



Tech Talk

Streamline Your Architecture with Red Hat Camel and AMQ Streams

Red Hat Team

Date 12-Aug-2024

# Table of Contents

|  |           |
|--|-----------|
| <b>1. Update the DevSpace plugin</b>   | <b>2</b>  |
| 1.1. Login to the openshift console using admin credentials.   | 2         |
| 1.2. Go to the Installed OperatorPage and look for Red Hat OpenShift Dev Spaces  | 2         |
| 1.3 Update the devspace cheCluster Yaml configurations.  | 3         |
| 1.4. Update the pluginRegistry configurations.   | 3         |
| 1.5. Save the configuration and verify the changes after reload.   | 4         |
| <b>2 Delete the previous workspace from the Devspace</b>   | <b>4</b>  |
| <b>3 Import the DeveloperHub template for this demo.</b>   | <b>5</b>  |
| 3.1. Go to the Create Software Template section  | 6         |
| 3.2. Register the new template   | 6         |
| 3.3. Finish the import   | 7         |
| <b>4 Create the Catalog from template</b>  | <b>7</b>  |
| 4.1 Go to the Template Create section  | 7         |
| 4.2. Choose the template "Create a Camel app for OpenShift"  | 8         |
| 4.3 Add the details, Name and Cluster Domain main copy from the browser url as shown below   | 8         |
| <b>5 Open the new workspace from the developer Hub.</b>  | <b>10</b> |
| 5.1 After waiting a few minutes for OpenShift Dev Spaces to finish setting up your workspace, You click the button Yes, I trust the authors. | 11        |
| 5.2. Verify all the extension has been installed correctly.  | 12        |
| <b>6 Module 1</b>  | <b>12</b> |
| 6.1 Update the application.properties  | 12        |
| 6.2 Commit the Changes   | 14        |
| 6.2.1. You click on the Source Control icon located in the left menu.  | 14        |
| 6.2.2. Then, you enter the commit message "updated application properties" and click on the Commit button to finalize your changes.          | 14        |
| 6.2.3. In the pop-up window that follows, you click Yes to stage your changes.   | 14        |
| 6.2.4. Finally, you click on the Sync Changes button.  | 15        |
| 6.2.5. In the pop-up that follows, you click OK to push your changes and complete the process.   | 15        |
| 6.3 Verify the pipeline  | 16        |
| 6.4 Open the application   | 17        |
| <b>7 Module 2</b>  | <b>17</b> |
| 7.1 Update Expression of SetBody   | 17        |
| 7.1.1. Go to src/main/java/resource/Module1.camel.yaml   | 17        |
| 7.2.2. Update the setBody component with below expression  | 17        |
| 7.2 Commit the Changes   | 18        |
| 7.3 Verify the pipeline has been completed successfully.   | 19        |
| 7.4 Verify the Output  | 19        |
| <b>8. Module 3</b>   | <b>19</b> |
| 8.1. Click on the 3 dot of the log component and select append   | 19        |

|   |           |
|---|-----------|
| 8.2. Select the xj component from the catalog                 | 20        |
| 8.2.1 Update the below configuration for the xj component     | 20        |
| 8.3 Commit the Changes  | 22        |
| 8.4 Verify the pipeline has been completed successfully.      | 22        |
| 8.5 Verify the Output   | 22        |
| <b>9 Module 4</b>   | <b>23</b> |
| 9.1. Click on the 3 dot of the XJ component and select append | 23        |
| 9.2. Select the kafka component from the catalog              | 24        |
| 9.2.1 Update properties for the Kafka component and close     | 24        |
| 9.3 Commit the Changes  | 25        |
| 9.4 Verify the pipeline has been completed successfully.      | 25        |
| 9.5 Verify the Output   | 25        |
| <b>10 Module 5</b>  | <b>26</b> |
| 10.1. Create a new route                                      | 26        |
| 10.2. Replace Timer with Kafka component.                     | 26        |
| 10.3. Select the kafka component from the catalog             | 27        |
| 10.3.1 Update properties for the Kafka component and close    | 28        |
| 10.4. Replace Log with AzureBlob Storage component.           | 28        |
| 10.5. Select the AzureBlob Storage component from the catalog | 29        |
| 10.5.1 Update parameters for the azure-storage-blob component | 30        |
| 10.6. Select both the route                                   | 32        |
| 10.7 Commit the Changes                                       | 32        |
| 10.8 Verify the pipeline has been completed successfully.     | 32        |
| 10.9 Verify the Output  | 32        |

# 1. Update the DevSpace plugin

Each of the following sections provides a brief description and a list of recommended actions. Additional explanations are provided where additional clarity is required.

## 1.1. Login to the openshift console using admin credentials.

UserName:- admin

Password:- <SameAsUser>

URL:- <https://console-openshift-console.<CLUSTER-DOMAINNAME>>

SAMPLE URL :- <https://console-openshift-console.apps.cluster-5kbh5.sandbox346.opentlc.com>

## 1.2. Go to the Installed OperatorPage and look for Red Hat OpenShift Dev Spaces

| Name                         | Managed Namespaces | Status                                      | Last updated        | Provided APIs   |
|------------------------------|--------------------|---|---------------------|---|
| Red Hat OpenShift Dev Spaces | All Namespaces     | Succeeded<br>Up to date                     | 7 Aug 2024, 9:59 pm | Red Hat OpenShift Dev Spaces instance Specification                         |
| DevWorkspace Operator        | All Namespaces     | Succeeded<br>Up to date                     | 7 Aug 2024, 9:59 pm | DevWorkspace<br>DevWorkspaceTemplate<br>DevWorkspaceOperatorConfig          |
| GitLab Runner                | All Namespaces     | Succeeded<br>Up to date                     | 7 Aug 2024, 9:58 pm | GitLab Runner   |
| Red Hat OpenShift GitOps     | All Namespaces     | Succeeded<br>Up to date<br>Plugin available | 7 Aug 2024, 9:27 pm | Argo CD<br>AnalysisRun<br>AnalysisTemplate<br>Application<br>View 6 more... |

## 1.3 Update the devspace cheCluster Yaml configurations.

Red Hat OpenShift Dev Spaces instance Specification

Click here

Next Click here

| Name         | Kind       | Namespace           | Status | Labels                           | Last updated        |
|--------------|------------|---------------------|--------|----------------------------------|---------------------|
| cc devspaces | CheCluster | openshift-devspaces | -      | rht-gitops.com/opens...=devsp... | 7 Aug 2024, 9:59 pm |

## 1.4. Update the pluginRegistry configurations.

pluginRegistry:

openVSXURL: '<https://open-vsx.org/>'

```

129 labels:
130   rht-gitops.com/openshift-gitops: devspaces
131 spec:
132   components:
133     cheServer:
134       debug: false
135       logLevel: INFO
136     dashboard:
137       logLevel: ERROR
138     devWorkspace: {}
139     devRegistry: {}
140     devEditor:
141       enable: false
142       spec: {}
143     metrics:
144       enable: true
145     pluginRegistry:
146       openVSXURL: 'https://open-vsx.org/' Update this registry
147     containerRegistry: {}
148     devEnvironments:
149       startTimeoutSeconds: 300
150       security: {}
151       secondsOfRunBeforeIdling: -1
152       maxNumberOfWorkspacesPerUser: -1
153       containerBuildConfiguration:
154         openShiftSecurityContextConstraint: container-build
155         defaultEditor: che-incubator/che-code/latest
156         maxNumberOfRunningWorkspacesPerUser: 2

```

Save Reload Cancel Download

1.5. Save the configuration and verify the changes after reload.

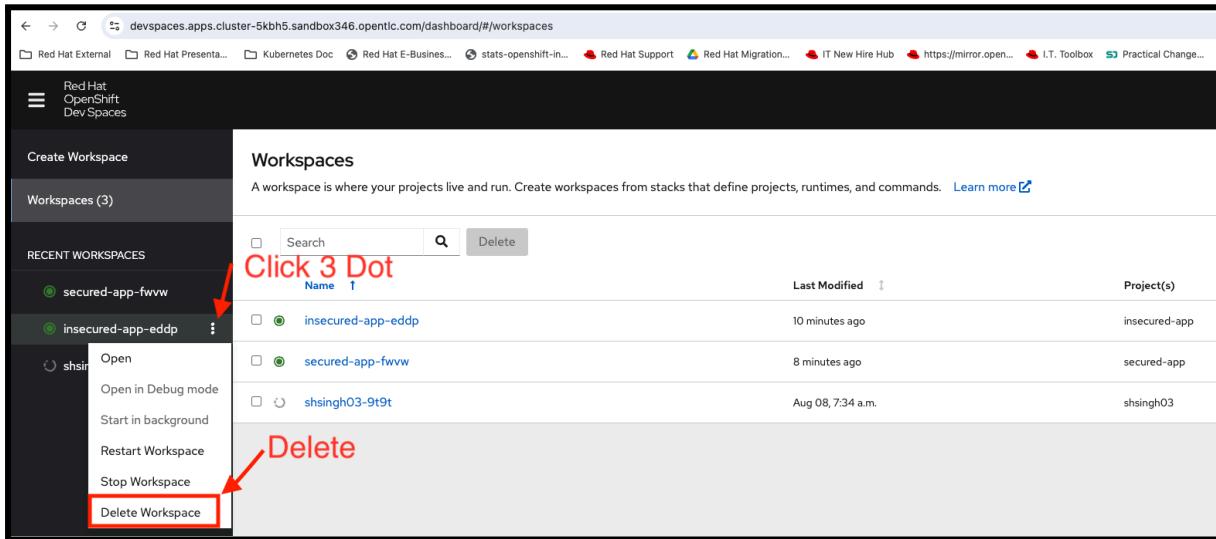
## 2 Delete the previous workspace from the Devspace

(Only when you have 2 Workspace)

URL:- <https://devspaces.<CLUSTER-DOMAINNAME>>

SAMPLE URL :-

<https://devspaces.apps.cluster-5kbh5.sandbox346.opentlc.com/>



The screenshot shows the Red Hat OpenShift Dev Spaces dashboard. On the left, there's a sidebar with 'Create Workspace' and 'Workspaces (3)'. Under 'RECENT WORKSPACES', there are three entries: 'secured-app-fwww', 'insecured-app-eddp', and 'shsingh03-9t9t'. A red arrow points to the three-dot menu next to 'insecured-app-eddp'. Another red arrow points to the 'Delete' button in the context menu for 'shsingh03-9t9t'. The main area is titled 'Workspaces' and contains a table with columns for 'Name', 'Last Modified', and 'Project(s)'. The table shows:

| Name               | Last Modified     | Project(s)    |
|--------------------|-------------------|---------------|
| insecured-app-eddp | 10 minutes ago    | insecured-app |
| secured-app-fwww   | 8 minutes ago     | secured-app   |
| shsingh03-9t9t     | Aug 08, 7:34 a.m. | shsingh03     |

3 Import the DeveloperHub template for this demo .



### 3.1. Go to the Create Software Template section

The screenshot shows the Red Hat Developer Hub interface. On the left, a dark sidebar lists various sections: Red Hat External, Red Hat Presentations, Kubernetes Doc, Red Hat E-Business, and stats-openshift-in... At the bottom of this sidebar is a red box highlighting the '+ Create...' button. To the right of the sidebar, the main content area has a title 'Software Templates' and a sub-section 'Available Templates'. A search bar at the top of this area contains the text 'Click Here'. Below the search bar are sections for 'Personal' (Starred, 0) and 'My Org' (All, 3). Further down are 'Categories' and 'Tags' dropdown menus. To the right of these filters is a 'Templates' sidebar listing 'Service', 'Secured Quarkus Service', and 'Supply Chain'. At the bottom of the sidebar are tags: recommended, java, maven, and rhdh.

### 3.2. Register the new template

Template URL:-

<https://github.com/shailendra14k/backstage-template/blob/main/backstage-camel/template.yaml>

**Register an existing component**

Start tracking your component in Red Hat Developer Hub

1 Select URL

URL \*  
https://github.com/shailendra14k/backstage-template/blob/n

Enter the full path to your entity file to start tracking your component.

Analyze

2 Import Actions  
Optional

3 Review

4 Finish

Click Here

Register an existing component

Enter the URL to your source code repository to add it to Red Hat Developer Hub.

Link to an existing entity file

Example: https://github.com/backstage/backstage/blob/master/catalog-info.yaml

The wizard analyzes the file, previews the entities, and adds them to the Red Hat Developer Hub catalog.

Learn more about the Software Catalog →

### 3.3. Finish the import

## 4 Create the Catalog from template

### 4.1 Go to the Template Create section

**Register an existing component**

Start tracking your component in Red Hat Developer Hub

1 Select URL

2 Select Locations  
Discovered Locations: 1

3 Review

4 Finish

The file was triggered for the following locations:  
https://github.com/shailendra14k/backstage-template/blob/n  
Entities: 2  
location:generated-7a843d349039a46b49276974835b17  
template:spring-boot-camel-on-openshift-template

Click Here

Register another

Register an existing component

Enter the URL to your source code repository to add it to Red Hat Developer Hub.

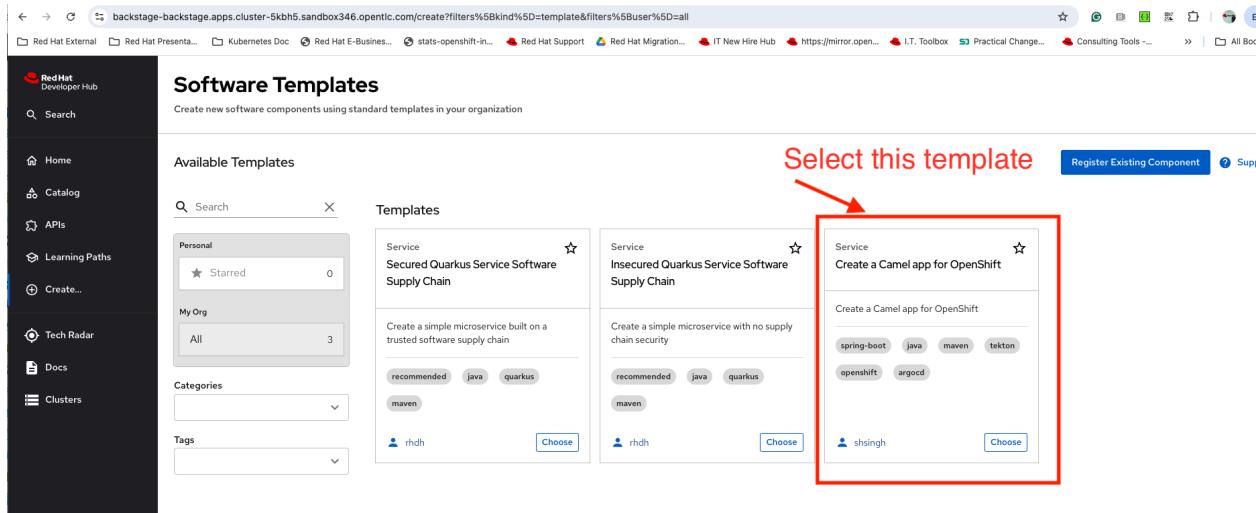
Link to an existing entity file

Example: https://github.com/backstage/backstage/blob/master/catalog-info.yaml

The wizard analyzes the file, previews the entities, and adds them to the Red Hat Developer Hub catalog.

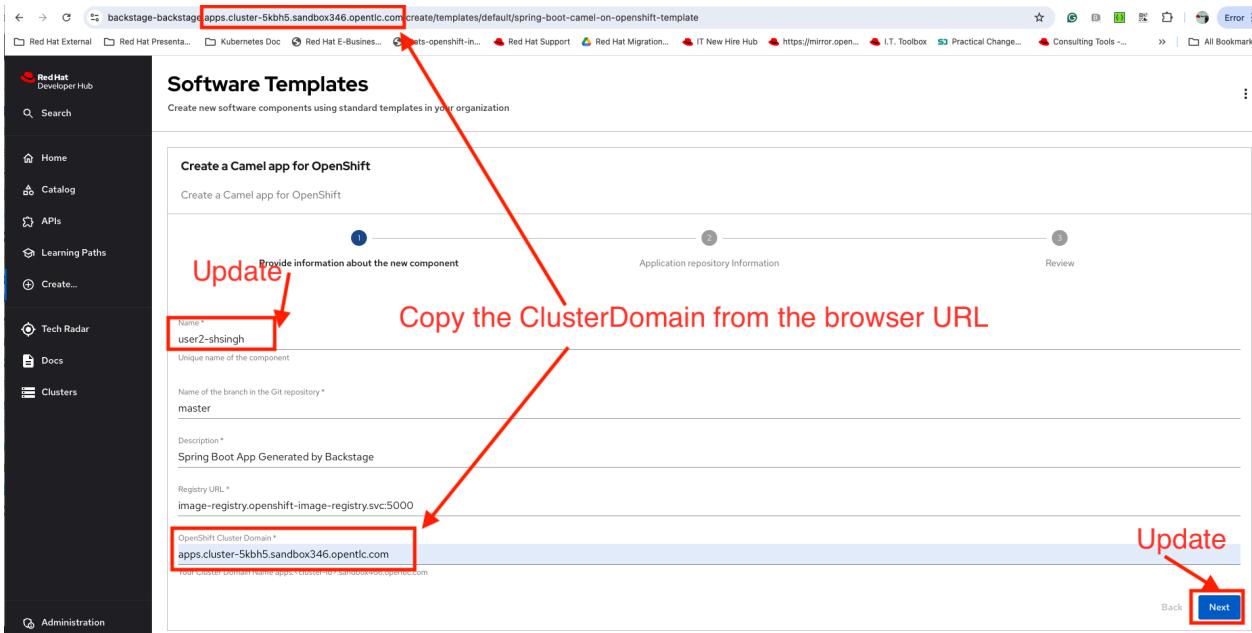
Learn more about the Software Catalog →

## 4.2. Choose the template “Create a Camel app for OpenShift”

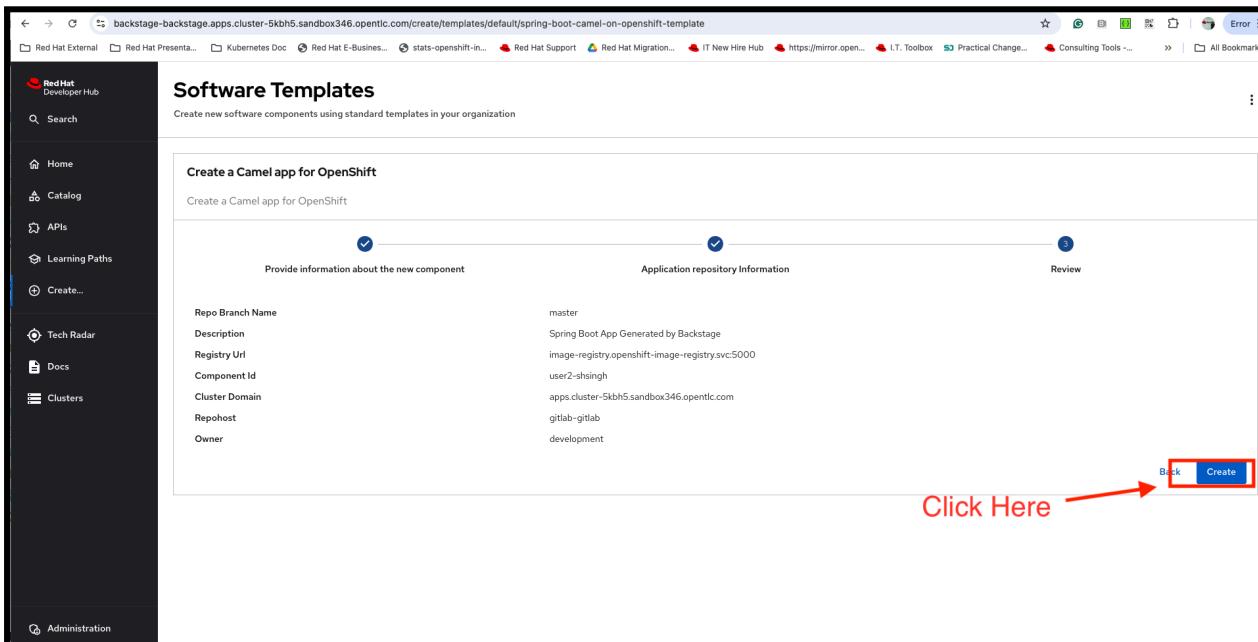


## 4.3 Add the details, Name and Cluster Domain main copy from the browser url as shown below

Name :- user1-<yourname> (no cpas)



## 7. Click Next and verify the repo details and create



## 8. The template pipeline will be executed for the initial setup

Run of spring-boot-camel-on-openshift-template

Task d859ee7e-23ca-494c-8c60-f1f8c235c918

Generating the Source Code Component 2 seconds Publishing to the Source Code Repository 3 seconds Registering the Catalog Info Component 1 second Generating the Config Code Component 2 seconds Publishing to the Config Code Repository 2 seconds Create ArgoCD Resources 3 seconds

**Click Here**

Open the Source Code Repository **Open the Catalog Info Component**

## 8. Verify the Catalog

COMPONENT ~ SERVICE

**user2-shsingh**

Overview Topology Issues Pull/Merge Requests CI CD Kubernetes API Dependencies Docs

This entity has relations to other entities, which can't be found in the catalog.  
Entities not found are: system/default/workshop-camel-demo

| Links                          |   | Merge requests statistics |               |  |  |  |  |
|--------------------------------|---|---------------------------|---------------|--|--|--|--|
| OpenShift Dev Spaces (VS Code) | OpenShift Dev Spaces (JetBrains IntelliJ) | Avg Time Until Merge      | Never         |  |  |  |  |
|                                |   | Merged To Total Ratio     | 0%            |  |  |  |  |
|                                |   | 20                        | Number of MRs |  |  |  |  |

| About           |  | Deployment summary          |               |               |        |                                |               |
|-----------------|--|-----------------------------|---------------|---------------|--------|--------------------------------|---------------|
| View Source     | View TechDocs  | ArgoCD App                  | Namespace     | Instance      | Server | Revision                       | Last deployed |
| Description     |  | Verify                      |               |               |        |                                |               |
| No description  | Owner: user1   | System: workshop-camel-demo | Type: service | user2-shsingh | main   | https://kubernetes.default.svc | d2aaada7      |
| Lifecycle: demo | Tags: spring-boot, java, maven, tekton, argoCD, renovate |                             |               | user2-shsingh | main   | https://kubernetes.default.svc | d2aaada7      |

5 Open the new workspace from the developer Hub.

Click here

**user1-test**

Links

- OpenShift Dev Spaces (VS Code)
- OpenShift Dev Spaces (JetBrains IntelliJ)

Merge requests statistics

|                       |       |
|-----------------------|-------|
| Avg Time Until Merge  | Never |
| Merged To Total Ratio | 0%    |
| Number of MRs         | 20    |

About

View View Source TechDocs

Deployment summary

5.1 After waiting a few minutes for OpenShift Dev Spaces to finish setting up your workspace, You click the button Yes, I trust the authors.

Welcome

Red Hat OpenShift Dev Spaces

with Microsoft Visual Studio Code - Open Source IDE

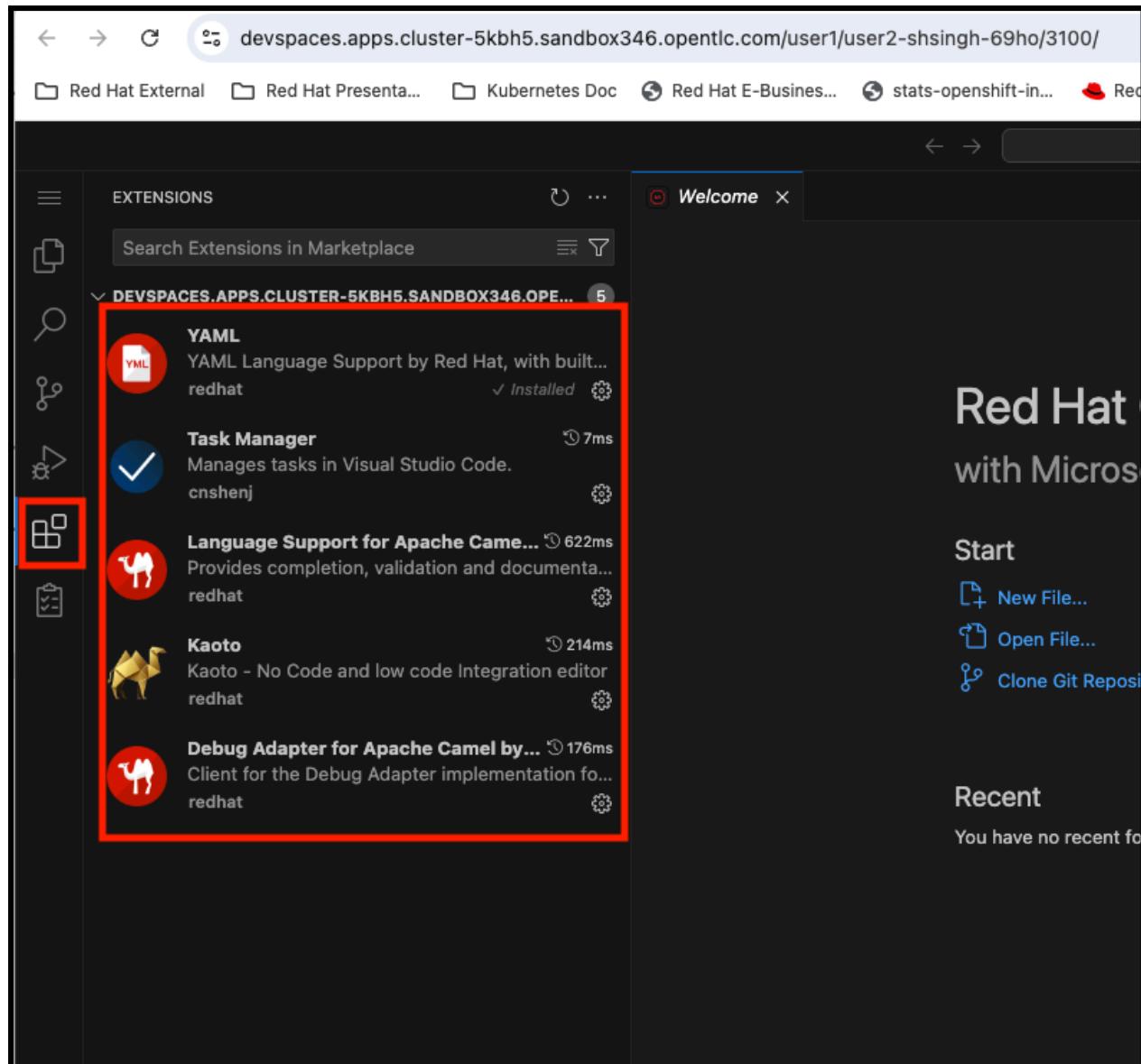
Do you trust the authors of the files in this workspace?

If you don't trust the authors of these files, we recommend to continue in restricted mode as the files may be malicious. See our docs to learn more.

No, I don't trust the authors      Yes, I trust the authors

Click here

## 5.2. Verify all the extension has been installed correctly.



## 6 Module 1

### 6.1 Update the application.properties

devspaces.apps.cluster-5kbh5.sandbox346.opentlc.com/user1/user2-shsingh-69ho/3100/

Red Hat External Red Hat Presenta... Kubernetes Doc Red Hat E-Busines... stats-openshift-in... Red Hat Support Red Hat Migration... IT New Hire Hub https://mirror.open... I.T. Toolb

application.properties 2

```

1 # the name of Camel
2 camel.springboot.name = MyCamel
3
4 camel.component.kafka.mapping.context-path=/camel/*
5 camel.component.kafka.name=CamelServlet
6
7 # what to say
8 greeting = Hello World
9
10
11 camel.component.kafka.security-protocol = SASL_SSL
12 camel.component.kafka.sasl-mechanism= SCRAM-SHA-256
13 camel.component.kafka.ssl-truststore-location = /opt/secrets/clienttruststore.jks
14 camel.component.azure-storage-blob.credential-type= SHARED_ACCOUNT_KEY
15
16 #Replace this
17 camel.component.kafka.brokers=
18 camel.component.kafka.sasl-jas-config=
19 camel.component.kafka.ssl-truststore-password =
20 camel.component.azure-storage-blob.access-key=
21 #####
22
23 camel.springboot.routes-include-pattern = classpath:route/Module1.camel.yaml
24
25
26
27 topicName.json=user2-shsingh-json
28
29 consumergroup=user2-shsingh-consumer
30
31 blobName=user2-shsingh-xml
32
33
34 # how often to trigger the timer
35 timer.period = 2000
36
37 # to automatic shutdown the JVM after a period of time
38 #camel.springboot.duration-max-seconds=60
39 #camel.springboot.duration-max-messages=100
40

```

Replace the Value

~~~~~

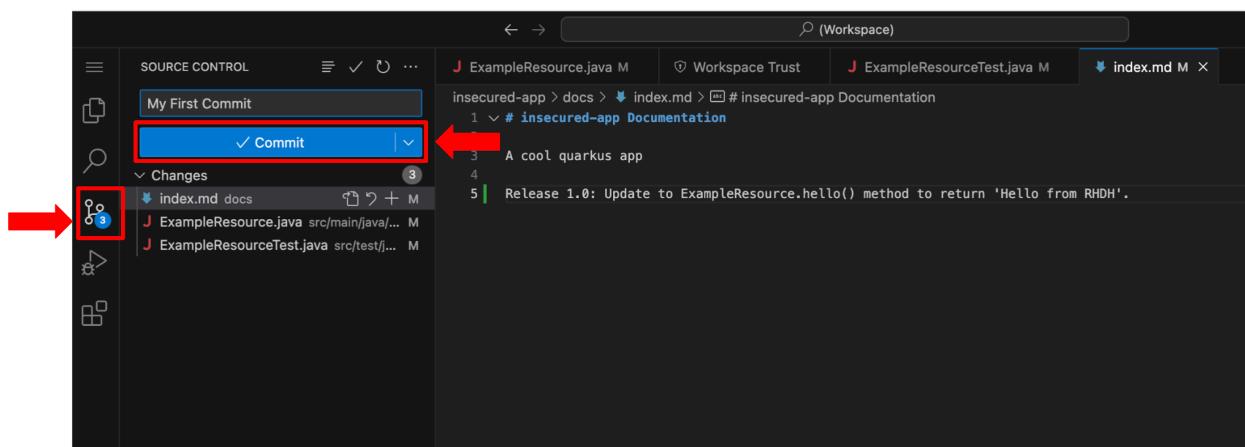
camel.component.kafka.brokers=my-cluster-kafka-route1-bootstrap-amq-stre  
ams-kafka.apps.shailendra14k.in:443  
camel.component.kafka.sasl-jas-config=  
org.apache.kafka.common.security.scram.ScramLoginModule required  
username="console-cluster-user1"  
password="JkP7hGT7rpJd1DdBQ3zb1DTUZ6Gau0zH";  
camel.component.kafka.ssl-truststore-password = indigo  
camel.component.azure-storage-blob.access-key=  
Y/fgFLVIPUGJ6MfQlWLxPWaILEE+BXV2FtzQwKwePp2XHKpF2djTaaNfc7utIHxD+m3OAxs8  
Ih4i+ASTY7QFJg==

~~~~~

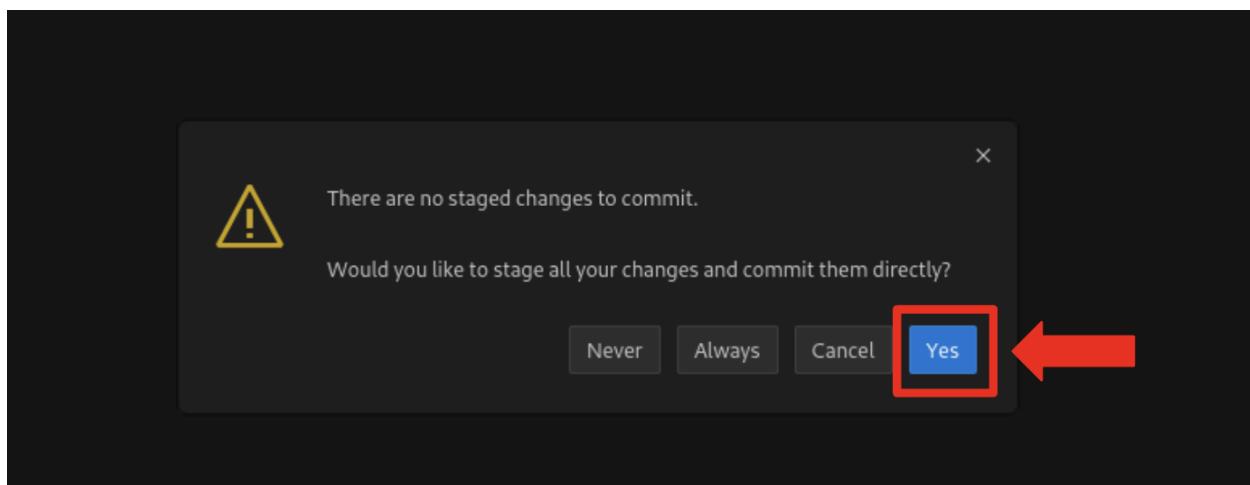
## 6.2 Commit the Changes

6.2.1. You click on the Source Control icon located in the left menu.

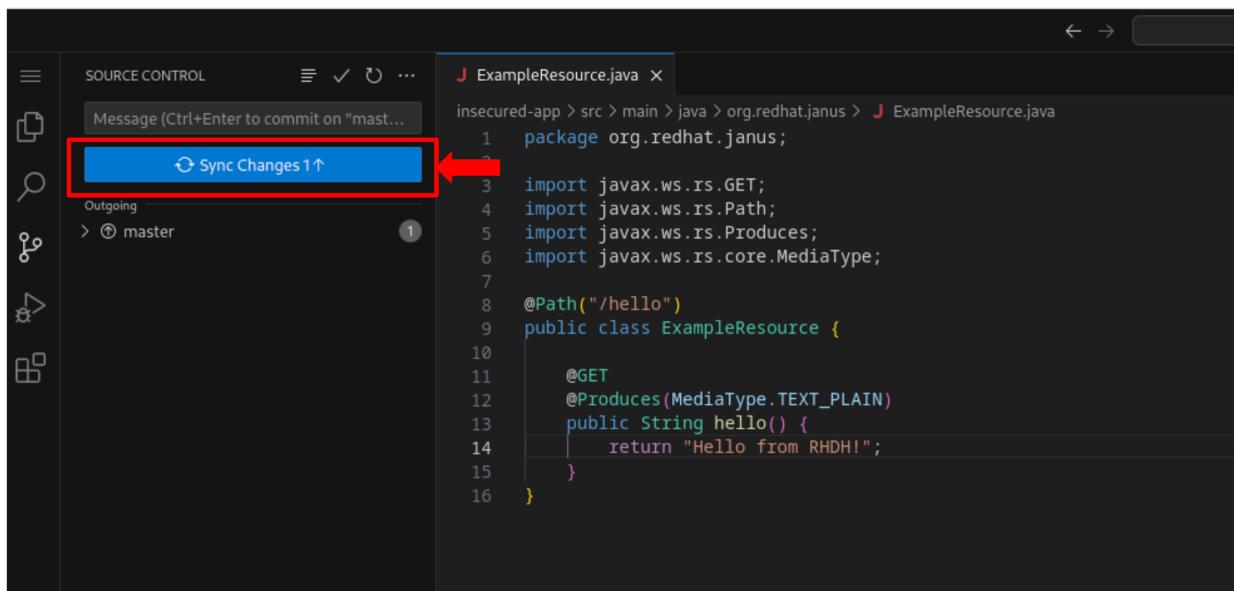
6.2.2. Then, you enter the commit message "updated application properties" and click on the Commit button to finalize your changes.



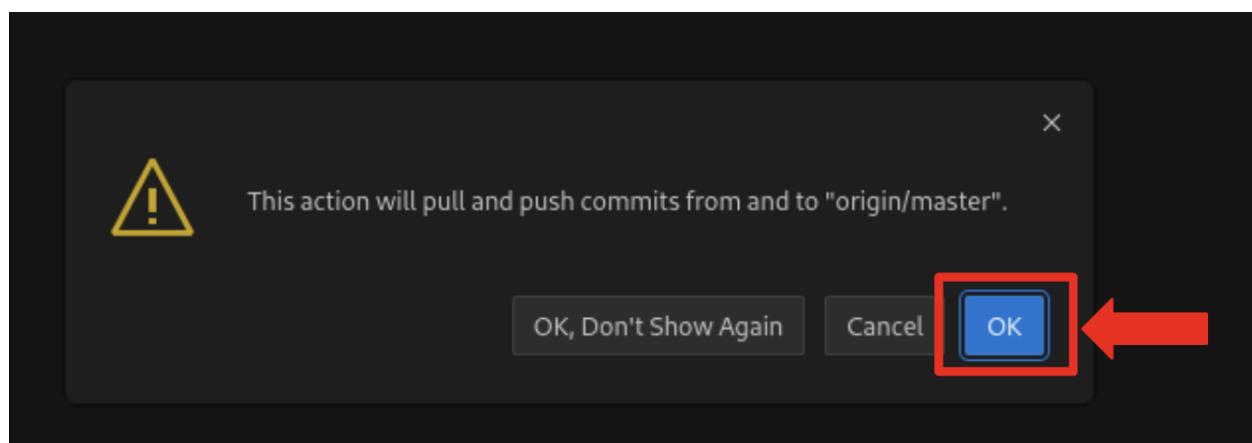
6.2.3. In the pop-up window that follows, you click Yes to stage your changes.



6.2.4.Finally, you click on the Sync Changes button.



6.2.5. In the pop-up that follows, you click OK to push your changes and complete the process.



In Case the commit fails like the below

The screenshot shows a code editor interface with a yellow border. In the center, there is a modal dialog box with a red close button. The dialog contains the text "Failed to authenticate to git remote:" followed by a URL: "https://gitlab-gitlab.apps.cluster-5kbh5.sandbox346.opentlc.com/development/user2-shsingh.git/". Below the URL are three buttons: "Show Command Output", "Cancel", and "Open Git Log". The background of the code editor shows a file named "application.properties" with several lines of configuration code.

```

application.properties 2  ! Module1.camel.yaml
user2-shsingh > src > main > resources > application.properties

11 component.kafka.security-protocol = SASL_SSL
12 component.kafka.sasl-mechanism= SCRAM-SHA-512
13 component.kafka.ssl-truststore-location = /mnt/secrets/clienttruststore.jks
14 component.azure-storage-blob.credential-type= SHARED_ACCOUNT_KEY
15
16 ce this
17 component.kafka.brokers=my-cluster-kafka-route1-bootstrap-amq-streams-kafka.apps.shailendra14k.in:443
18 component.kafka.sasl-jaas-config= org.apache.kafka.common.security.scram.ScramLoginModule required username="console-cluster-user1" password=
19 component.kafka.ssl-truststore-password = indigo
20 component.azure-storage-blob.access-key= Y/fgFLVlPUGJ6MfQlWlxPWaILEE+BXVFtzQwKwePp2XHKpF2djTaaNfc7utIHxD+m30Axs8Ih4i+AStY7QFJg==

23
24
25 springboot.routes-include-pattern = classpath:route/Module1.camel.yaml
26
27
28 lame.json=user2-shsingh-json
29
30 iergroup=user2-shsingh-consum
31 me=user2-shsingh-xml
32
33
34
35 often to trigger the timer
period = 2000
36
37 utomatic shutdown the JVM after a period of time
38 .springboot.duration-max-seconds=60
39 .springboot.duration-max-messages=100
40
41 for example: @repeatCount=5 to the timer endpoint to make Camel idle
42 .springboot.duration-max-idle-seconds=15
43
44
45 use actuator endpoint via HTTP
46 .management.endpoints.web.exposure.include=info,health,camelroutes
47
48 on actuator health check
49 .management.endpoint.health.enabled = true
50
51

```

Delete all the workspaces from the Devspace URL and start the new workspace from the same software catalogue.

## 6.3 Verify the pipeline

The screenshot shows the Red Hat Developer Hub interface. On the left, there is a sidebar with various navigation options like Home, Catalog, APIs, Learning Paths, Create..., Tech Radar, Docs, and Clusters. The main area displays a project titled "user2-shsingh" with a star icon. The "CI" tab is selected. Below it, there is a "Pipeline Runs" section. A single pipeline run is listed with the name "PLR user2-shsingh-run-v2knq". The status of this run is "Succeeded", indicated by a green bar. The "STARTED" timestamp is "08/08/2024, 15:00:37" and the "DURATION" is "4 minutes 57 seconds". Below the run, the pipeline stages are shown as a sequence of green circles: "git-clone" (1/1), "maven" (1/1), and "s2i-java" (1/1). At the bottom of the pipeline run card, there are icons for search, refresh, and other actions.

## 6.4 Open the application

URL:-

`https://<PROJECT-NAME>-<PROJECT-NAME>.<CLUSTER-DOMAINNAME>/camel/demo`

SAMPLE URL :-

`https://user2--shsingh03.apps.cluster-5kbh5.sandbox346.opentlc.com/camel/demo`



## 7 Module 2

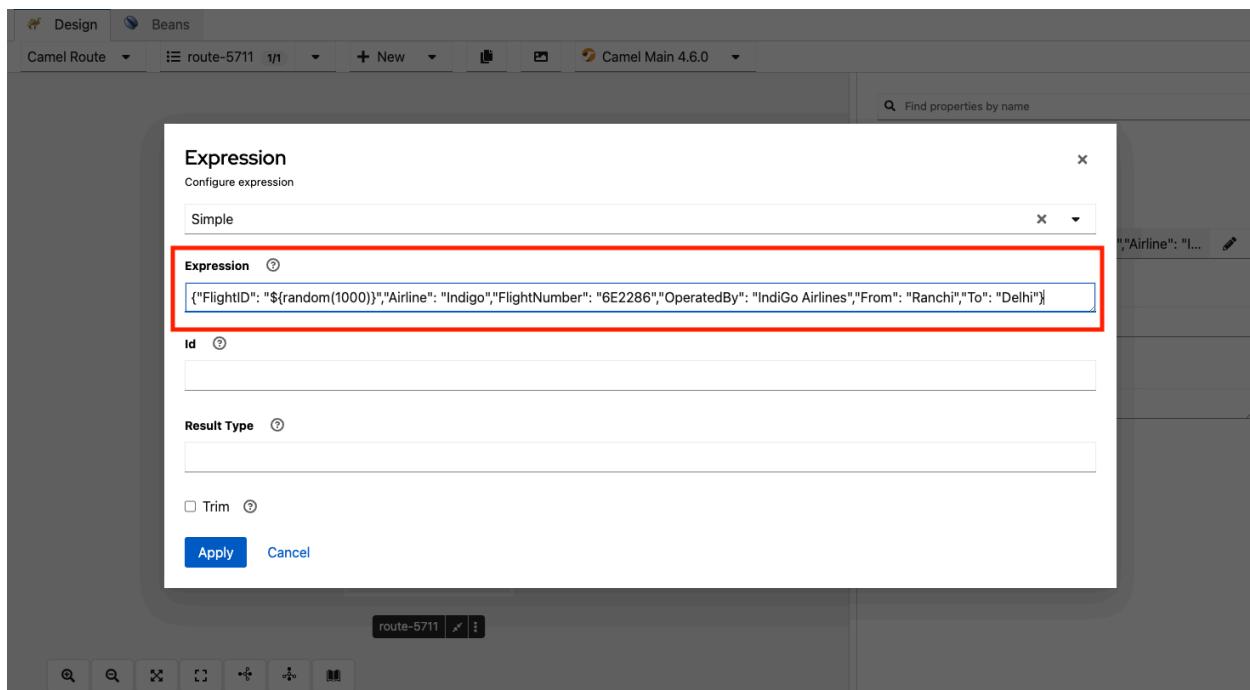
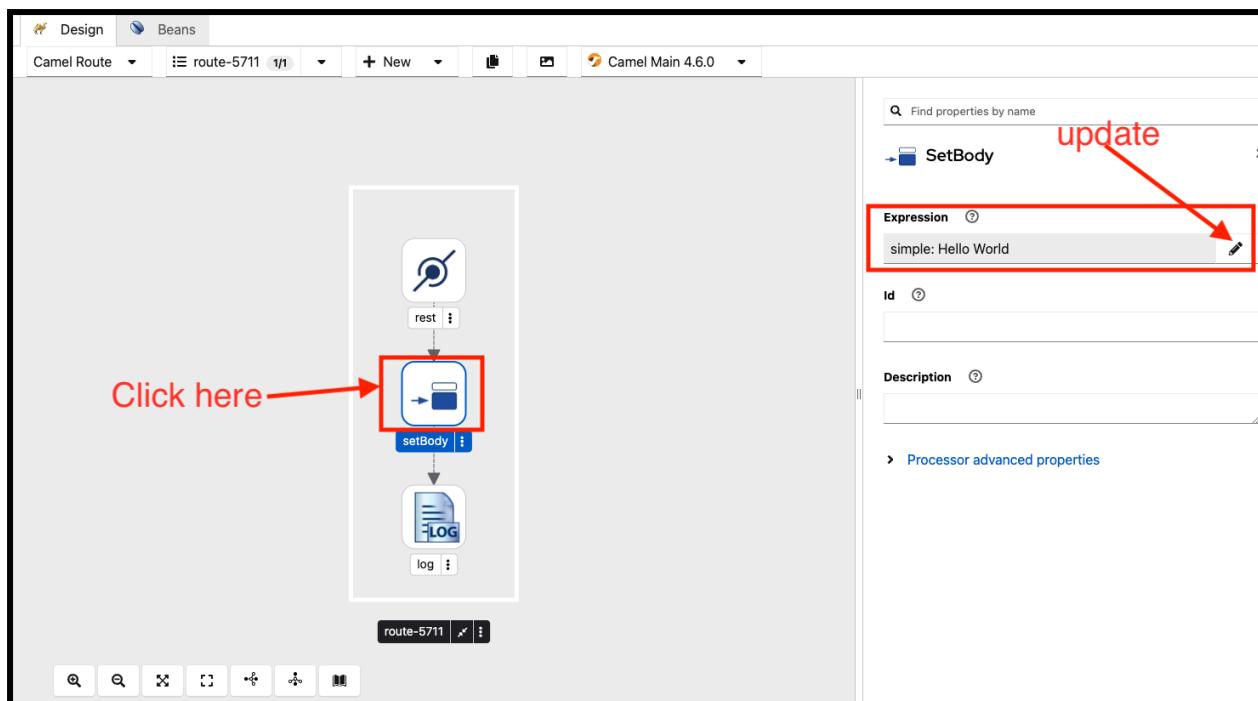
### 7.1 Update Expression of SetBody

7.1.1. Go to `src/main/java/resource/Module1.camel.yaml`

7.2.2. Update the setBody component with below expression

Expression:-

```
{"FlightID": "${random(1000)}", "Airline": "Indigo", "FlightNumber":  
"6E2286", "OperatedBy": "IndiGo Airlines", "From": "Ranchi", "To": "Delhi"}
```



## 7.2 Commit the Changes

[Follow Module1 steps](#)



## 7.3 Verify the pipeline has been completed successfully.

[Follow Module1 steps](#)

## 7.4 Verify the Output

URL :-

<https://<PROJECT-NAME>-<PROJECT-NAME>.<CLUSTER-DOMAINNAME>/camel/demo>

SAMPLE URL :-

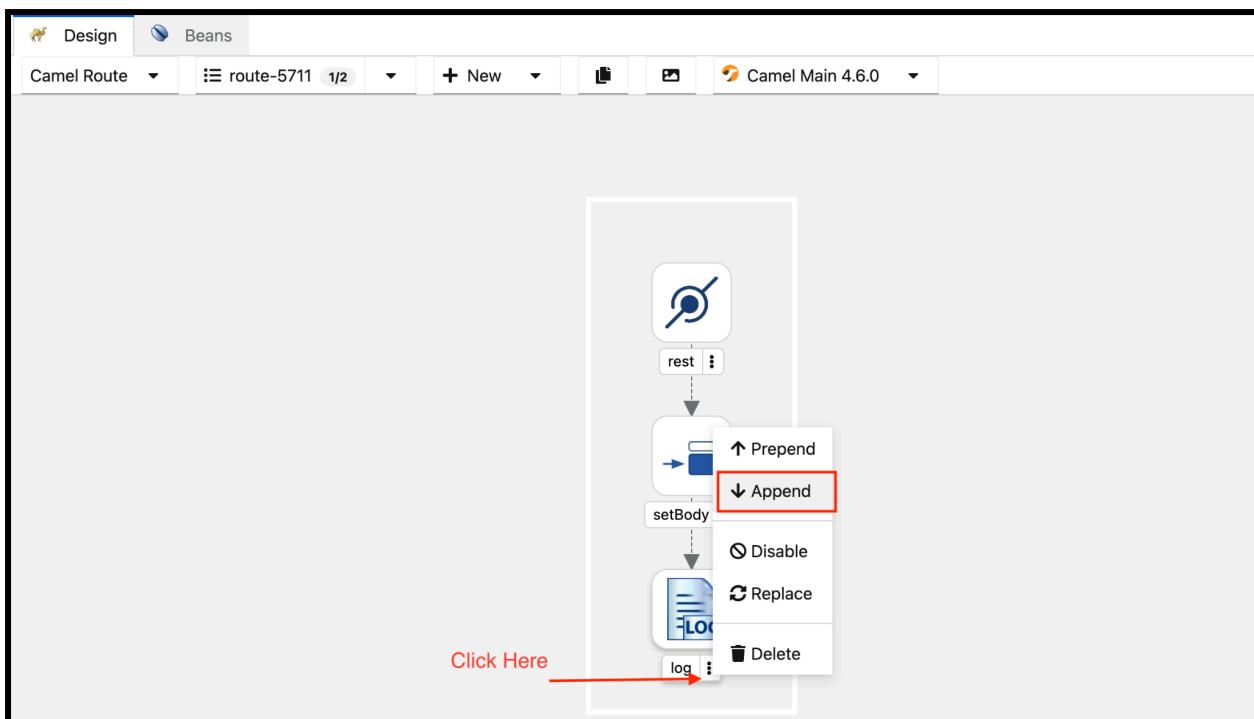
<https://shsingh03-shsingh03.apps.cluster-5kbh5.sandbox346.opentlc.com/camel/demo>

The screenshot shows a browser window with the URL `shsingh03-shsingh03.apps.cluster-5kbh5.sandbox346.opentlc.com/camel/demo`. The page displays a JSON response:

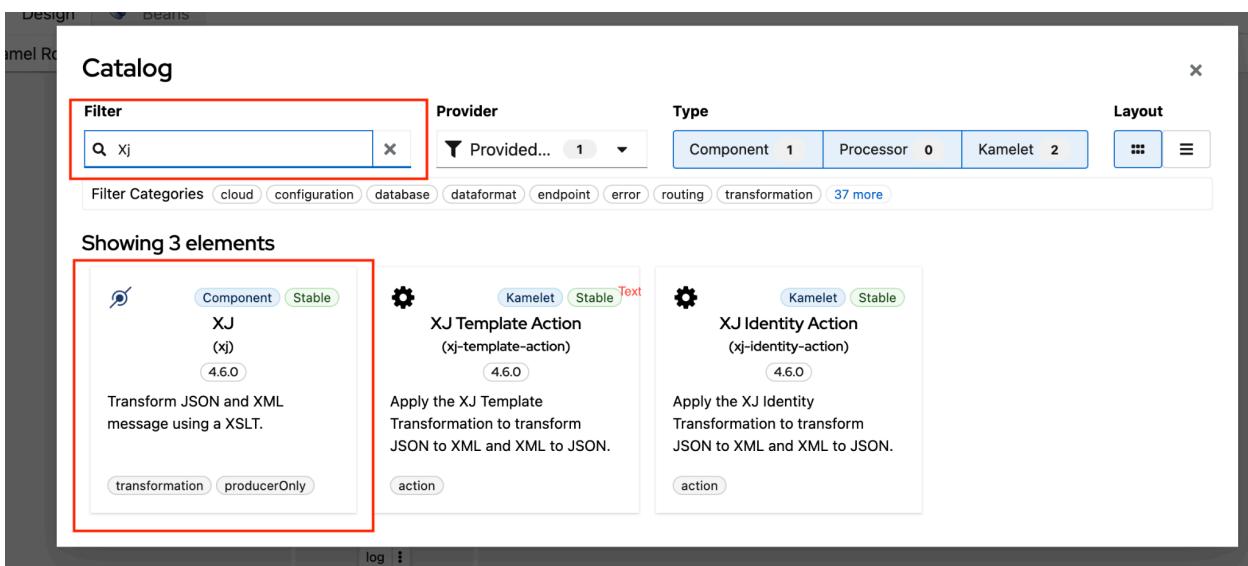
```
{"FlightID": "820", "Airline": "Indigo", "FlightNumber": "6E2286", "OperatedBy": "IndiGo Airlines", "From": "Ranchi", "To": "Delhi"}
```

## 8. Module 3

### 8.1. Click on the 3 dot of the log component and select append

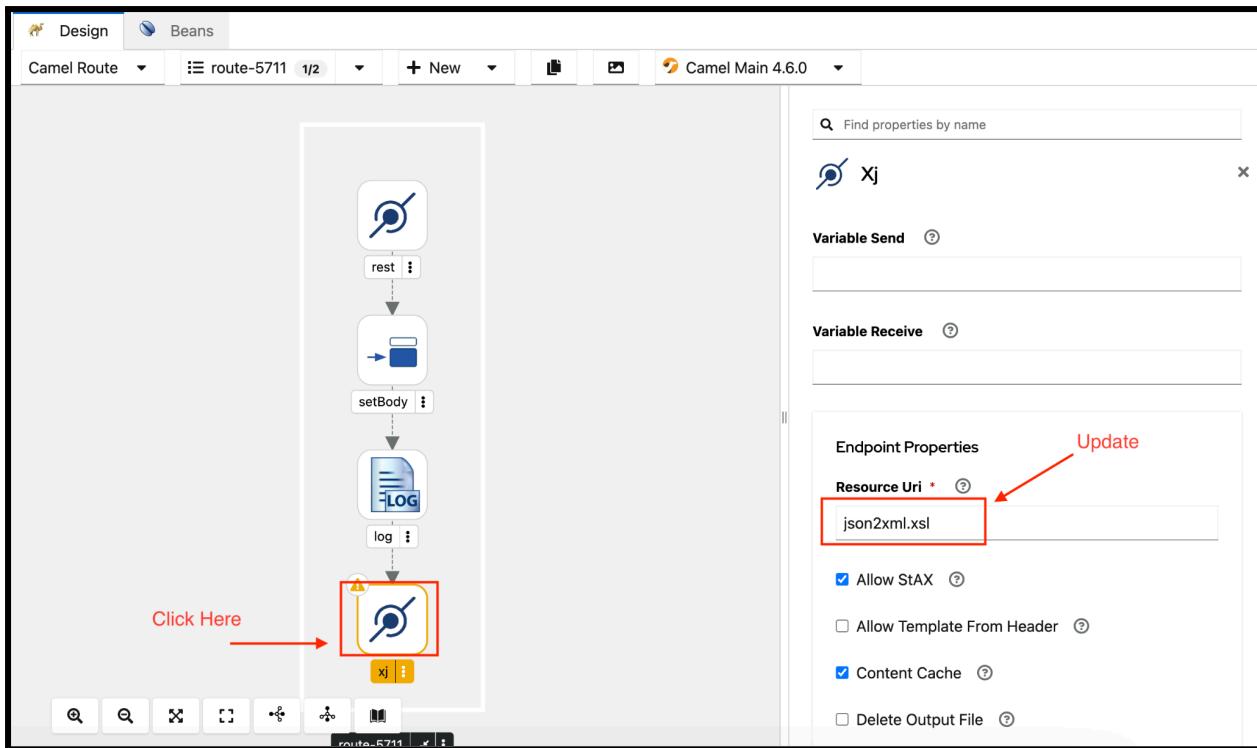


## 8.2. Select the xj component from the catalog

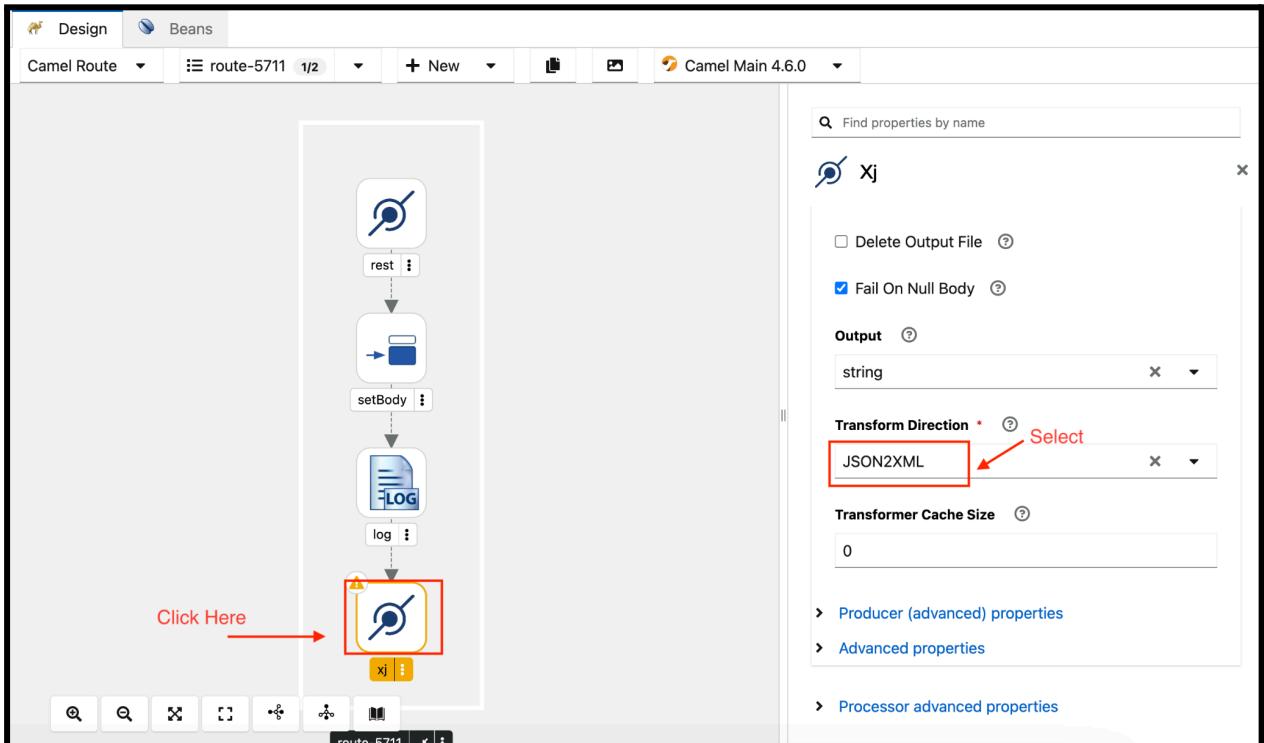


### 8.2.1 Update the below configuration for the xj component

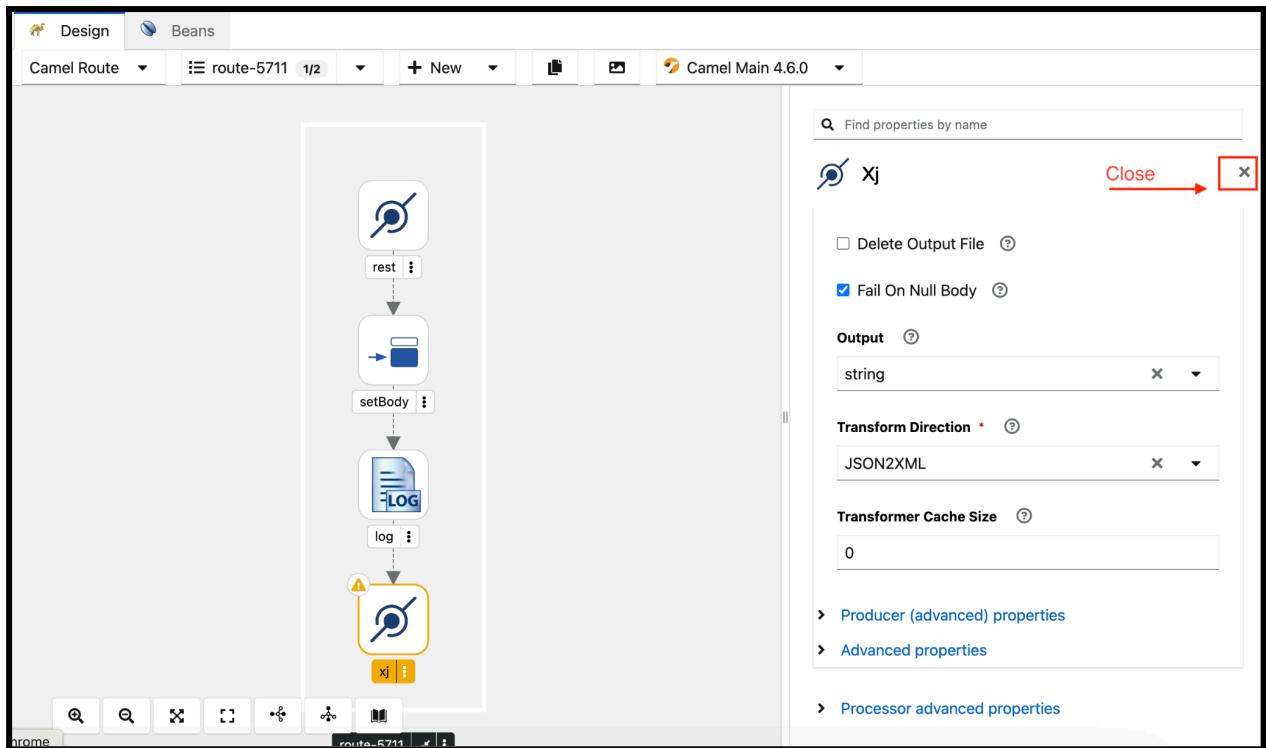
- Resource Uri --> json2xml.xsl



- Transform Direction ---> JSON2XML



- Close



### 8.3 Commit the Changes

[Follow Module1 steps](#)

### 8.4 Verify the pipeline has been completed successfully.

[Follow Module1 steps](#)

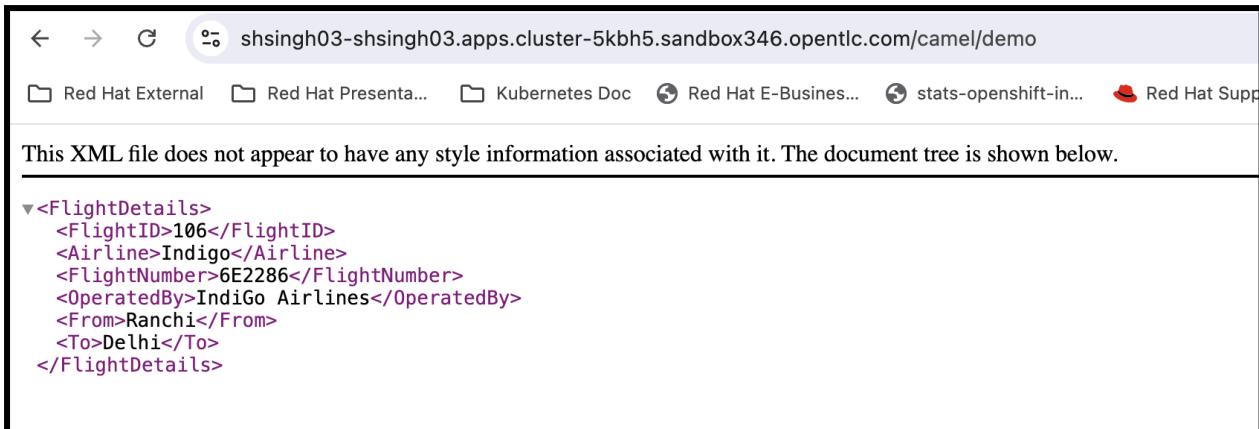
### 8.5 Verify the Output

URL:-

<https://<PROJECT-NAME>-<PROJECT-NAME>.<CLUSTER-DOMAINNAME>/camel/demo>

SAMPLE URL :-

<https://shsingh03-shsingh03.apps.cluster-5kbh5.sandbox346.opentlc.com/camel/demo>

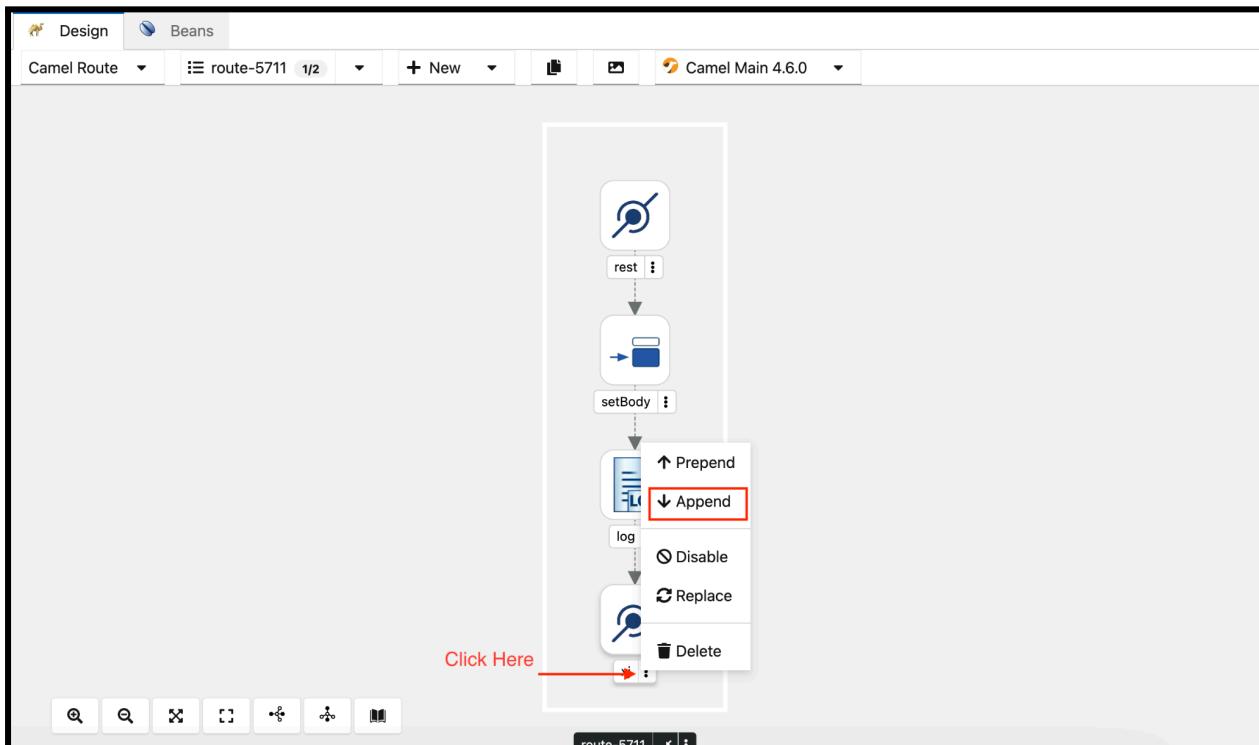


This XML file does not appear to have any style information associated with it. The document tree is shown below.

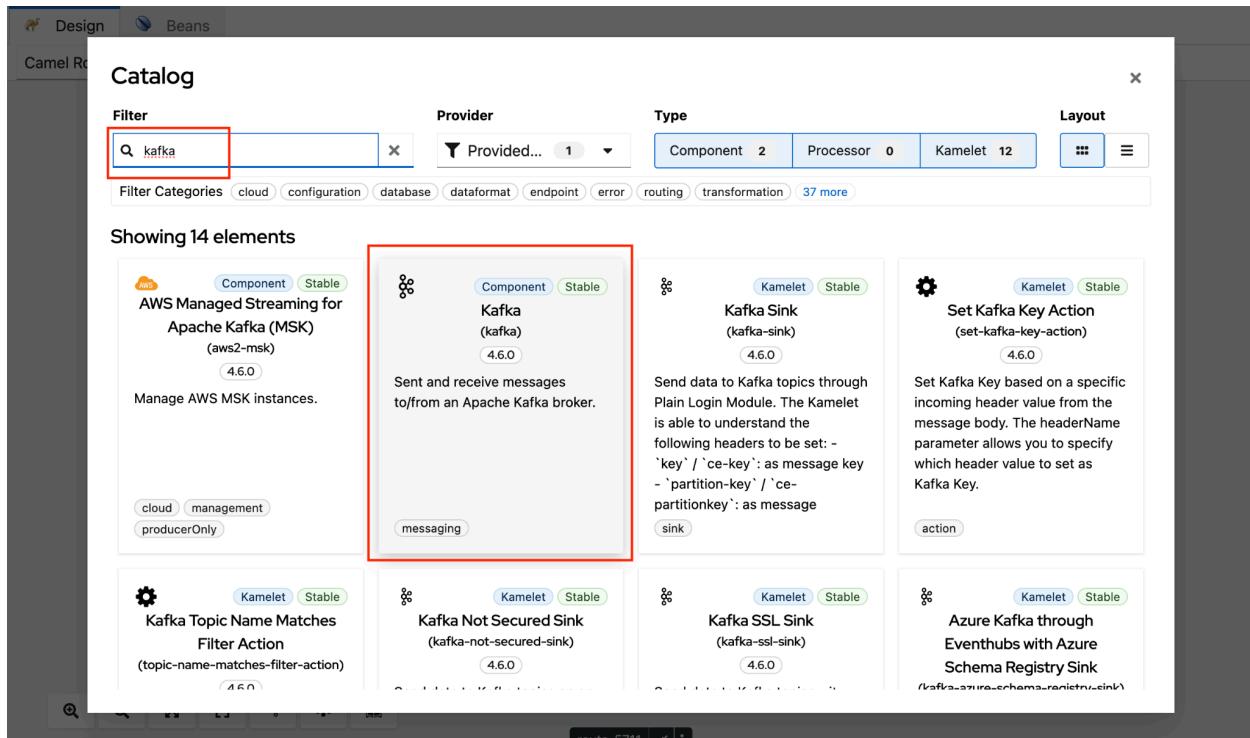
```
<?xml version="1.0"?>
<FlightDetails>
    <FlightID>106</FlightID>
    <Airline>Indigo</Airline>
    <FlightNumber>6E2286</FlightNumber>
    <OperatedBy>IndiGo Airlines</OperatedBy>
    <From>Ranchi</From>
    <To>Delhi</To>
</FlightDetails>
```

## 9 Module 4

### 9.1. Click on the 3 dot of the XJ component and select append

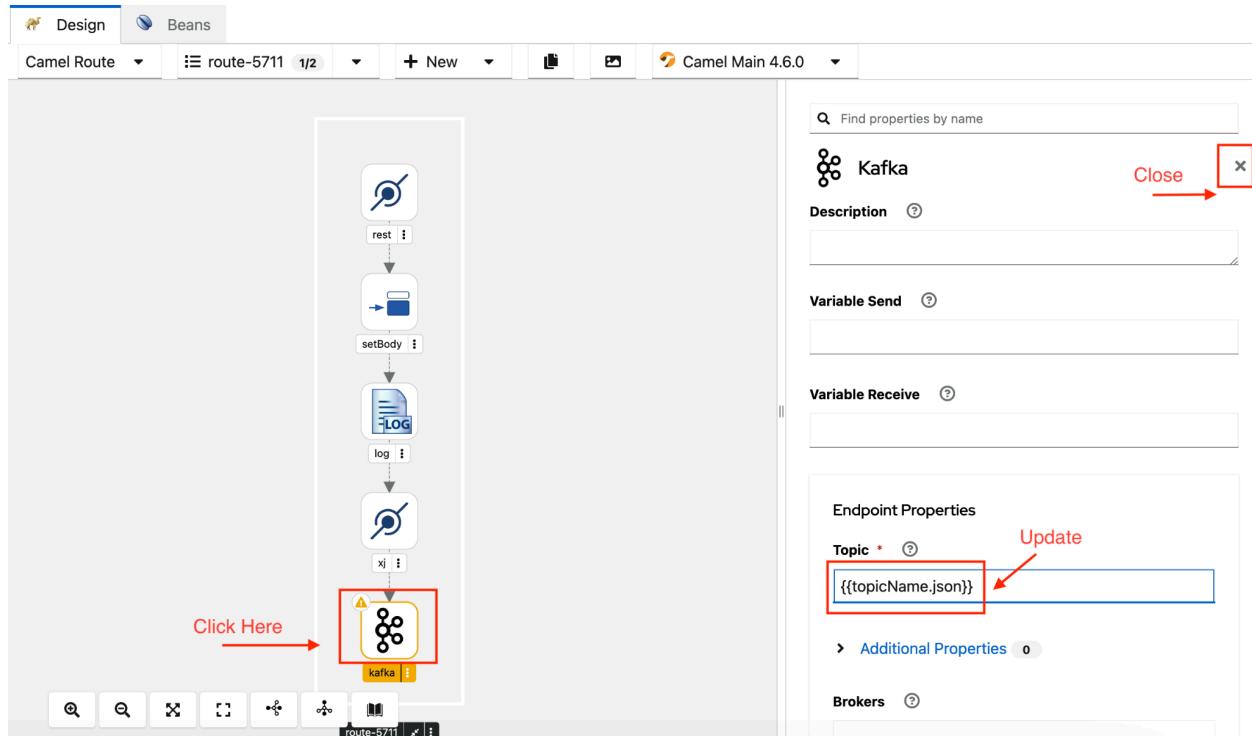


## 9.2.Select the kafka component from the catalog



### 9.2.1 Update properties for the Kafka component and close

- Topic --> {{topicName.json}}



### 9.3 Commit the Changes

[Follow Module1 steps](#)

### 9.4 Verify the pipeline has been completed successfully.

[Follow Module1 steps](#)

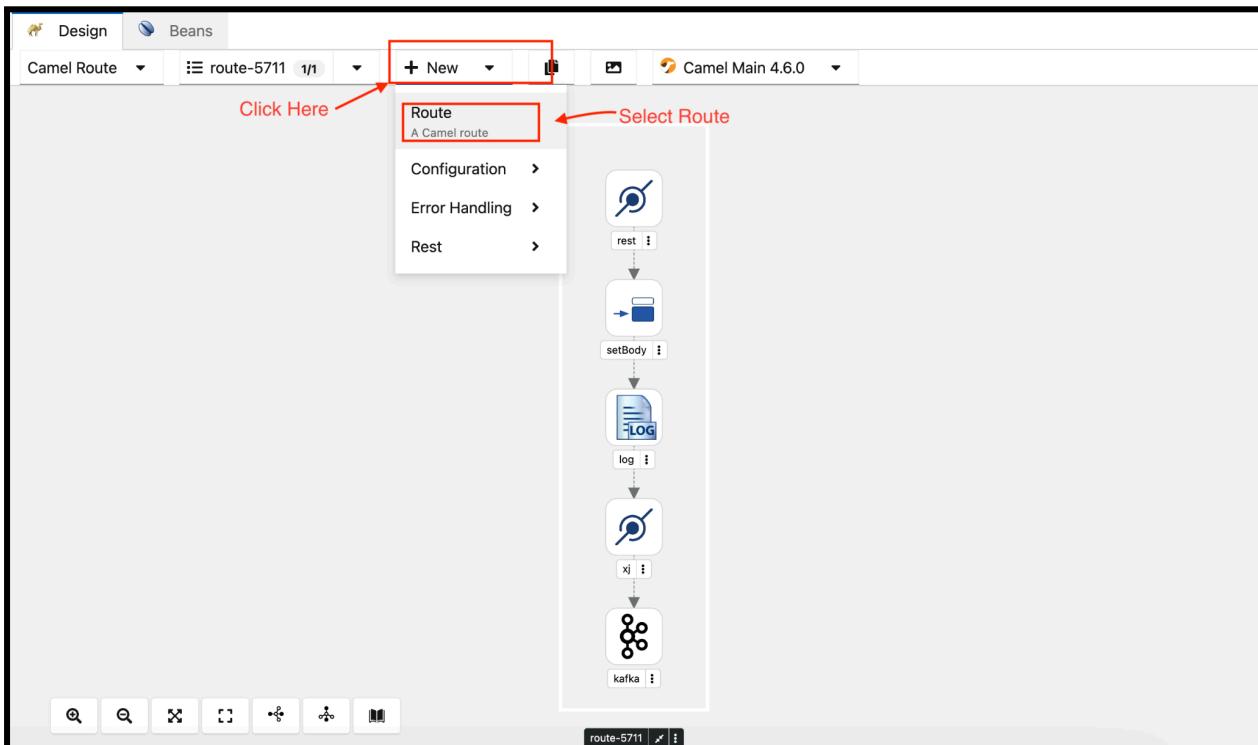
### 9.5 Verify the Output

AMQ Stream (Kafka) access will be with the Instructor.

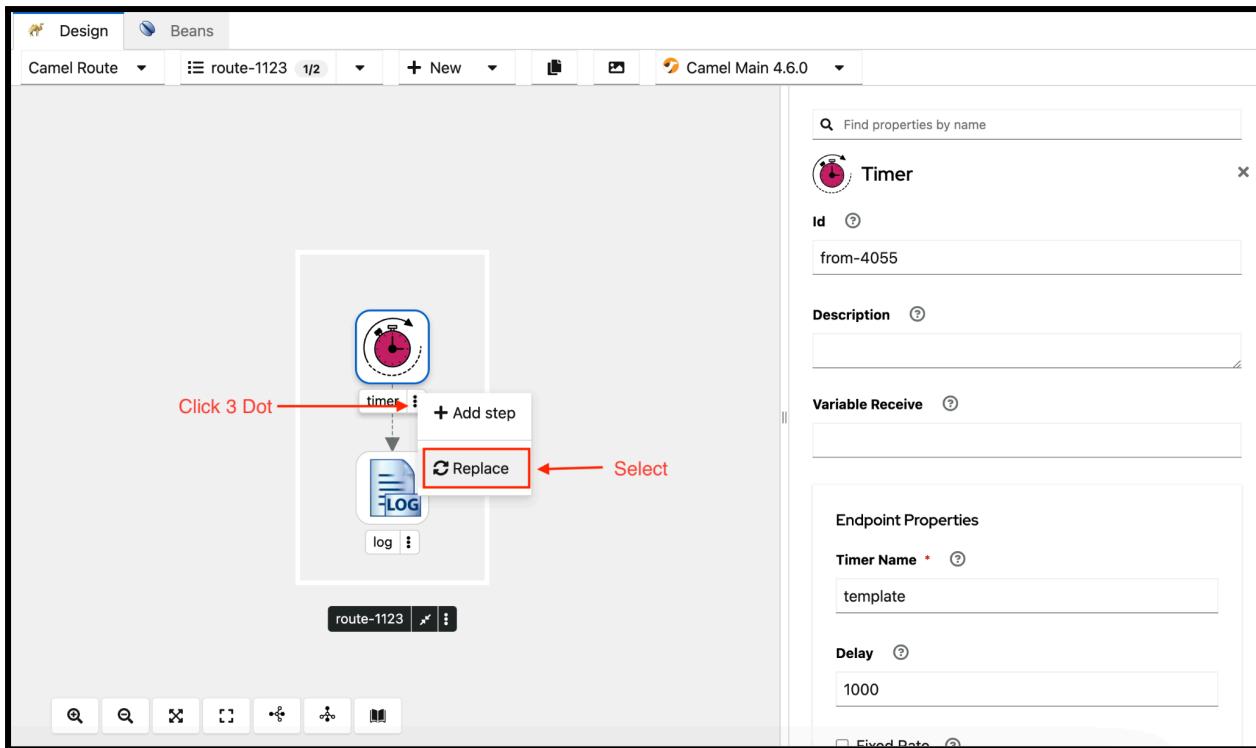
Topic with the <projectName>-json will be created and will have all the messages. The screenshot below is for user2.

## 10 Module 5

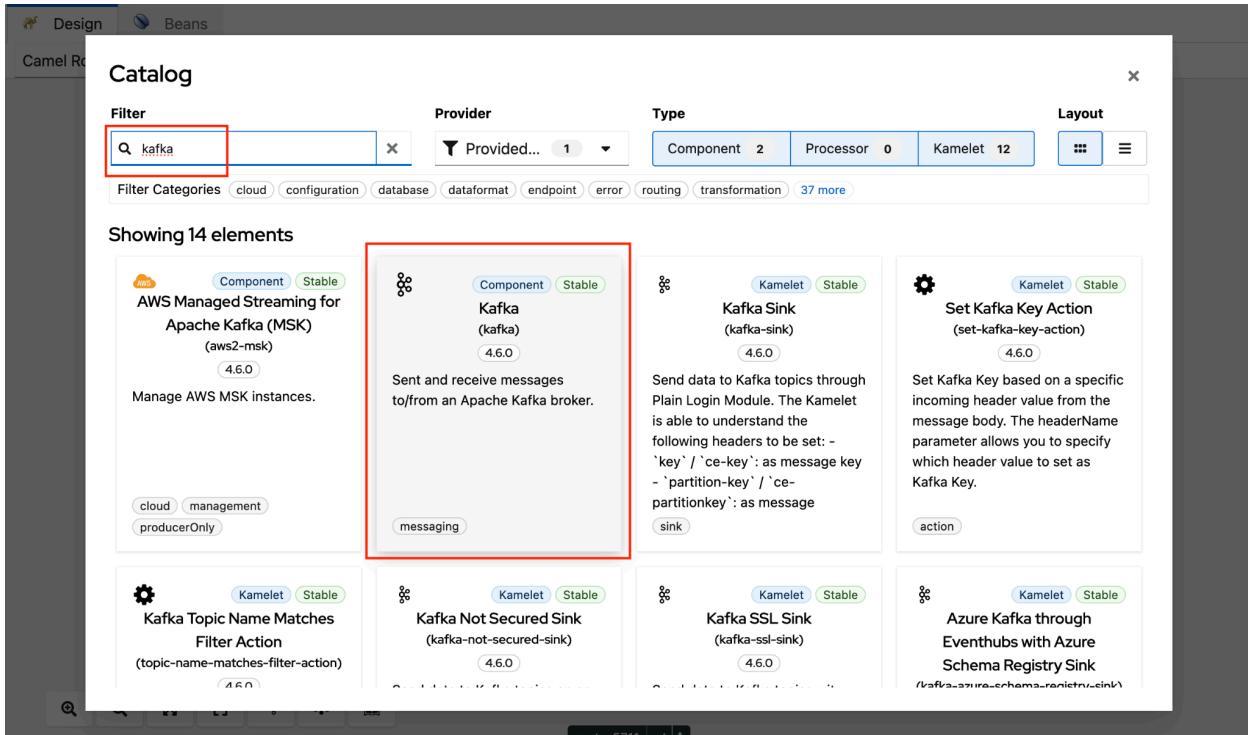
### 10.1. Create a new route



### 10.2. Replace Timer with Kafka component.

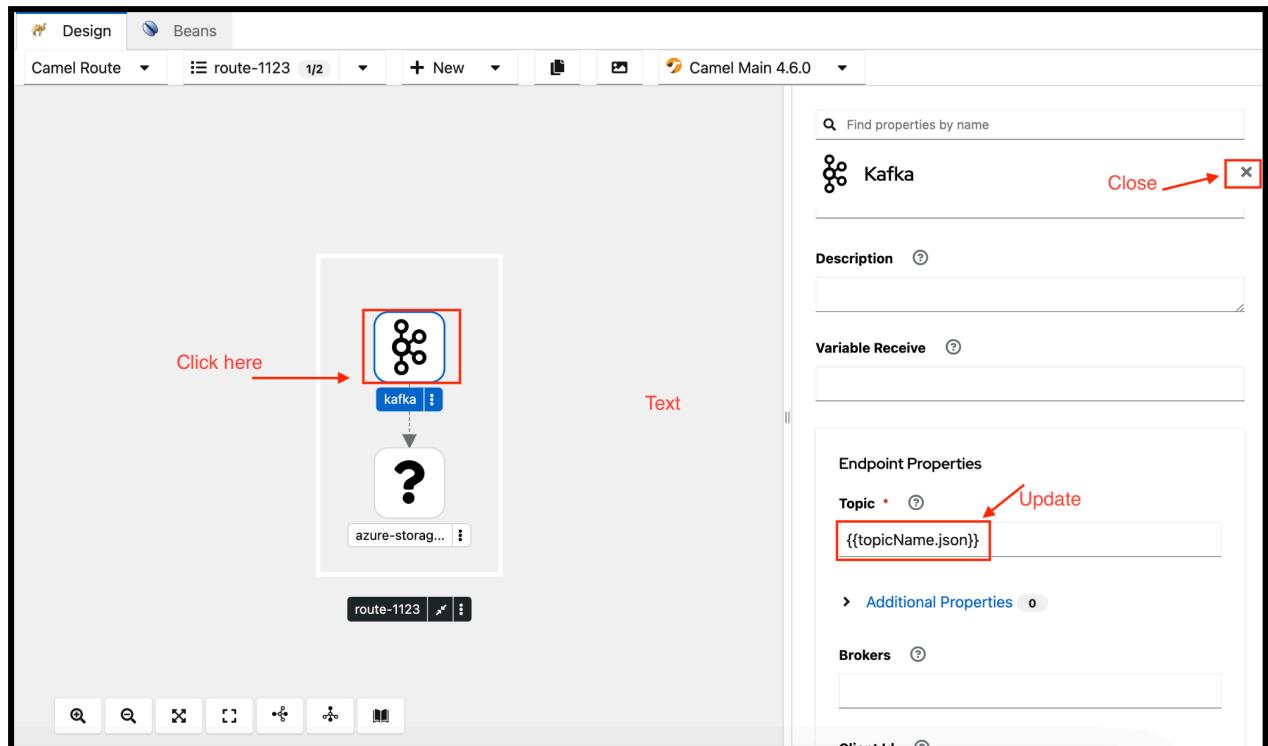


### 10.3.Select the kafka component from the catalog

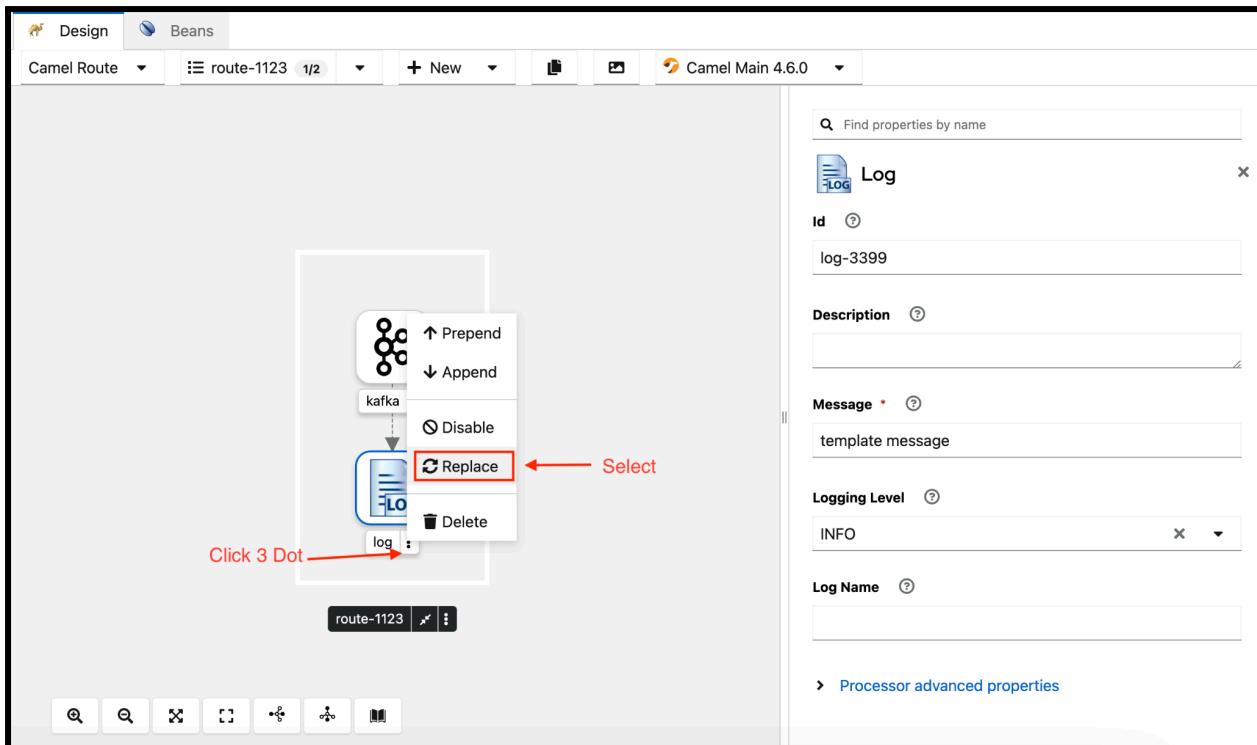


### 10.3.1 Update properties for the Kafka component and close

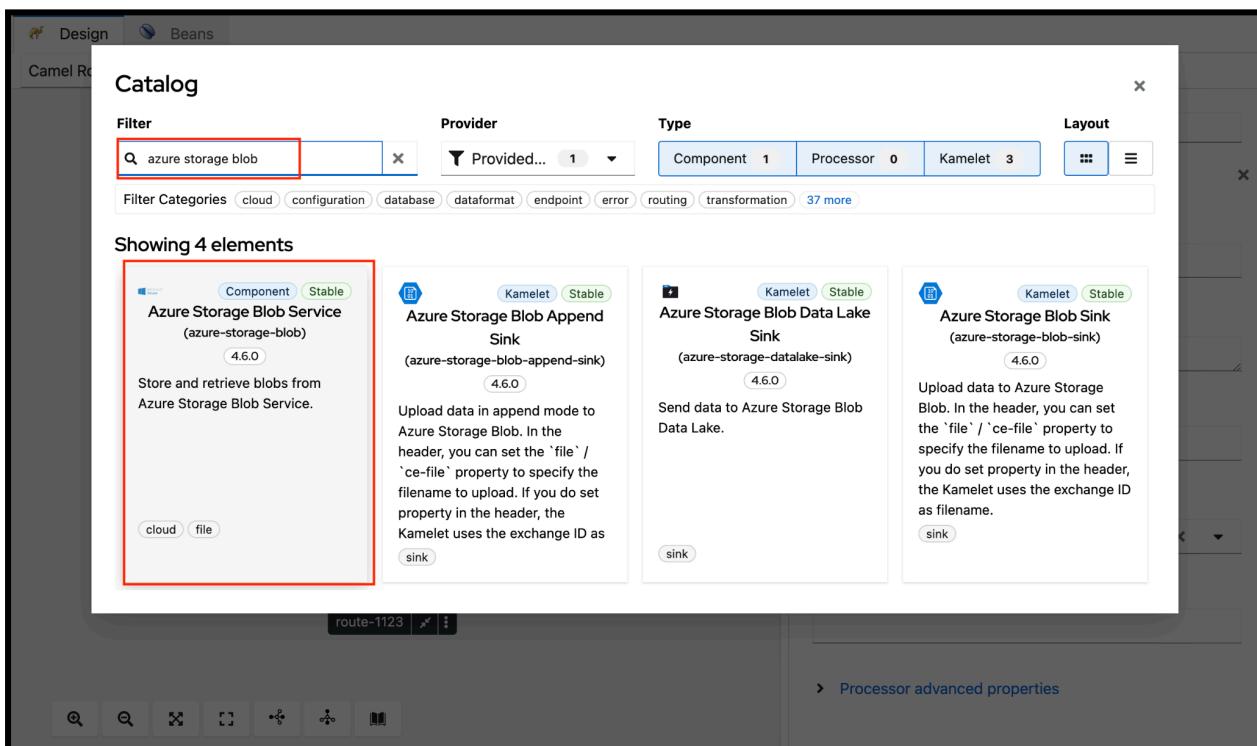
- Topic --> {{topicName.json}}



### 10.4. Replace Log with AzureBlob Storage component.

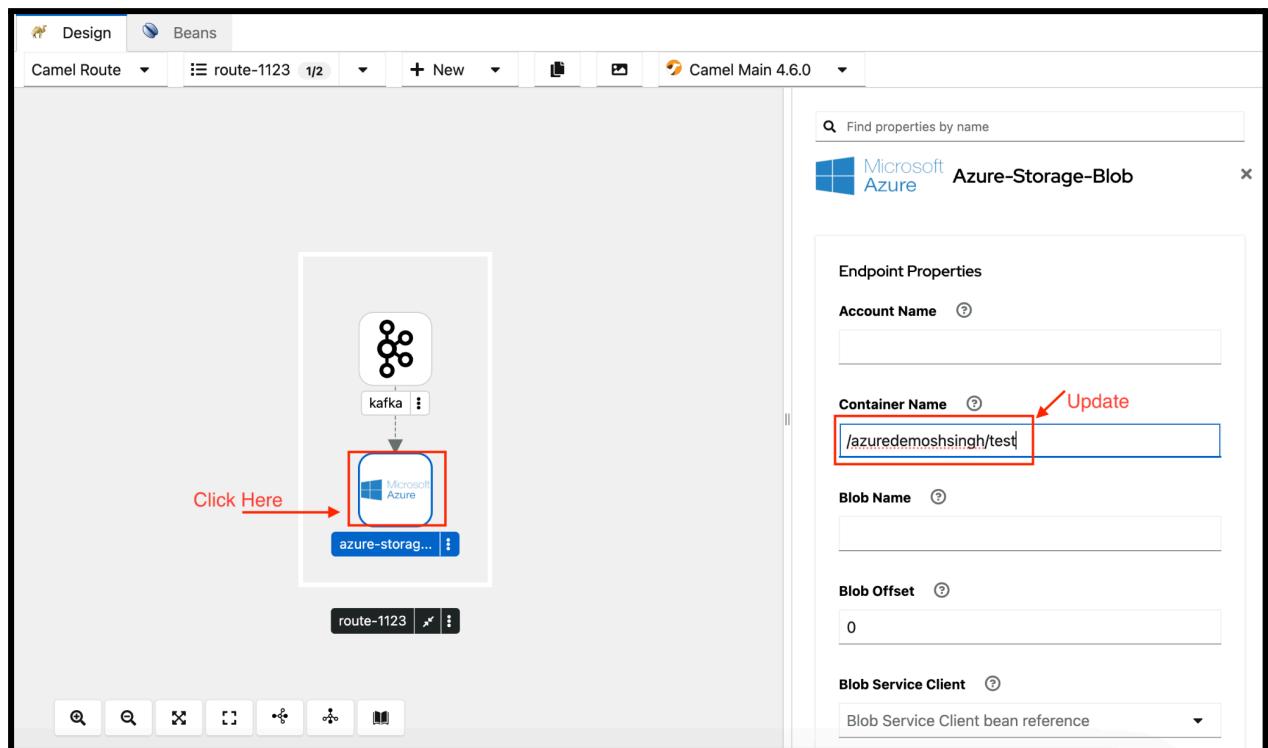


## 10.5. Select the AzureBlob Storage component from the catalog

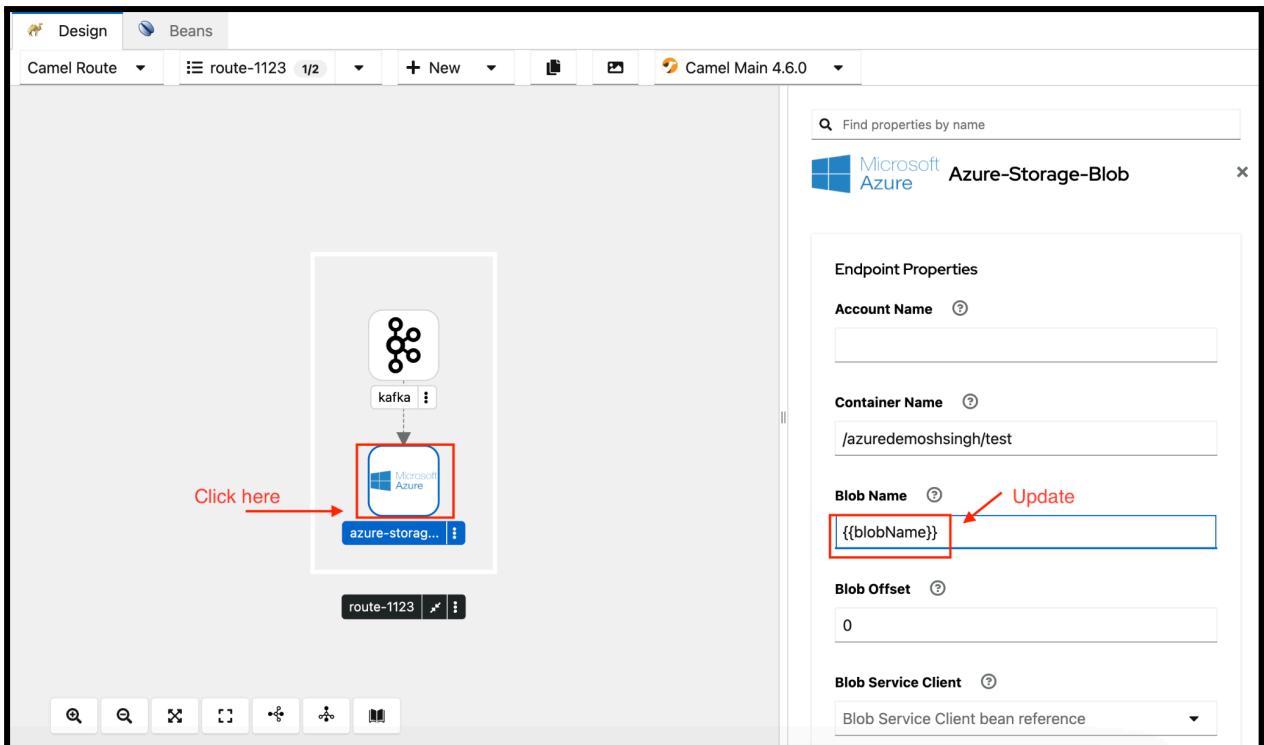


### 10.5.1 Update parameters for the azure-storage-blob component

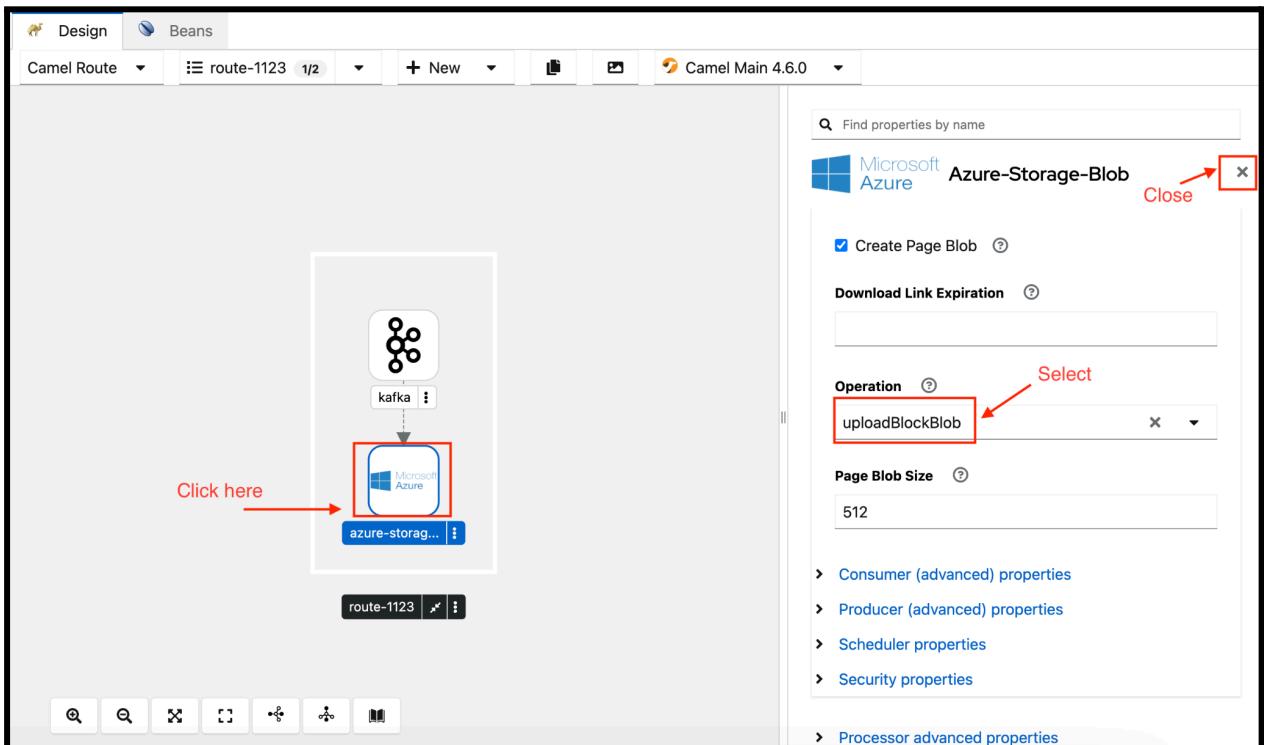
- Container Name ---> /azuredemoshsingh/test



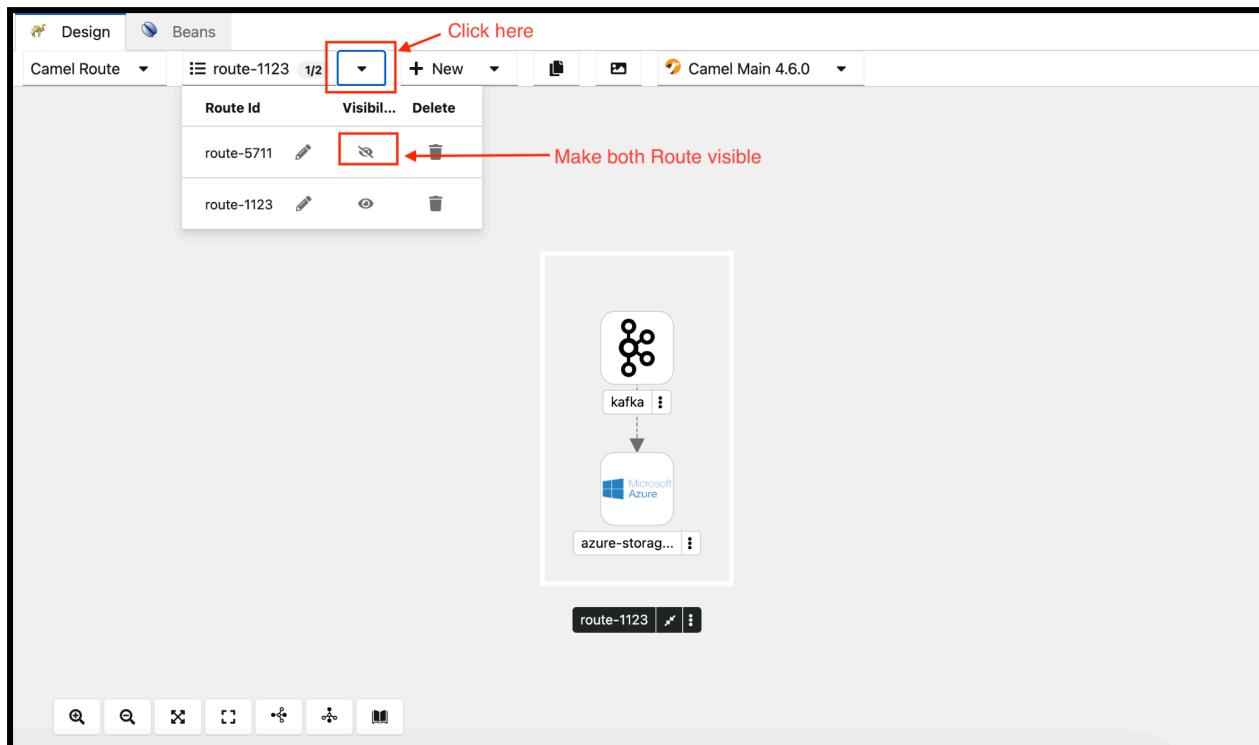
- Blob Name ---> {{blobName}}



- Operation ---> uploadBlockBlob



## 10.6. Select both the route



## 10.7 Commit the Changes

[Follow Module1 steps](#)

## 10.8 Verify the pipeline has been completed successfully.

[Follow Module1 steps](#)

## 10.9 Verify the Output

Azure Storage account access will be with the Instructor.

File with the name <projectName>-xml will be created on the azure storage. The screenshot below is for user2.