create a **mapper class** with a relevant name (**UpdateAcBalInfo**) and Processor interface. In the generated mapper java file, **comment the constructor** (if not required). Only the **process** method will be used.

**b.** To map the fields of the **provider API**, we must first receive the header and JSON body content (containing header and body section of JSON response) of the provider API after ensuring the http response code is **200**. Below is the corresponding code line. On confirming it is a successful response, we obtain the JSON response – header and body section.

```
public void process(Exchange exchange) throws Exception {
    // TODO Auto-generated method stub
    String HttpStatusCode =
exchange.getIn().getHeaders().get("CamelHttpStatusCode").toString();
    Message in = exchange.getIn();
    String response = (String) in.getBody(String.class);
    JSONObject = new JSONObject(response);
    JSONObject jsonHeader = jsonObject.getJSONObject("header");
    String updateResponse = null;
```

```

        if (HttpServletResponse.getStatusCode().equals("200")) {

            JSONObject jsonBody =
                jsonObject.getJSONObject("body").getJSONObject(0);

            Receive the header and body content of provider JSON response.

            String responseCode = null;
            String accountNo = null;
            String alternateAccountNo = null;
            String currency = null;
            String balance = null;

            try {
                responseCode = jsonBody.getString("response");
                accountNo = jsonBody.getString("accountId");
                alternateAccountNo = jsonBody.getString("accountIBAN");
                currency = jsonBody.getString("currency");
                balance = jsonBody.getString("availableBalance");
            } catch (Exception e) {}

            jsonBody.remove("accountId");
            jsonBody.remove("accountIBAN");
            // jsonBody.remove("currency");
            jsonBody.remove("availableBalance");

            jsonBody.put("accountNo", accountNo);
            jsonBody.put("alternateAccountNo", alternateAccountNo);
            jsonBody.put("balance", balance);

            jsonHeader.remove("audit");
            jsonHeader.remove("page_start");
            jsonHeader.remove("page_token");
            jsonHeader.remove("total_size");
            jsonHeader.remove("page_size");
            jsonHeader.remove("status");

            jsonHeader.put("statusCode", responseCode);
            updateResponse = "{\"header\": " + jsonHeader + ", \"body\": " + jsonBody
+ "}\"";
        }

        ObjectMapper om = JSONHelper.createObjectMapper();
        JsonNode modifiedResponse = om.readTree(updateResponse);
        in.setBody(modifiedResponse);
    }

```

### Sample Mapper Class:



UpdateAcInfo.java



UpdateAcBallInfo.java



UpdateAcBallInfo.java

c. Edit the **service.xml** of the publisher API by adding this **ResponseMapper** ([UpdateAcBallInfo](#)) as a process after redirecting it to the provider API.

```

party-v1.0.0-published-myacbal-service-v1.0.0.xml
<camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published">
  <onException>
    <exception>java.lang.Exception</exception>
    <handled>
      <constant>true</constant>
    </handled>
    <process ref="exceptionHandler"/>
  </onException>
  <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1.0.0-sw">
    <rest path="/v1.0.0/party/v1.0.0" produces="application/json" id="party-v1.0.0.ser">
      <get uri="/party/accounts/{accountId}" id="getAcBal">
        <param name="accountId" type="path" required="true"/>
        <!-- Publisher API Uri -->
        <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
      </get>
    </rest>
    <route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
      <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
      <!-- Provider API Uri -->
      <to uri="direct-vm:party-accounts.v1.0.0.getAcBal"/>
      <process ref="UpdateAcBalInfo"/>
    </route>
    <route id="direct.mockResponder">
      <from uri="direct:mockResponder"/>
      <process ref="mockResponder"/>
    </route>
  </camelContext>
  <bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo"/>
</beans>

```

```

<get uri="/party/accounts/{accountId}" id="getAcBal">
  <param name="accountId" type="path" required="true"/>
  <!-- Publisher API Uri -->
  <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
</get>
</rest>
<route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
  <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
  <!-- Provider API Uri -->
  <to uri="direct-vm:party-accounts.v1.0.0.getAcBal"/>
  <process ref="UpdateAcBalInfo"/>
</route>
<route id="direct.mockResponder">
  <from uri="direct:mockResponder"/>
  <process ref="mockResponder"/>
</route>

```

**Add the bean: OPTIONAL**

```
<bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo"/>
```



party-v1.0.0-published-myacbal-service-v1.0.0.xml



party-v1.0.0-published-myacbal-service-v1.0.0.xml

**Special Note:** We must redirect the URI of the other versions of the service xml's of the Publisher API by changing like this, otherwise war file deployment will be **failed**.

```

<route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
  <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
  <!-- OWN PUBLISHER API Uri -->
  <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
</route>

```

```

party-v1.0.0-published-myacbal-service-v1.0.xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans/spring-beans.xsd
         http://camel.apache.org/schema/spring
         http://camel.apache.org/schema/spring/camel-spring.xsd">
  <camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published-myAcBal.service.v1.0">
    <onException>
      <exception>java.lang.Exception</exception>
      <handled>
        <constant>true</constant>
      </handled>
      <process ref="exceptionHandler"/>
    </onException>
    <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1.0-swagger.json" bindingMode="auto">
      <rest path="/v1.0/party/v1.0.0" produces="application/json" id="party-v1.0.0.service.restlet">
        <get uri="/party/accounts/{accountId}" id="getAcBal">
          <param name="accountId" type="path" required="true"/>
          <to uri="direct-vm:party-v1.0.0.v1.0.getAcBal"/>
        </get>
      </rest>
      <route id="direct-vm.party-v1.0.0.v1.0.getAcBal">
        <from uri="direct-vm:party-v1.0.0.v1.0.getAcBal"/>
        <!-- OWN PUBLISHER API Uri -->
        <to uri="direct-vm:party-v1.0.0.v1.0.getAcBal"/>
      </route>
    </camelContext>
  </beans>

```

```

party-v1.0.0-published-myacbal-service-v1.xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
         http://www.springframework.org/schema/beans/spring-beans.xsd
         http://camel.apache.org/schema/spring
         http://camel.apache.org/schema/spring/camel-spring.xsd">
  <camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published-myAcBal.service.v1">
    <onException>
      <exception>java.lang.Exception</exception>
      <handled>
        <constant>true</constant>
      </handled>
      <process ref="exceptionHandler"/>
    </onException>
    <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1-swagger.json" bindingMode="auto">
      <rest path="/v1/party/v1.0.0" produces="application/json" id="party-v1.0.0.service.restlet">
        <get uri="/party/accounts/{accountId}" id="getAcBal">
          <param name="accountId" type="path" required="true"/>
          <to uri="direct-vm:party-v1.0.0.v1.getAcBal"/>
        </get>
      </rest>
      <route id="direct-vm.party-v1.0.0.v1.getAcBal">
        <from uri="direct-vm:party-v1.0.0.v1.getAcBal"/>
        <!-- OWN PUBLISHER API Uri -->
        <to uri="direct-vm:party-v1.0.0.v1.0.getAcBal"/>
      </route>
    </camelContext>
  </beans>

```



party-v1.0.0-published-myacbal-service-v1.0.xml



party-v1.0.0-published-myacbal-service-v1.xml

## 11. Maven Install the **publisher API** project(s).



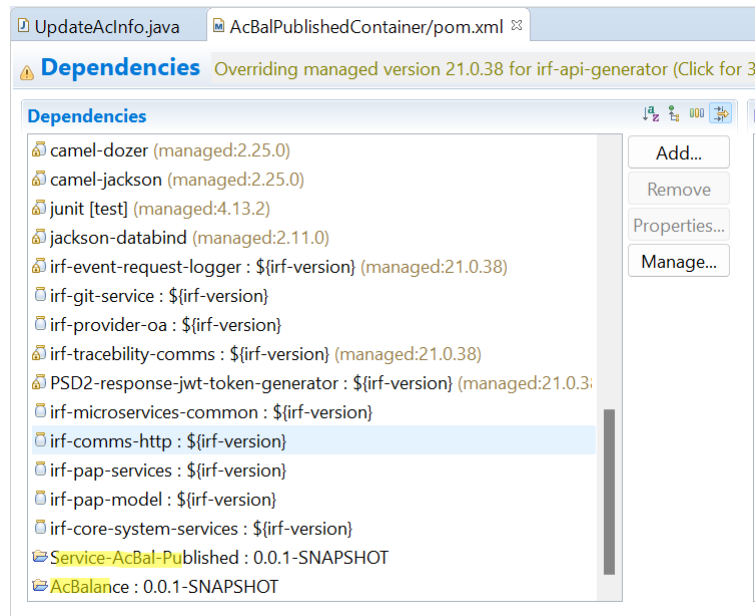
Published-AcBal-0.0.1-SNAPSHOT.jar



Published-AcBal-0.0.1-SNAPSHOT.jar



12. Include the **publisher API** and **provider API** project(s) dependency in the **container** project's pom.xml and perform maven install.



```
<dependency>
  <groupId>com.api.service</groupId>
  <artifactId>Service-AcBal-Published</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
<dependency>
  <groupId>AcBal</groupId>
  <artifactId>AcBalance</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

14. Create a new bean with **any relevant name** for "id" attribute in **applicationContext.xml** of the **container project** for the mapper class created. Enable Basic Auth/JWT required.

```
<!-- Response mapper added -->
  <bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo">
</bean>
```

```
<!-- Response mapper added -->
<bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo">
</bean>

<!-- <bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.NullBean" /> -->

<!-- <bean id="t24SecurityFilter" class="com.temenos.irf.security.t24.T24SpringSecurityContextFilter" /> -->

<!-- Comment the above bean with id t24SecurityFilter and uncomment below bean for Security Filter -->
<!--<bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.T24Security" /> -->

<!-- Comment the above bean with id t24SecurityFilter and uncomment below bean for Basic Authentication -->
<bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.T24BasicAuthenticationCheck"/>

<bean id="serviceLocatorProperties" class="com.temenos.irf.config.StandardPropertyReader">
  <property name="path" value="classpath:/irf-config/service-locator.properties" />
</bean>

<bean id="mockDataMgmtProcessor" class="com.temenos.irf.mock.MockDataMgmtProcessor">
  <!-- basePath should be set to the base Directory where mockFiles folder is placed. -->
  <property name="basePath" value="classpath:/irf-config/"></property>
</bean>
```



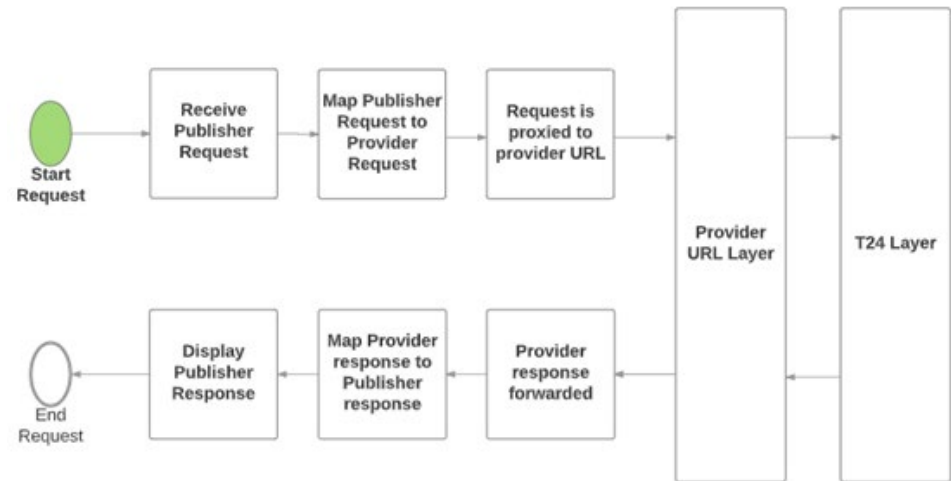
SAVE it.



PublishedAcBalContainer.war



PublishedAcBalContainer.war



#### 14. Deploy war in application server (or) jetty server.

This PC > Local Disk (D:) > Temenos > jboss > standalone > deployments > Search deployments				
Sort View ...				
	Name	Date modified	Type	Size
	PublishedAcBalContainer.war	18/02/2024 3:25 PM	WAR File	142,515 KB
	PublishedAcBalContainer.war.deployed	18/02/2024 3:25 PM	DEPLOYED File	1 KB

15. APIs can now be accessed with structure of **published API JSON** and the corresponding response is now formed as per requirement.

API Framework Workbench

Dashboard APIs Vocabulary Settings

mb-api-v1.0.0-swagger v1.0.0

API docs

party-accounts-mb-api

GET /v1.0.0/party/accounts/{accountId}

published-myAcBal-v1.0.0-swagger v1.0.0

API docs

party-v1.0.0-published-myAcBal

GET /v1.0.0/party/v1.0.0/party/accounts/{accountId}

Below screenshots show the JSON requests and responses for the examples explained in this document.

HTTP ... / http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accou...

GET http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accounts/120456 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (9) Test Results 200 OK 23.65 s 565 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "header": {
3     "status": "success"
4   },
5   "body": {
6     "accountId": "120456",
7     "currency": "USD",
8     "availableBalance": 2047214
9   }
10 }
```



AcBalanceContainer\_response.json

HTTP ... / http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accou...

GET http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accounts/120855 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (9) Test Results 200 OK 1877 ms 648 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "header": {
3     "status": "success"
4   },
5   "body": {
6     "accountId": "120855",
7     "alternateAccountNo": "GB04DEM060161300120855",
8     "balance": "81080",
9     "accountNo": "120855",
10    "currency": "USD",
11    "availableBalance": 81080
12  }
13 }
```

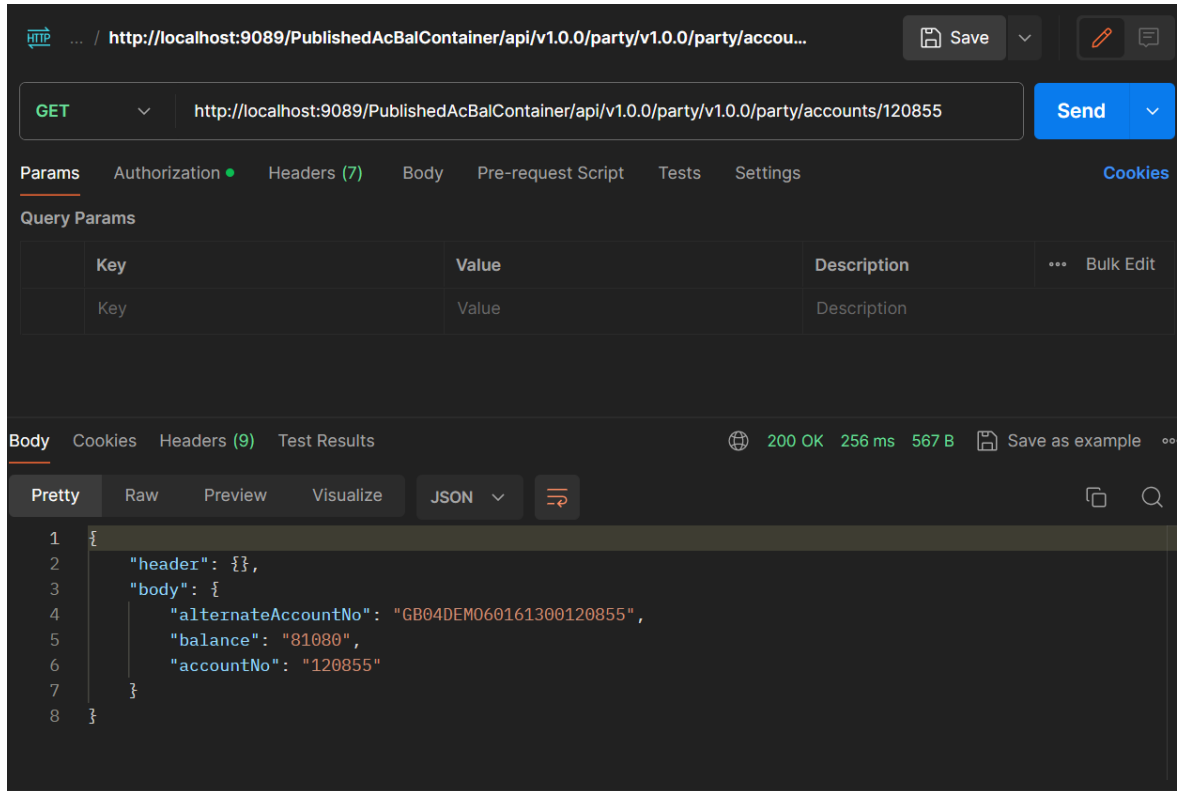


UpdatedAcBalanceContainer\_response.json

After changing the Java Code to meet our expectation.



UpdateAcBallInfo.java



The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accou...`
- Method:** `GET`
- Path:** `http://localhost:9089/PublishedAcBalContainer/api/v1.0.0/party/v1.0.0/party/accounts/120855`
- Status:** `200 OK`, `256 ms`, `567 B`
- Body (JSON):**

```
{
  "header": {},
  "body": {
    "alternateAccountNo": "GB04DEM060161300120855",
    "balance": "81080",
    "accountNo": "120855"
  }
}
```



Final\_Published\_response.json