IBM Business Automation Manager Open Editions
Demos and Labs 2025

Exploring BAMOE Developer Tools for VS Code

V 1.0 (for IBM BAMOE 9.2)

Raul Mariano
raul.mariano@ibm.com

NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Table of Contents

1 Introduction

These hands-on lab exercises are designed to guide you through the essential aspects of process automation using BAMOE Developer Tools for VS Code. Whether you're a developer or an architect, these labs will equip you with the skills you need to use VS Code to effectively build workflow and decision automation projects on modern, cloud-native business automation solutions.

*Includes **six exercises**. We recommend performing them sequentially.*

***Duration:** Approximately 3 hours.*

***Audience:** Anyone who wants to learn how to use IBM Business Automation Manager Open Editions.*

1.1 IBM Business Automation Manager Open Editions

IBM Business Automation Manager Open Editions (IBM BAMOE) is a powerful open-source solution that serves as a foundation platform for tailoring long-lasting business automation solutions for the hybrid cloud.

With a developer-centric approach, this comprehensive and flexible platform makes it easy for teams to collaborate through Open Standards and efficient development tools suited for different personas.
Each automation solution can be shaped to perfectly address each scenario: business applications are flexible and can effortlessly integrate with external systems of your existing architecture.

Designed for the hybrid cloud, IBM Business Automation Manager Open Editions, accelerates the application modernization and cloud adoption journeys, as the lightweight design tools, business applications and other product components can be containerized and deployed with popular technologies such as Kubernetes and OpenShift.

For more information, see IBM documentation and other useful links:

- IBM Business Automation Manager Open Editions Documentation
- Open Editions Community

2 Lab Setup Instructions

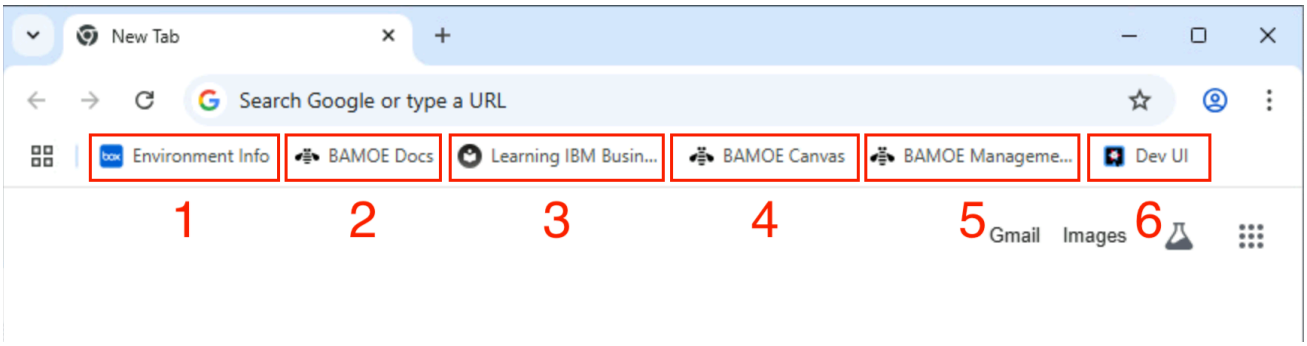
2.1 Access the environment

You received this email with instructions on how to access the environment using your IBMid.

If necessary, this is the Windows credential:

```
User: techzone
Password: IBMDem0s!
```

2.1.1 Useful links:



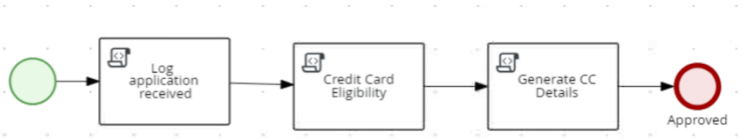
Item	Description
1	**“Environment Info”:** Access this document available in Box.
2	**“BAMOE Docs”:** Access the official product documentation.
3	**“Learning IBM Business Automation Open Edition”:** A great guide for users who are trying IBAMOE for the first time. Recommended getting started guide.
4	**“BAMOE Canvas”:** You can access BAMOE Canvas through the URL: *http://localhost:9090*
5	**“BAMOE Management Console”:** You can access BAMOE Management Console through the URL: *http://localhost:7070*
6	**“Dev UI”:** For projects run by VS Code, you can access the Dev UI via the URL: *http://localhost:8080/q/dev-ui*

3 The use case

We will work with a fictitious credit card request process, our goal is to demonstrate how to integrate Workflows with Decision services, in addition to exploring multiple scenarios for each data entry.

You will use a pre-configured BAMOE project, and throughout the exercises we will modify the Workflow and execute it in practice, setting up the following scenarios:

- **Automatic Approval:** If the credit card applicant's data meets the minimum requirements for approval, the card will be automatically generated.
- **Automatic Rejection:** If the applicant's data does not meet the minimum requirements, it will be automatically rejected.
- **Manual Approval:** There is a scenario where the application analysis must be done manually by a user, who may also be Approved or Rejected.



Now, in this lab, we will work with BAMOE Developer Tools for VS Code, which brings together graphical editors for BPMN, DMN and SCESIM files. Access more in the BAMOE Developer Tools for VS Code documentation.

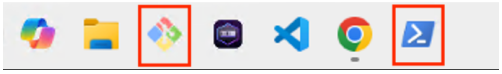
Note: you know that we will use VS Code for this Lab, if you want to know more about it, access BAMOE Developer Tools for VS Code documentation.

And if you're an **experienced Java developer**, you may have your own preferences when creating new projects. Click here to learn more.

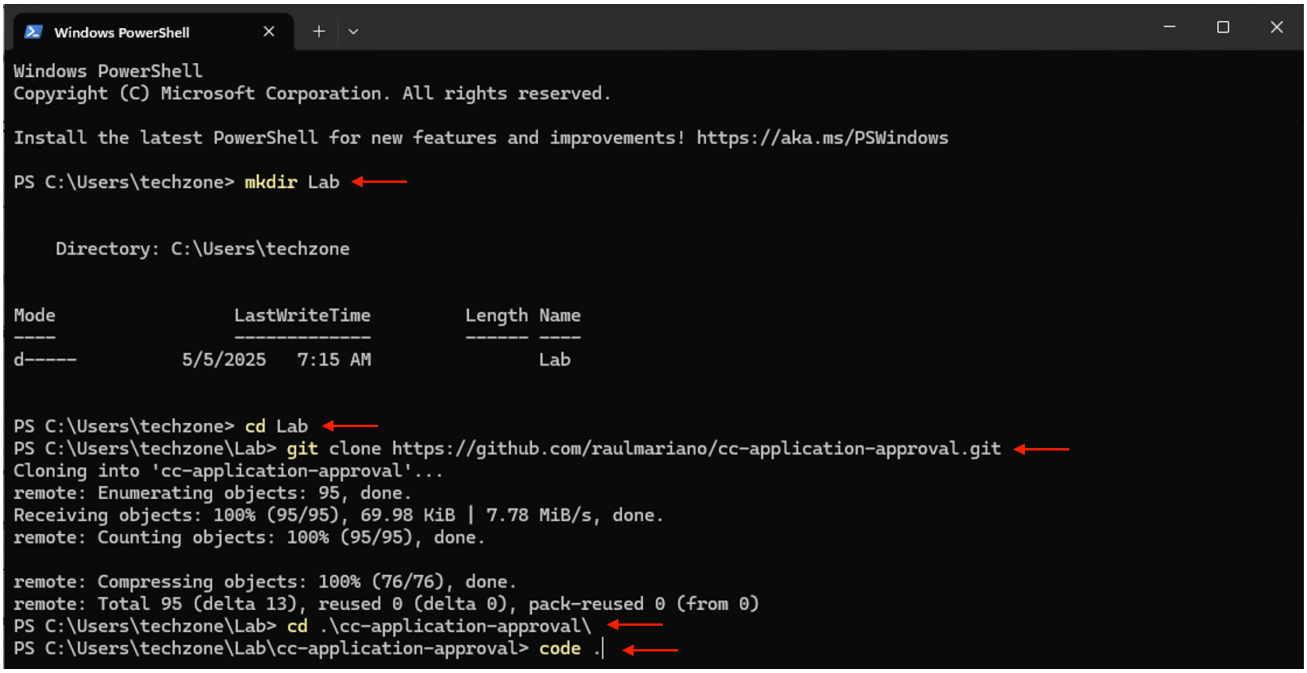
Next, the first exercise begins. Happy studying!

4 Exercise 1: Exploring the Dev-UI console in more detail

As mentioned earlier, we will now work with a pre-built project. From your taskbar, open Git or Terminal.



Let's create a folder to download the pre-built project from GitHub:



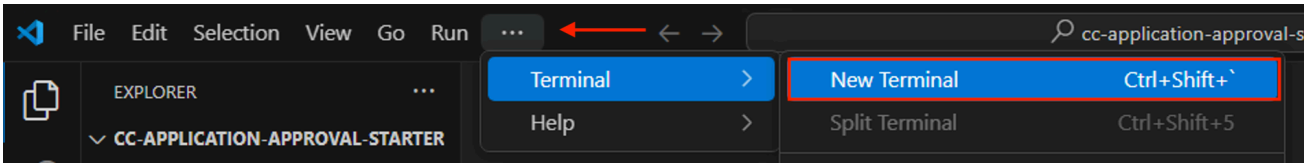
#	Description
1	Create a new folder: `mkdir Lab`
2	Enter the new ***"Lab"*** folder `cd Lab`
3	Enter the command to clone the project: `git clone https://github.com/raulmariano/credit-card-application-approval.git`
4	Access the project folder: `cd credit-card-application-approval`
5	Open VS Code using the command: `code .`

In short, the Kogito project structure focuses on separating the definition of business logic (rules, processes, decisions) from the technical implementation of the Java application, allowing for greater clarity, agility, and ease of maintenance.

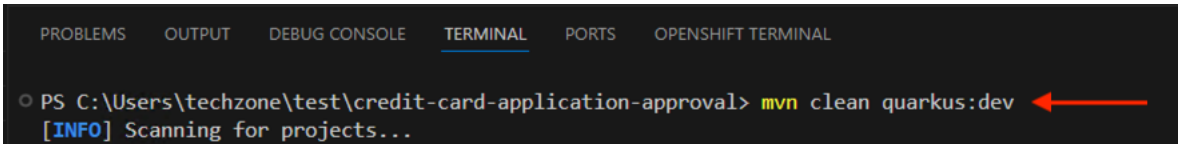
Explore the project structure and see what features have already been created in this use case.

```
<project-name>/
├── src/
│   ├── main/
│   │   ├── java/          # Java application code (models, services, etc.)
│   │   └── resources/     # Business logic definitions (.bpmn, .dmn, .drl) and others
│   └── test/
│       ├── java/         # Unit and integration tests
│       └── resources/    # Testing Resources
├── pom.xml             # Maven configuration file
```

Let's compile the project to explore DEV-UI. Is a centralized development interface to understand, debug and interact with the intelligence of your application, in addition to standard information (configuration, health, logs), it offers visualization of BPMN processes, monitoring of instances and tasks.

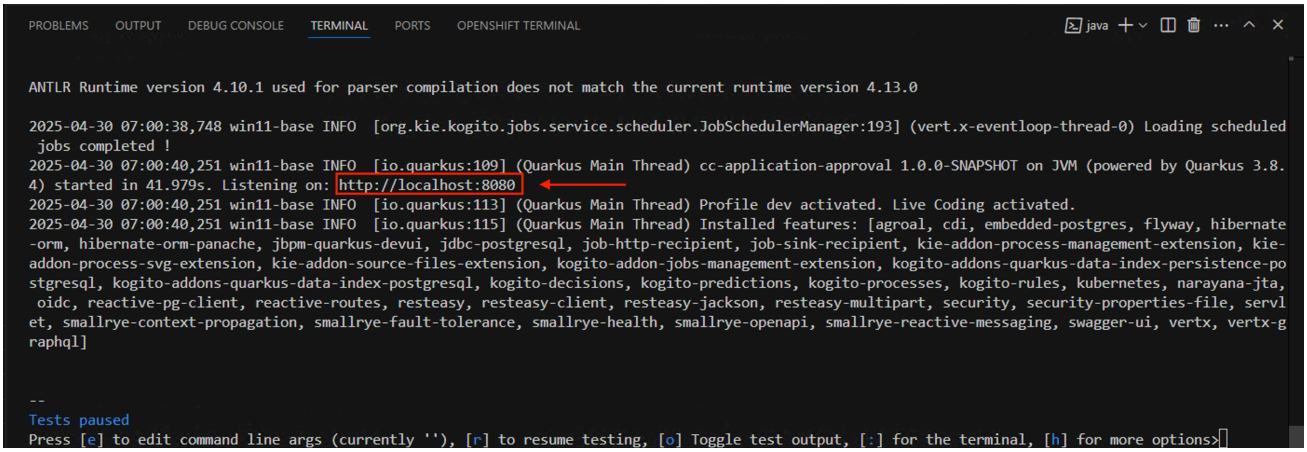


Item	Description
1	From the toolbar, go to ***Terminal > New Terminal***

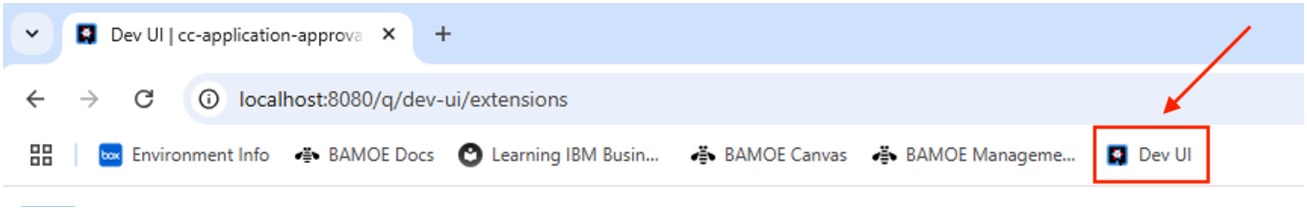


Item	Description
1	Run the following command: `mvn clean quarkus:dev`

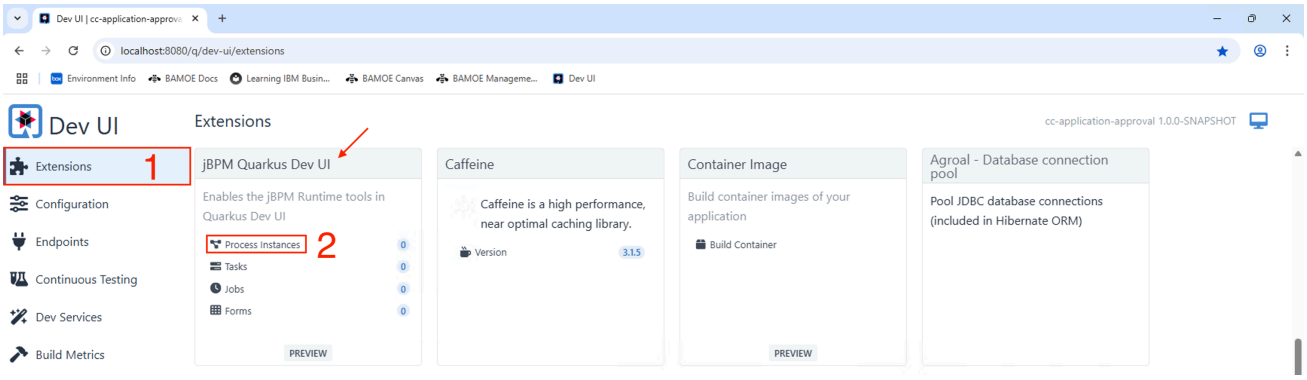
Wait for the process to start. You should see the message Listening on: *http://localhost:8080*. This may take a few minutes.



Open Google Chrome, and go to http://localhost:8080/q/dev-ui.

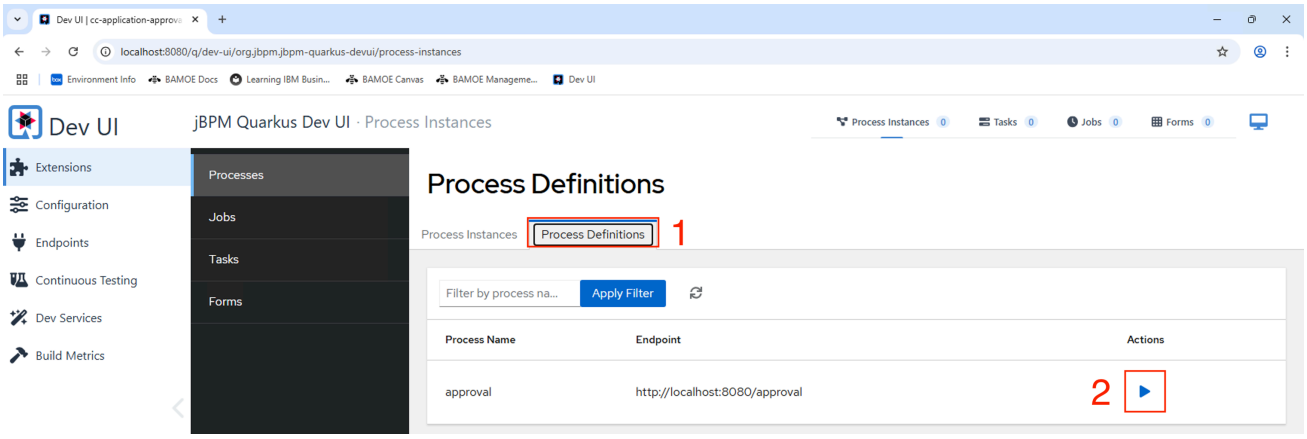


Let's understand more about the use case by accessing the process instances.



Item	Description
1	On the main ***"Extensions"*** page, locate the ***"jBPM Quarkus Dev UI"*** extension
2	Click on ***"Process Instances"***

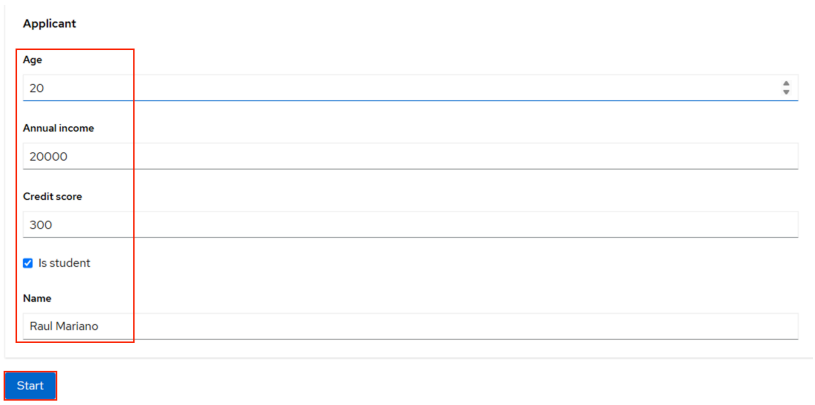
Note that the layout is like the BAMOE Management Console.
We continue by creating a new instance to analyse what we have built.

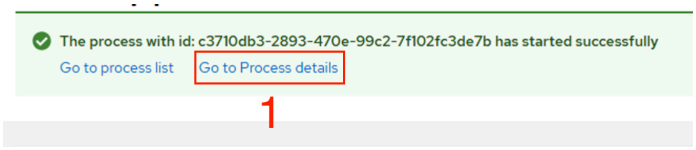


Item	Description
1	Go to the ***"Process Definitions"*** tab
2	Click to perform the action

We will perform several tests on this form, to facilitate documentation, we will replace the screen print with a table with the test data. See:

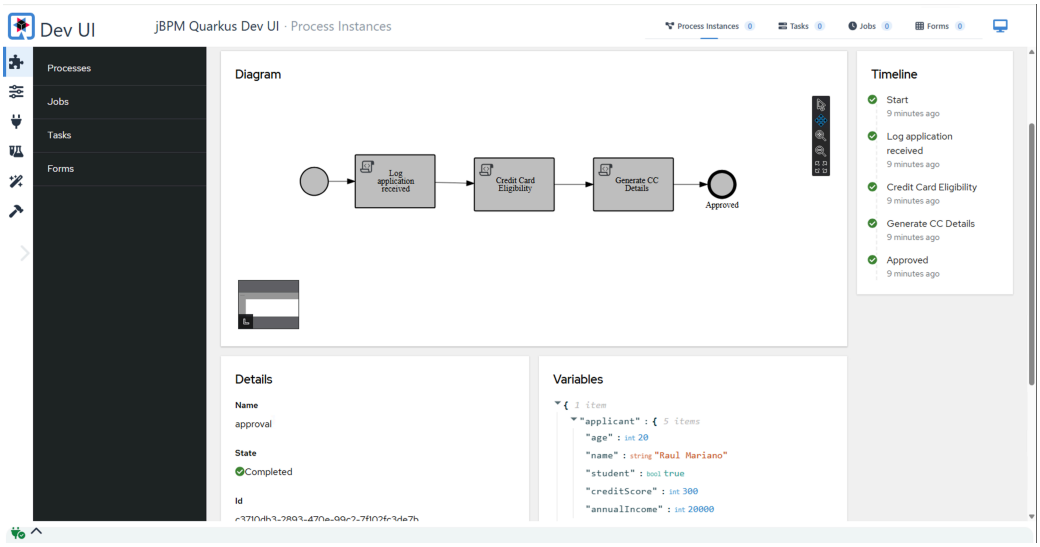
AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	NAME
20	*20000*	*300*	*YES*	*[YOUR-NAME]*





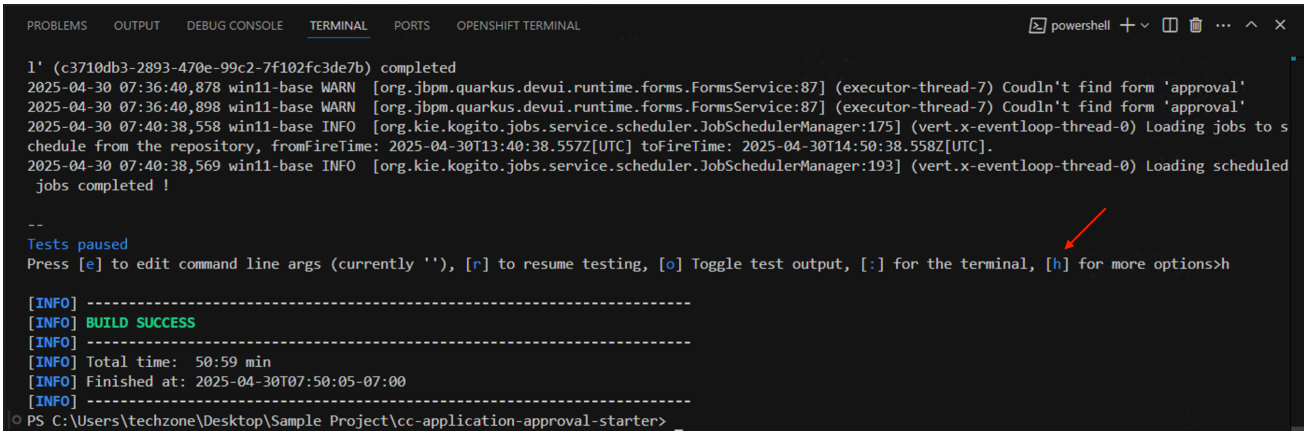
Item	Description
1	Click to access process details

Here you can see what was done in the process, a very straightforward and simple process that performed several "simple" script tasks. In the next exercise, we will start improving this!



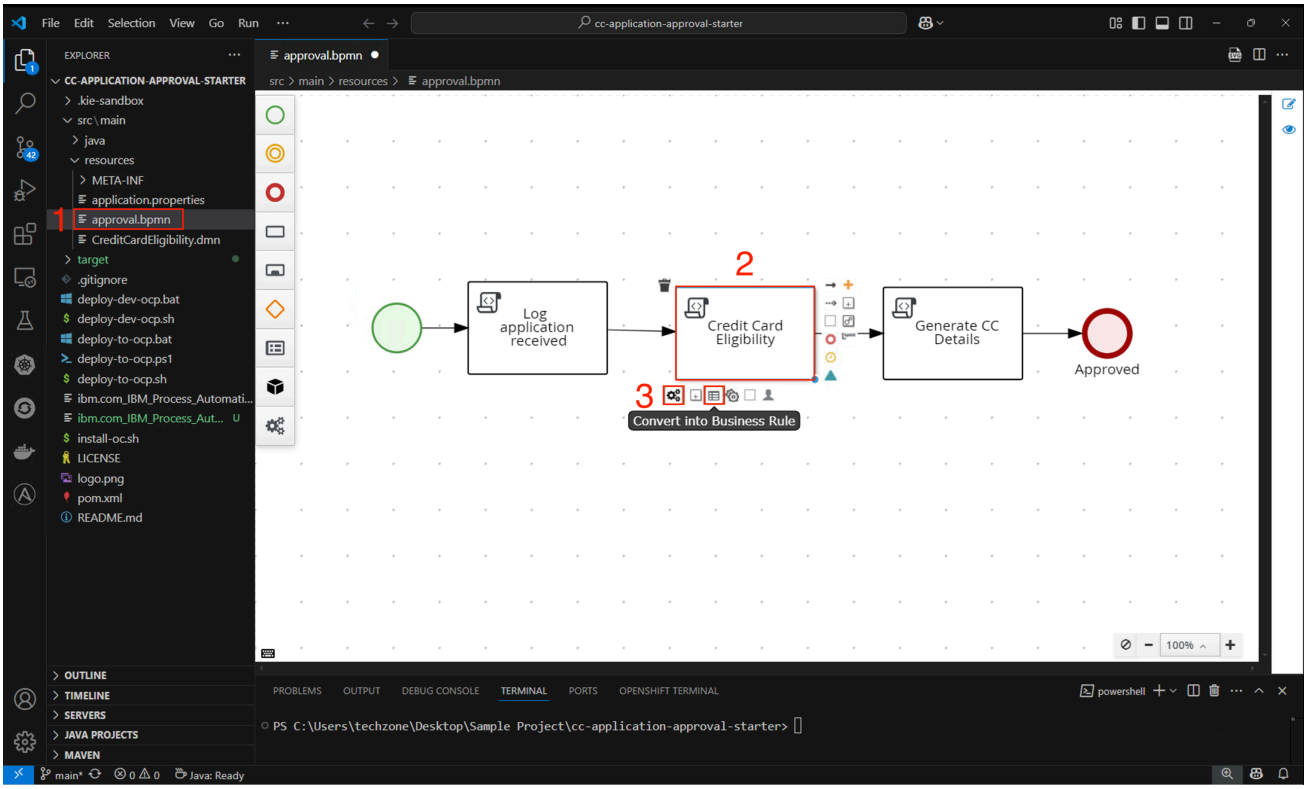
Close the browser. Let's work some more in VS Code and then review again when we're done.

Back in the VS Code terminal, stop the execution by pressing the options: [h] then [q].

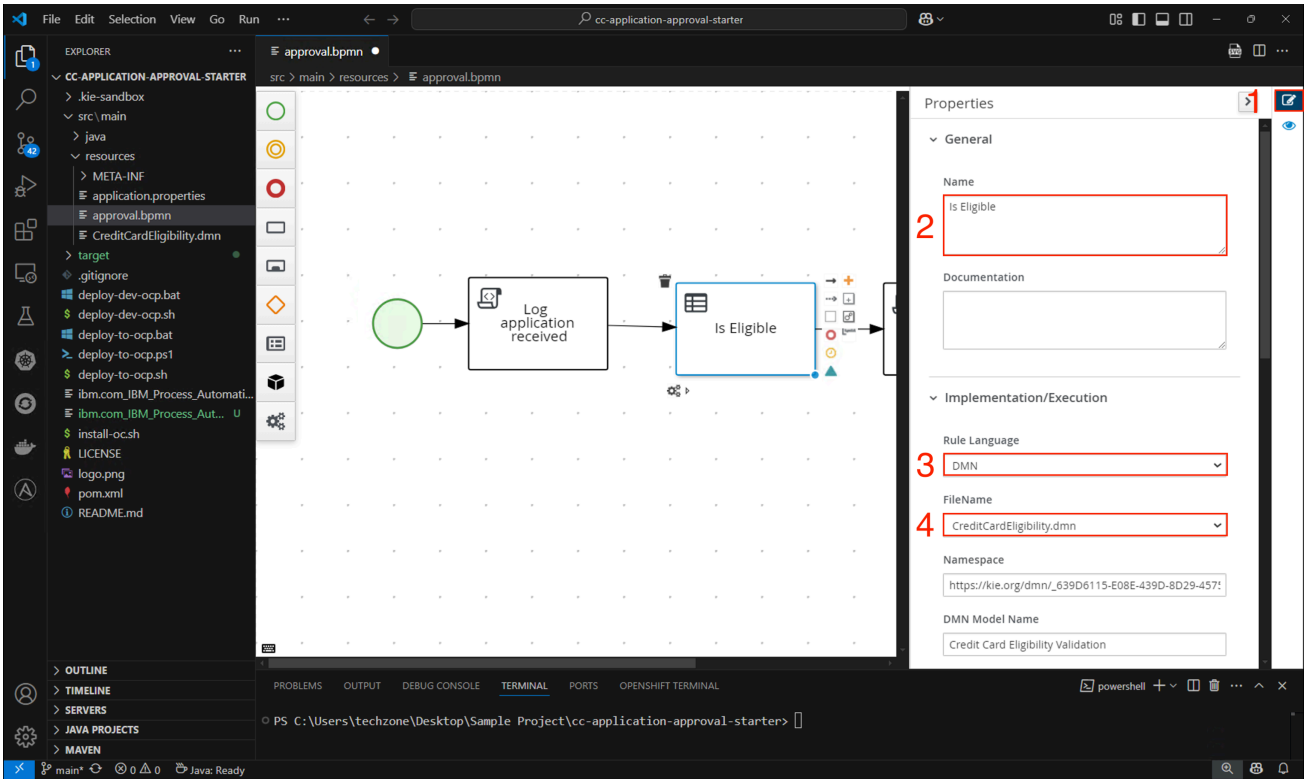


5 Exercise 2: Using DMN in the Process

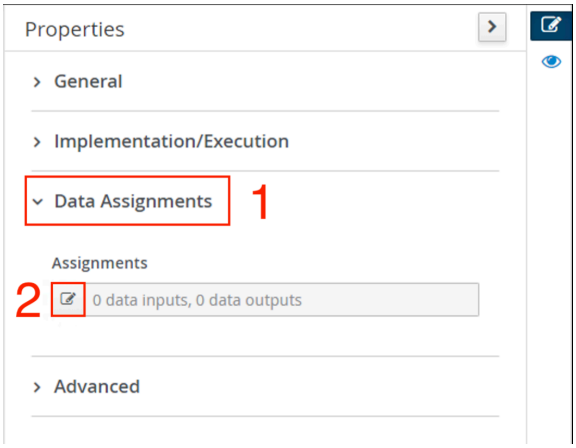
In this exercise, we will modify the ***"Credit Card Eligibility"*** task of the “Approval” process, to incorporate a decision we make (DMN). That is, this task will send the input data to the DMN to execute the business rule, and the DMN result will be used in the next steps of the process.



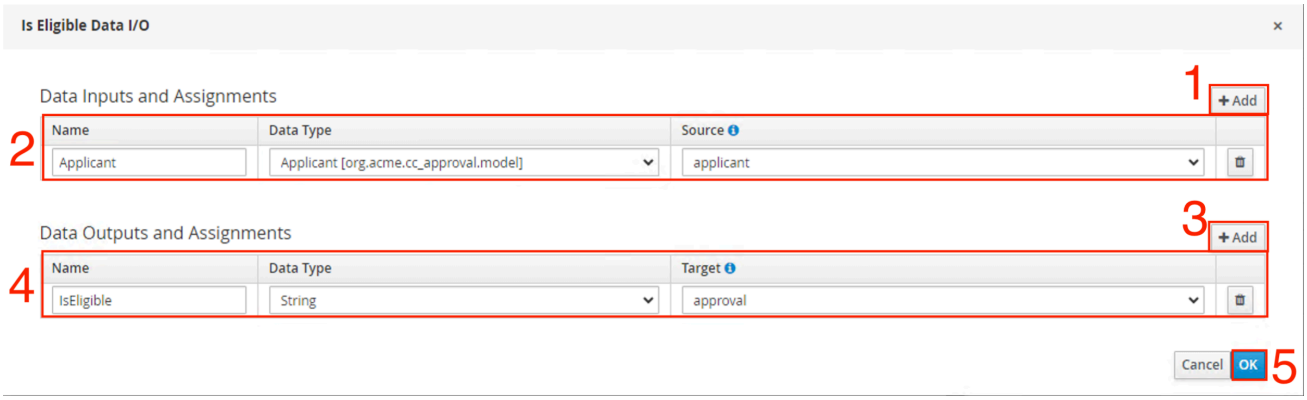
Item	Description
1	Open the ***approval.bpmn*** process, which is located at ***src/main/resources/approval.bpmn***. Click to open it.
2	Select the ***"Credit Card Eligibility"*** task
3	Click on the **gear** then select the second option to **convert into business rule**.



Item	Description
1	Access the element **properties** menu (similar to BAMOE Canvas)
2	Change the task name to ***"Is Eligible"***
3	In the ***"Implementation/Execution"*** section, select ***"DMN"*** in the ***"Rule Language"*** field
4	And in the ***"File Name"*** field select **"CreditCardEligibility.dmn"**
5	The remaining fields will be filled in automatically.



Item	Description
1	Scroll down in the properties window and expand ***"Data Assignments"***
2	Click the **pencil icon** to create them.



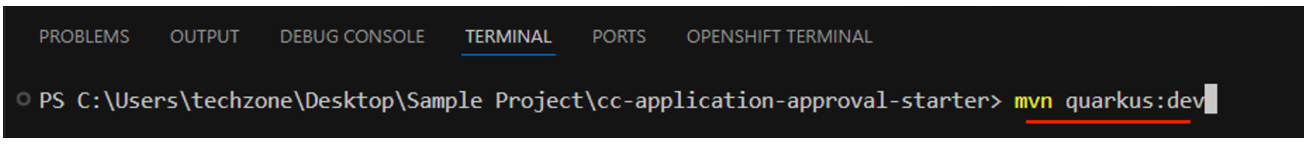
Item	Description
1	Click ***"+ Add"***
2	Fill in the ***Data Input*** fields: **FIELD** / **DATA** Name: *Applicant* Data Type: *Applicant[org.acme.cc_approval.model]* Source: *applicant*
3	Click ***"+ Add"***
4	Fill in the ***Data Output*** fields: **FIELD** / **DATA** Name: *IsEligible* Data Type: *String* Source: *approval*
5	Click "***OK***"

You can and should also open the DMN in parallel, to explore the business rules built, and understand how card eligibility will be carried out.

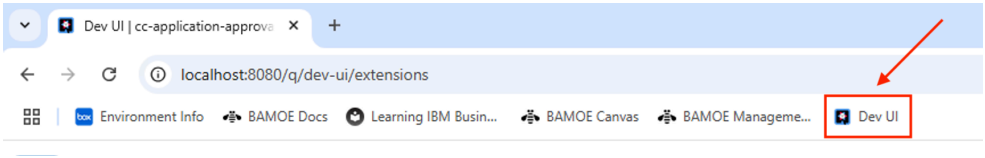
Before running the project, make sure the project has been saved ***(CTRL + S)***, and click the SVG button  (in the top right corner), to update the process image, as it will be used in the process instance details in DEV-UI.

Back in the terminal, run the command to run the project.

```
mvn quarkus:dev
```



Now in the browser, open DEV-UI: http://localhost:8080/q/dev-ui



Note: And since we have already navigated this area before, I believe you are already familiar with navigation, if you have any questions, go back and check the steps.

Navigate to the ***"Process Definitions"*** tile to run the following test scenarios. Use the form and repeat the test with different data that will result in different decision outputs.

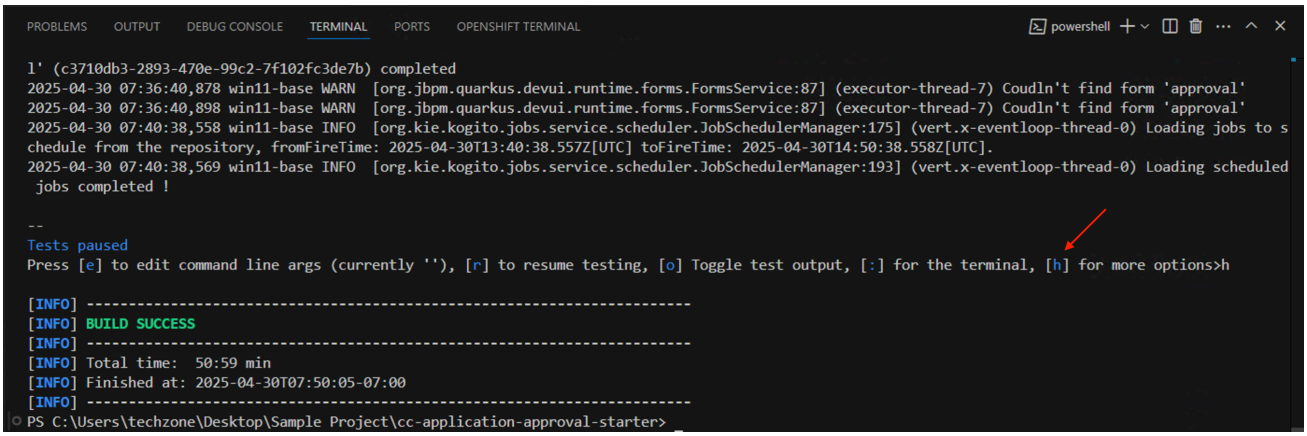
AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	SCENARIO
25	*15000*	*750*	*NO*	*Automatic approval*
17	*15000*	*750*	*NO*	*Automatic rejection*
20	*19000*	*600*	*NO*	*Manual Review*

Validate by variables if the result of each scenario was like this:

Automatic approval	Automatic rejection	Manual Review
Variables { 2 items "applicant" : { 5 items "name" : string "Raul" "student" : bool false "annualIncome" : int 15000 "creditScore" : int 750 "age" : int 25 } "approval" : string "approved" }	Variables { 2 items "applicant" : { 5 items "name" : string "Angelito" "student" : bool false "annualIncome" : int 15000 "creditScore" : int 750 "age" : int 17 } "approval" : string "rejected" }	Variables { 2 items "applicant" : { 5 items "name" : string "Manoel" "student" : bool false "annualIncome" : int 19000 "creditScore" : int 600 "age" : int 20 } "approval" : string "manual" }

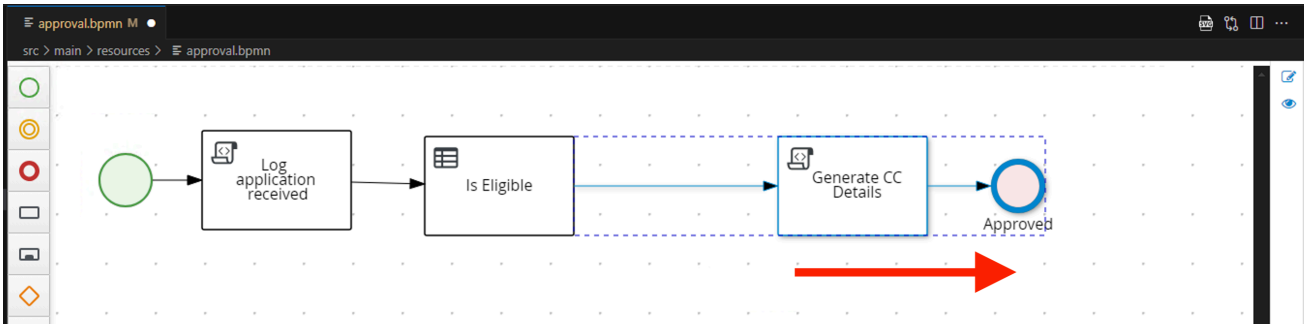
If you got the same result, congratulations! Let's move on to the next exercise.

Close the browser. Let's work some more in VS Code and then review again when we're done.
Back in the VS Code terminal, stop the execution by pressing the options: [h] then [q].

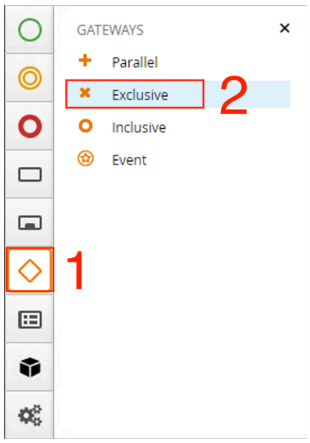


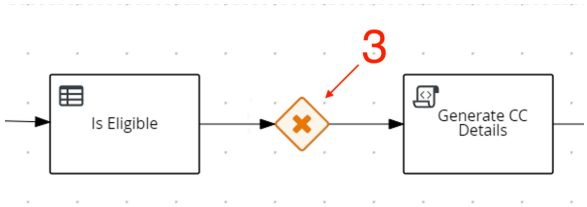
6 Exercise 3: Adding possible process scenarios

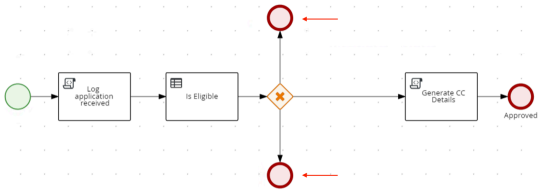
Now let's add the gateway that will divide the process into three possible scenarios: **Automatic Approval, Automatic Rejection,** and **Manual Approval.** Go back to **VS Code** to the "***appoval.bpmn***" file.



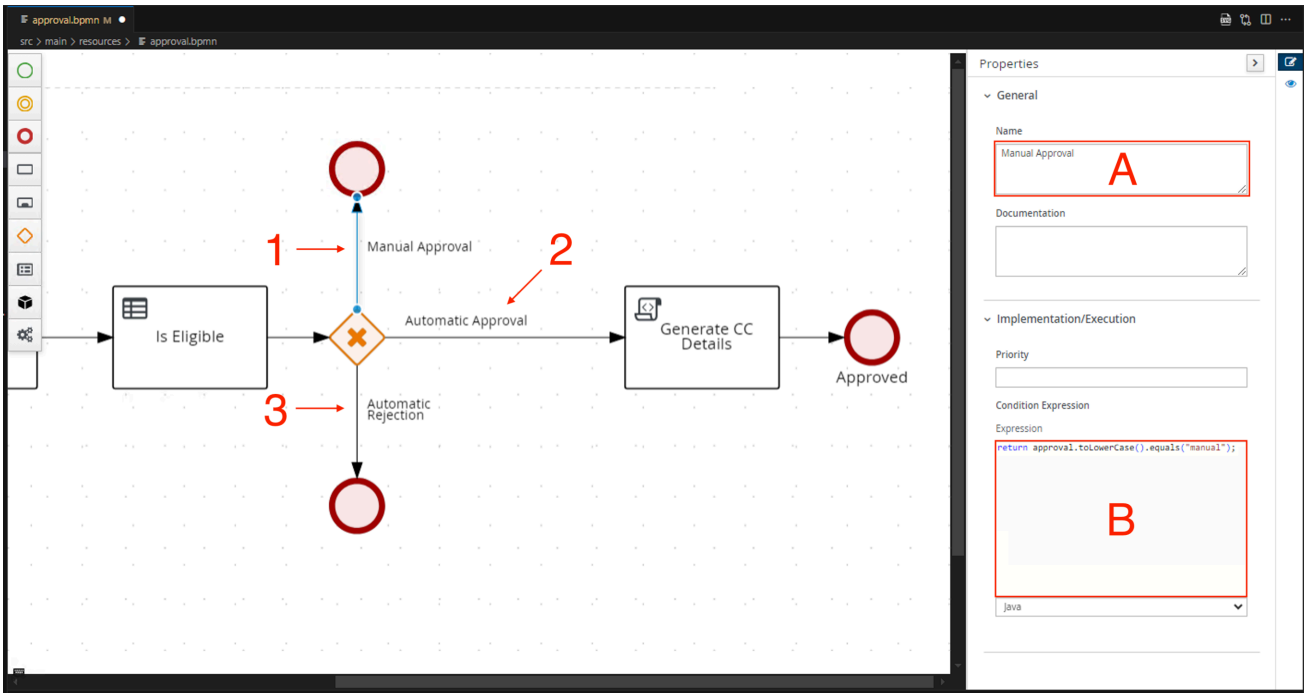
Item	Description
1	Select the ***"Generate CC Details"*** task and the ***"Approved"*** event, and drag to the right.





Item	Description
1 and 2	In the side panel, select a Gateway of type ***"Exclusive"***
3	Drag the Exclusive Gateway to the line separating ***"Is Eligible"*** and ***"Generate CC Details".*** *If positioned correctly, the gateway will turn the line blue and new arrows will form. If not, simply reconnect the arrows between the two existing nodes.*
4	From this Gateway, add two new paths to ***"Final Event"***, see: 

Now, let's configure the path by specifying the condition to be followed for each scenario.



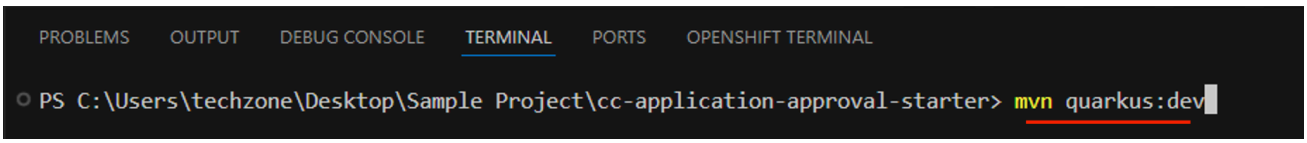
*Fill in fields **A** and **B** with their respective values:*

Item	Description
1	Click to select the path, and fill in the fields on the Properties screen: **FIELD** / **DATA** Name: *Manual Approval* Expression: *return approval.toLowerCase().equals("manual");*
2	Click to select the path, and fill in the fields on the Properties screen: **FIELD** / **DATA** Name: *Automatic Approval* Expression: *return approval.toLowerCase().equals("approved");*
3	Click to select the path, and fill in the fields on the Properties screen: **FIELD** / **DATA** Name: *Automatic Rejection* Expression: *return approval.toLowerCase().equals("rejected");*

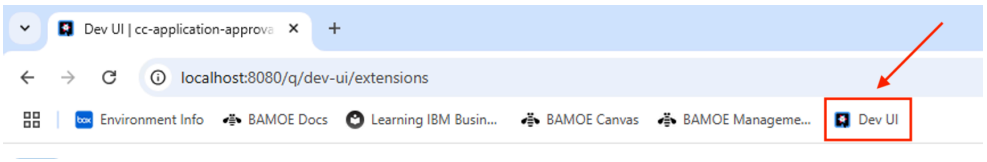
Before running the project, make sure the project has been saved ***(CTRL + S)***, and click the SVG button  (in the top right corner), to update the process image, as it will be used in the process instance details in DEV-UI.

Back in the terminal, run the command to run the project.

```
mvn quarkus:dev
```



Now in the browser, open DEV-UI: http://localhost:8080/q/dev-ui

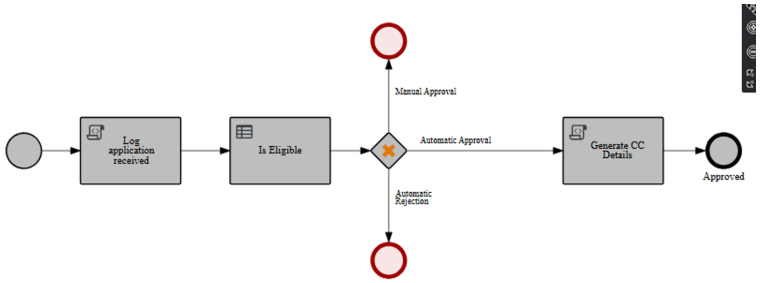
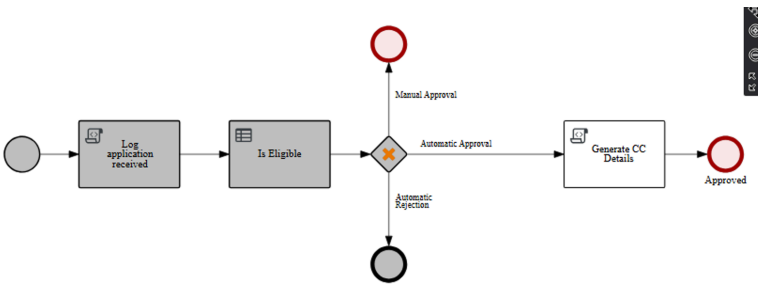


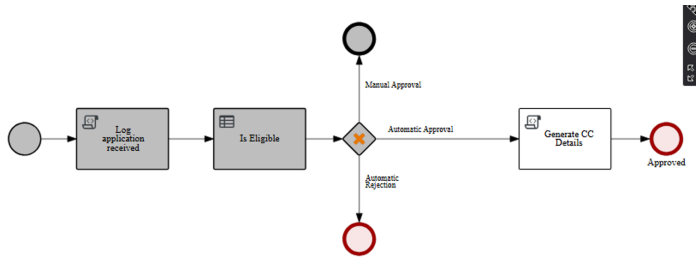
Note: And since we have already navigated this area before, I believe you are already familiar with navigation, if you have any questions, go back and check the steps.

Navigate to the ***"Process Definitions"*** tile to run the following test scenarios. Use the form and repeat the test with different data that will result in different decision outputs.

AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	SCENARIO
25	*15000*	*750*	*NO*	*Automatic approval*
17	*15000*	*750*	*NO*	*Automatic rejection*
20	*19000*	*600*	*NO*	*Manual Review*

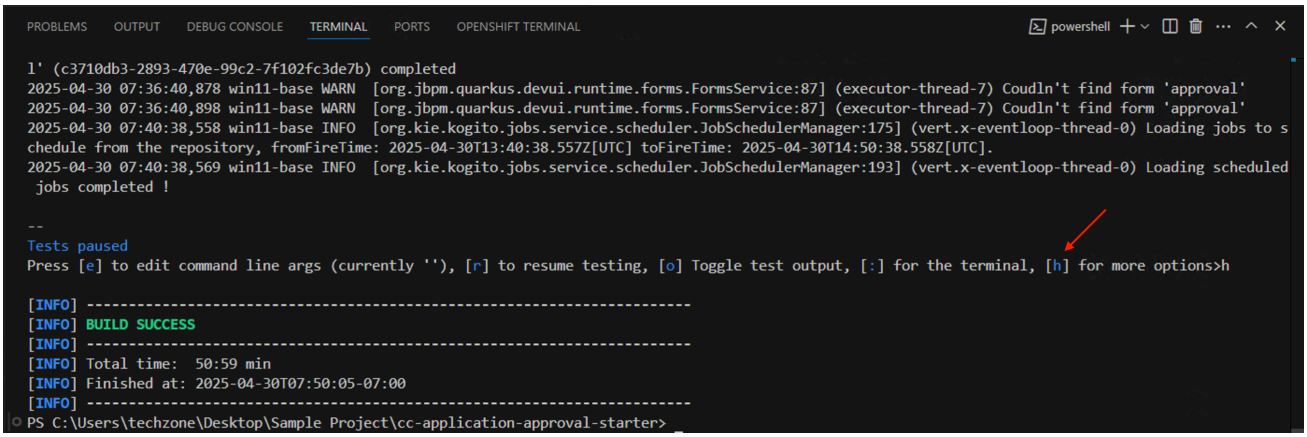
Check the diagram for each scenario, where the path taken after applying the process rules:

SCENARIO	DIAGRAM
Automatic approval	
Automatic rejection	

Manual Review	

If you got the same result, congratulations! Let's move on to the next exercise.

Close the browser. Let's work some more in VS Code and then review again when we're done. Back in the VS Code terminal, stop the execution by pressing the options: [h] then [q].



7 Exercise 4: Setting up process for Automatic Approval

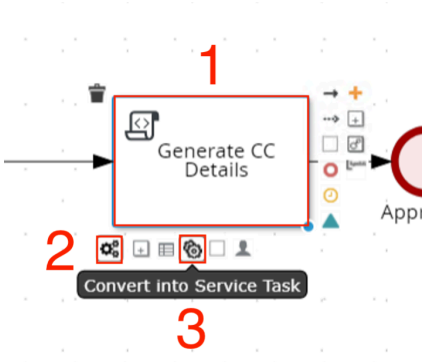
In this exercise we will configure the process for Automatic Approval cases.

When a credit card application is approved, we need to generate the card details. For this lab, we will start a Service Task that will invoke a Java method, but this could easily call a different service. This is the flexibility that the jBPM runtime can offer with Java services.

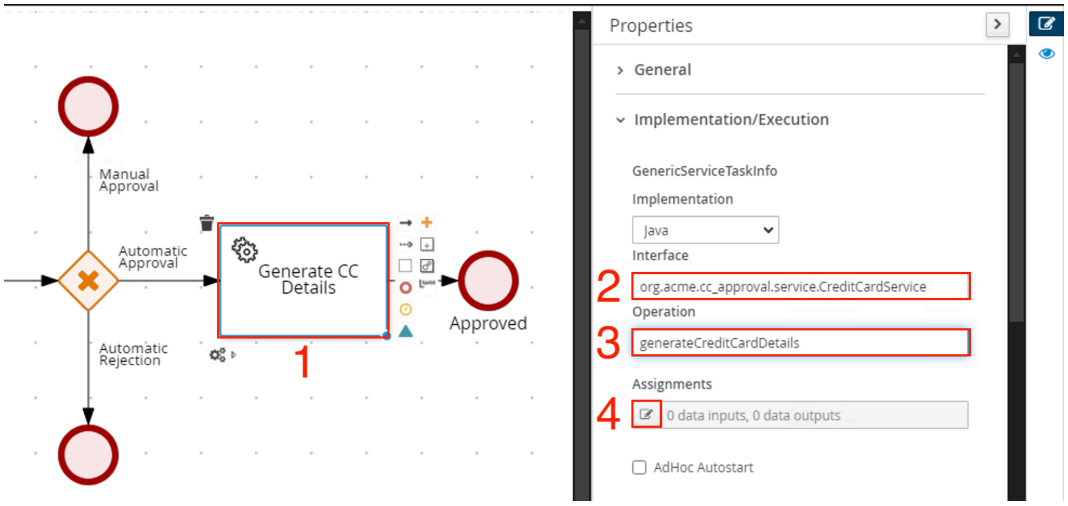
There is a Java class already created to process the credit card details for the approved account. It can be opened by going to ***src/main/java/org/acme/service/CreditCardService.java***. This class contains the logic to generate the credit card details:

This service calculates a credit limit based on 30% of the applicant's annual income and creates a new Credit Card object with the applicant's name and the calculated credit limit.

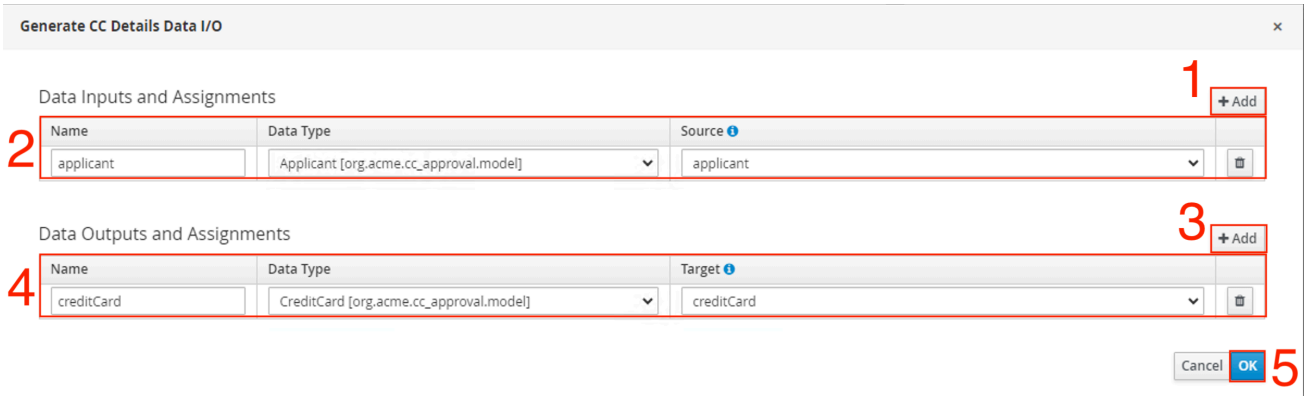
To incorporate custom Java code into your process using a Service Task, follow these steps:



Item	Description
1	Select the ***"Generate CC Details"*** task
2	Select the button with the "gears"
3	Select the option to ***"Convert into Service Task"***



Item	Description
1	Continue with task ***"Generate CC Details"*** selected
2	In properties > ***"Implementation/Execution",*** fill in the ***"Interface"*** field with: `org.acme.cc_approval.service.CreditCardService`
3	Fill in the ***"Operation"*** field with: `generateCreditCardDetails`
4	Click the button to configure the inputs and outputs, in the next step



Item	Description
1	Click ***"+ Add"***
2	Fill in the ***Data Input*** fields: **FIELD** / **DATA** Name: *applicant* Data Type: *Applicant[org.acme.cc_approval.model]* Source: *applicant*
3	Click ***"+ Add"***

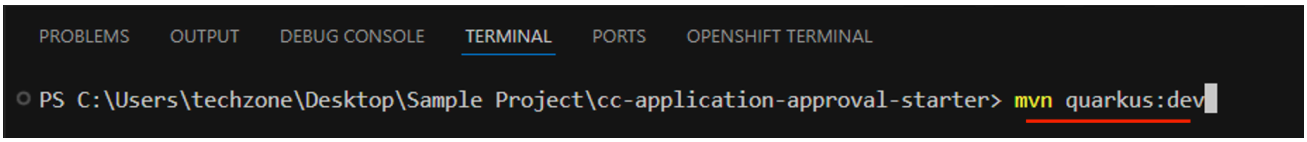
<table>
<tr><td>4</td><td>Fill in the Data Output fields:

<table><tr><th>FIELD</th><th>DATA</th></tr><tr><td>Name</td><td>creditCard</td></tr><tr><td>Data Type</td><td>CreditCard[org.acme.cc_approval.model]</td></tr><tr><td>Source</td><td>creditCard</td></tr></table></td></tr>
<tr><td>5</td><td>Click “OK”</td></tr>
</table>

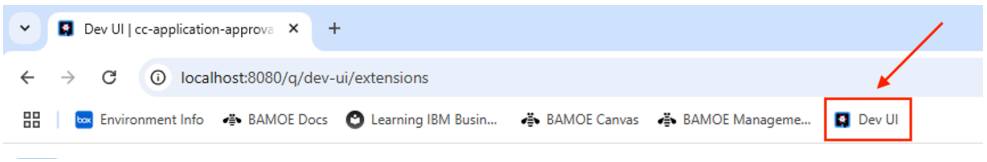
Before running the project, make sure the project has been saved ***(CTRL + S)***, and click the SVG button  (in the top right corner), to update the process image, as it will be used in the process instance details in DEV-UI.

Back in the terminal, run the command to run the project.

```
mvn quarkus:dev
```



Now in the browser, open DEV-UI: *http://localhost:8080/q/dev-ui*



Note: And since we have already navigated this area before, I believe you are already familiar with navigation, if you have any questions, go back and check the steps.

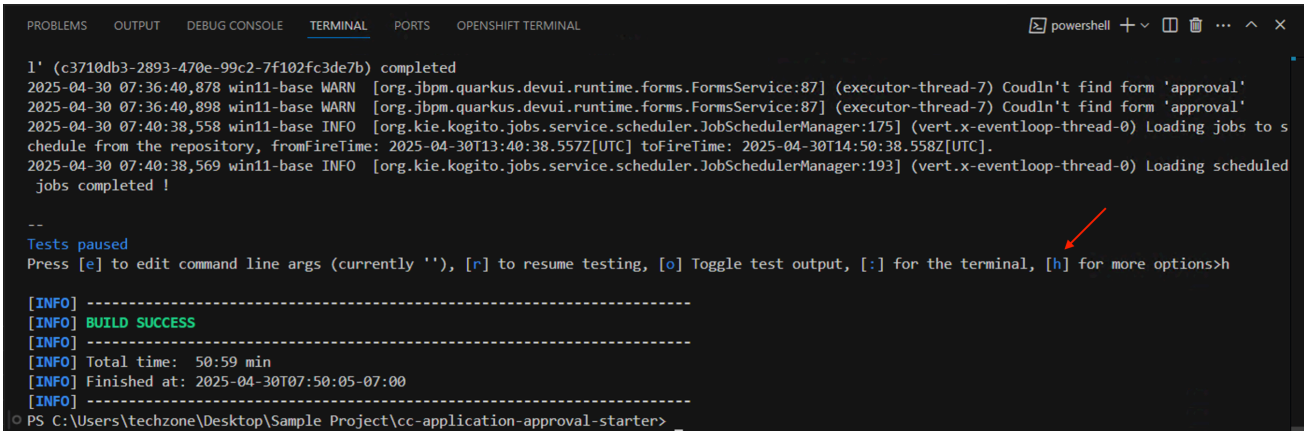
Navigate to the ***"Process Definitions"*** tile to run the following test scenarios. Use the form and repeat the test with different data that will result in different decision outputs.

AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	SCENARIO
25	*15000*	*750*	*NO*	*Automatic approval*

Validate by variables if the result of each scenario was like this:

If you got the same result, congratulations! Let's move on to the next exercise.

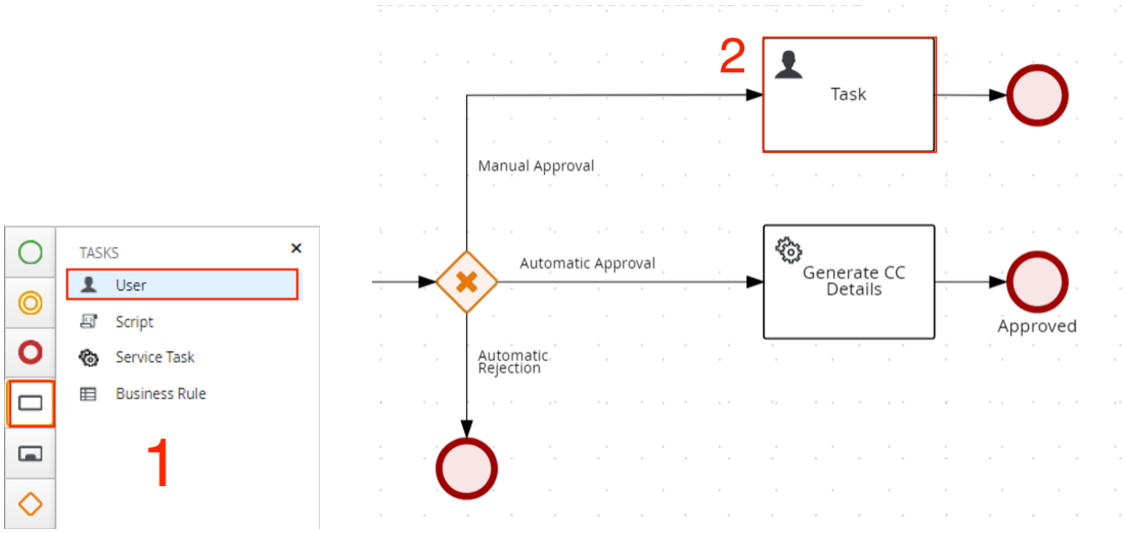
Close the browser. Let's work some more in VS Code and then review again when we're done. Back in the VS Code terminal, stop the execution by pressing the options: [h] then [q].



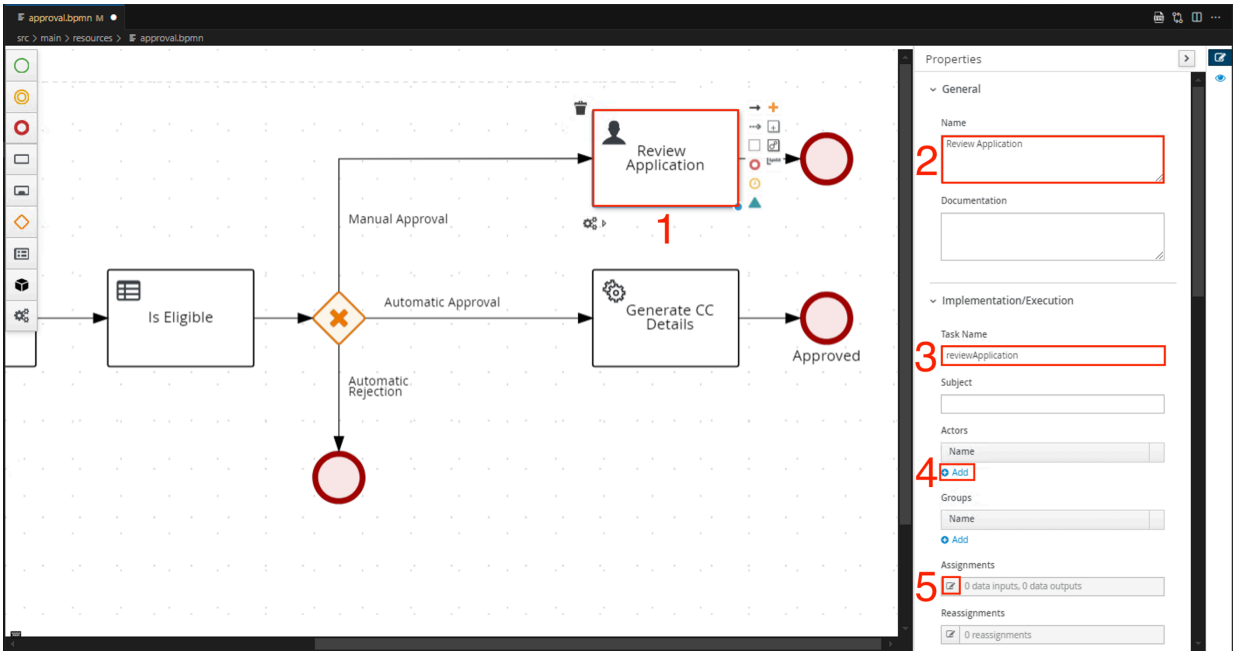
8 Exercise 5: Setting up process for Manual Approval

Now we can set up the second possible scenario, when the decision result indicates the need for manual approval, the flow should advance to a user task. Let's add this manual step to the process and see it in action!

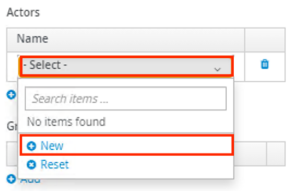
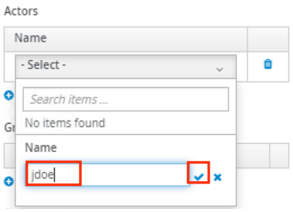
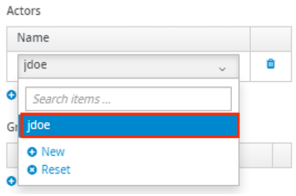
We will be adding new elements to the "Manual Approval" flow:

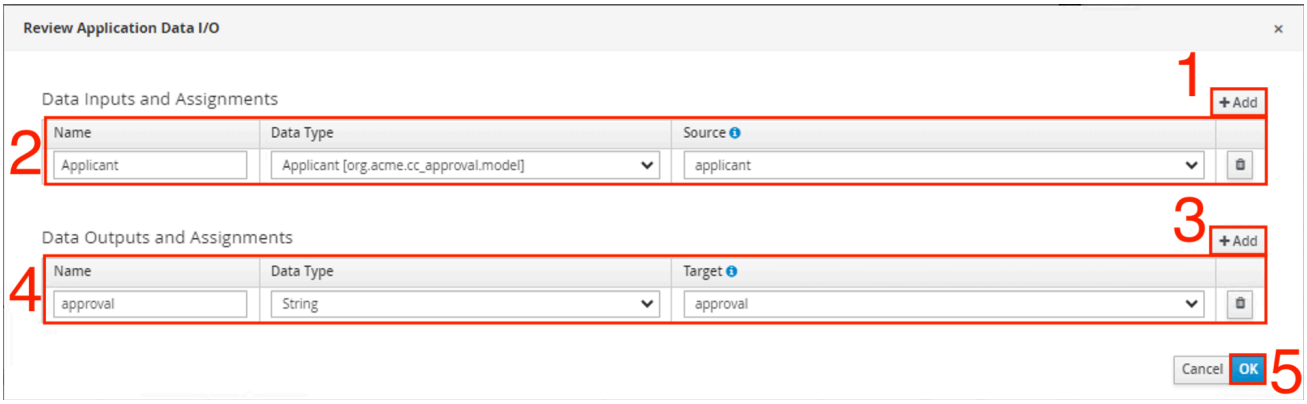


Item	Description
1	Add a task of type ***"User"***
2	Reorganize the ***"Manual Approval"*** flow to add the user task after the gateway and end event



Item	Description
1	Select the task user and open the properties panel

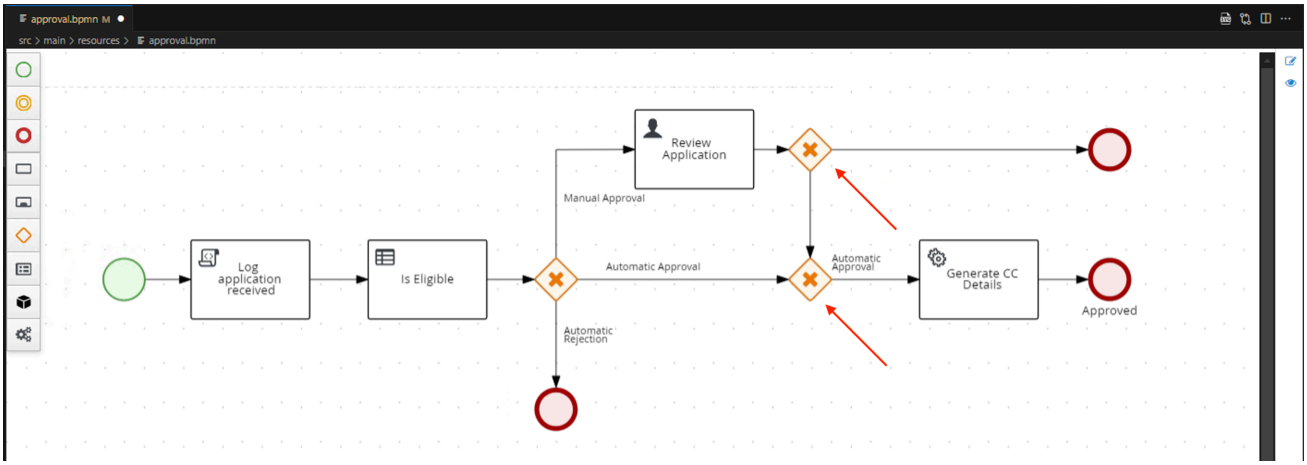
<table>
<tr><td>2</td><td>Rename the task and enter "Review Application"</td></tr>
<tr><td>3</td><td>In the "Task Name" field, enter "reviewApplication"</td></tr>
<tr><td>4</td><td>Let's link a user responsible for the task, click on "Add":
<table>
<tr><td> 
</td><td>Click on the "- Select -" listbox, then click on the "New" button.</td></tr>
<tr><td> 
</td><td>Enter the default user name of our project "jdoe" and click the button to confirm.</td></tr>
<tr><td> 
</td><td>Select to finish</td></tr>
</table></td></tr>
<tr><td>5</td><td>Click the button to configure the task inputs and outputs.</td></tr>
</table>

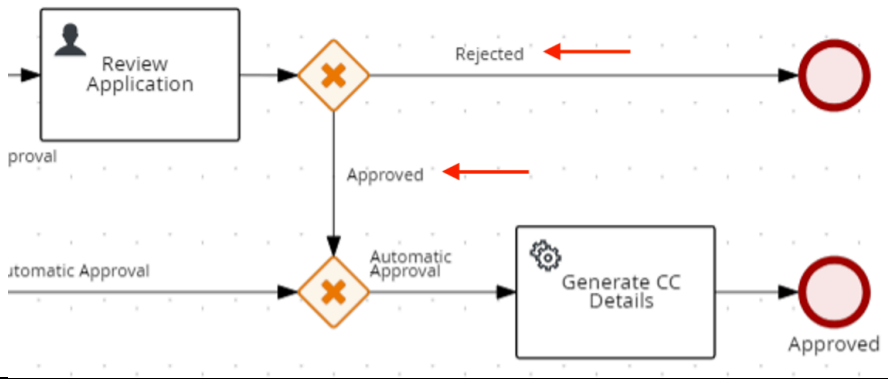


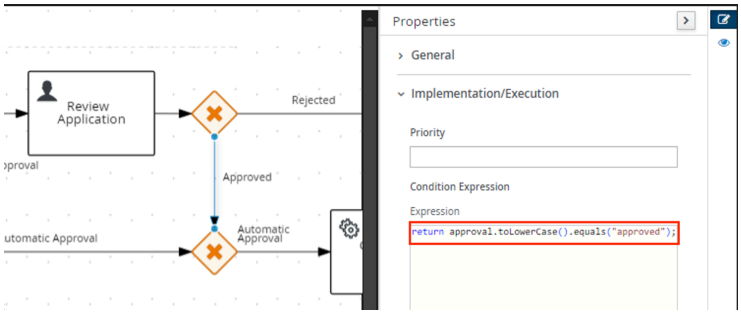
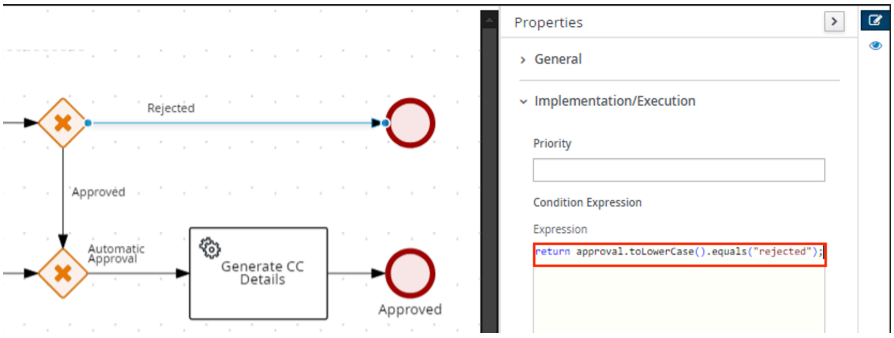
Item	Description
1	Click ***"+ Add"***
2	Fill in the ***Data Input*** fields: **FIELD** / **DATA** Name — *Applicant* Data Type — *Applicant[org.acme.cc_approval.model]* Source — *applicant*

3	Click ***"+ Add"***
4	Fill in the ***Data Output*** fields: **FIELD** / **DATA** Name: *approval* Data Type: *String* Source: *approval*
5	Click "***OK***"

The process diagram after the Review Application Human task can be configured many different ways, below is just an example of a possible solution, you do not have to have all of the exclusive gateways coming together to simplify if you choose not to. That is ultimately your choice!



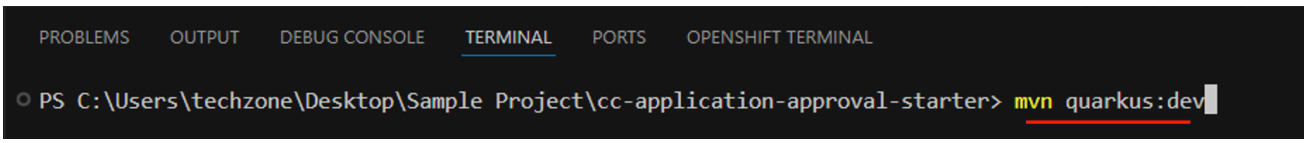
Item	Description
1	Please note that I have reorganized the flow and added 2 new Exclusive Gateways
2	And after the "Review Application" task, we will have two new paths: "Approved" and the other "Rejected" 

3	For the ***"Approved"*** path, set the condition to: `return approval.toLowerCase().equals("approved");` 
4	For the ***"Rejected"*** path, you can set it to be the default path after the manual approval so that if it doesn't explicitly say approved, the process will proceed to the rejected applications path. If you create an exclusive gateway after the Review Application human task, you can select the default route under Implementation/Execution to be Rejected Requests or you can set the exact condition if you'd prefer. To manually set the condition it would need to be set to: `return approval.toLowerCase().equals("rejected");` 

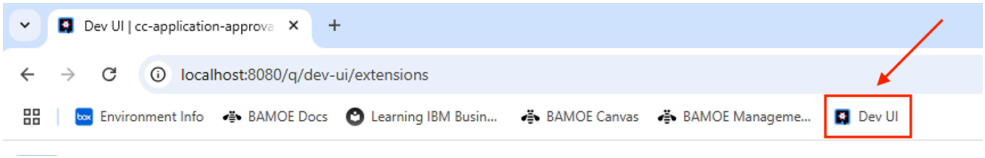
Before running the project, make sure the project has been saved ***(CTRL + S)***, and click the SVG button  (in the top right corner), to update the process image, as it will be used in the process instance details in DEV-UI.

Back in the terminal, run the command to run the project.

```
mvn quarkus:dev
```



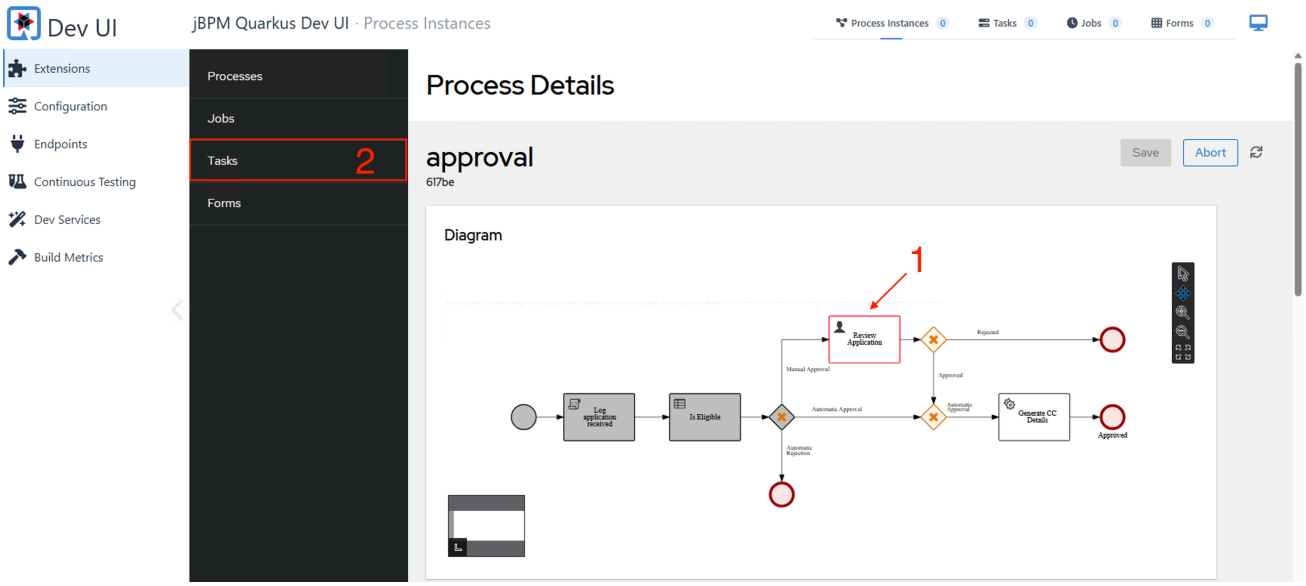
Now in the browser, open DEV-UI: http://localhost:8080/q/dev-ui



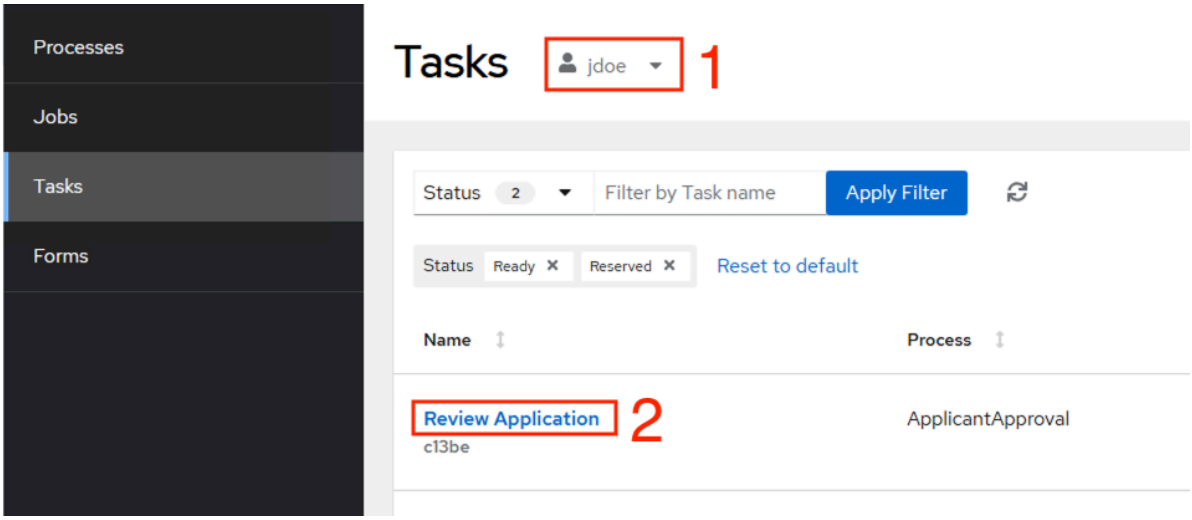
Note: And since we have already navigated this area before, I believe you are already familiar with navigation, if you have any questions, go back and check the steps.

Navigate to the ***"Process Definitions"*** tile to run the following test scenarios. Use the form and repeat the test with different data that will result in different decision outputs.

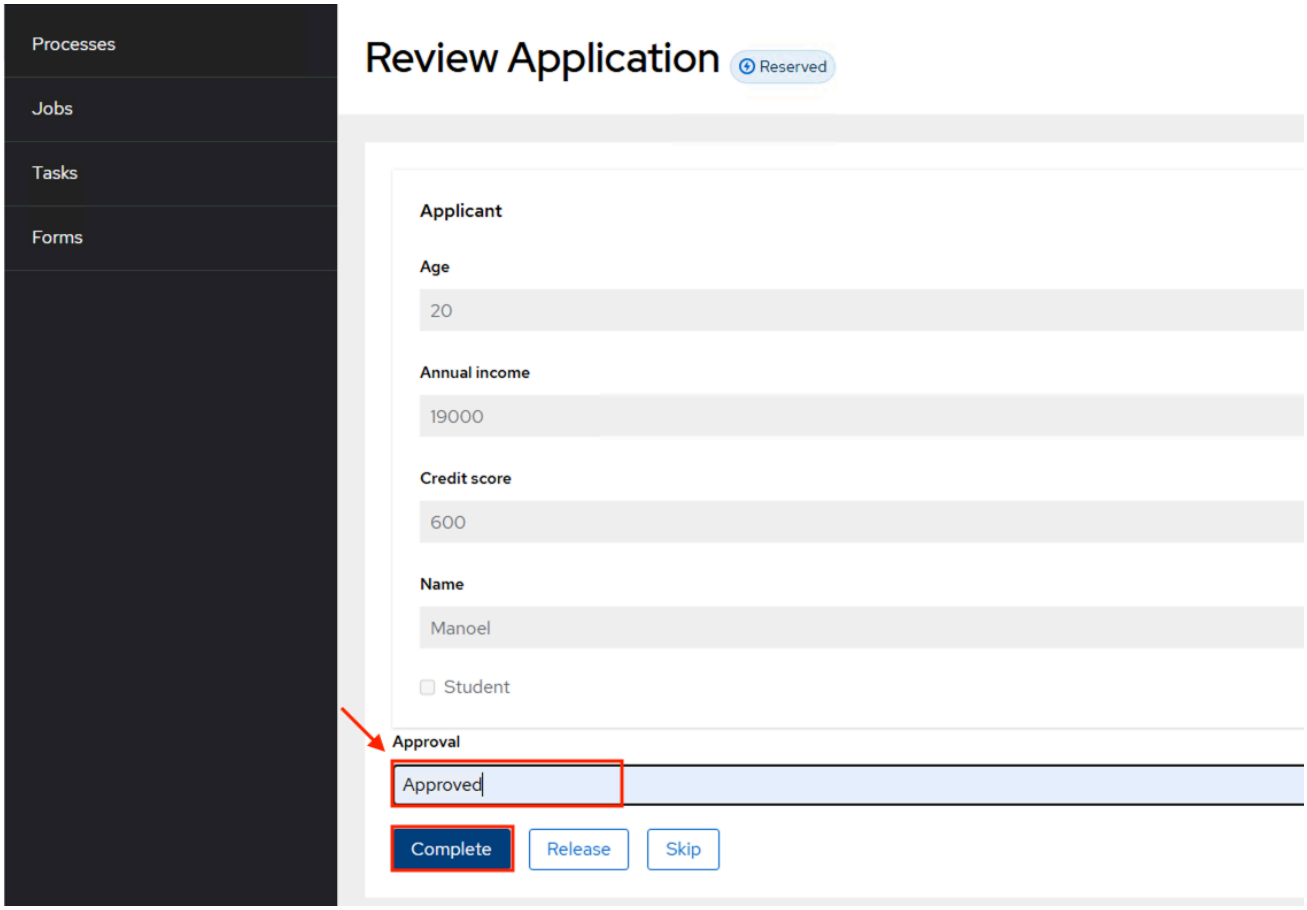
AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	SCENARIO
20	*19000*	*600*	*NO*	*Manual Review*



Item	Description
1	Note that the instance is already waiting for the action of the user ***"jdoe",*** to approve or not this application.
2	Access the ***"Tasks"*** screen in the side menu

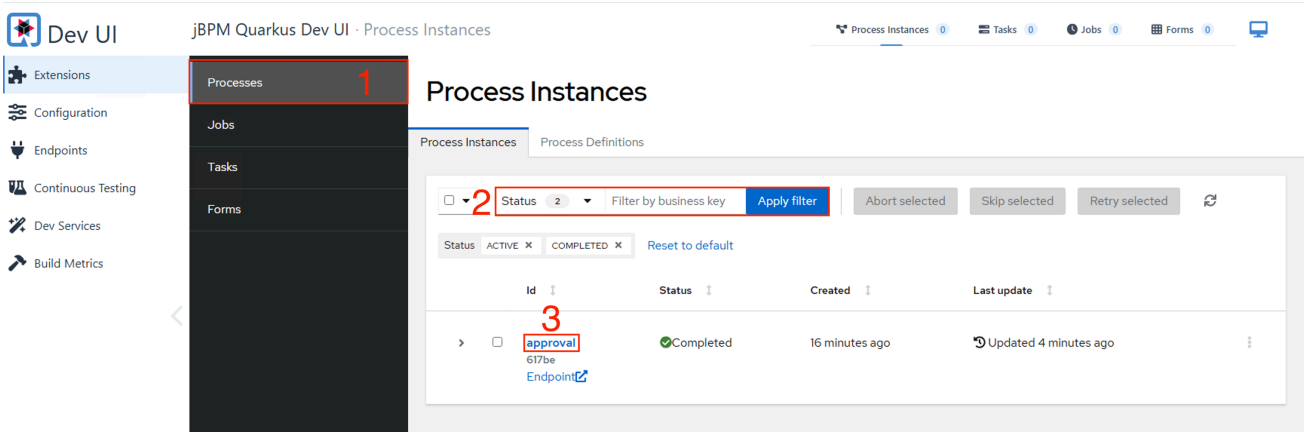


Item	Description
1	Select user ***"jdoe"***
2	Access the instance by clicking on ***"Review Application"***



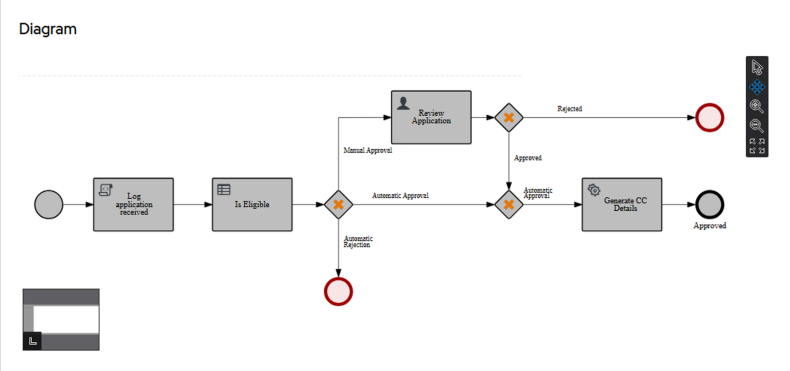
Item	Description
1	Type ***"approved"*** so we can evaluate the result
2	Select the ***"Complete"*** button

Now, let's return to "***Process Instances***" to evaluate the result.

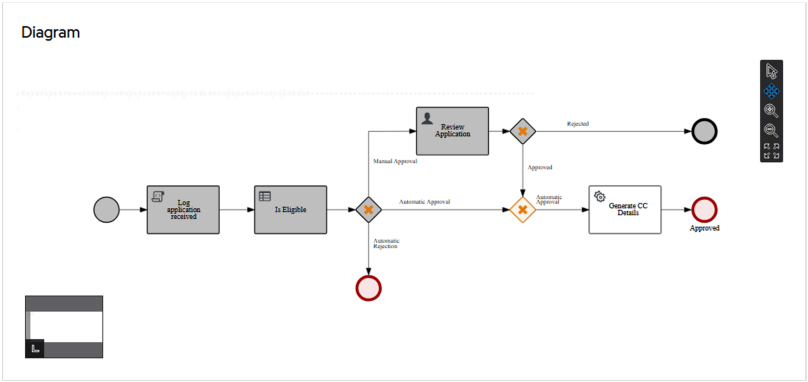


Item	Description
1	Click on ***"Processes"***
2	Don't forget to add the ***"Completed"*** status to the filters.
3	Select the instance by clicking on ***"approval"***

Validate the result using the diagram and variables, where you can see the path taken and the **approval result**:



Now, repeat the same test, but in the user action, reject the user application, to evaluate the flow.
*Note: To reject, just write **"rejected"** in the **"jdoe"** action.*



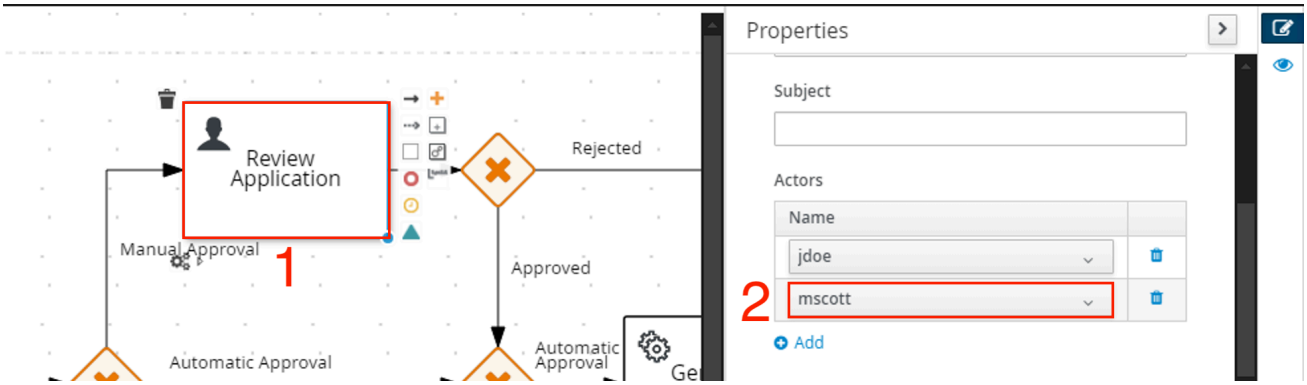
If you got the same result, congratulations! Let's move on to the next exercise.

Close the browser. Let's work some more in VS Code and then review again when we're done.
Back in the VS Code terminal, stop the execution by pressing the options: [h] then [q].

9 Exercise 6: Adding a new user

Now this process right now is assigned to just ***"jdoe"***, ideally, we should use a group, but for this lab we're going to attach a user. Let's say instead of ***"jdoe"***, the tasks will now go to ***"mscott"***. We need to make changes for the Dev-UI to work with this.

Note: Before proceeding be sure to close the Chrome instance of the Dev-UI console.



Item	Description
1	Back in the workflow, select the user task ***"Review Application"***
2	Repeat the same process to add a new user, including ***"mscott"***

If you were to go into Dev-UI now and repeat the steps from the previous section, you would see that a task has been created, but there is no ***"mscott"*** actor in Dev-UI to use. How can we do this? Very easily, a quick change to the properties will allow it to be added to the Dev-UI console.

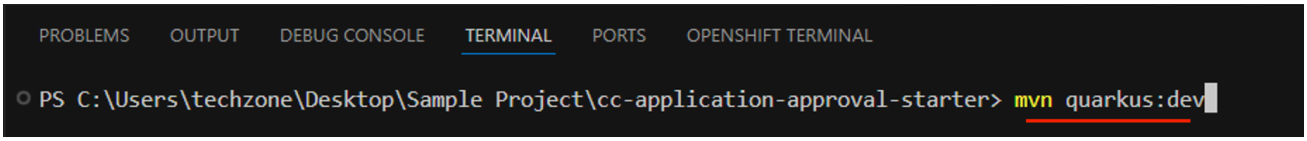
```
102
103  #############
104  # Misc. dev #
105  #############
106
107  %dev.quarkus.smallrye-openapi.path=/docs/openapi.json
108  %dev.quarkus.http.test-port=0
109  %dev.quarkus.dev-ui.cors.enabled=false
110  %dev.quarkus.swagger-ui.always-include=true
111  %dev.quarkus.kogito.data-index.graphql.ui.always-include=true
112  %dev.jbpm.devui.users.admin.groups=admin
113  %dev.jbpm.devui.users.jdoe.groups=admin,HR,IT
114  %dev.jbpm.devui.users.mscott.groups=admin,HR,IT
```

Item	Description
1	In VSCode, open "***src/main/resources/application.properties***"
2	If you notice on line 113 there is a property that says ***"jdoe"*** is the user. Copy this line, changing ***"jdoe"*** to "***mscott"***. Now ***"mscott"*** will have the same groups as ***"jdoe"***.

Before **running** the project, make sure the project has been saved ***(CTRL + S)***.

Back in the terminal, run the command to run the project.

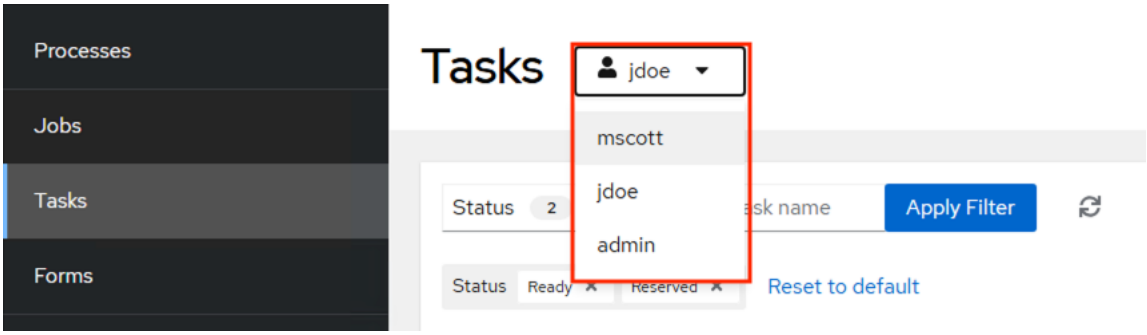
```
mvn quarkus:dev
```



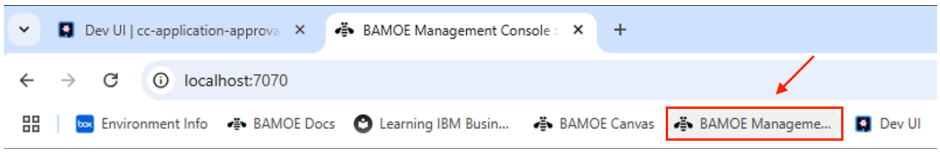
Now in the browser, open DEV-UI: *http://localhost:8080/q/dev-ui*

Create a manual approval task and verify that it is now assigned to ***"mscott"***

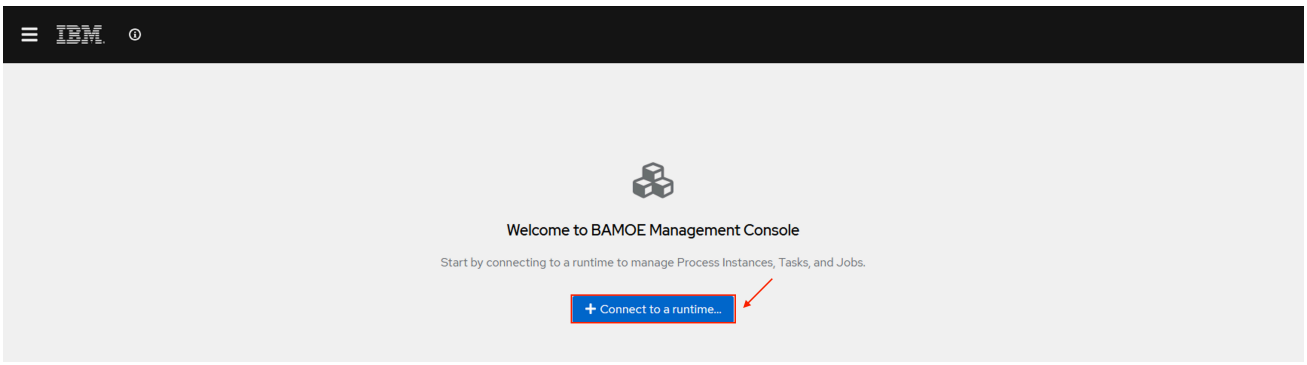
AGE	ANNUAL INCOME	CREDIT SCORE	IS STUDENT	SCENARIO
20	*19000*	*600*	*NO*	*Manual Review*

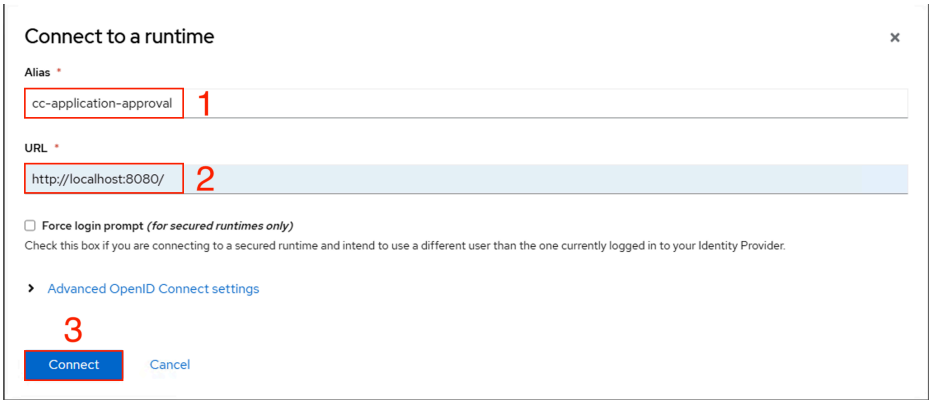


You can also connect your application to the BAMOE Management Console.
Go to http://localhost:7070/

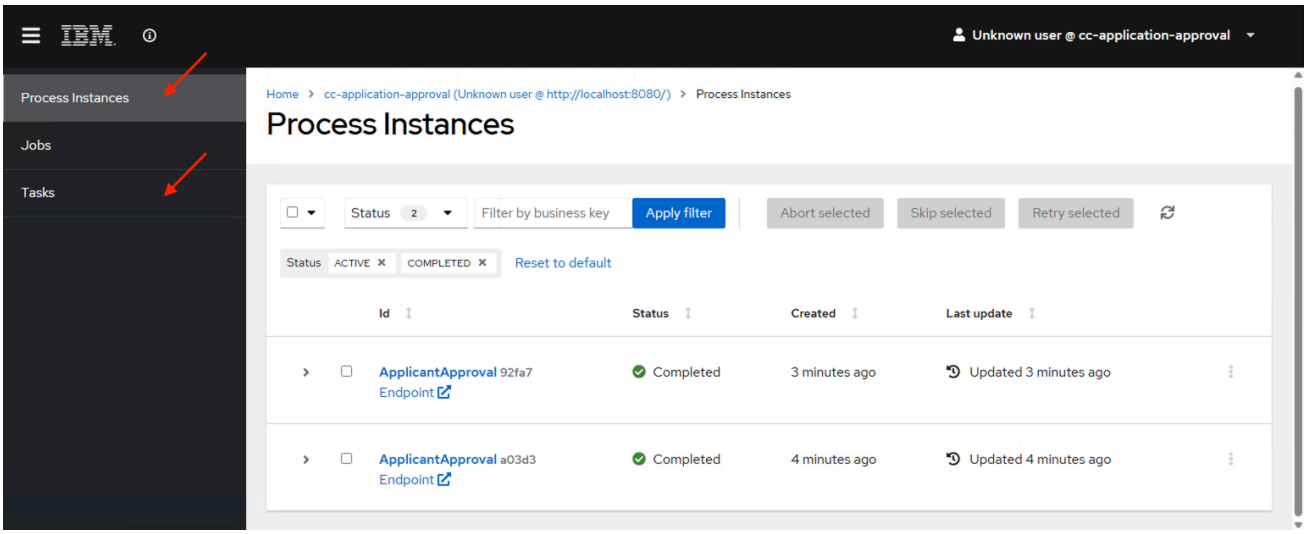


Click "Connect to a runtime..."





Item	Description
1	Let's use the same project name ***"cc-application-approval"***
2	Enter ***http://localhost:8080/***
3	Click on ***"Connect"***



Now you can experiment with how the user can perform their tasks, try creating a new instance through Dev-UI or making a call through Swagger, and see the result in the Management Console. Access the IBM documentation for more information.

Congratulations! You have completed this lab, where we explored a practical use case with Workflow and Decision. I hope that from now on you will be able to create and explore your own cases and other BAMOE features.

10 Consult Documentation and Communities

- IBM BAMOE Official Documentation
- IBM Business Automation Community: Open Editions