Exam : **Salesforce-MuleSoft-Developer-I**

Title : Salesforce Certified MuleSoft Developer (Mule-Dev-201)

Vendor : Salesforce

Version : DEMO

NO.1 What is the main purpose of flow designer in Design Center?

A. To design and develop fully functional Mule applications in a hosted development environment

B. To design API RAML files in a graphical way

C. To design and mock Muleapplication templates that must be implemented using Anypoint Studio

D. To define API lifecycle management in a graphical way

Answer: A

Its primary function is to design and develop fully functional Mule applications in a hosted development environment.

NO.2 Refer to the exhibits.

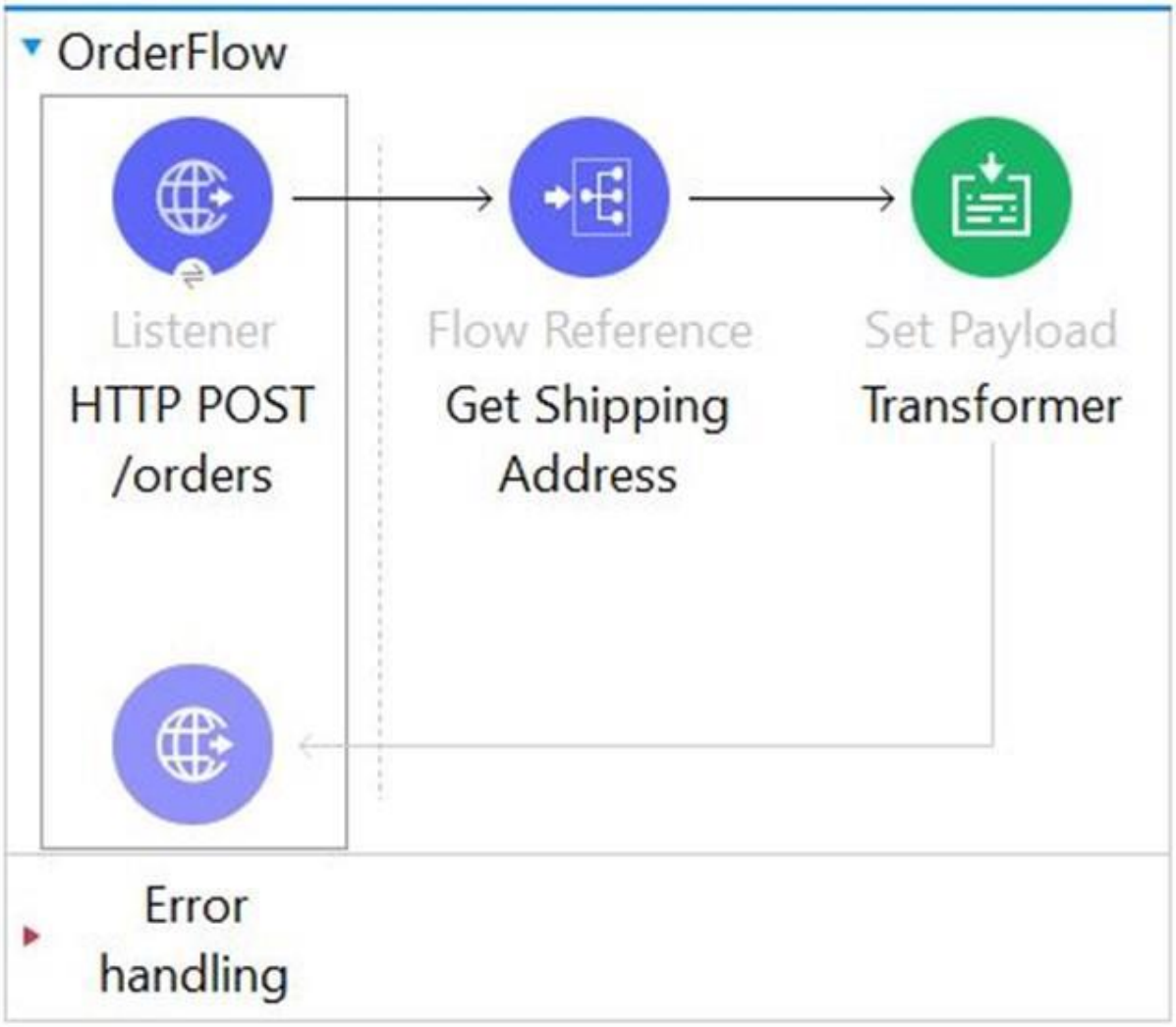
```
{
    "name": "Rohan Kulkarni"
    "order": [
              {"product": "laptop", "amount":"25000", "currency" :"INR"}
             ]
}
```

This main mule application calls a separate flow called as ShippingAddress which returns the address corresponding to the name of the user sent to it as input. Output of this ShippingAddress is stored in a target variable named address.

Next set of requirement is to have a setPayload transformer which will set below two values

1) orderkey which needs to set to be equal to the order element received in the original request payload.

2) addressKey which needs to be set to be equalto the address received in response of ShippingAddress flow What is the straightforward way to properly configure the Set Payload transformer with the required data?




```
<flow name="OrderFlow" doc:id="db8246ce-55ce-4e71-83c7-55c2a256aaba" >
    <http:listener doc:name="HTTP POST /orders" doc:id="4abfbf51-a921-45af-b162-f97508def591" config-ref="HTTP_Listener_config"
    path="/orders" allowedMethods="POST"/>
    <flow-ref doc:name="Get Shipping Address" doc:id="efb25eef-1ba1-4318-b550-71e5e27698ad" name="ShippingAddress" target="address"/>
    <set-payload value='#[output application/json
---
{
orderkey: "?????",
addresskey: "?????"
}]'_doc:name="Transformer" doc:id="3ce06091-db7d-4462-9a6f-1713c4969445" />
</flow>
```

A mule application is being developed which will process POST requests coming fromclients containing the name and order information. Sample request is as below

A. 1. 1. {2. 2. orderkey: "payload.order",3. 3. addresskey: "vars.address"4. 4. }

B. 1. 1. {2. 2. orderkey: "attributes.shippingaddress.order",3. 3. addresskey: "payload"4.}

C. 1. 1. {2. 2. orderkey: "payload.order",3. 3. addresskey: "address"4. }

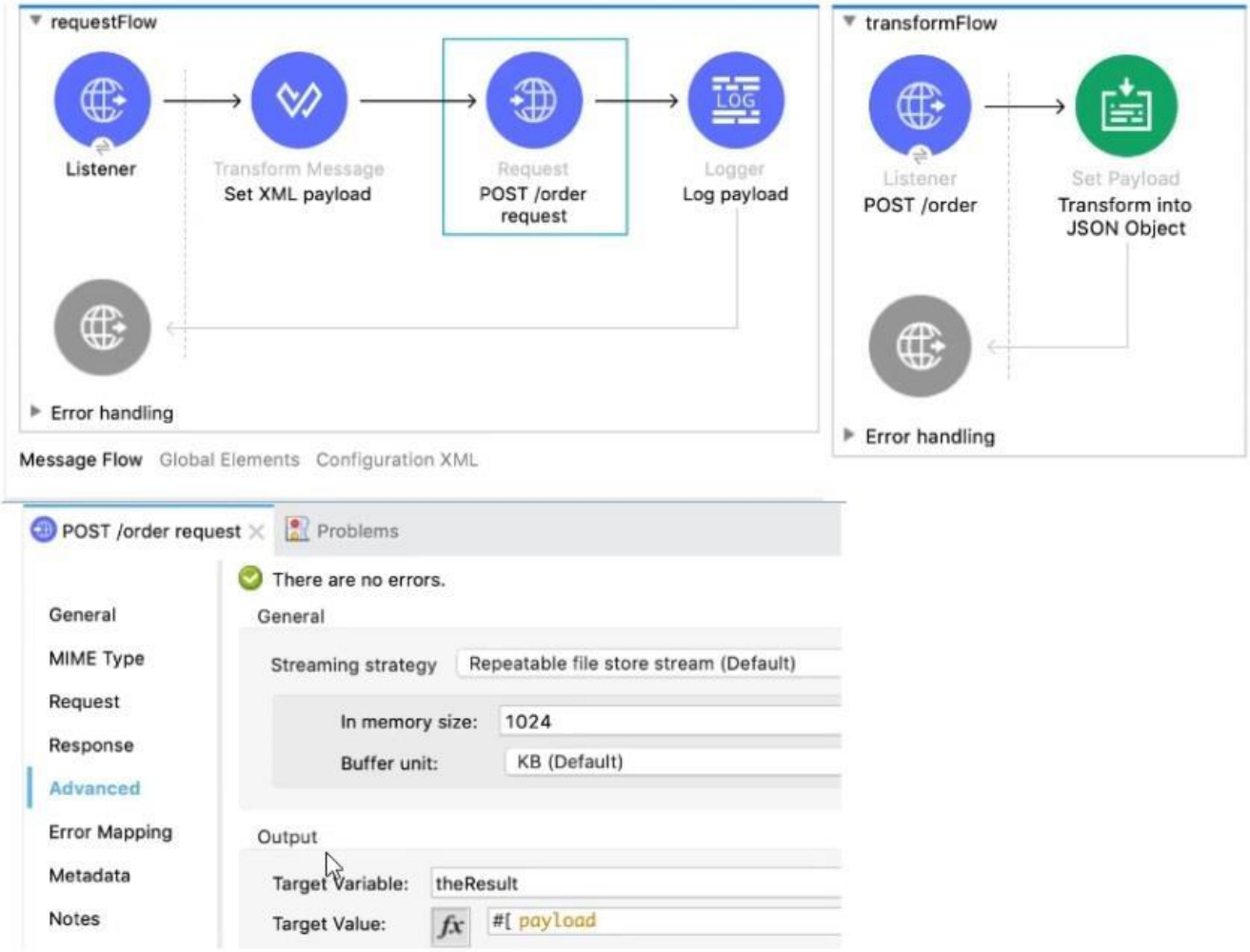
D. 1. 1. {2. 2. orderkey: "attributes.order",3. 3. addresskey: "vars.address"4. }

Answer: A

Correct answer is as below. In this case address will be stored in a variable. Hence payload will not be overwritten and will contain order details

{
orderkey: "payload.order",
addresskey: "vars.address"
}

NO.3 Refer to the exhibits.



In the requestFlow an HTTP Request operation is configured to send an HTTP request with an XML payload.
The request is sent to the HTTP Listener in the transform Flow.
That flow transforms the incoming payload into JSON format and returns the response to the HTTP request.
The response of the request is stored in a target variable named the Result.
What is the payload at the Logger component after the HTTP Request?

A. A non-empty Java object

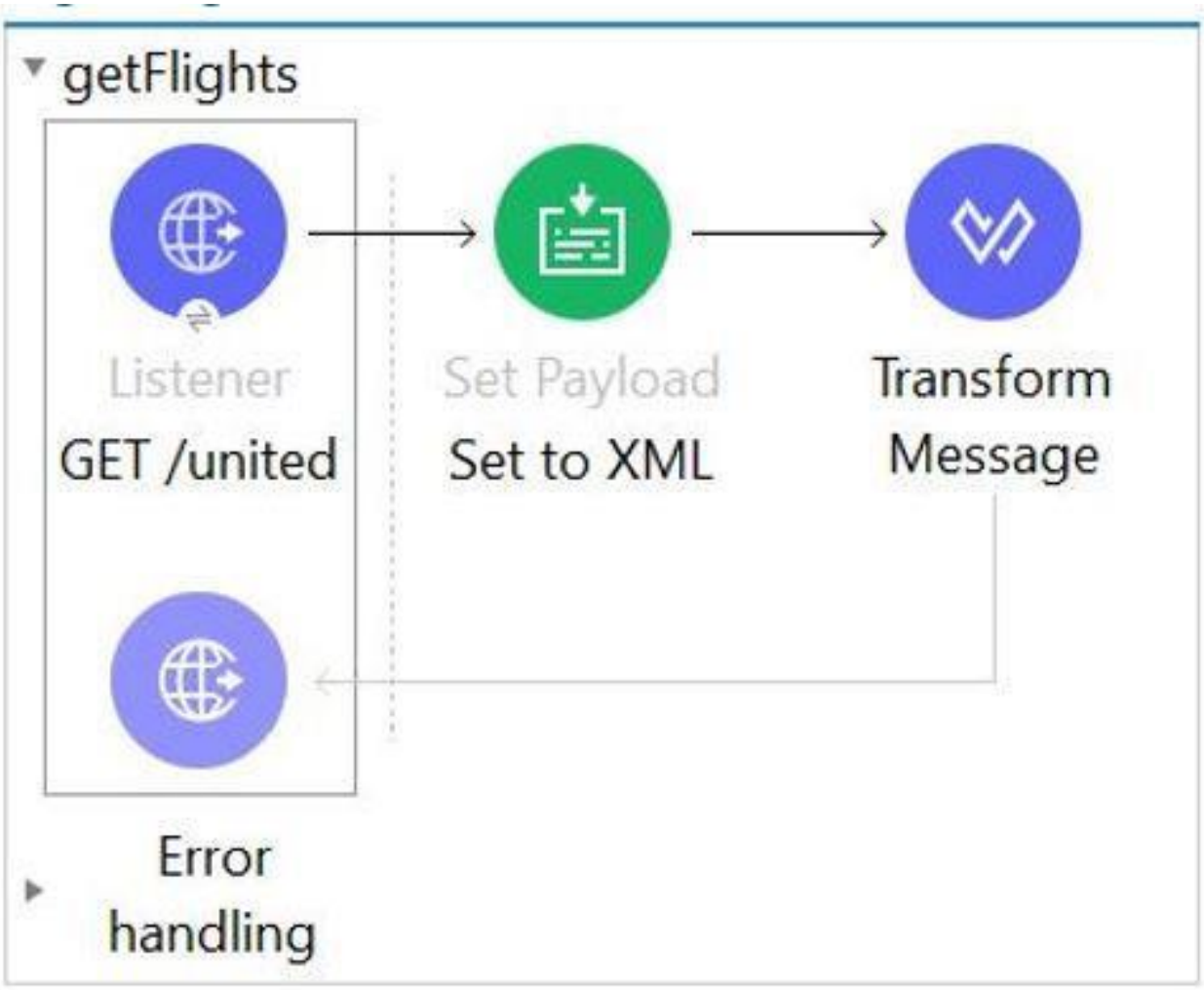
B. The original XML payload

C. null

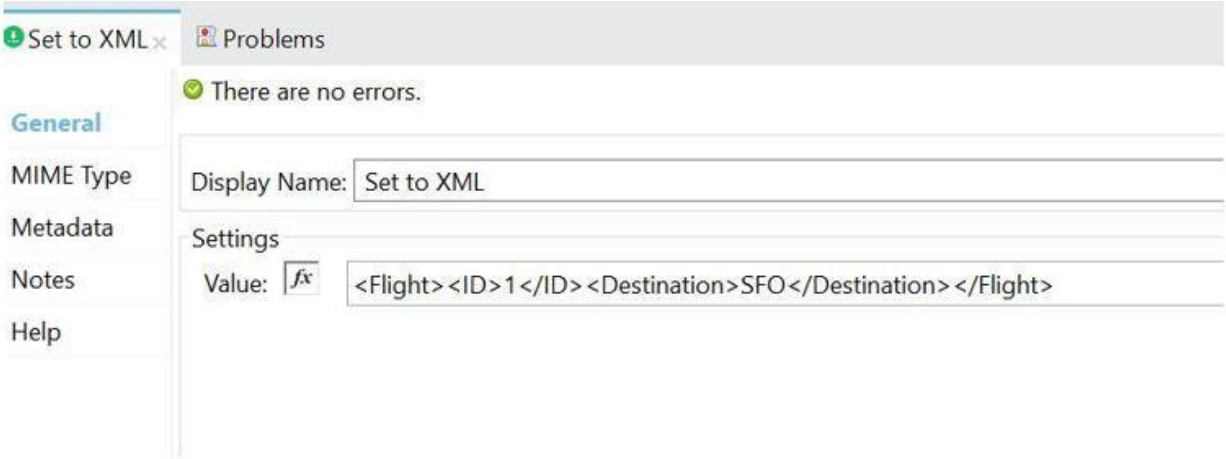
D. The returned JSON response

Answer: B

NO.4 Refer to the exhibits.
A web client submits a request to below flow. What is the output at the end of the flow?
Diagram Description automatically generated



Graphical user interface, text, application, email Description automatically generated



Graphical user interface, text, application, email Description automatically generated

```
    <flow name="getFlights" doc:id="f89731e6-ea03-4d69-a66c-df99584e7c08" >
        <http:listener doc:name="GET /united" doc:id="f331134b-91f8-4c18-8b98-7d5a8ecd4c5b" config-ref="
        <set-payload value="&lt;Flight&gt;&lt;ID&gt;1&lt;/ID&gt;&lt;Destination&gt;SFO&lt;/Destination&g
        <ee:transform doc:name="Transform Message" doc:id="3de3829e-843c-41cf-94fd-5b2ead45f9af" >
            <ee:message >
                <ee:set-payload ><![CDATA[%dw 2.0
output application/json
---
typeOf(payload)
]]></ee:set-payload>
            </ee:message>
        </ee:transform>

</flow>
```

A. String

B. Object

C. Java

D. XML

Answer: A

String is the correct answer as XML is of an Object type String

NO.5 What is output of Dataweave flatten function?

A. Object

B. Map

C. Array

D. LInkedHashMap

Answer: C

Correct answer isArray.

Flatten turns a set of subarrays (such as [[1,2,3], [4,5,[6]], [], [null]]) into a single, flattened array (such as [

1, 2, 3, 4, 5, [6], null]).

This example defines three arrays of numbers, creates another array containing those three arrays, and then uses the flatten function to convert the array of arrays into a single array with all values.

Source

%dw 2.0

output application/json

var array1 = [1,2,3]

var array2 = [4,5,6]

var array3 = [7,8,9]

var arrayOfArrays = [array1, array2, array3]

---

flatten(arrayOfArrays)

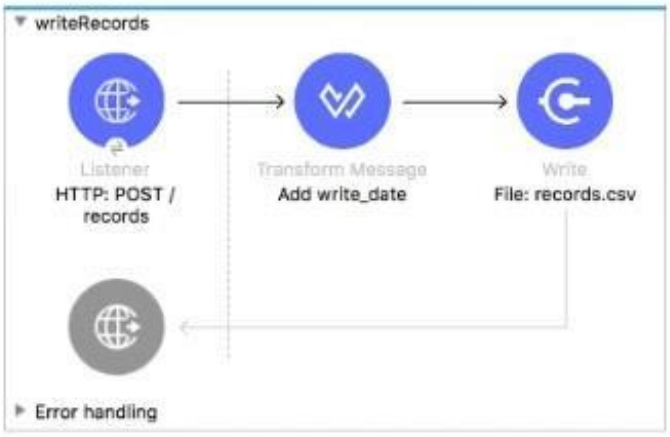
Output

[1,2,3,4,5,6,7,8,9]

NO.6 Refer to the exhibits.

Payload

```
{
    "transaction_id": "SS-4848-44KK-4SYQ",
    "account_id": "KA-382-SKD44",
    "name": "Max Mule",
    "position": "sell"
}
```




```
<flow name="writeRecords" >
    <http:listener doc:name="HTTP: POST /records" config-ref="HTTP_Listener_config"
        path="/records" allowedMethods="POST"/>
    <ee:transform doc:name="Add write_date">
        <ee:message >
            <ee:set-payload ><![CDATA[%dw 2.0
            output application/json
            ---
            payload ++ {"write_date": now()}]]>
            </ee:set-payload>
        </ee:message>
    </ee:transform>
    <file:write doc:name="File: records.csv" path="file-store/records.csv">
        <file:content ><![CDATA[#[payload]]]></file:content>
    </file:write>
</flow>
```

What is written to the records.csv file when the flow executes?

A. The JSON payload

B. An error message

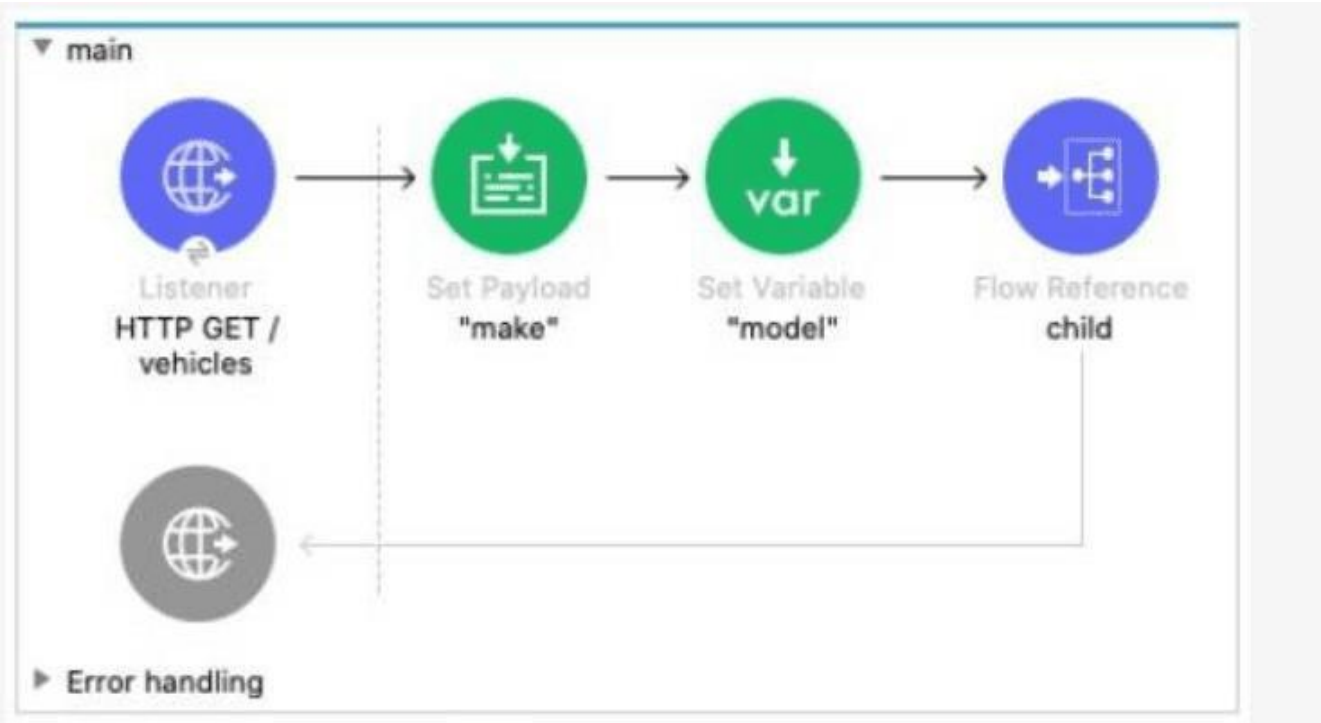
C. Nothing

D. The payload convert to CVS

Answer: A

Transform Message Add write_date is coverting payload in JSON format and same JSON payload is avaialble to file write processor. However, if the payload is a different format (for example, not CSV) , you can place the transformation inside the Write operation to generate content that will be written without producing a side effect on the message in transit. This is not done in this case. By default, the connector writes whatever is in the message payload. Hence JSON payload will be written to file.

NO.7 Refer to the exhibits.



The main flow contains a Flow Reference component configured to callthe child flow What part(s) of a Mule event passed to the Flow Reference component are available in the child flow?

A. The payload and all attributes

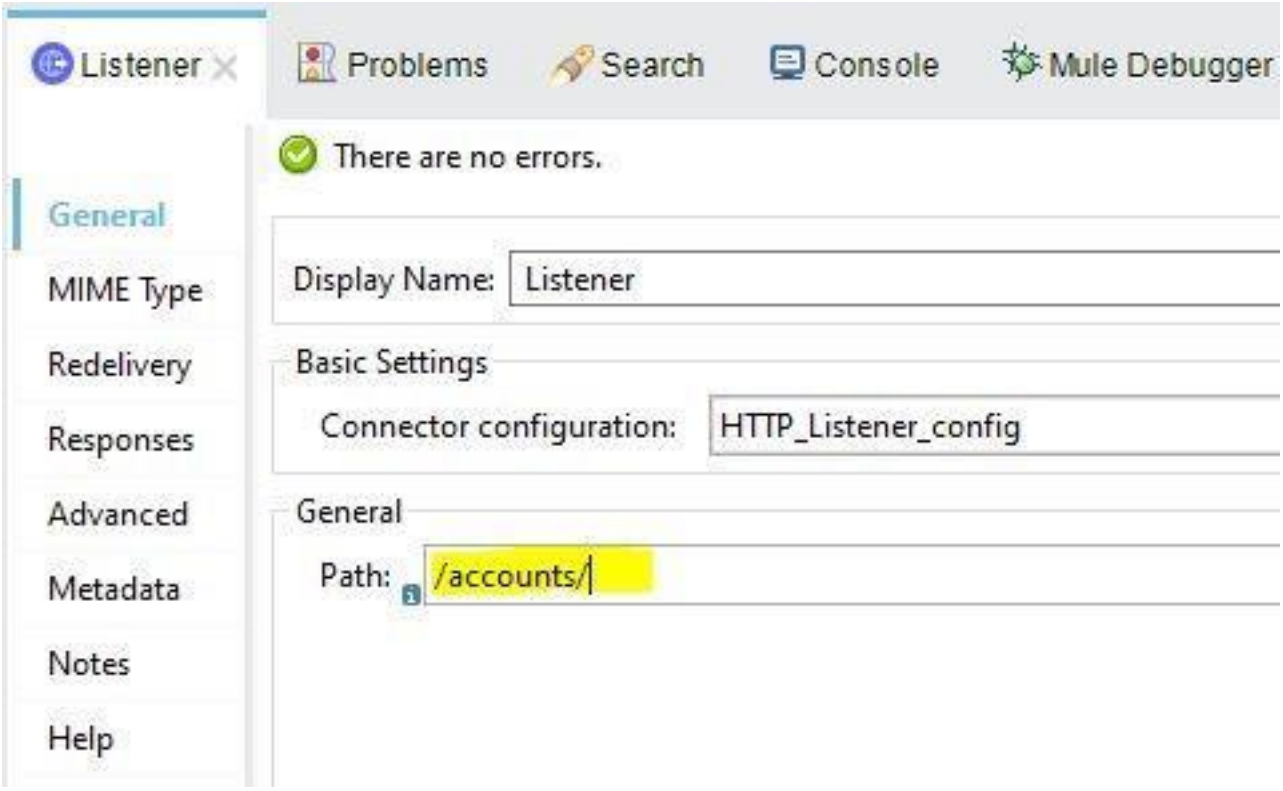
B. The payload and all variables

C. The entire Mule event

D. The payload

Answer: B

NO.8 What is the correct Syntax to add a customer ID as a URI parameter in the HTTP listener's path attribute?

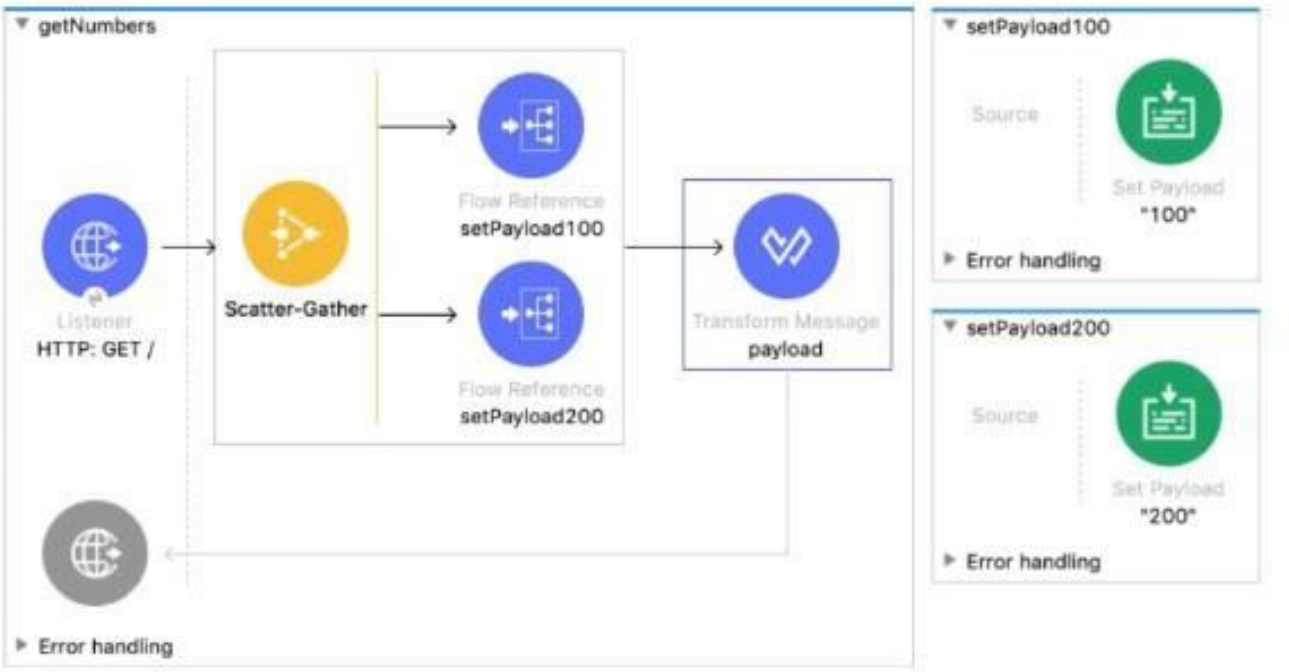


A. #[customerID]
B. $[customerID]
C. {customerID}
D. (customerID)
Answer: C
URL parameters are always accessed using { } like => {customerID}

NO.9 What asset cannot be created using Design Center?
A. Mule Applications
B. API fragments
C. API specifications
D. API portals
Answer: D
API portal are created by API Exchange and cannot be created by Design Center

NO.10 What statement is a part of MuleSoft's description of an application network?
A. Creates and manages high availability and fault tolerant services and infrastructure
B. Creates reusable APIs and assets designed to be consumed by other business units
C. Creates and manages a collection of JMS messaging services and infrastructure
D. Leverages Central IT to deliver complete point-to-point solutions with master data management
Answer: B
Creates reusable APIs and assets designed to beconsumed by other business units

NO.11 Refer to the exhibits.




```
<flow name="getNumbers" >
    <http:listener doc:name="HTTP: GET /" config-ref="HTTP_Listener_config" path="/"/>
    <scatter-gather doc:name="Scatter-Gather" >
        <route >
            <flow-ref doc:name='setPayload100' name='setPayload100'/>
        </route>
        <route >
            <flow-ref doc:name="setPayload200" name="setPayload200"/>
        </route>
    </scatter-gather>
    <ee:transform doc:name="payload">
        <ee:message >
            <ee:set-payload ><![CDATA[%dw 2.0
            output application/json
            ---
            payload]]></ee:set-payload>
        </ee:message>
    </ee:transform>
</flow>
<flow name="setPayload100" ><set-payload value='#["100"]' doc:name='"100"' /></flow>
<flow name="setPayload200" ><set-payload value='#["200"]' doc:name='"200"' /></flow>
```

The input array of strings is processed by the batch job that processes, fitters, and aggregates the values.

What is the last message logged by the Logger component after thebatch job completes processing?

A.

```
[
    {
        "attributes": ...,
        "payload": "100"
    },
    {
        "attributes": ...,
        "payload: "200"
    }
]
```

B.

```
{
    "0": "100",
    "1": "200"
}
```

C.

```
["100", "200"]
```

D.

```
{
    "0": {
        "attributes": ...,
        "payload": "100"
    },
    "1": {
        "attributes": ...,
        "payload: "200"
    }
}
```

Answer: D

NO.12 An HTTP Request operation sends an HTTP request with a non-empty JSON object payload to

an external HTTP endpoint. The response from the external HTTP endpoint returns an XML body. The result is stored in a target named the Result.
What is the payload at the event processor after the HTTP Request?

A. The XML response body

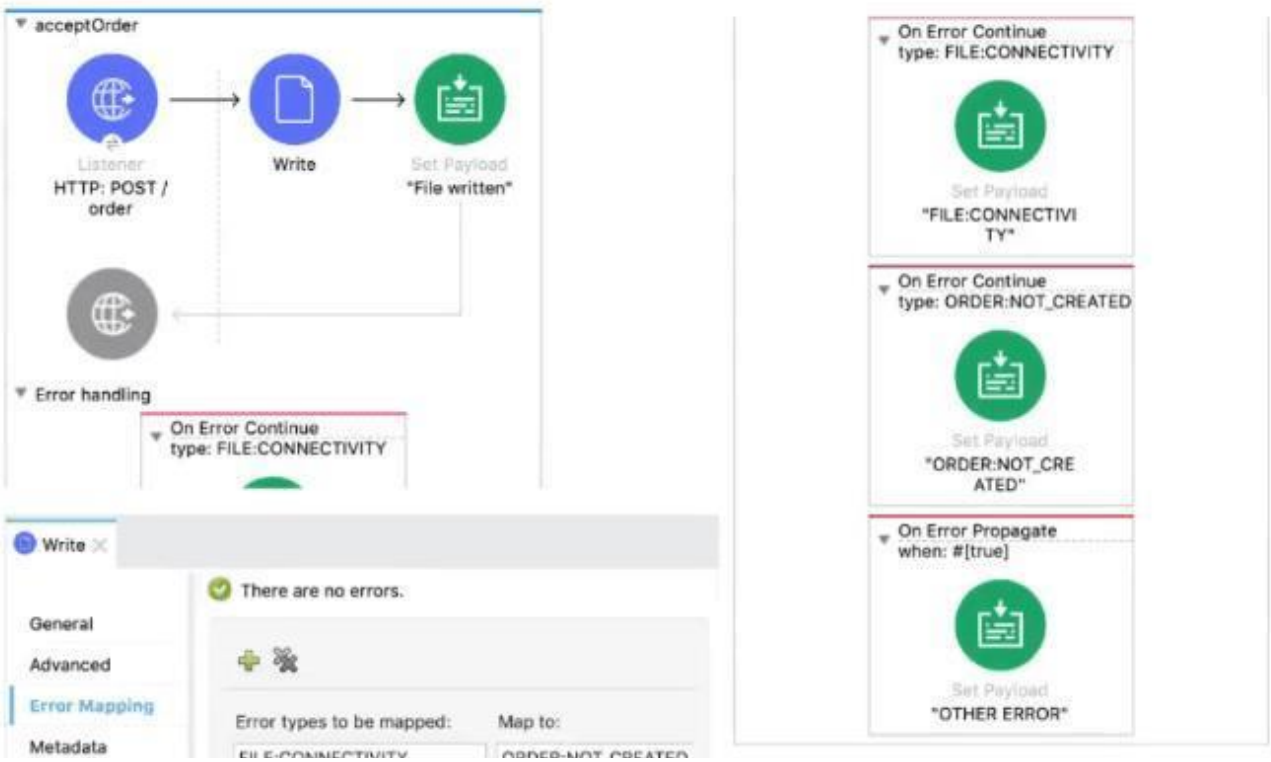
B. null

C. The original JSON request body

D. A non-empty Java object

Answer: C

NO.13 Refer to the exhibits.




```
<flow name="acceptOrder">
    <http:listener doc:name="HTTP: POST /order" config-ref="HTTP_Listener_config"
        path="/order" allowedMethods="POST">
        <http:error-response >
            <http:body ><![CDATA[#[output text/plain --- payload]]]></http:body>
        </http:error-response>
    </http:listener>
    <file:write doc:name="Write" config-ref="File_Config" path="newOrder.json">
        <error-mapping sourceType="FILE:CONNECTIVITY" targetType="ORDER:NOT_CREATED" />
        <file:content ><![CDATA[#[output appliation/json --- payload]]]></file:content>
    </file:write>
    <set-payload value='#["File written"]' doc:name='"File written"' />
```

A web client sends a POST request with the payload {"oid": "1000", "itemid": "AC200", "qty": "4" } to the Mule application. The File Write operation throws a FILE:CONNECTIVITY error.
Whatresponse message is returned to the web client?

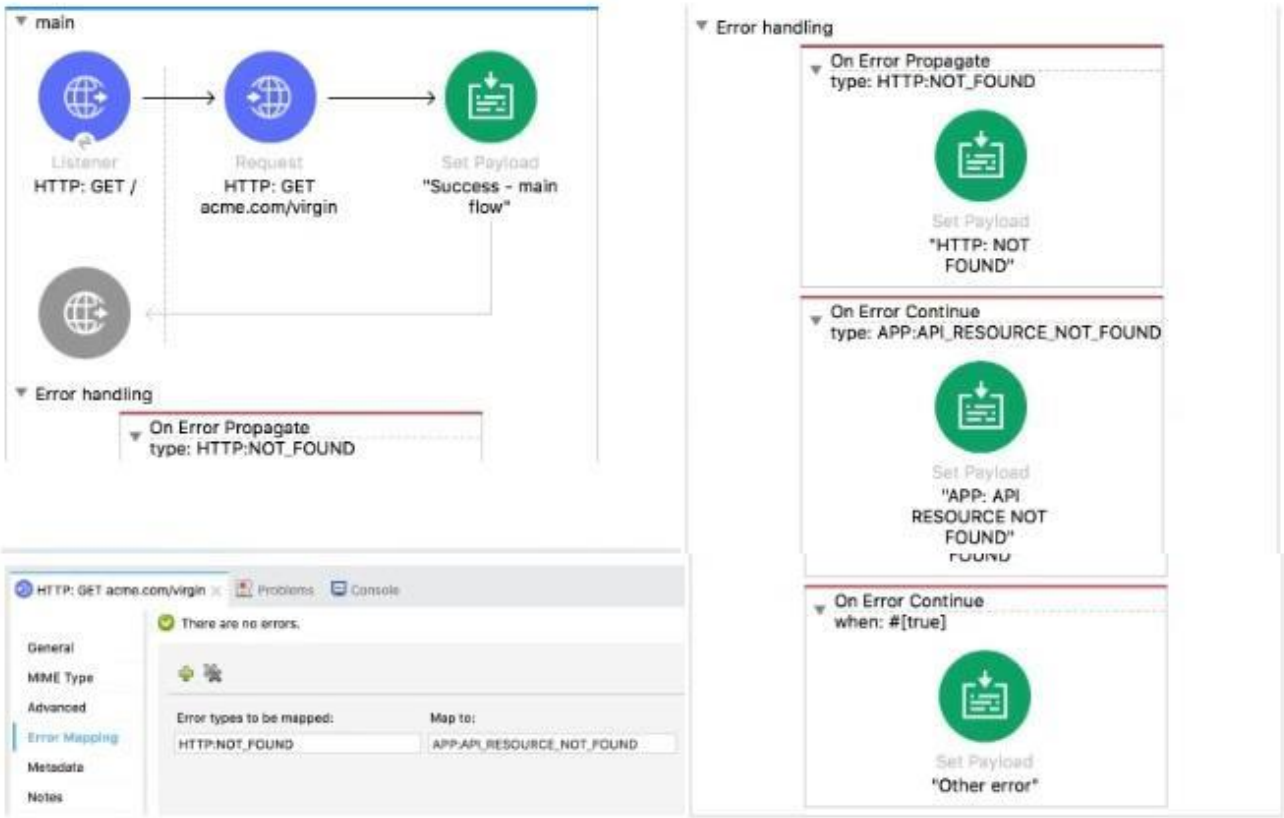
A. "FILE:CONNECnvnY'

B. "ORDER:NOT_CREATED"

C. "OTHER ERROR"

D. "File written"
Answer: B

NO.14 Refer tothe exhibit.



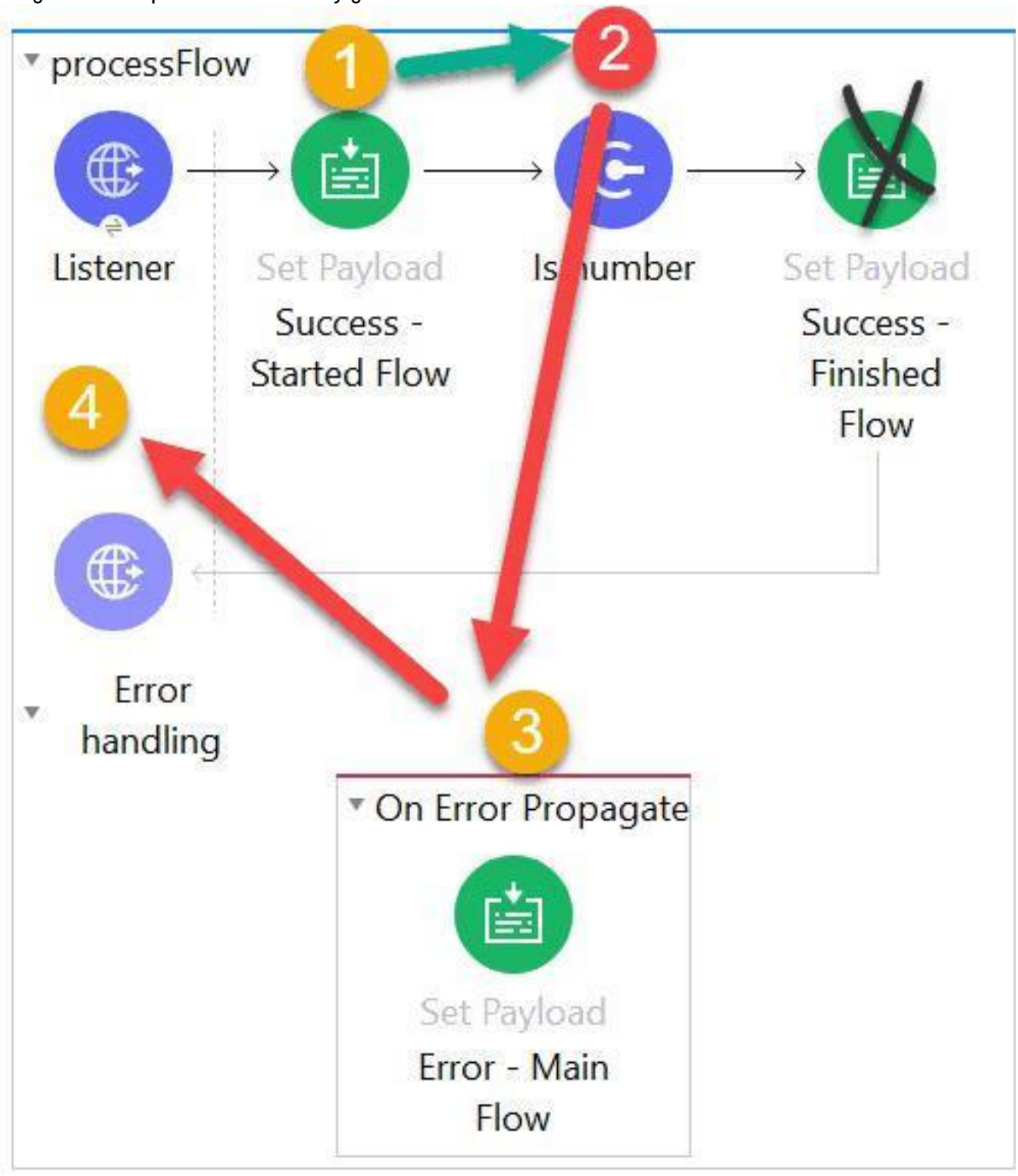

```
<http:listener-config name="HTTP_Listener_config" doc:name="HTTP Listener config">
    <http:listener-connection host="0.0.0.0" port="8081" />
</http:listener-config>

<flow name="main">
    <http:listener doc:name="HTTP: GET /" config-ref="HTTP_Listener_config" path="/" />
    <http:request method="GET" doc:name="HTTP: GET acme.com/virgin" url="http://acme.com/virgin" >
        <error-mapping sourceType="HTTP:NOT_FOUND" targetType="APP:API_RESOURCE_NOT_FOUND" />
    </http:request>
```

The main flow is configured with their error handlers. A web client submit a request to the HTTP Listener and the HTTP Request throws an HTTP:NOT_FOUND error.
What response message is returned?''
What response message is returned?
A. APP:API RESOURCE NOT FOUND
B. HTTP: NOT FOUND
C. other error
D. success - main flow
Answer: A
Correct answer is APP: API RESOURCE NOT FOUND

--
1) A web client submits the request to the HTTP Listener.
2) The HTTP Request throws an "HTTP:NOT_FOUND" error, execution halts.

3) The On Error Propagate error Handler handles the error. In this case ,HTTP:NOT_FOUND error ismapped to custom error APP:API_RESOURCE_NOT_FOUND. This error processor sets payload to APP:
API_RESOURCE_NOT_FOUND.
4) "APP:API_RESOURCE_NOT_FOUND. " is the error message returned to the requestor in the body of the HTTP request with HTTP Status Code: 500 Reference Diagram:
Diagram Description automatically generated with medium confidence

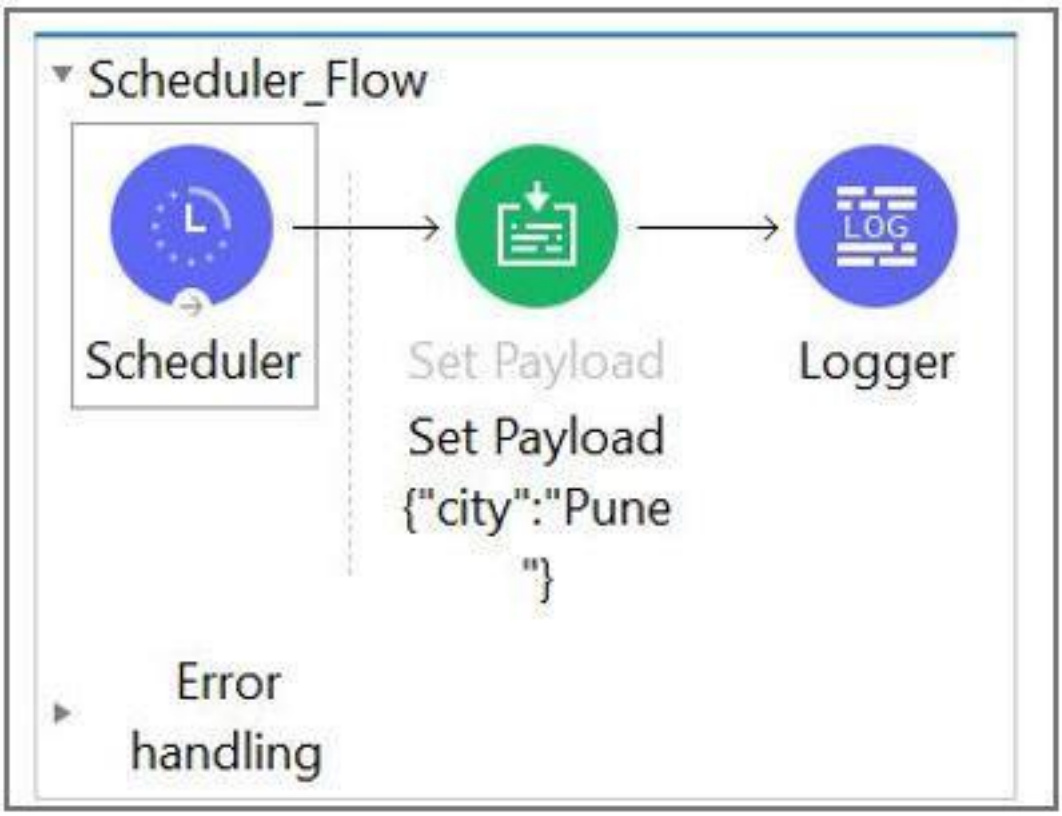


NO.15 A Database On Table Row listener retrieves data from a CUSTOMER table that contains a

primary key userjd column and an increasing kxjin_date_time column. Neither column allows duplicate values.
How should the listener be configured so it retrieves each row at most one time?
A. Set the watermark column to the bgin_date_time column
B. Set the target value to the last retrieved login_date_time value
C. Set the target value to thelast retrieved user_jd value
D. Set the watermark column to the user_Id column
Answer: A
* Watermark allows the poll scope to poll for new resources instead of getting the same resource over and over again.
* The database table must be ordered so that the"watermark functionality" can move effectively in the ordered list. Watermark stores the current/last picked up "record id."
* If the Mule application is shut down, it will store the last picked up "record id" in the Java Object Store and the data will continue to exist in the file. This watermark functionality is valuable and enables developers to have increased transparency.
* Developers do not need to create code to handle caching; it is all configurable!
* There are two columns and both are unique butuser_id can't guaranty sequence whereas date_time will always be in increasing order and table content can easily be ordered on the basis of last processed date_time.
So correct answer is: Set the watermark column to the date_time column

NO.16 The new RAML spec has been published to Anypoint Exchange with client credentials.
What is the next step to gain access to the API?
A. Request access to the API in Anypoint Exchange
B. Email the owners of the API
C. Create a new client application
D. No additional steps needed
Answer: A
Correct answer is Request access to the API in Anypoint Exchange. This way we can get clientId and Client secret which we can use to access the API

NO.17 Refer to exhibits.



What message should be added to Logger component so that logger prints "The city is Pune" (Double quote should not be part of logged message)?

A. #["The city is" ++ payload.City]

B. The city is + #[payload.City]

C. The city is #[payload.City]

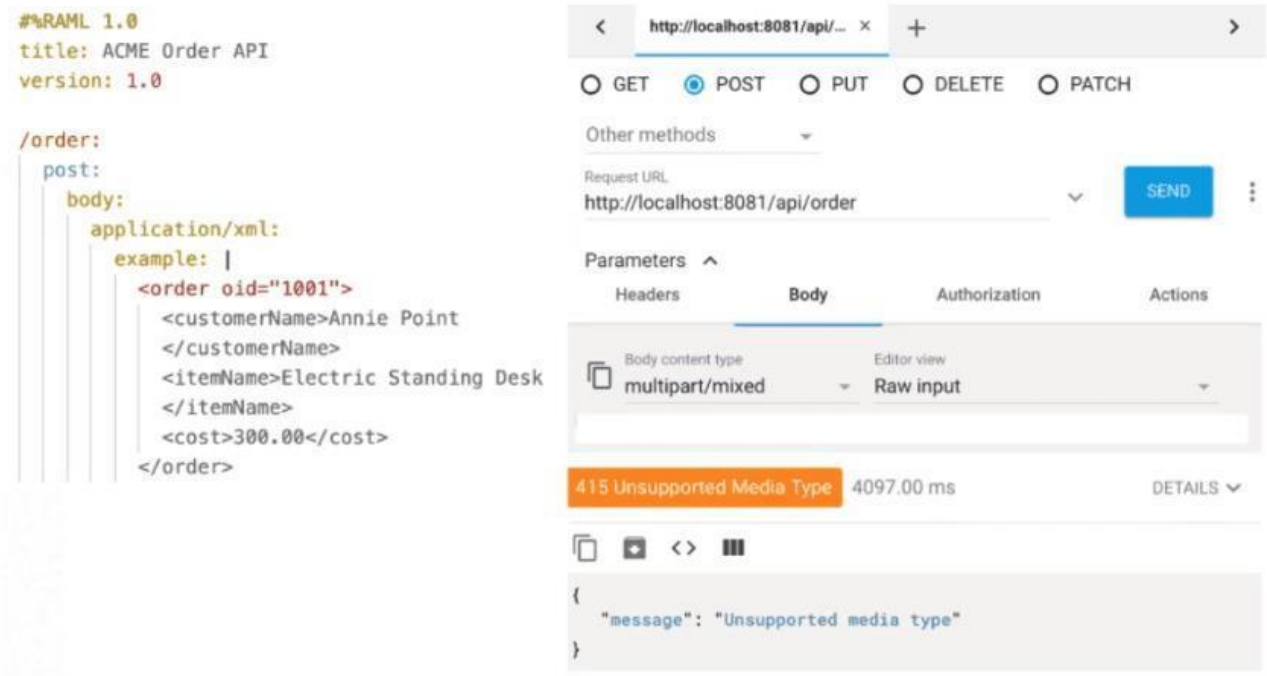
D. #[The city is ${payload.City}

Answer: C

Correct answer is The city is #[payload.City]

Answer can get confused with the option #["The city is" ++ payload.City] But note that this option will not print the space between is and city name. This will print The city isPune

NO.18 Refer to the exhibits.



The web client sends a POST request to the ACME Order API with an XML payload. An error is returned.
What should be changed in the request so that a success response code is returned to the web client?

A. Set a request header with the name Content-Type to a value of applicatron/octet-stream

B. Set a request header with the name Content-Type to a value of application/xml

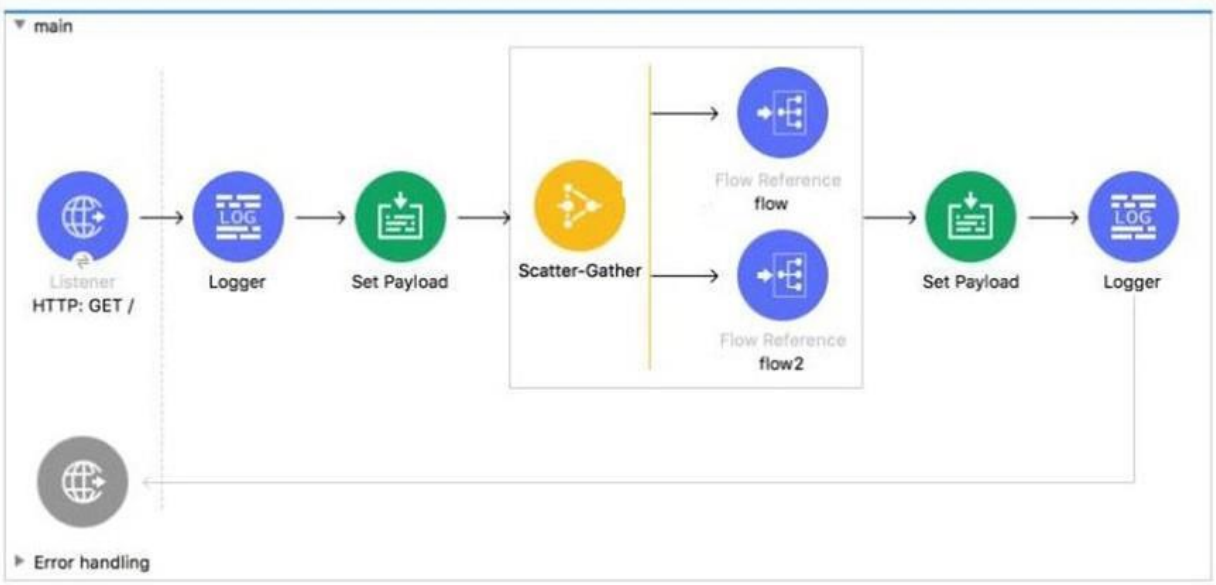
C. Set a response header with the name Content-Type to a value of applkation/xml

D. Set a response header with the name Content-Type to a value of application/octet-stream

Answer: B

The HTTP 415 Unsupported Media Type client error response codeindicates that the server refuses to accept the request because the payload format is in an unsupported format. The format problem might be due to the request's indicated Content-Type or Content-Encoding , or as a result of inspecting the data directly. Asper RAML input is expected in application/xml.
Hence correct answer is Set a request header with the name Content-Type to a

NO.19 Refer to exhibits.



In the execution of the Scatter-Gather , the flow route completes after 10 seconds and the flow2 route completes in 40 seconds. How many seconds does it take for the Scatter-Gather to complete?

A. 10

B. 50

C. 40

D. 20

Answer: C

Scatter-Gather sends the event to each routes concurrently. Hence both route in this example will start in parallel. So total time to complete processing is 40 seconds The Scatter-Gather component is a routing event processor thatprocesses a Mule event through different parallel processing routes that contain different event processors. Each route receives a reference to the Mule event and executes a sequence of one or more event processors. Each of these routes uses a separate thread to execute the event processors, and the resulting Mule event can be either the same Mule event without modifications or a new Mule event with its own payload, attributes, and variables. The Scatter-Gather component then combines the Mule events returned by each processing route into a new Mule event that is passed to the next event processor only after every route completes successfully.

The Scatter-Gather component executes each route in parallel, not sequentially. Parallel execution of routes can greatly increase the efficiency of your Mule application and may provide more information than sequential processing.

Mule Ref Doc : https://docs.mulesoft.com/mule-runtime/4.3/scatter-gather-concept

NO.20 There are three routes configured for Scatter-Gather and incoming event has a payload is an Array of three objects. How routing will take place in this scenario?

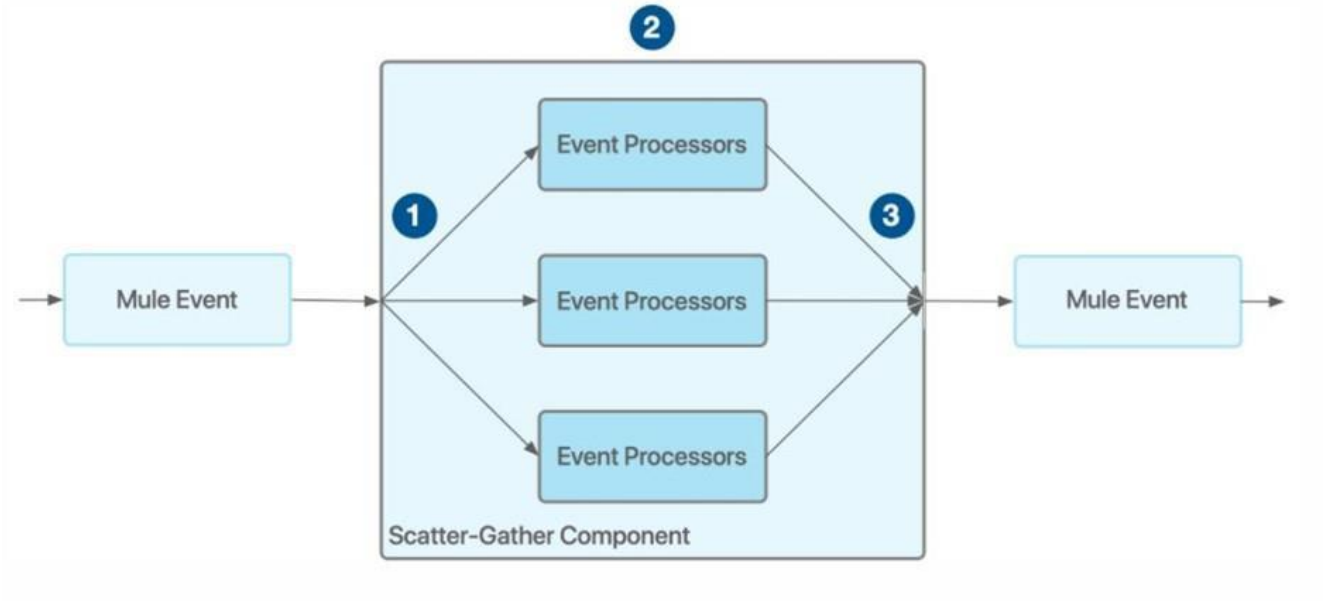
A. Incoming array objects would be split into three and each part would be sent to one route each in sequential manner

B. Incoming array objects would be split into three and each part would be sent to one route each in parallel

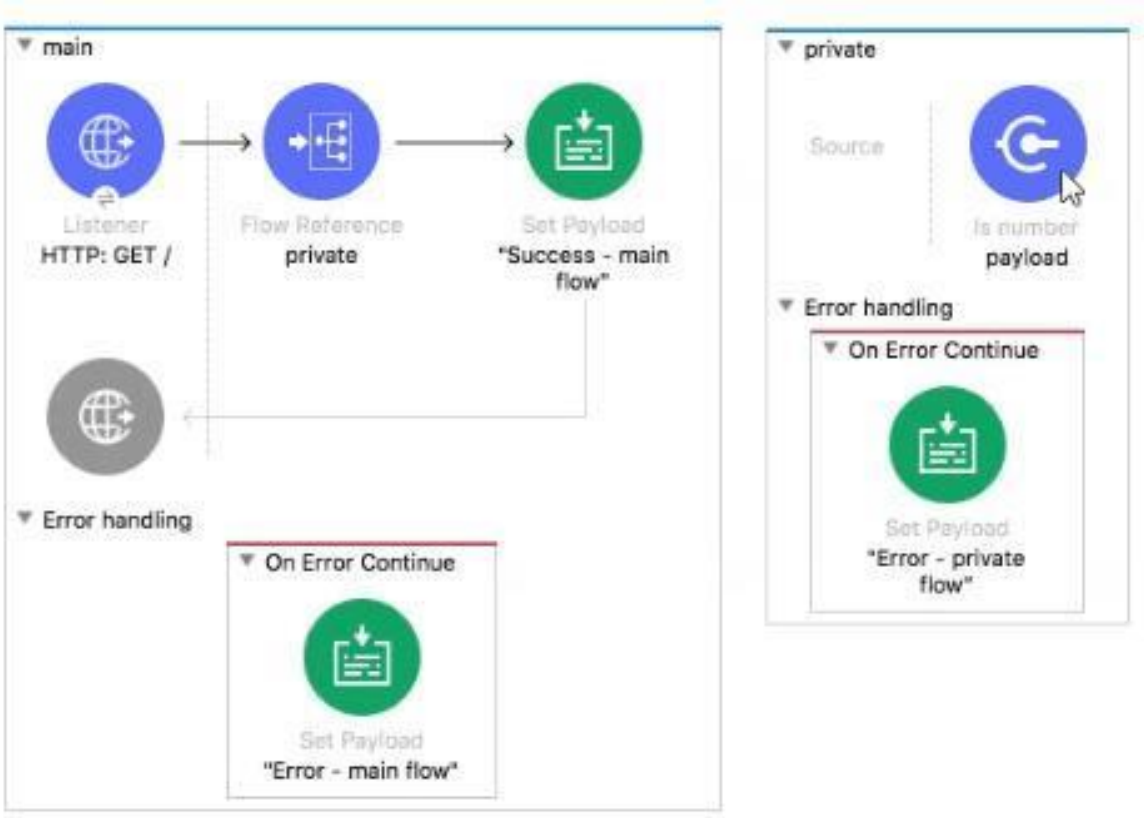
C. Entire event would be sent to each route sequentially

D. Entire event would be sent to each route in parallel

Answer: D
Entire event would be sent to each route in parallel.
Scatter-Gather works as follows :
- TheScatter-Gather component receives a Mule event and sends a reference of this Mule event to each processing route.
- Each of the processing routes starts executing in parallel. After all processors inside a route finish processing, the route returns a Muleevent, which can be either the same Mule eventwithout modifications or a new Mule event created by the processors in the route as a result of the modifications applied.
- After all processing routes have finished execution, the Scatter-Gather component creates a new Mule event that combines all resulting Mule events from each route, and then passes the new Mule event to the next component in the flow.
Diagram Description automatically generated



NO.21 Refer to the exhibits.




```
<flow name="main" >
    <http:listener doc:name="HTTP: GET /" config-ref="HTTP_Listener_config" path="/" />
    <flow-ref doc:name="private" name="private"/>
    <set-payload value="Success - main flow" doc:name='"Success - main flow"' />
    <error-handler>
        <on-error-continue enableNotifications="true" logException="true" doc:name="On Error Continue" >
            <set-payload value="Error - main flow" doc:name='"Error - main flow"' />
        </on-error-continue>
    </error-handler>
</flow>

<flow name="private" >
    <validation:is-number numberType="INTEGER" doc:name="payload" value="#[payload]"
    message="Validation Error" />
    <error-handler >
        <on-error-continue enableNotifications="true" logException="true" doc:name="On Error Continue" >
            <set-payload value="Error - private flow" doc:name='"Error - private flow"' />
        </on-error-continue>
    </error-handler>
</flow>
```

The Validation component in the private flow throws an error. What response message is returnedto a client request to the main flow's HTTP Listener?

A. Error - private flow

B. Error - main flow

C. Success - main flow

D. Validation Error

Answer: B

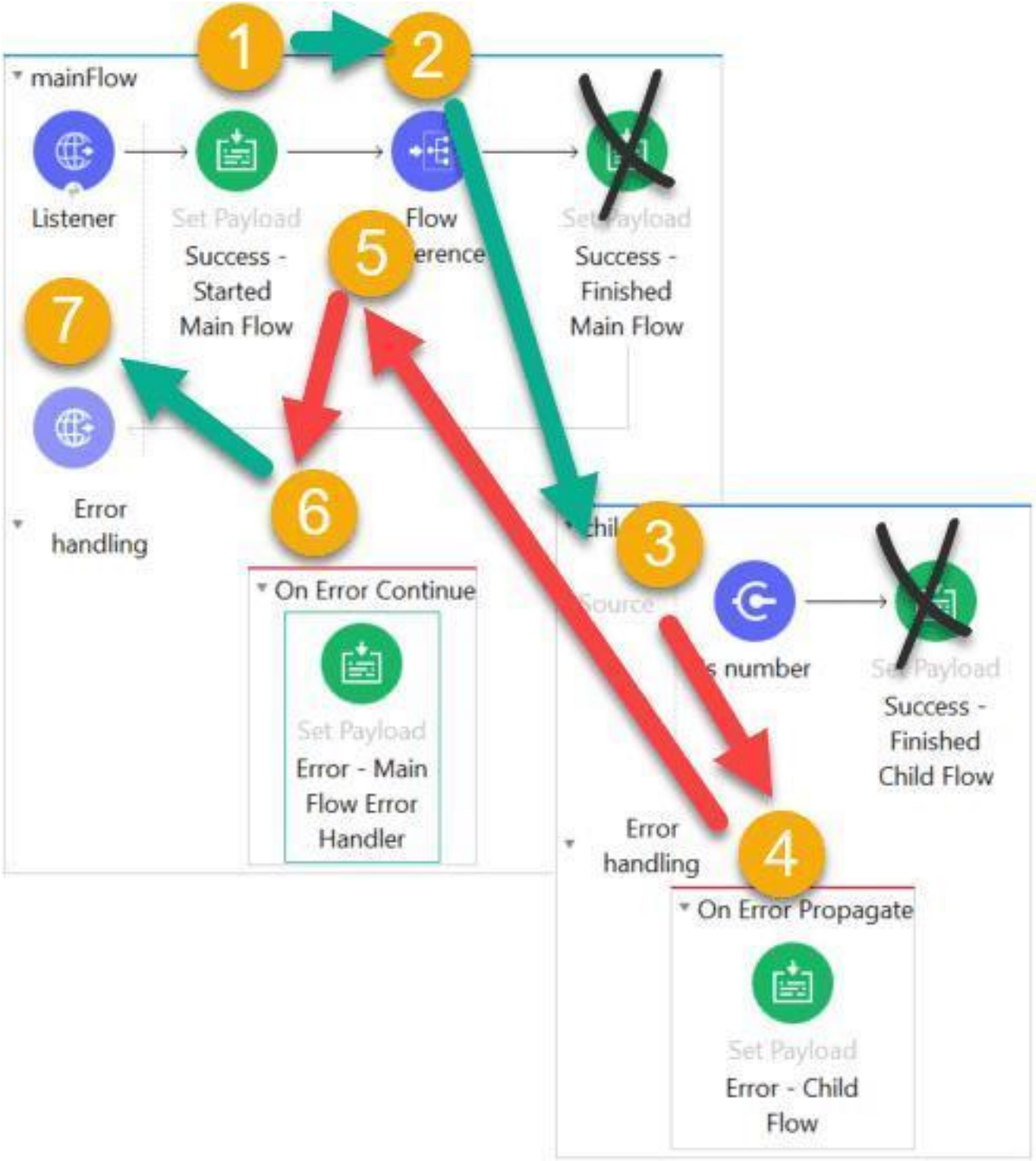
Error in validation component will get processed by Processer level On Error Propagate block and thenerror will be rethrown which will get processed by flow level error handler which will set payload to "Error- main flow". Hence correct answer is Error - main flow

1) Request is received by HTTP listener

2) Try scope gets executed
3) The validator component in the Try scope creates an Error Object because the payload is not null.
4) The On Error Propagate handles the error. The payload is set to "Error - Try scope"
6) "Error - Try scope" is returned to the 'On Error Continue' block. Main flow execution stops. Payload is set to "Error - main flow"
7) "Error - main flow" is returned to the requestor in the body of the HTTP request. HTTP Status Code: 200

- -
- -
- - - - - - - - - - - - - Reference Diagram though not exactly same, conditions are similar. They will help you answer any new question on error handling in real exam:

Timeline Description automatically generated



https://docs.mulesoft.com/mule-runtime/4.3/on-error-scope-concept#on-error-continue

NO.22 What HTTP method in a RESTful web service is typically used to completely replace an existing resource?

A. GET

B. PATCH

C. PUT

D. POST

Answer: C

PUT replaces the original version of the resource, whereas the PATCH method supplies a set of instructions to modify theresource

NO.23 How are query parameters dynamically passed to an outbound REST request using an HTTP Request operation?

A. As query parameters in the HTTP Request operation

B. As URI parameters in the HTTP Request operation

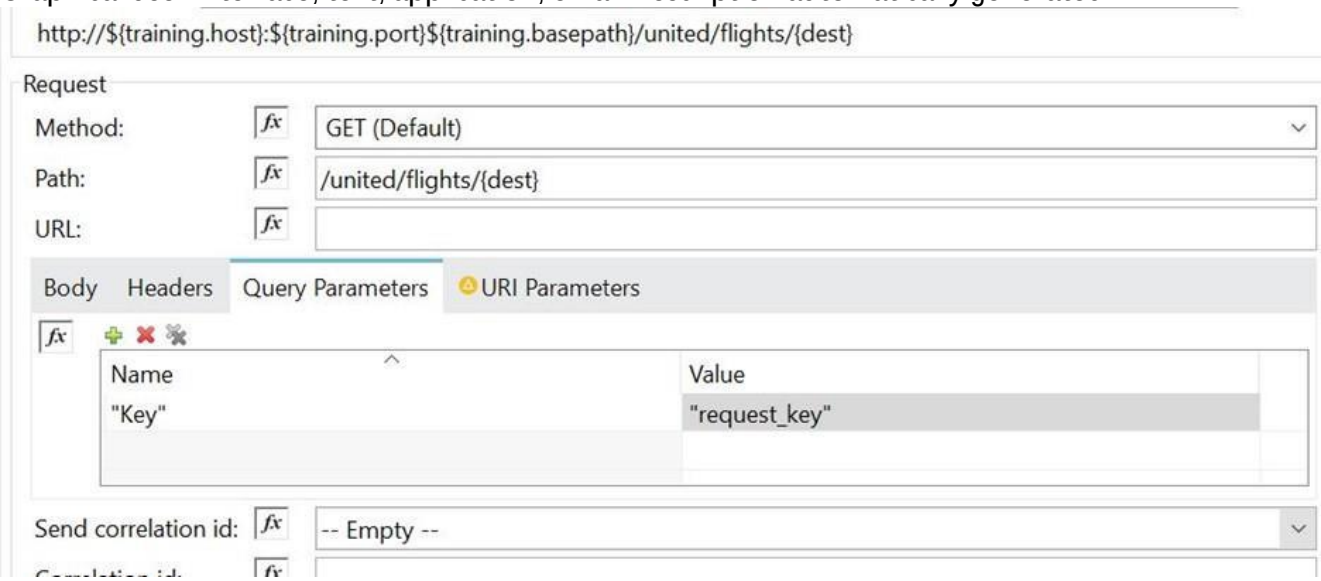
C. In the Mule event's payload

D. As flow variables

Answer: A

In General > Request > Query Parameters, click theplus icon (+) to add a parameter to a request. Type a name and value for the parameter or use a DataWeave expression to define the name and value.

Graphical user interface, text, application, email Description automatically generated



NO.24 Refer to the exhibits. A company has defined this Book data type and Book example to be used in APIs. What is valid RAML for an API that uses this Book data type and Book example?

```
#%RAML 1.0 DataType
# bookDataType.raml

type: object
properties:
  ID?: integer
  title: string
  author: string
  publisher?: string
  year: integer
  ISBN:
    type: string
    required: true
```

```
#%RAML 1.0 NamedExample
# bookExample.raml

bookExample:
  ID: 101
  title: Shakespeare
  author: Encyclopaedia Britannica
  publisher: John Wiley & Sons
  year: 2007
  ISBN: "0471767840"
```

A.

```
#%RAML 1.0
title: Books

Book: BookDataType.raml

/books:
    post:
      body:
        application/json:
          type: Book
          examples:
            input: BookExample.raml
       responses:
         201:
           body:
             application/json:
               example:
                 message: Book added
```

B.

```
#%RAML 1.0
title: Books

Book: !include BookDataType.raml

/books:
    post:
      body:
        application/json:
          type: Book
          examples:
            input: !include BookExample.raml
      responses:
        201:
          body:
            application/json:
              example:
                message: Book added
```

C.

```
#%RAML 1.0
title: Books

types:
  Book: ABC/DataTypes/BookDataType.raml

/books:
    post:
      body:
        application/json:
          type: Book
          examples:
            input: ABC/Examples/BookExample.raml
      responses:
        201:
          body:
            application/json:
              example:
                message: Book added
```

D.

```
#%RAML 1.0
title: Books

types:
  Book: !include BookDataType.raml

/books:
    post:
      body:
        application/json:
          type: Book
          examples:
            input: !include BookExample.raml
      responses:
        201:
          body:
            application/json:
              example:
                message: Book added
```

Answer: D