Oracle® Cloud

Using Agentic AI in Oracle Integration 3

G41284-05
January 2026

Oracle Cloud Using Agentic AI in Oracle Integration 3,

G41284-05

Primary Author: Oracle Corporation

Contents

4 Design and Run AI Agents

About This Content

Using Agentic AI in Oracle Integration 3 describes how to automate your workflows using AI agents and agentic AI tools in Oracle Integration.

Audience

Using Agentic AI in Oracle Integration 3 is intended for users who want to automate workflows with AI agents and agentic AI tools in Oracle Integration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Resources

For more information, see these Oracle resources:

- Oracle Integration documentation on the Oracle Help Center.

Conventions

The following text conventions are used in this document.

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

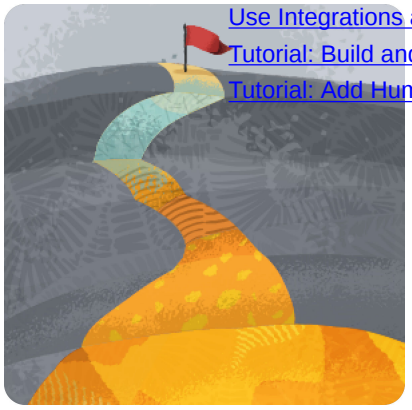
Welcome to Agentic AI

Use AI agents in Oracle Integration when you need to orchestrate different integrations in an adaptable and flexible automation. AI agents use integrations as tools to communicate with external systems and perform tasks.

Explore Agentic AI

Image	Links
	How AI Agents and Integrations Fit Together

Automate with AI Agents

Image	Links
	Use Integrations as Tools in an MCP Server Tutorial: Build and Run Your First AI Agent Tutorial: Add Human Approval to Your AI Agent

2

Get Started

Get started with Oracle Integration and agentic AI.

Topics:

How Integrations and AI Agents Fit Together

You use AI agents in Oracle Integration when you need to orchestrate different integrations in an adaptable and flexible automation. You're looking to achieve a complex business goal. AI agents orchestrate integrations as tools to fit your business needs.

Integrations Define a Set Order of Steps

With integrations, you define a specific order of steps. The steps are run the exact same way every single time. You use integrations when you want to control the exact steps and order, and to get enterprise connectivity.

In some cases though, you may have too many possibilities and designing an integration would be too complex. For those cases, you can use AI agents.

Agents Adapt: the Integrations Used Depend on the Goal

An AI agent is a software program that uses a Large Language Model (LLM) to reason without human intervention to achieve a specific goal. AI agents decide which steps to take and in which order, depending on what is happening in the environment. AI agents are more adaptable.

Integrations are Tools for AI Agents

AI agents use tools to communicate with the world. In Oracle Integration, integrations become agentic AI tools. AI agents use integrations as agentic AI tools to connect to the external world. AI agents determine which integrations to use and in which order to achieve the goal that you define. You create AI agents in projects.

AI Agents Use Thinking Patterns to Reason and Make Decisions

AI agents use specific thinking patterns to reason and make decisions. Patterns that are available by default are:

- ReAct
- Plan and Execute

You can also customize the existing thinking patterns or create your own for your AI agents.

Integrations are Discoverable from MCP Clients and Agent Frameworks that Support MCP

You can create AI agents in Oracle Integration, or you can create AI agents in other agent frameworks such as AI Agent Studio for Fusion Applications, Langflow, or others. Discover and use integrations as tools in any third-party agent framework that supports MCP.

Regardless of the agent framework that you use, any integration can be used as an agentic AI tool. You can define which integrations in a projects to expose as tools as an MCP server. Each project becomes an MCP server.

Involve Humans for Approval with Human in the Loop

Human in the loop enables human intervention in key decision areas, when errors occur, and to ensure quality and reliability. Human in the loop ensures people are always in control.

For example, you have an AI agent that automates employee onboarding. If important documents are missing or a background check raises a flag, the AI agent can ask Human Resources for review before continuing.

You use Human in the loop:

- When approval is needed for an AI agent to execute a particular tool
- When the AI agent needs approval before doing something else
- When the AI agent doesn't know what to do. Instead of ending, the AI agent can ask a human what the next steps should be
- For error handling, when an AI agent calls a tool that results in an error, the agent can tell a human about it who can provide feedback on what to do

Prompt Templates Facilitate Dynamic Prompting

Prompt templates provide a ready-to-use system prompt with placeholders for variables. Prompt templates are populated with values from the payload, then sent to the LLM. An AI agent can be associated with one or more prompt templates.

For example, if you had a weather agent to get the weather for a particular city, you could have different prompt templates depending on the country:

- `Get the weather for {{ payload.city }} in USA`
- `Get the weather for {{ payload.city }} in FRANCE`
- `Get the weather for {{ payload.city }} in BRAZIL`

Corporate Documents become Knowledge Bases for Agent Actions

Corporate documents are no longer silos and difficult to find and consult. Any corporate document can become a knowledge base that AI agents and humans can query and use as reference. You add documents to a RAG knowledge base that exists within Oracle Integration and query those documents from within an integration, or use the knowledge base as a tool for AI agents.

Tutorial: Build and Run Your First AI Agent

This tutorial walks you through creating a very simple AI agent with one integration as an agentic AI tool. We'll also run the AI agent with different inputs and monitor the runs.

Background

We'll create a simple AI agent for expense reimbursement. Any expense that is less than $50 USD can be automatically approved. Any expense above $50 USD is rejected.

There's one tool in this AI agent:

- Auto Approve Expense Report

What do You Need?

- Access to Oracle Integration. If you're not yet using Oracle Integration, get a free trial of Oracle Integration.
- Connection information for the Large Language Model (LLM) to which you'll be connecting. You'll need:
  - Base URL for the LLM. Example: `https://api.openai.com`
  - Model to use. Example: `gpt-40-mini`
  - API Key for the LLM

What Do I Learn from this Tutorial?

This tutorial walks you through performing the following tasks:

- Creating a project for your AI agent.
- Creating a simple tool to automatically approve an expense report.
- Creating an AI agent that analyzes the input and uses the tool to approve the expense report.
- Running the AI agent.
- Monitoring the AI agent run.

Next Step: Steps to Build an AI Agent

Steps to Build an AI Agent

Here's a summary of the steps we'll follow to build the AI agent.

Step	Tasks
Complete Prerequisites	Get connection information for the Large Language Model (LLM) you'll be using. You'll need this information to configure your LLM for your AI agent. Required information: • Base URL for the LLM. Example: `https://api.openai.com` • Model to use. Example: `gpt-40-mini` • API Key for the LLM
1	Plan Your AI Agent. Take a few moments to plan your AI agent. This will help you to clearly define the AI agent's purpose and which tools it will need.
2	Create the Project for the AI Agent. You create AI agents inside projects.
3	Create integrations optimized for AI agents. In order to use an integration as an agentic AI tool, the integration must have a REST trigger, a POST verb, and JSON input and output. 1. Create the integration, activate it, and register it as an agentic AI tool. a. Create the Integration: Auto Approve Expense Report b. Register the Integration as an Agentic AI Tool
4	Build the AI agent. 1. Configure the Agentic AI Thinking Pattern. Define which thinking pattern your AI agent will use and configure your LLM connection information. 2. Add and Configure the AI Agent.
5	Run and Test the AI Agent.

Next step: Plan Your AI Agent.

Plan Your AI Agent

Before we start creating the AI agent, let's identify the agent purpose and which tools the AI agent will use. This will help you have all the required information when creating the AI agent and tools in Oracle Integration.

What's the Purpose of the AI agent?

For this tutorial we're creating a simple AI agent to approve or reject expense claims.

AI Agent	Purpose
Process Expense Claims	Approve or reject expense claims. Automatically approve expenses equal to or less than $50 USD. Reject the expense if the expense is more than $50 USD.

What Tools does the AI Agent Need?

The AI agent needs one integration to automatically approve expense claims. Since all the reasoning is done in the AI agent, the integration can be very simple.

Identify which tools are needed, their purpose, input and output parameters.

Here's an overview of the tools we need for the AI agent:

Tool Name	Purpose	Input Parameters	Output Parameters
Auto Approve Expense Report	Automatically approve submitted expenses.	`AmountClaimed`: string. This is the dollar amount submitted for reimbursement. Example JSON: `{ "AmountClaimed" : "" }`	`AutoApproveResponse` : string. Indicates whether the expense has been approved. Value is always Approved since this integration is only used for automatic approval. Example JSON: `{ "AutoApproveResponse" : "" }`

Next step: Create the Project for the AI Agent. Now that we know what the agent does and the tools that we need, we're ready to start building the components.

Create the Project for the AI Agent

Before you can create an AI agent, you need to create a project.

1. In the navigation pane, select **Projects**.
2. Click **Add**.

3. Select **Create**.

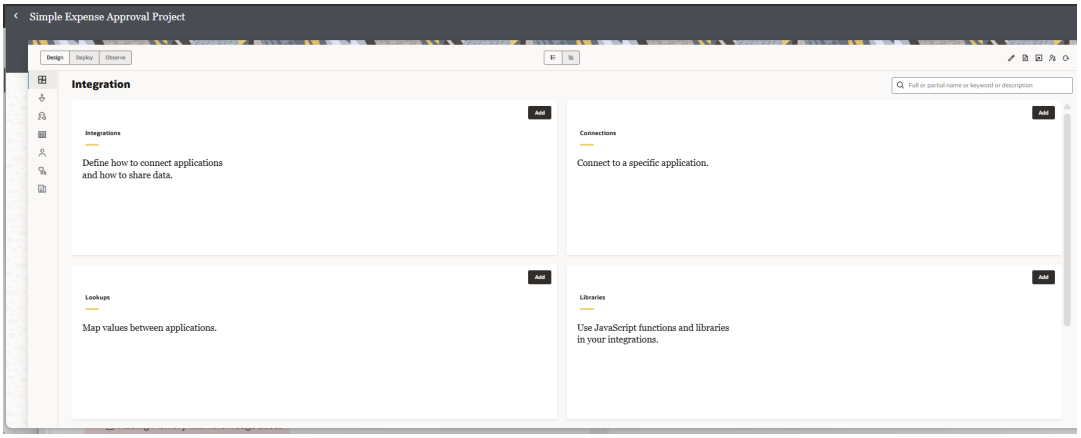
   The Start a new project page is displayed.

4. Under **New project**, click **Choose**.

   The Create projects panel is displayed.

5. In **Name:**, specify `Simple Expense Approval Project` and click **Create**.

   The project detail page is displayed.



Next step: Create the Automatic Expense Approval Tool. You can now create integrations to use as agentic AI tools for your agent.

Create the Automatic Expense Approval Tool

Integrations become tools for AI agents. Create the integration, activate it, then register it as an agentic AI tool.

1. Create the Integration: Auto Approve Expense Report.
2. Register the Integration as an Agentic AI Tool.

Create the Integration: Auto Approve Expense Report

Let's create the integration to automatically approve expense claims. Since all the reasoning is done in the AI agent, the integration can be very simple.

Requirements for integrations to be used as tools:

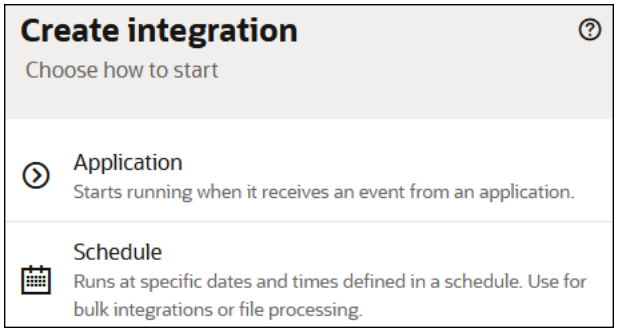
Any integration can be used as an agentic AI tool, but the integration must meet the following criteria:

- The integration must be part of a project.
- The integration must be Active. We'll be activating the integration after we create it.
- The first connection in your integration must be a REST trigger connection with the REST Adapter.
- The REST trigger connection in your integration must have:
  - JSON payload
  - POST verb

1. In your project, in the **Integrations** box, click **Add**.

   The Add integration panel opens.

2. Click **Create**, then select **Application**.



3. In the **Name** field, enter **`Auto Approve Expense Report`**, leave all other fields as they are, then click **Create**.

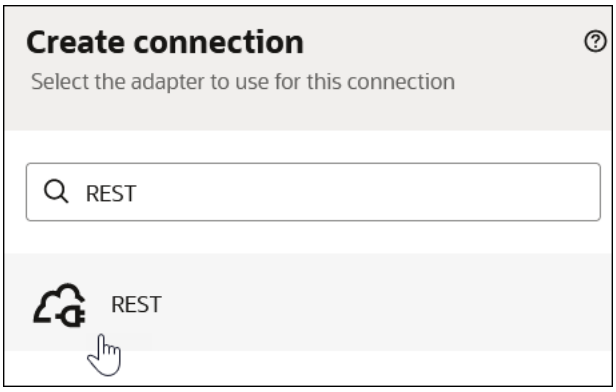
   The integration canvas opens with an option for creating a trigger connection.

4. Create the trigger connection. You need a trigger connection for an integration to be used as a tool.

   a. Click **Add trigger**.

      The Create connection panel opens.

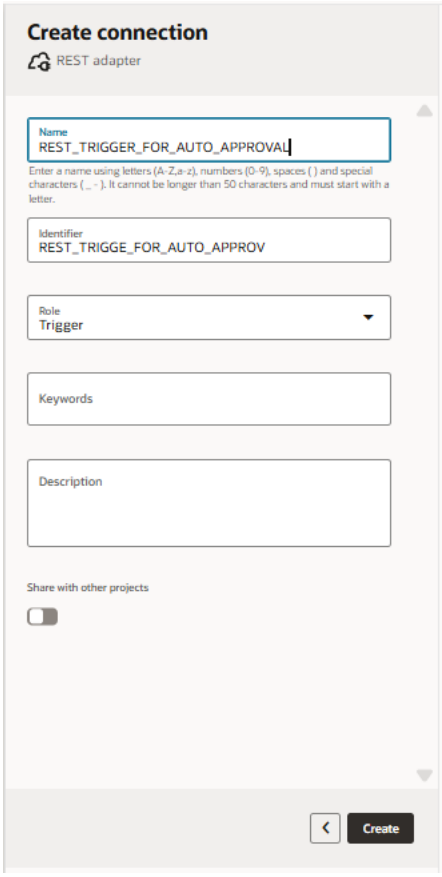
   b. Enter **`REST`** in the search field and select **REST** as the adapter connection to use.



      The Create Connection panel for configuring the REST Adapter connection opens.

   c. In Create connection, configure the **Name** and **Role** fields, leave all other fields blank, and click **Create**.

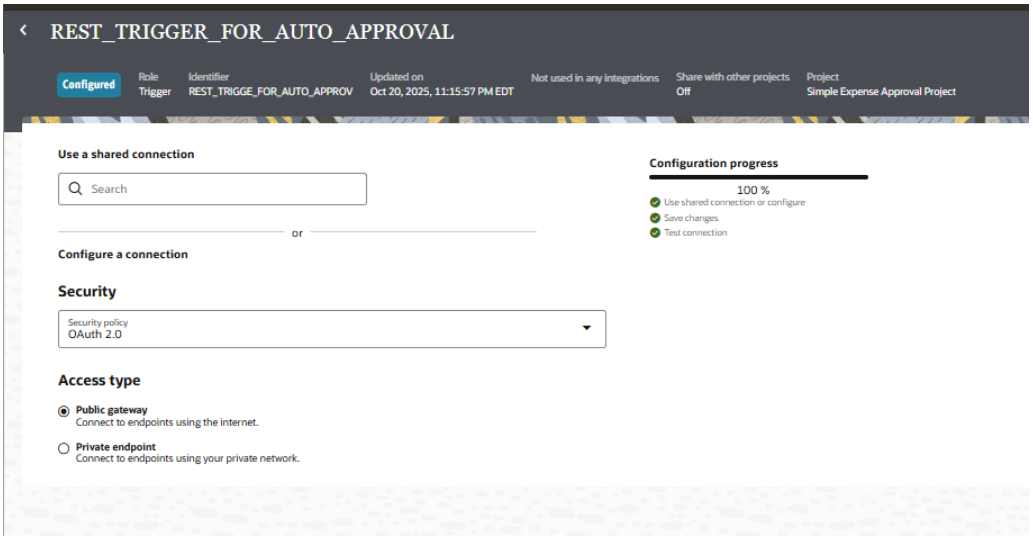
      - **Name**: For the connection name, specify **`REST_TRIGGER_FOR_AUTO_APPROVAL`**.
      - **Role**: Select **Trigger** so that an inbound application can run the integration.



The page for configuring security and delivery methods for the REST Adapter is displayed.

d. Configure connection security.

- **Security**: From the list, select **OAuth 2.0 Or Basic Authentication** to secure incoming messages.
- **Access type**: Ensure **Public gateway** is selected. This access type uses the public internet to receive messages.

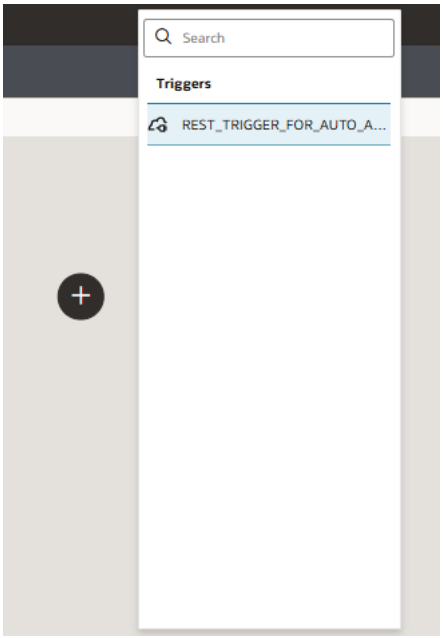


e. Click **Test** to test the connection.

f. Click **Save**.

g. Click **Back** to return to the integration canvas.

5. Click **Add** to display the **Trigger connections** dialog and select REST_TRIGGER_FOR_AUTO_APPROVAL.

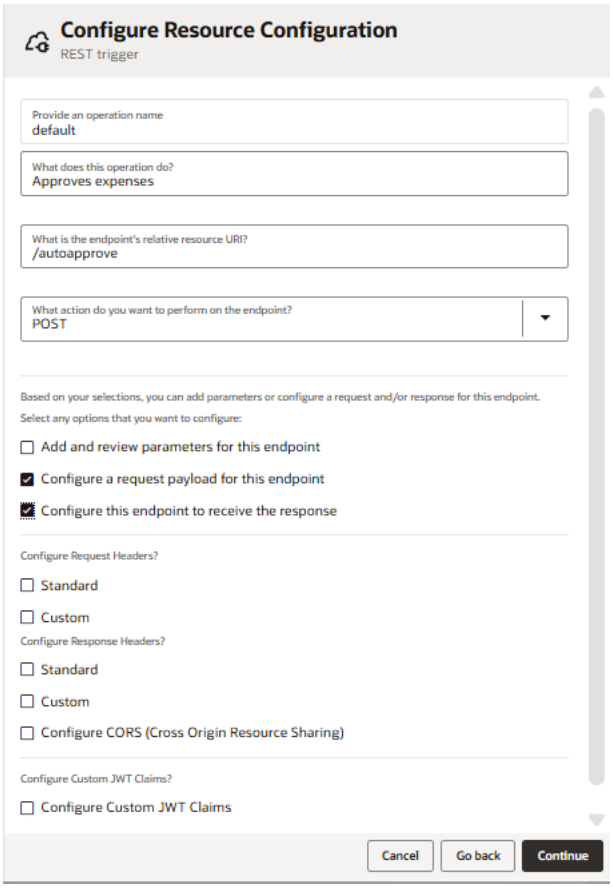


The Configure Basic Info page is displayed.

6. In **Configure Basic Info**, in the field **What do you want to call this endpoint** enter `AutoApproveExpense` and click **Continue**.

The Configure Resource Configuration page is displayed.

7. In **Configure Resource Configuration**, complete the fields as indicated, then click **Continue**:



- **What does this operation do?**: `Approves expenses`.
- **What is the endpoint relative resource URI?**: `/autoapprove`.
- **What action do you want to perform on this operation?**: `POST` . An integration must have a REST trigger with a POST verb to be used as a tool in AI agents.
- **Configure a request payload for this endpoint**: Make sure it has a checkmark.
- **Configure this endpoint to receive the response**: Make sure it has a checkmark.

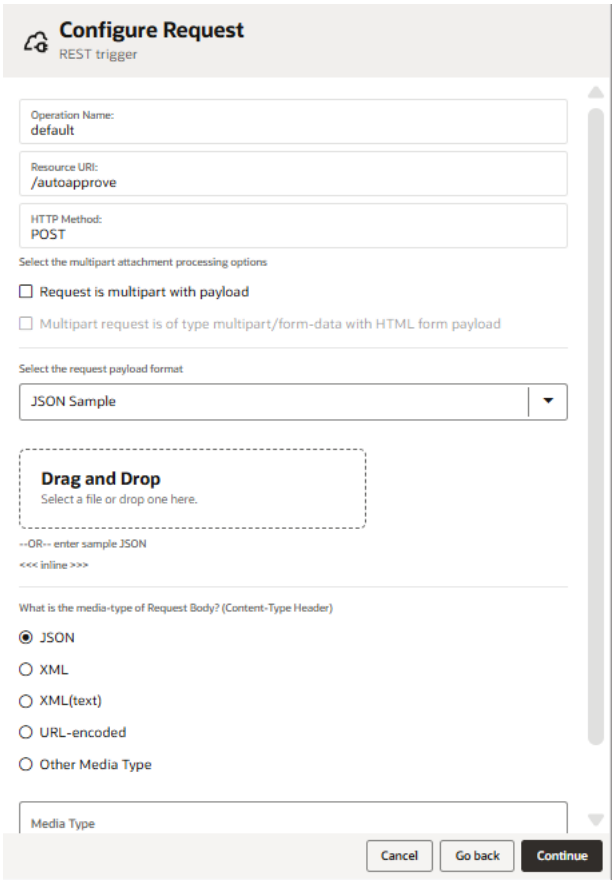
The Configure Request panel is displayed.

8. In Configure Request, complete the fields as indicated, then click **Continue**.

   The trigger connection must have JSON payload to be used as an agentic AI tool.

   - **Select the request payload format**: `JSON Sample`.
   - **<<<inline>>>**: Click and specify the following request format:

     ```
     { "AmountClaimed" : "" }
     ```



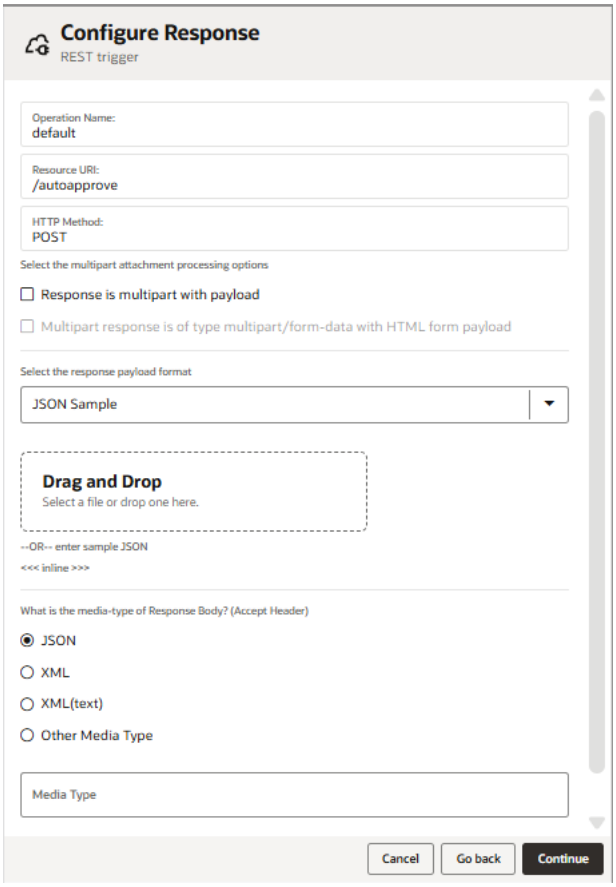
The Configure Response panel is displayed.

9. In Configure Response, complete the fields as indicated, then click **Continue**.

   The trigger connection must have JSON payload to be used as an agentic AI tool.

   - **Select the response payload format**: `JSON Sample`.
   - **<<<inline>>>**: Click and specify the following response format:

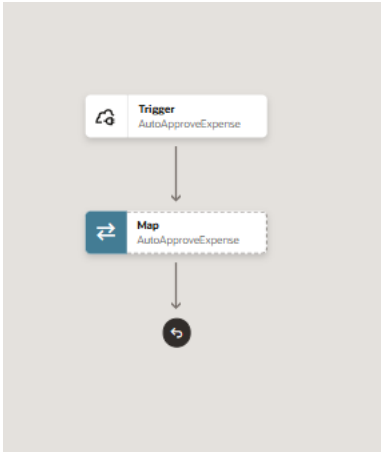
     ```
     {
       "AutoApproveResponse": ""
     }
     ```



The Summary page is displayed.

10. Click **Finish** to complete the trigger connection configuration.

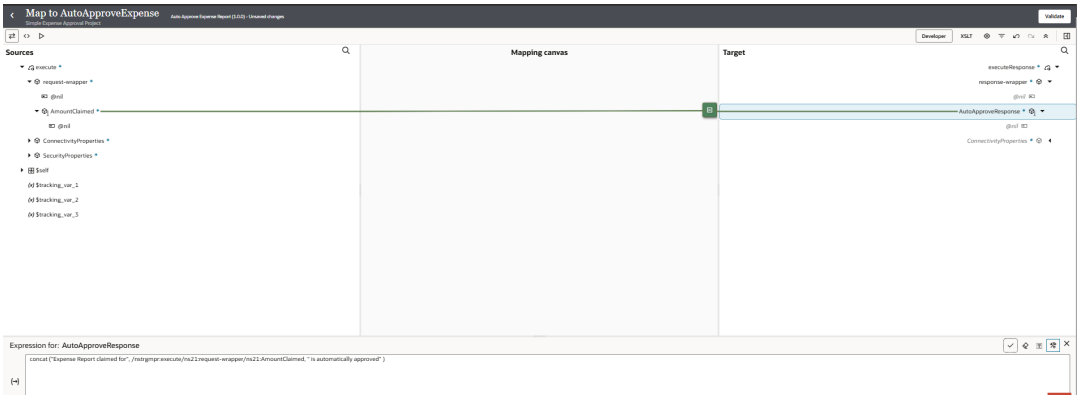
    Your integration should now look like this:



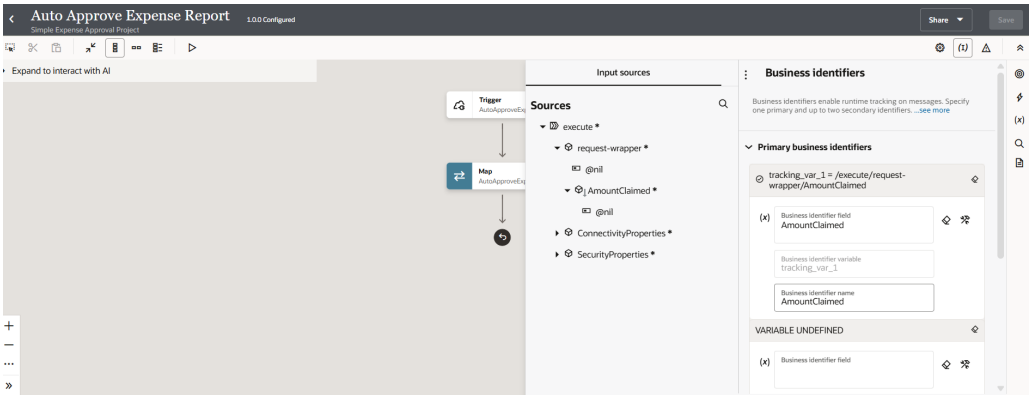
11. In the map action, click **Actions** ⋯, then select **Edit** ✎.
12. Under Request Wrapper, drag **AmountClaimed** to **AutoApproveResponse** to map the fields and add enclose the expression that is there with `concat ("Expense Report claimed for", , "is automatically approved")`

For example:

```
concat ("Expense Report claimed for", /nstrgmpr:execute/ns21:request-
wrapper/ns21:AmountClaimed, " is automatically approved" )
```



13. Click **Validate**.
14. Click **Back** to return to the integration canvas.
15. Assign a business identifier.
    a. Above the integration, click **Business Identifier**.
    b. Drag **AmountClaimed** as the primary business identifier.



16. Click **Save**.

    You have completed creating the integration that automatically approves an expense.

17. Click **Back** to return to the Project page.
18. Activate the integration. Only Active integrations can become agentic AI tools.
    - In the Integrations box, next to the integration **Auto Approve Expense Report**, click **Actions**, and select **Activate**.

Next step: Register the Integration as an Agentic AI Tool.

Register the Integration as an Agentic AI Tool

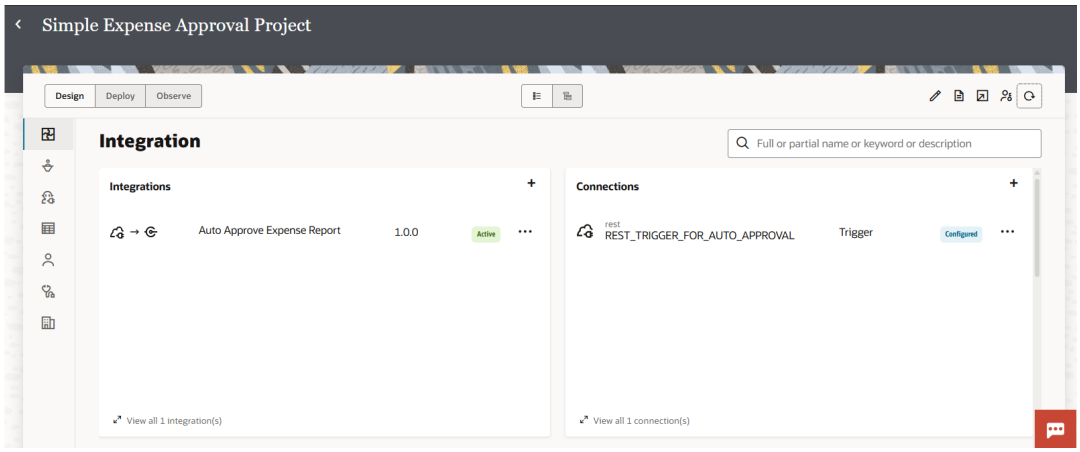
An integration can become an agentic AI tool for AI agents. AI agents can then invoke the integration as a tool to accomplish a specific task.

Let's register the integration as an agentic AI tool so that it can be used by the AI agent.

Prerequisites:

Create the Integration: Auto Approve Expense Report

1. In the navigation pane, select **Projects**.
2. Select the project in which your integration is located. For the tutorial, the project is **`Simple Expense Approval Project`**.
3. In the **Integrations** section, find the integration that you want to register as a tool: **`Auto Approve Expense Report`**.



4. Check that your integration is Active. If it's not active, activate it by clicking **Actions** ⋮, and selecting **Activate**.
5. Register the integration as a tool.
   - Next to **`Auto Approve Expense Report`**, select **Actions** ⋮, and select **Create agentic AI tool**.

   The Create Tool panel is displayed.
6. Enter information for the tool.

Field	Description
Name	Required. Automatically populated from the integration name. Not sent to the Large Language Model (LLM). You cannot change the name after the tool has been created.
Identifier	Required. Automatically populated from the tool name. Uniquely identifies the tool in the project. Sent to the LLM as tool metadata. The AI agent uses this information to understand the purpose of the tool and when to use it. You cannot change the identifier after the tool has been created

Field	Description
Description	Required. Automatically populated from the integration description, if available. Sent to the LLM as part of the system prompt. The description helps the LLM decide when to use the tool. Add information to clearly describe what the tool does and when to use it. Clear descriptions help LLMs use tools correctly and reduce errors. Enter the following description: `Automatically approves submitted expenses.`

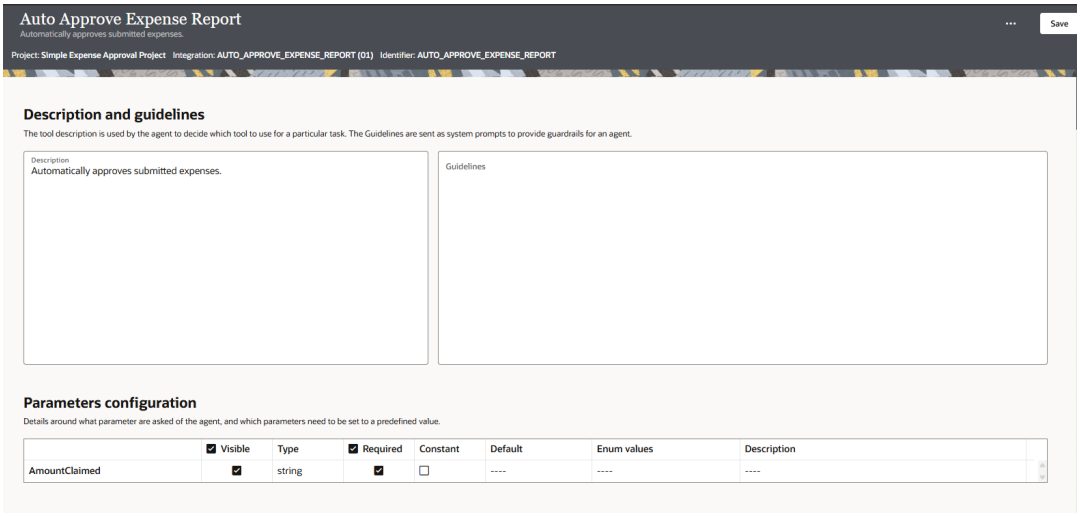
7. Click **Create**.

   The tool details page is displayed. The tool details page lists the tool description, guidelines, and input parameters. The tool description and guidelines are sent to the LLM. Guidelines are constraints to limit tool behavior and respect corporate policies. Guidelines influence the LLM decision process.

8. Review tool parameters.

   When we take a look at the Parameters configuration, we see integration input parameters. The `AmountClaimed` parameter is listed and it is sent to the LLM because it's marked as Visible. We don't need to make any changes.

   If you were working with an integration that had many parameters, you could make visible to the LLM only parameters relevant to the tool.



9. Click **Save** to save your changes.

10. Click **Back** to return to the Project page.

    The integration can now be used in an AI agent.

Next step: Build the AI Agent.

Build the AI Agent

Now that you have an agentic AI tool registered from the integration, you are ready to create your AI agent.

Prerequisites:

1. Create the Integration: Auto Approve Expense Report.
2. Register the Integration as an Agentic AI Tool.

Steps to Build the AI Agent:

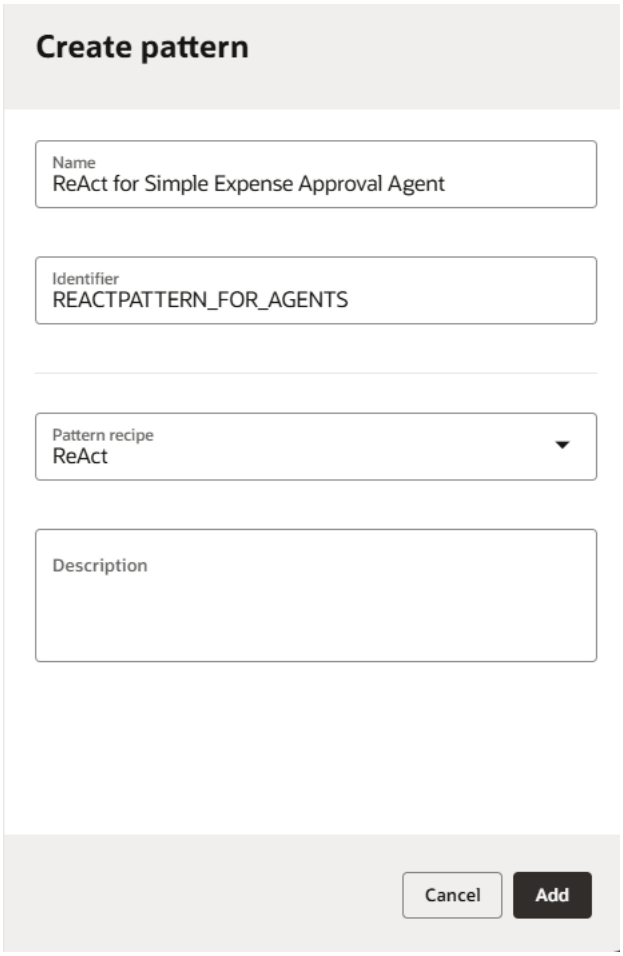
1. Configure the Agentic AI Thinking Pattern.
2. Add and Configure the AI Agent.

Configure the Agentic AI Thinking Pattern

AI agents can use different thinking patterns for reasoning. Before you can create an AI agent, you need to create the thinking pattern the agent will use.

1. In the left navigation pane, select **Projects**.
2. Select the Simple Expense Approval project that you created.
3. In the left navigation pane, select **AI Agents** .
4. In the **Agent patterns** box, click **Add**.

   The Create pattern panel is displayed.
5. In **Create pattern**, configure required fields for your pattern:



- **Name:** Name for the thinking pattern. For our tutorial, enter the name **`ReAct for Simple Expense Approval Agent`**. Currently, there are two patterns available in Oracle Integration. ReAct and Plan and Execute. You might want to identify your pattern with those names for easier understanding.
- **Identifier:** Specify `REACTPATTERN_FOR_AGENTS`.
- **Pattern recipe:**
  Value sent to the LLM as a system prompt to define how the agent reasons. Select the **ReAct** pattern for the tutorial. A pattern recipe is implemented as an integration that contains the thinking pattern for the AI agent. Oracle Integration ships with two ready-to-use patterns:
  - ReAct: Reasoning and Action. The AI agent alternates between reasoning and acting. The AI agent updates its plans based on new information that it learns. The AI agent prompts the LLM for the next action, acts, learns, and then repeats.
  - Plan and Execute: The AI agent prompts the LLM to generate a complete plan, then executes that plan step by step.

6. Click **Create**.

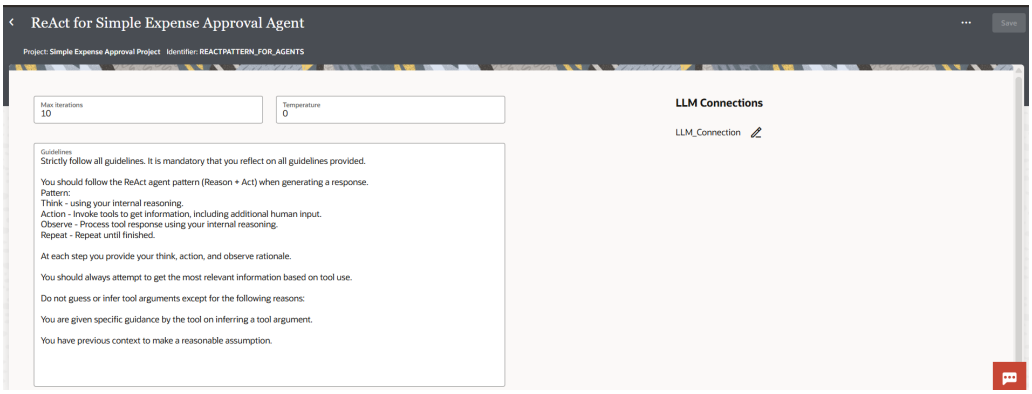
   The Pattern details page is displayed.

7. Configure the Pattern and LLM.

   a. Leave the following fields with the default number:

- **Max iterations**: Leave **Max iterations** as the default number. This limits the number of reasoning and action steps the agent can take before providing a final answer or exiting. This field prevents infinite loops.
- **Temperature**: Leave as the default number. A higher temperature means the LLM can be more creative in its thinking.

b. In the **Guidelines** box, enter the following guidelines for the ReAct thinking pattern:



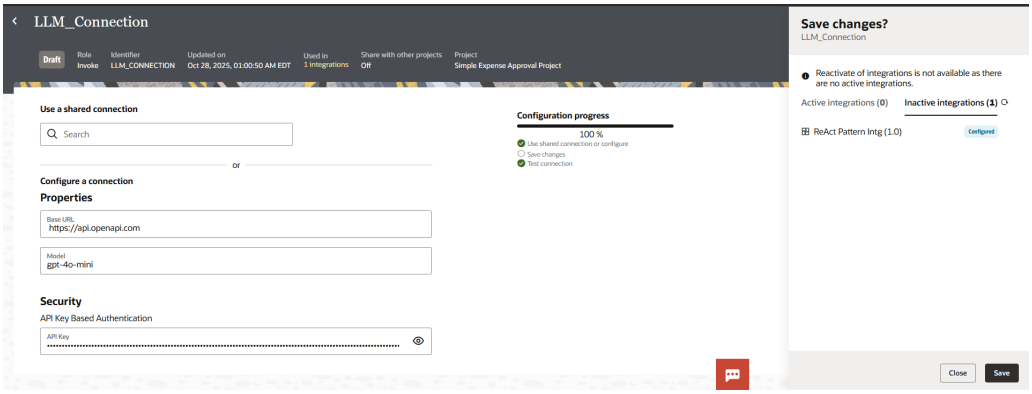

```
Strictly follow all guidelines. It is mandatory that you reflect on all
guidelines provided.

You should follow the ReAct agent pattern (Reason + Act) when
generating a response.
Pattern:
Think - using your internal reasoning.
Action - Invoke tools to get information, including additional human
input.
Observe - Process tool response using your internal reasoning.
Repeat - Repeat until finished.

At each step you provide your think, action, and observe rationale.

You should always attempt to get the most relevant information based on
tool use.

Do not guess or infer tool arguments except for the following reasons:

You are given specific guidance by the tool on inferring a tool
argument.

You have previous context to make a reasonable assumption.
```

c. Click **Save**.

d. Under **LLM Connections**, click  to configure the LLM connection.

A new LLM connection is created and is displayed.

e. Configure connection information for your LLM.



- **Base URL:** URL to connect to your LLM. For example: `https://api.openai.com`.
- **Model:** Model to use for your AI agent. For example: `gpt-4o-mini`
- **API Key Based Authentication:** Specify your API key to connect to the LLM.

f. Click **Test** to test that your connection works, then click **Save**.

When you save, Oracle Integration creates an integration that implements the ReAct pattern called **ReAct Pattern Intg**. This system-created integration is automatically activated for the agent.

8. Click **Back** to return to the Pattern details page.

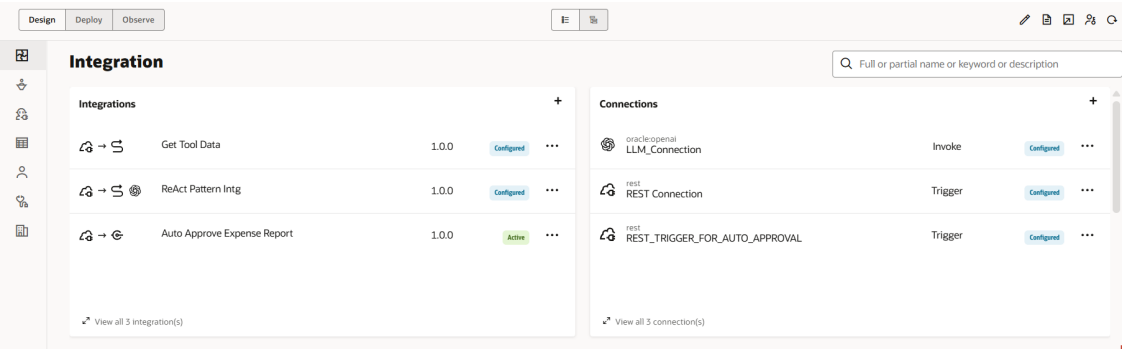
9. Click **Back** to return to the AI Agents page.

10. Activate the thinking pattern. The pattern must have the status Active before you can specify it in your agent.

- In the **Agent patterns** section, next to **ReAct for Simple Expense Approval agent**, click **Actions** ⋯, and select **Activate**.

Once your thinking pattern is created for your AI agent, Oracle Integration automatically creates two integrations in your project: ReAct Pattern Intg and Get Tool Data. Select the Integration section to see these integrations. These integrations are system-generated and are required for your AI agent. You don't need to change or edit them.

The LLM connection is also automatically created for you.



Next step: Add and Configure the AI Agent

Add and Configure the AI Agent

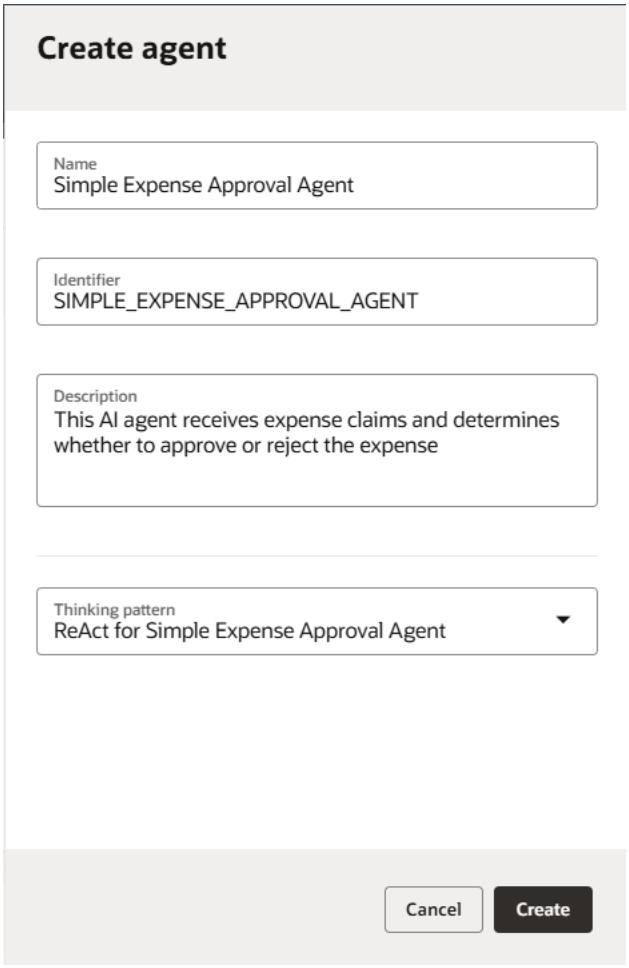
We now have all that we need to create the Simple Expense Approval agent. We created an integration and registered it as a tool, configured the ReAct agentic AI pattern, and configured the LLM.

Prerequisites:

- Integrations: Create the integration that you want to use in the AI agent and register the integration as an agentic AI tool. See Register the Integration as an Agentic AI Tool.
- Agentic AI thinking pattern: Create the thinking pattern your agent will use to reason. See Configure the Agentic AI Thinking Pattern.

1. In the navigation pane, select **Projects**.
2. Select the project in which to create your AI agent. For this tutorial, select the **Simple Expense Approval** project.
3. In the left navigation pane, select **AI Agents**  .
4. In the **AI Agents** box, click **Add** .

   The **Create agent** panel is displayed.
5. Configure required fields for your agent:

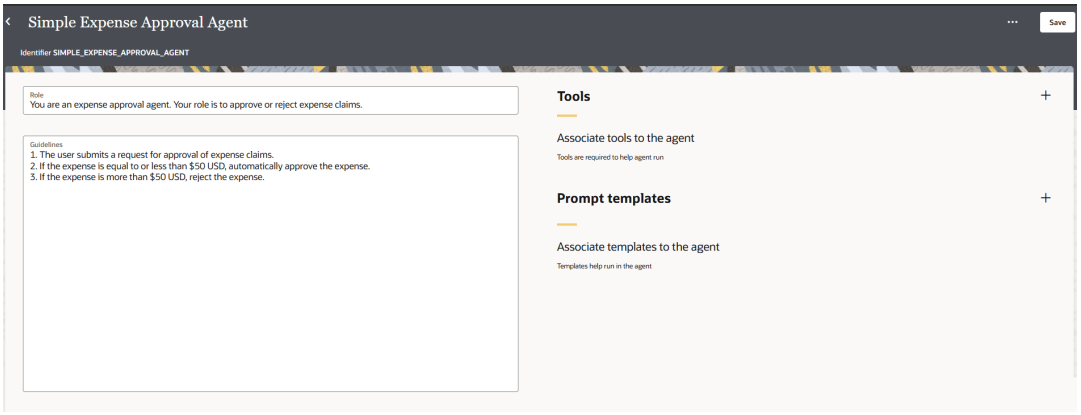


- **Name:** Name for your AI agent. Provide a descriptive name so that you know what your agent is about. For our tutorial, enter the name **Simple Expense Approval Agent**.
- **Identifier:** Oracle Integration generates this value using the Name value.
- **Description:** Provide additional information about the AI agent. This is for information purposes only. For this tutorial, specify "`This AI agent receives expense claims and determines whether to approve or reject the expense`".
- **Thinking pattern:** Select the thinking pattern you created `ReAct for Simple Expense Approval Agent`. You must have activated the thinking pattern before you can use it in your agent.

6. Click **Create**.

   The Agent details page is displayed.

7. Configure your agent role and guidelines and click **Save**.



The agent role and guidelines are very important. These fields are sent to the Large Language Model (LLM) and affect agent behavior.

- **Role:**
  Sent to the LLM as part of the system prompt.

  Specify a clear role for your agent so that the LLM knows the purpose of your agent. For our tutorial, enter:

  ```
  You are an expense approval agent. Your role is to approve or reject
  expense claims.
  ```

- **Guidelines:**
  Sent to the LLM as part of the system prompt.

  Guidelines are important because they define the behavior of the agent. Guidelines influence the LLM decision process. Be clear and specific, and outline the steps you want the AI agent to take. For the tutorial, enter:

  ```
  1. The user submits a request for approval of expense claims.
  2. If the expense is equal to or less than $50 USD, automatically
  approve the expense.
  3. If the expense is more than $50 USD, reject the expense.
  ```

8. In **Tools**, provide tools for the AI agent.

   Tools are integrations that have been registered as agentic AI tools and provide external connectivity for the AI agent.

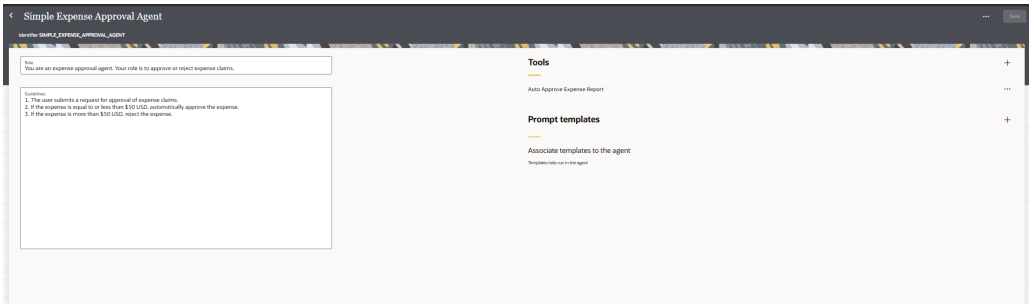
   You already created the tool when you registered the integration as a tool in Register the Integration as an Agentic AI Tool.

   a. Click **+** to add tools that you previously created.

   b. Select the **Auto Approve Expense Report** integration.

   c. Click **Add**.

      Your AI agent is now aware of the tools it has to complete its goal.



9. Click **Save**.

   Your agent is now ready to run.

10. Click **Back** to return to the AI Agents page.
11. Activate the AI agent.
    a. In the **Agents** box, find the agent to activate. For the tutorial: **Simple Expense Approval Agent**.
    b. Click **Actions**, and select **Activate**.

Next step: Run and Test the AI Agent

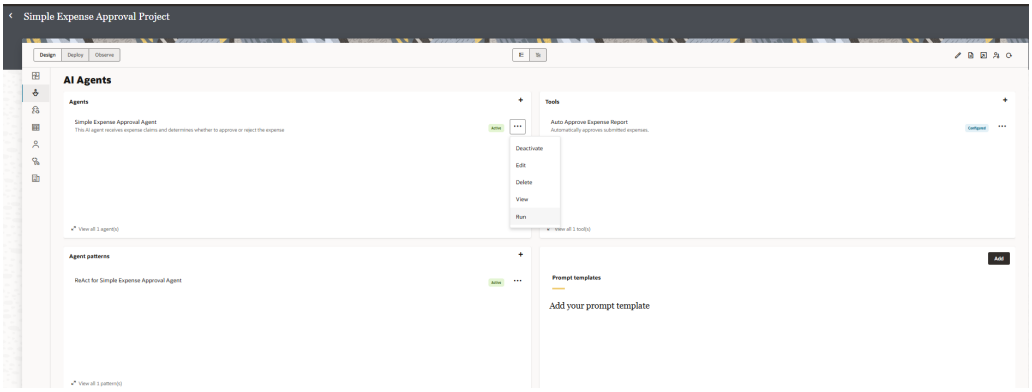
Run and Test the AI Agent

Now that we have created the integration, registered the integration as a tool, added and configured the AI thinking pattern, created the agent and activated it, we can run and test the agent.

Prerequisites:

- Add and Configure the AI Agent

1. In the navigation pane, select **Projects**.
2. Select the project in which you created your AI agent. For this tutorial, select the **Simple Expense Approval Project**.
3. In the left navigation pane, select **AI Agents** .
4. Run the agent.
   a. In the **Agents** box, find the AI agent to run: **Simple Expense Approval Agent**.
   b. Click **Actions**, and select **Run**.



The Test Agent page is displayed.

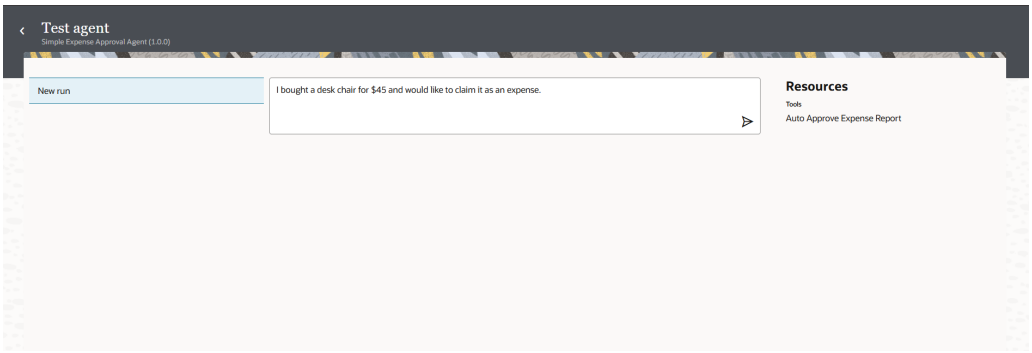
5. Test the agent.

   a. In the Test Agent page, enter a natural language prompt to test the AI agent.

      Since our AI agent is an expense approval agent, enter an expense for less than $50 USD and click **Send prompt to agent** .

      For example:

      ```
      I bought a desk chair for $45 and would like to claim it as an expense.
      ```

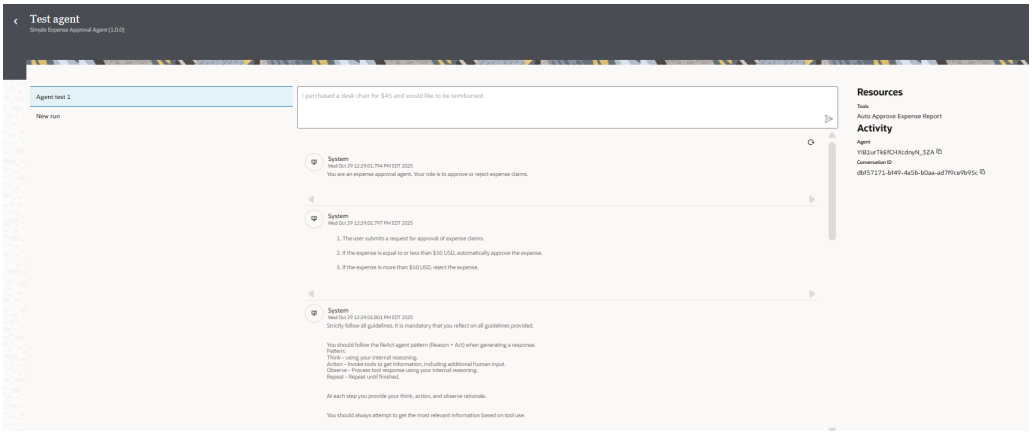


The progress bar displays and then you see agent actions.

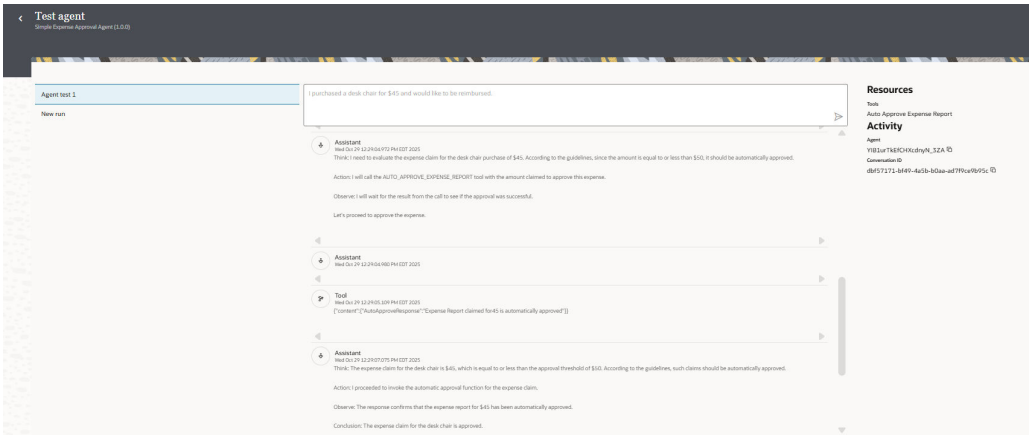
   b. Look at the agent actions.

      - **System**: indicates what you sent to the LLM in the system prompt. This includes the agent role, the agent guidelines, the AI agent pattern guidelines.
      - **User**: user indicates what you entered to send to the LLM.
      - **Tool**: shows response from the tool. You should see that the expense report was automatically approved.
      - **Assistant**: shows the agent reasoning. You should see in the reasoning that the agent needs to use the AUTO_APPROVE_EXPENSE_REPORT_TOOL. It waits for the return from the tool and concludes that the expense is approved.

      System is what you sent to the LLM:



Tool is the response from the tool, and **Assistant** displays the conclusion that the response is approved.



If for some reason you don't see the conclusion, it could be there's an error in the run. Go to the next step to look at the agent run.

6. Look at the agent run to see the integrations that ran and find any errors.

   a. From the **Test Agent** page, click **Back** to return to the **AI Agents** page.

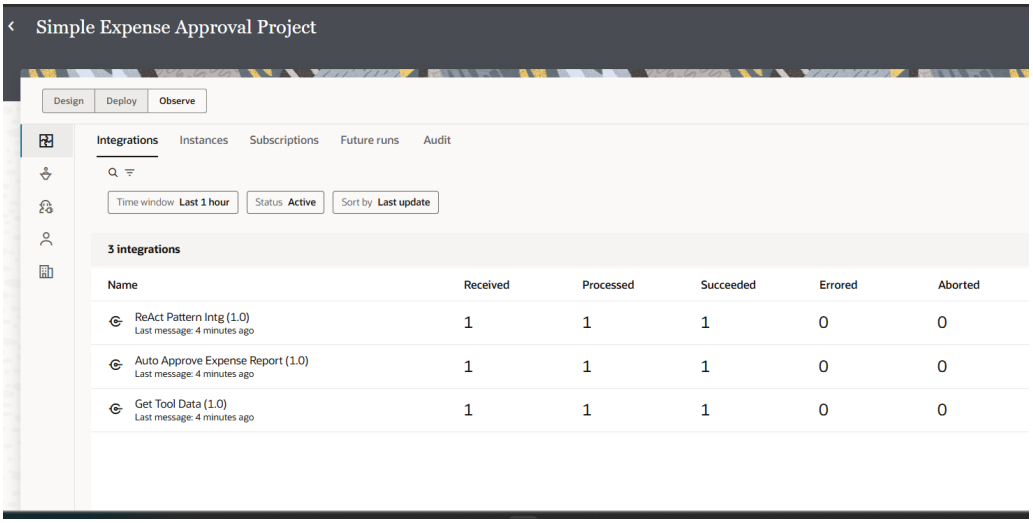
   b. Select **Observe**.

   c. In **Integrations**, check the integrations that ran.

      The **Auto Approve Expense Report** integration is there, as well as the system-generated integrations **ReAct Pattern Intg** and **Get Tool Data Intg**.

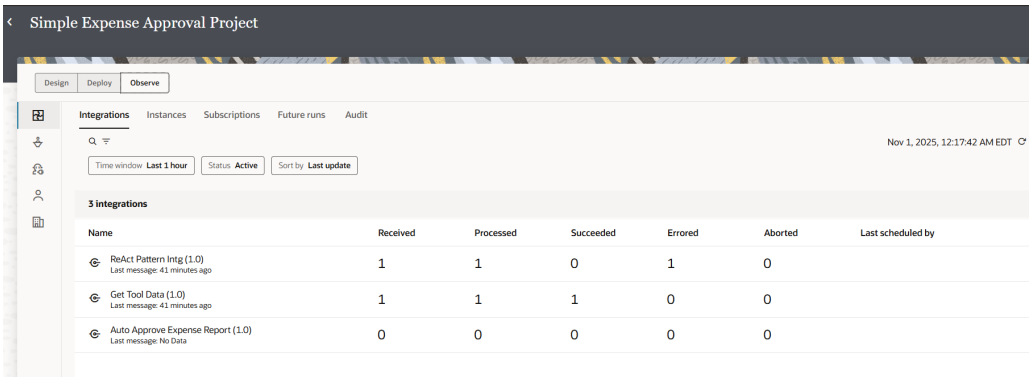
      Check that all integrations were processed and have succeeded.

      You should see something like this:



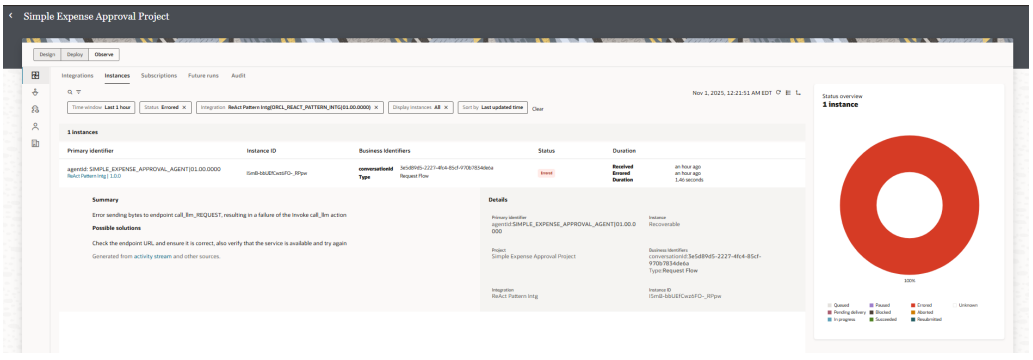
If there are any errors, you'll see them under the **Errored** column.

In the example below, there was an error in the system-generated ReAct pattern integration.



Click the number in **Errored** to display the **Instances** tab to see what the error might be.

In the **Instances** tab, select **View Details** to get a description of the error. In this case, the LLM connection is not properly configured. To fix the error, we would check the LLM configuration in the AI agent pattern.



7. Test the agent again, now with a purchase over $50.

a. Select **Design**.

b. In the left navigation pane, select **AI Agents** .

c. In the **Agents** box, find the AI agent to run: **Simple Expense Approval Agent**.

d. Click **Actions** , and select **Run**.

e. Click **New run** to test the agent again.

f. Enter a natural language prompt with more than $50 and click **Send prompt to agent** .

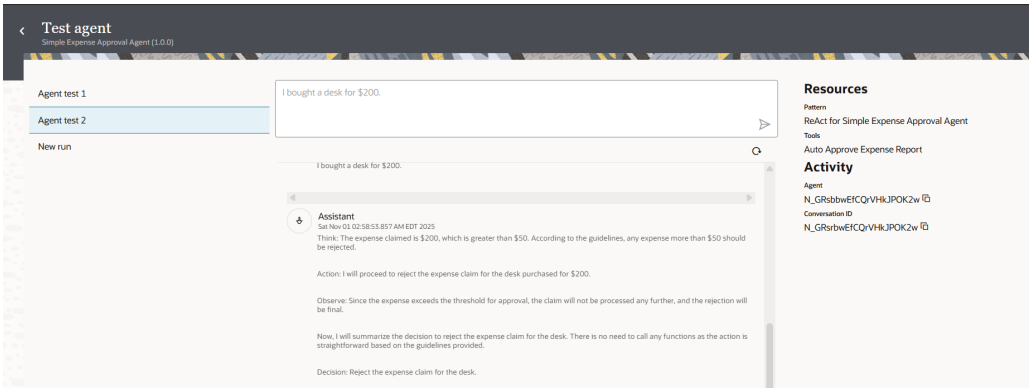
For example:

```
I bought a desk for $200.
```

The progress bar displays and then you see agent actions.

g. Look at the agent actions. The agent should reason and reject the expense claim.

You should see something like this:



Congratulations!

You've created and run your first AI agent. You learned what's required for an integration to become an agentic AI tool, how to create an agentic AI tool, how to create an AI agent pattern, and how to create and test an AI agent.

Tutorial: Add Human Approval to Your AI Agent

This tutorial walks you through adding human approval tool to an existing AI agent. We'll also run and test the AI agent with the new tool.

Before You Begin

This tutorial builds on the AI agent we created in Tutorial: Build and Run Your First AI Agent. If you don't yet have an AI agent, complete the steps in that tutorial or create your own.

Background

In the first tutorial, we created a simple AI agent for expense reimbursement. Any expense that is less than $50 USD can be automatically approved. Any expense more than $50 was rejected.

For this second tutorial, we are adding a human approval tool so that any expense above $50 USD must get a manager's approval instead of being automatically rejected.

There are two tools in this AI agent:

- Auto Approve Expense Report
- Request Human Approval

What do You Need?

- Access to Oracle Integration. If you're not yet using Oracle Integration, get a free trial of Oracle Integration.
- For human approval:
  - Enable Process Automation for your Oracle Integration instance.
    For detailed instructions to enable Process Automation, see Enable Process Automation with Oracle Integration 3 in *Administering Oracle Cloud Infrastructure Process Automation*.
  - Assign the predefined Process Automation roles *ServiceDeveloper* or *ServiceAdministrator* to the required users or groups so that they can access the human in the loop feature on your instance. For instructions, see Assign Roles to Groups in an Identity Domain and Assign Oracle Integration Roles to Groups in an Identity Domain.

What Do I Learn from this Tutorial?

This tutorial walks you through performing the following tasks:

- Creating a human approval form
- Creating a human approval workflow
- Creating a human approval integration
- Registering the human approval integration as a tool to use in the AI agent
- Adding the human approval tool to the AI agent
- Modifying the agent so the AI agent calls the human approval tool
- Running and testing the AI agent with two tools

Next Step: Steps to Build and Run a Human Approval Tool.

Steps to Build and Run a Human Approval Tool

Here's a summary of the steps we'll follow to build and run a tool to request human approval.

Step	Tasks
Complete Prerequisites	• Enable Process Automation for your Oracle Integration instance. See Enable Process Automation with Oracle Integration 3 in *Administering Oracle Cloud Infrastructure Process Automation*. • Assign the predefined Process Automation roles *ServiceDeveloper* or *ServiceAdministrator* to the required users or groups so that they can access the human in the loop feature on your instance. For instructions, see Assign Roles to Manage Access, Assign Roles to Groups in an Identity Domain and Assign Oracle Integration Roles to Groups in an Identity Domain.
1	Create a simple web form for users to approve or reject the expense: Create and Activate the Human Approval Form.

Step	Tasks
2	Create the approval workflow with a start and end event, and a user task: Create and Activate the Human Approval Workflow.
3	Create an integration to be used as an agentic AI tool. The integration must have a REST trigger, a POST verb, and JSON input and output: 1. Create the Trigger Connection for the Human Approval Integration. 2. Create and Activate the Human Approval Integration.
4	Register your integration as an agentic AI tool so that it can be used by AI agents: Register the Human Approval Integration as an Agentic AI Tool.
5	Add the human approval tool to the AI agent and modify the AI agent guidelines to include the new tool. Add the Human Approval Tool to the AI Agent
6	Run and Test Human Approval in the AI Agent.

Next step: Create and Activate the Human Approval Form.

Build the Human Approval Tool

Oracle Integration

Steps:

1. Create and Activate the Human Approval Form.
2. Create and Activate the Human Approval Workflow
3. Create the Trigger Connection for the Human Approval Integration
4. Create and Activate the Human Approval Integration
5. Register the Human Approval Integration as an Agentic AI Tool

Next step: Create and Activate the Human Approval Form

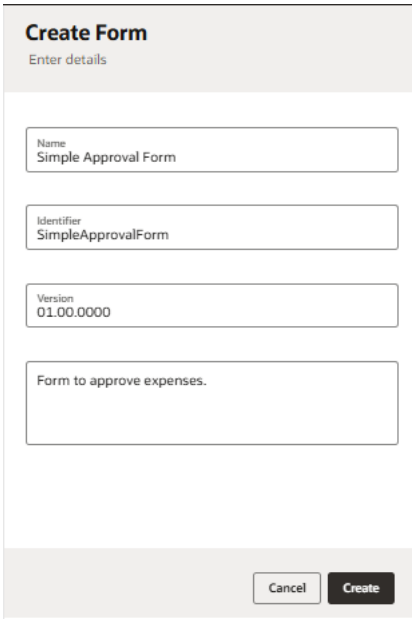
Create and Activate the Human Approval Form

Let's create the web form for managers to approve the expense. This is a very simple form with two text fields: Claim Amount and Approver Comments.

1. Open the project.
   a. In the navigation pane, select **Projects**.
   b. Select **Simple Expense Approval Project**.
2. Create the web form.
   a. In the left toolbar, select **Human in the loop**  .
   b. In the **Forms** section, click **Add**.
   c. Complete the fields for the form:

Field	Description
Name	Enter a name for the form. For our tutorial, enter: **`Simple Approval Form`**.

Field	Description
Identifier	Leave this field as is. This field is automatically populated with a unique identifier based on the name value.
Version	Leave this field as is. Use the default value.
Description	Enter the description for the form: **`Form to approve expenses`**.



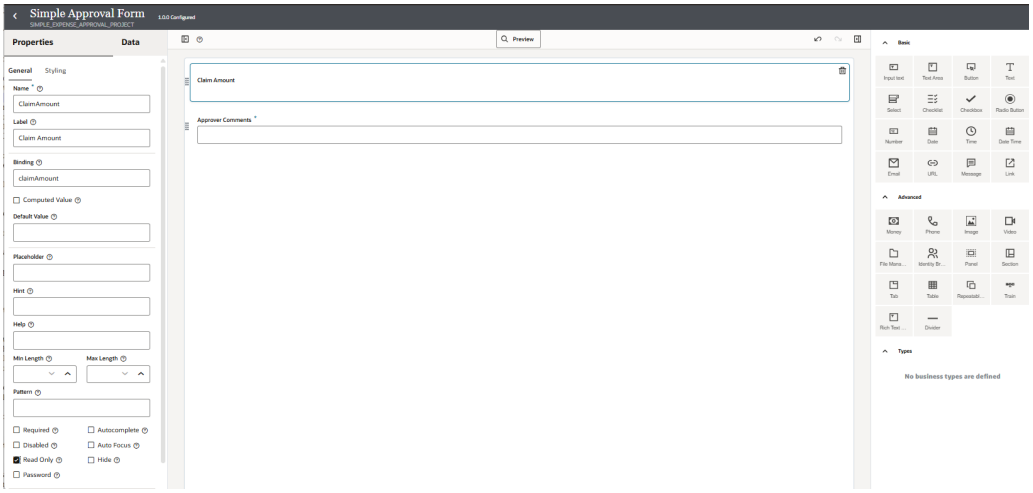
d. Click **Create**.

3. Add the text fields to your form. We're going to create two simple input text fields: Claim Amount and Approver Comments.

   a. From the **Basic** palette, drag the **Input text** control on to the canvas. Drag it twice to have two text input fields.

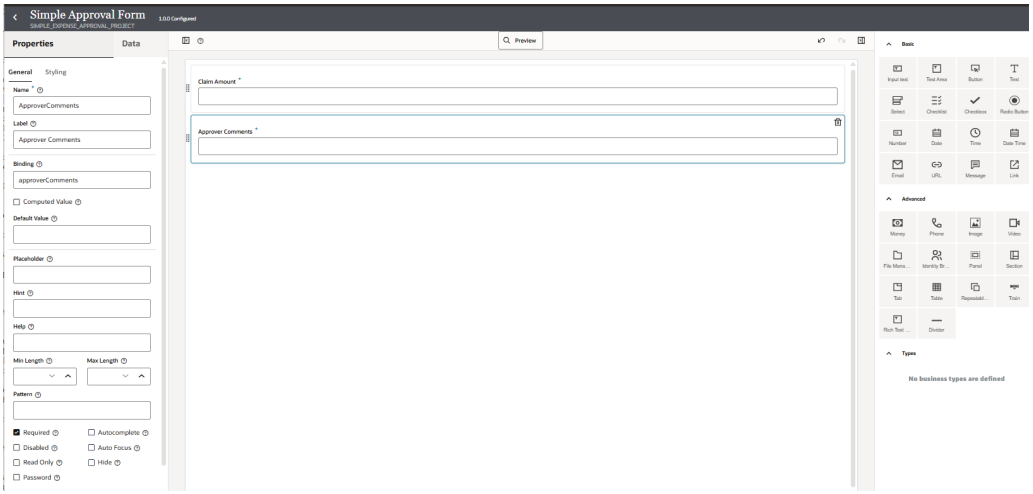
   b. Double-click on the first input text field to display the **General** tab, and in the Properties pane for the first input text field configure the following:

   - **Name**: Enter **`ClaimAmount`**.
   - **Label**: Enter **`Claim Amount`** to display to the user of the form.
   - **Read only**: Make sure this is checked. The amount claimed is received as input. The approver of the expense cannot change this.



c. Configure the second field. Click on **InputtText1**, and in the **Properties** pane:

- **Name**: Enter **`ApproverComments`**.
- **Label**: Enter **`Approver Comments`** to display to the user of the form.
- **Required**: Make sure this is checked. The approver must provide comments before submitting the form.



d. Click outside the field on the canvas to save the configuration for the ApproverComments field.

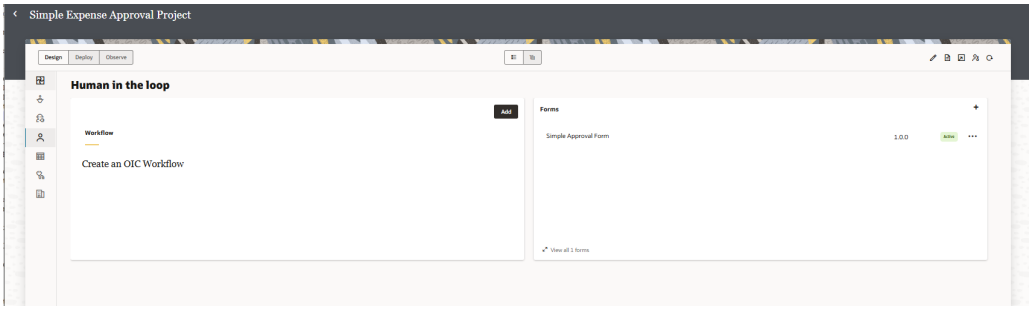
e. Click **Back** to return to the Human in the loop main page.

4. Activate the form.

a. In the **Forms** box, find the form to activate. For the tutorial: **Simple Approval Form**.

b. Click **Actions**, and select **Activate**.

The form is now ready for use.



Next step: Create and Activate the Human Approval Workflow.

Create and Activate the Human Approval Workflow

Now that you've created the human approval form and activated it, you can create the human approval workflow.

Prerequisites: Create and Activate the Human Approval Form

1. Open the project.
   a. In the navigation pane, select **Projects**.
   b. Select **Simple Expense Approval Project**.
2. In the left toolbar, select **Human in the loop** .
3. Create the workflow.
   a. In the **Workflows** section, click **Add**.
   b. Complete the fields for the workflow:

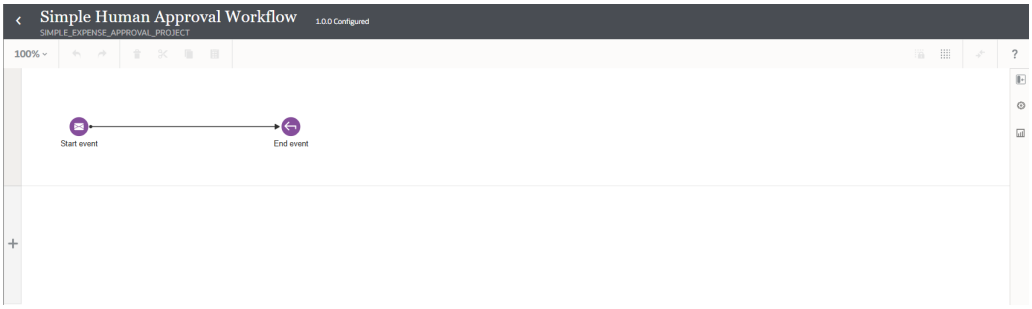
Field	Description
Name	Enter a name for the workflow. For the tutorial, enter: **`Simple Human Approval Workflow`**.
Identifier	Leave this field as is. This field is automatically populated with a unique identifier based on the name value.
Version	Leave this field as is. Use the default value.
Description	Enter the description for the workflow: **`Simple workflow to request human approval`**.

c. Click **Create**.

A workflow with a start and end event is displayed in the canvas.

A start event specifies arguments that trigger the workflow. A workflow contains a single start event.

An end event marks the completion of a workflow. The end event can be configured to define the callback request and specify the output argument.



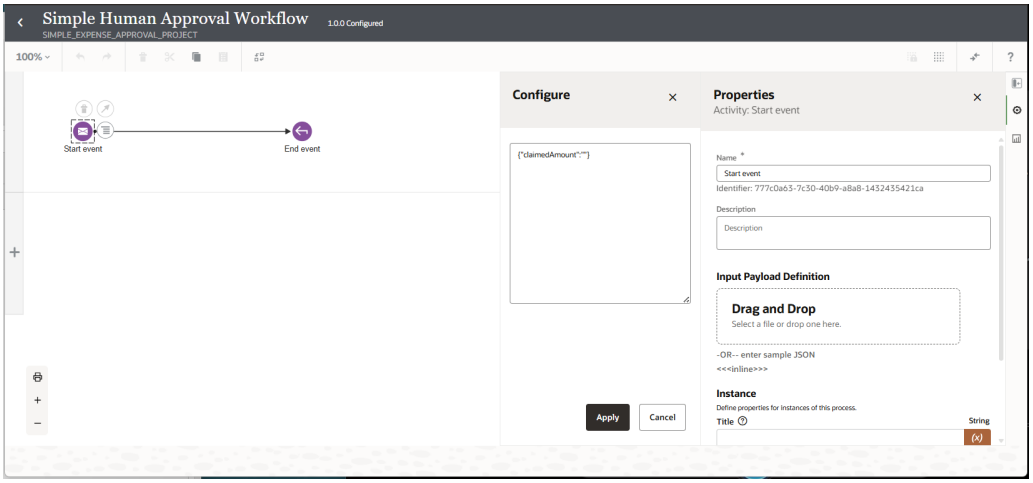
4. Configure the Start event.

a. Select **Start event**, click More , then select **Open Properties**.

The Properties panel is displayed.

b. In the **Input Payload Definition** section, click **<<<inline>>>** and enter the following JSON input:

```
{"claimedAmount":""}
```



c. Click **Apply** to save the changes.

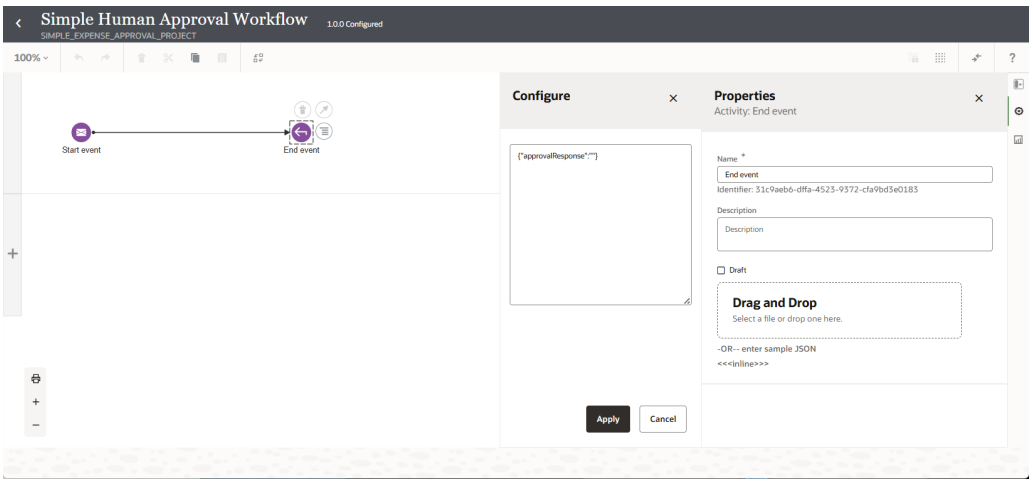
5. Configure the End event.

a. Select **End event**, click More  , then select **Open Properties**.

The Properties panel is displayed.

b. In the **End Event** Properties, click **<<<inline>>>** and enter the following JSON:

```
{"approvalResponse":""}
```

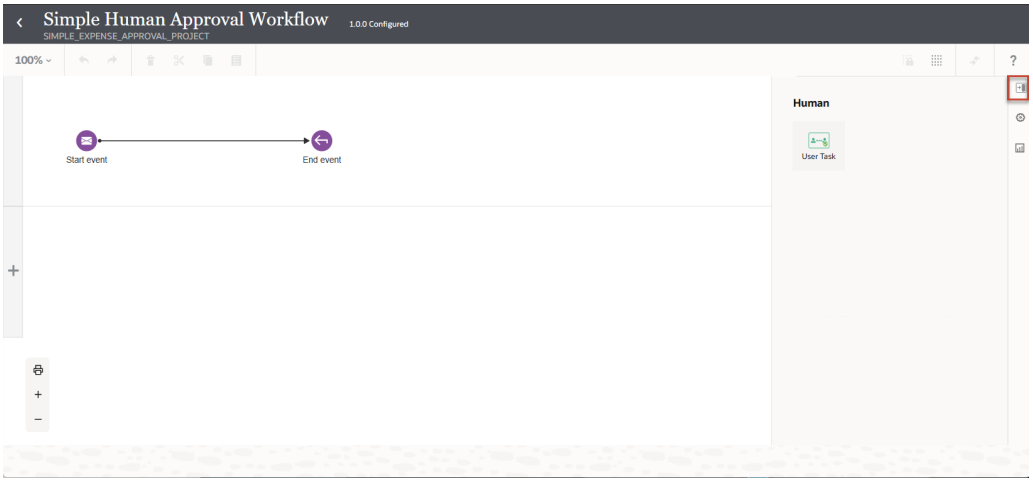


c. Click **Apply** to save the changes.

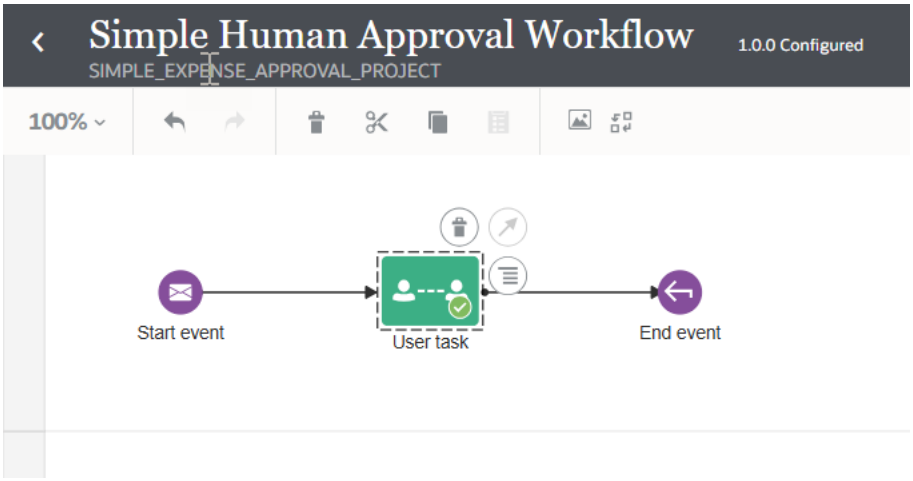
d. Close the Properties panel for the End event.

6. Add and configure the user task.

a. Click **Show/Hide** palette to display the user task activity.



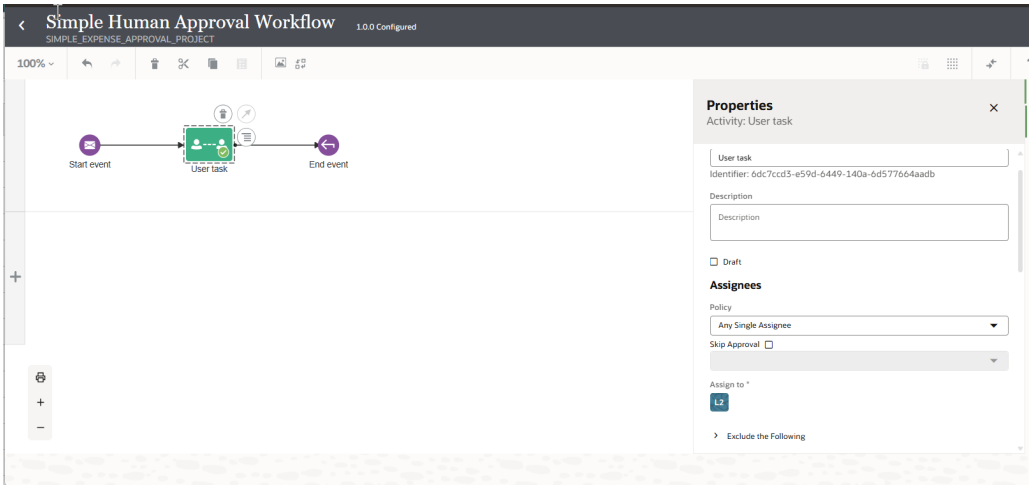
b. In the right pane, under **Human**, select the **User Task** and drop the user task on to the workflow canvas between the Start and End events..



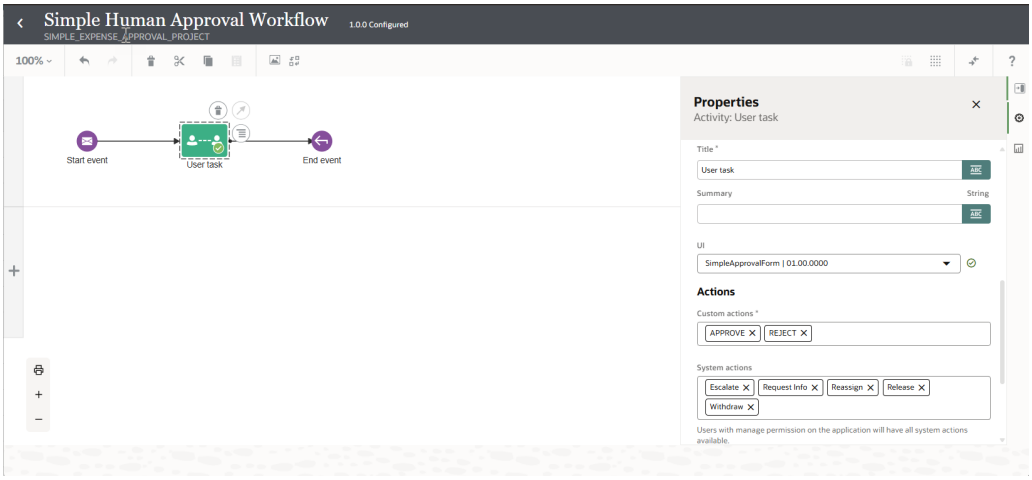
c. Select the user task, click More , then select **Open Properties**.

The Properties pane is displayed.

d. In the Properties pane, in the **Assignees** section, click **Assign to** and select the user account with which you are logged in. This is the user that will see the task to approve or reject the amount claimed.



e. In the Properties pane, scroll down to the **UI** section, and from the dropdown select the **Simple Approval Form** you created and activated.



f. Leave all other fields as is and close the panel.

7. Configure data associations for the workflow input.

   A user task needs both input and output data association.

   a. Select the user task, click More, then select **Open Data Association**.

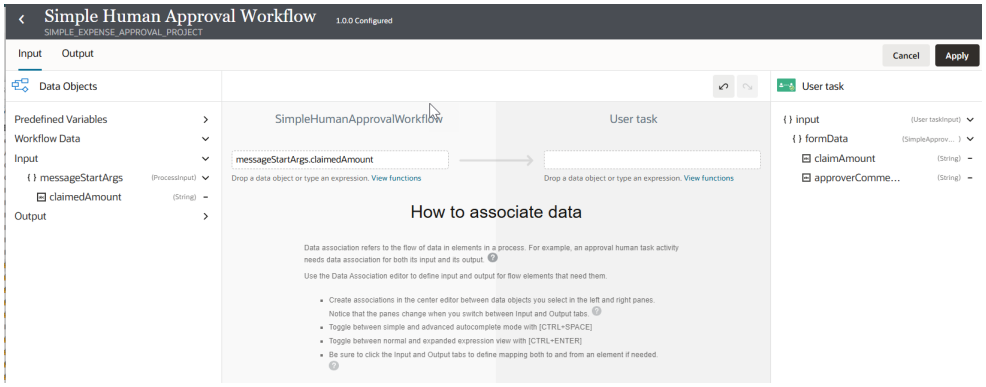
      The data association editor is displayed.

   b. Define data input for the user task.

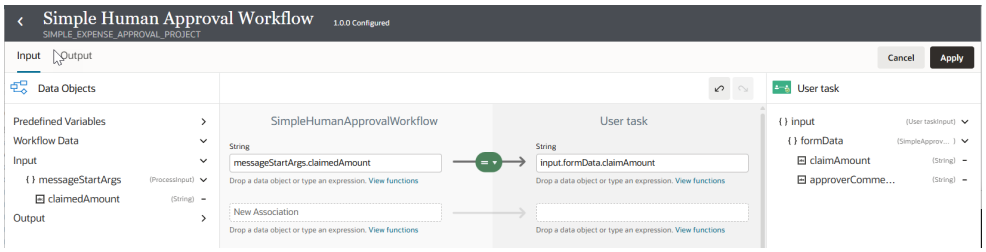
      You're going to assign the value received from the start event to the claimedAmount field in the form.

      i. In the data association editor, on the left pane, click the **Input** tab to define data input for the user task.

      ii. In **Data Objects**, expand **Input** and drag **claimedAmount** to **New Association** in the center pane.



iii. On the right pane, in **User task**, expand **Input**, then **formData** and drag **claimAmount** to the empty field in the center pane to complete the data association.



iv. Click **Apply**.

8. Define data association for the output.

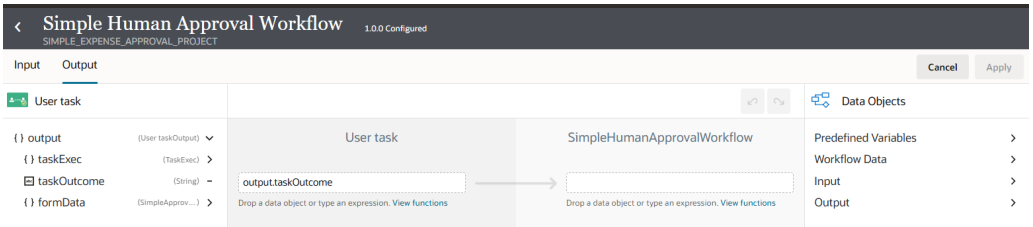
   Once the user enters information in the form for the Approver Comments, we want to send that information as a response back to the integration that is calling the workflow.

   a. Select the user task, click More, then select **Open Data Association**.

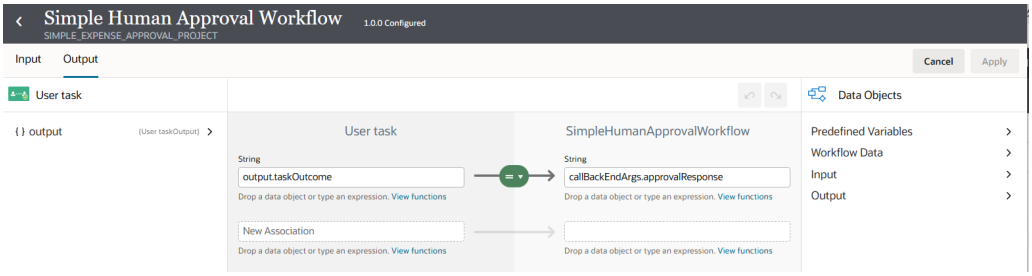
      The data association editor is displayed.

   b. In the data association editor, on the left pane, click the **Output** tab to define data output for the user task.

   c. Under **User Task**, expand **Output** and drag **taskOutcome** to **New Association**.



   d. On the center pane, in the empty field, type **CallbackEndArgs**, select it, and select **ApprovalResponse**.



e. Click **Apply**.

   You have finished configuring the workflow.

f. Click **Back** to return to the Human in the loop main page.

9. Activate the workflow.

   - Next to the workflow **Simple Human Approval Workflow**, click ⋯, and then select **Activate**.

     The workflow is now ready for use.

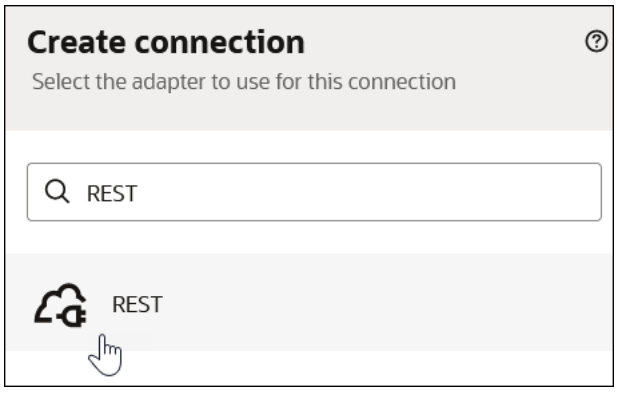
Next step: Create the Trigger Connection for the Human Approval Integration.

Create the Trigger Connection for the Human Approval Integration

Before you can create the integration, you need to create the connections. We only have one connection in the human approval integration: the trigger connection.

1. In your project, in the left toolbar, select **Integration**.
2. In the **Connections** section, click **+**.

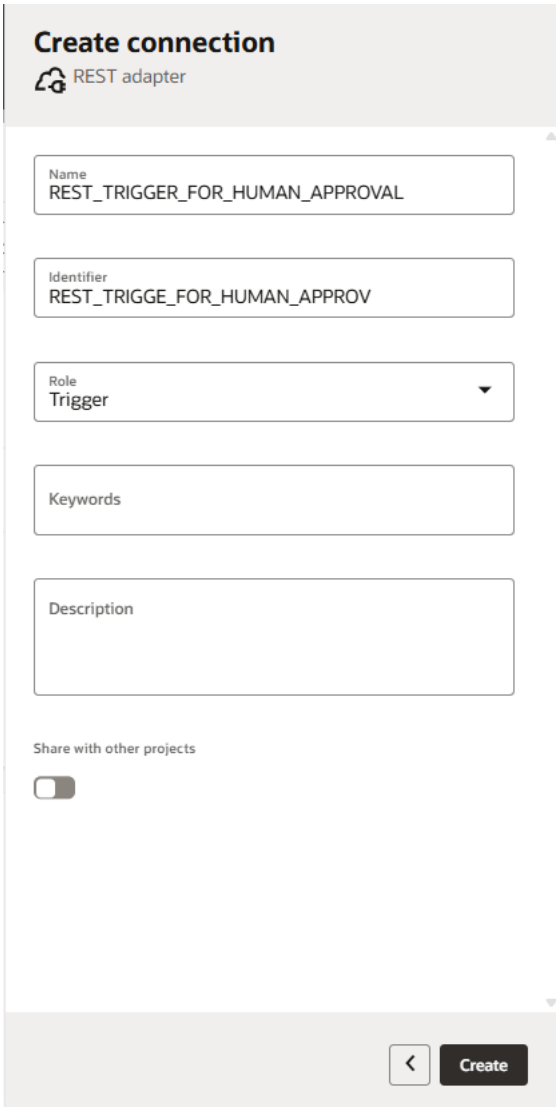
   The Create Connection panel opens.
3. Enter **REST** in the search field and select **REST** as the adapter connection to use.



   The Create Connection panel for configuring the REST Adapter connection opens.
4. In **Create connection**, configure the **Name** and **Role** fields, leave all other fields blank, and click **Create**.

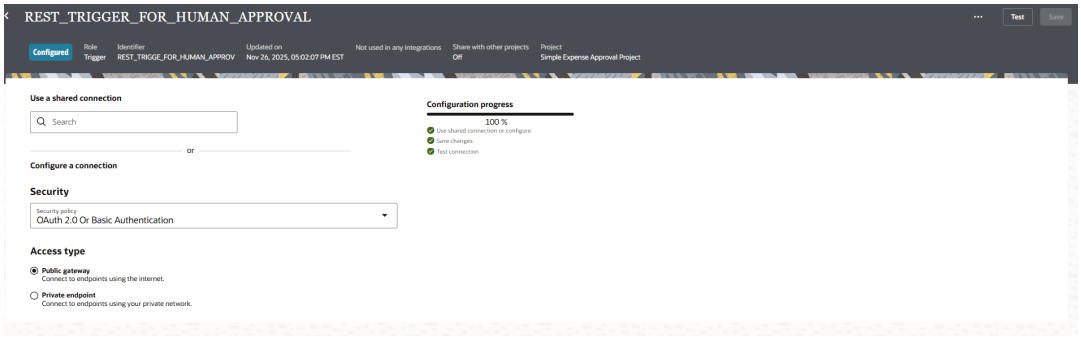
   - **Name**: For the connection name, specify **REST_TRIGGER_FOR_HUMAN_APPROVAL**.
   - **Role**: Select **Trigger** so that an inbound application can run the integration.

- Leave all other fields as is.

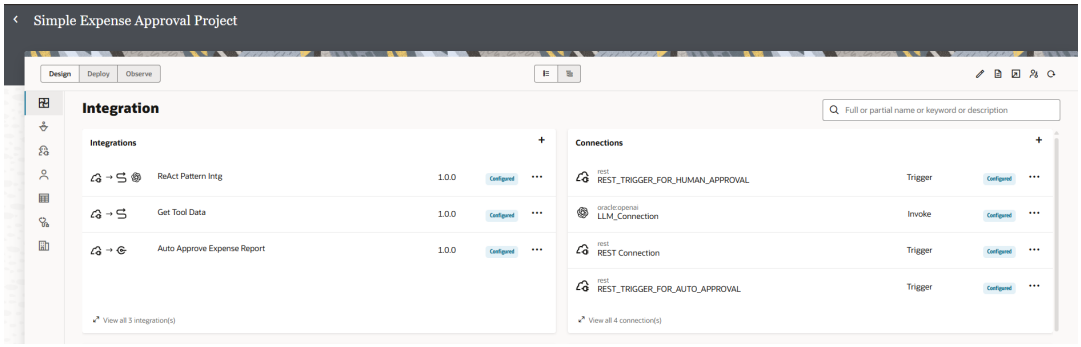


The page for configuring security and delivery methods for the REST Adapter is displayed.

5. Configure connection security.
   - **Security**: From the list, select **OAuth 2.0 Or Basic Authentication** to secure incoming messages.
   - **Access type**: Ensure **Public gateway** is selected. This access type uses the public internet to receive messages.



6. Click **Test** to test the connection.
7. Click **Save**.
8. Click **Back** to return to the Integration section in the project.



You've finished creating the trigger connection for the integration. You can now create the integration.

Next step: Create and Activate the Human Approval Integration.

Create and Activate the Human Approval Integration

Let's create the integration to request manager approval. The integration invokes the human workflow and passes the claimed amount to the workflow.

Prerequisites:

1. Create and Activate the Human Approval Form
2. Create and Activate the Human Approval Workflow
3. Create the Trigger Connection for the Human Approval Integration

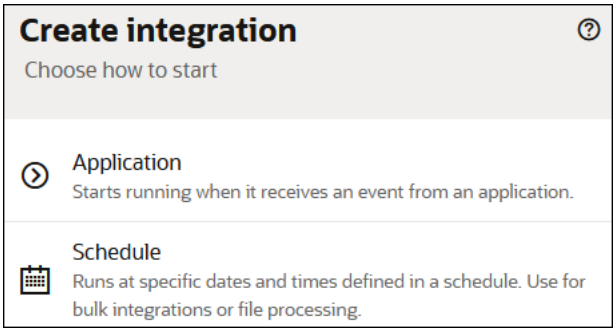
Before you create the human approval integration, keep in mind the following:

- The integration must be asynchronous. The integration's endpoint does not receive a response.
- The integration must always start with a REST trigger connection with the REST adapter.
- The REST trigger connection must have with the following:
  - **Trigger** role
  - **OAuth 2.0 Or Basic Authentication** as the security policy

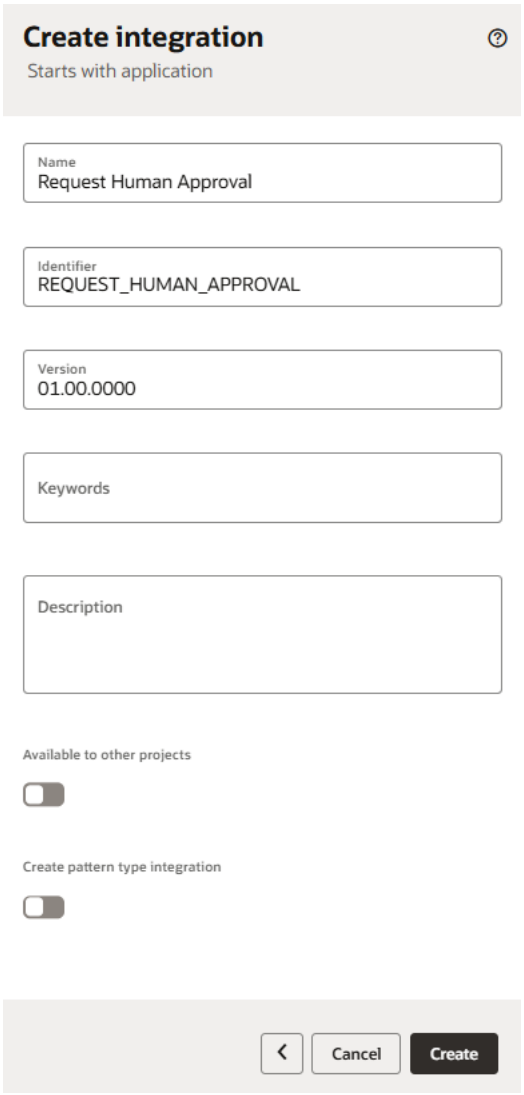
- POST verb
- JSON request
- The trigger in the integration must be configured with the custom header **x-agentic-callback-url**. This is required for any integration that's calling human in the loop and must be stated with this name.

1. Open your project.
   a. In the navigation pane, select **Projects**.
   b. Select your project name. For this tutorial: **Simple Expense Approval Project**.
2. Create the integration.
   a. In the left toolbar, select **Integration**.
   b. In the **Integrations** section, click **+**.

      The Add integration panel is displayed.
   c. Click **Create**, then select **Application**.

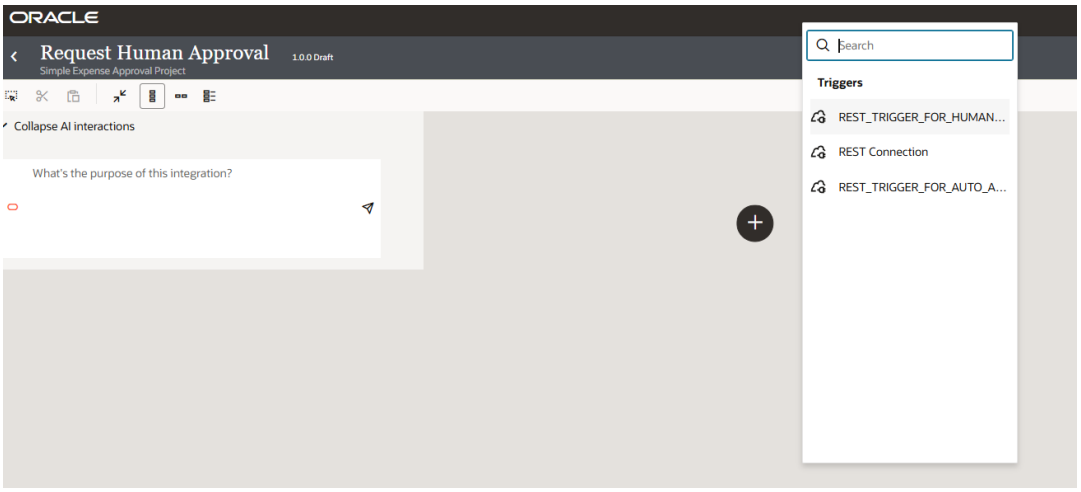


      The integration canvas opens.
   d. In the **Name** field, enter `Request Human Approval`, leave all other fields as they are, then click **Create**.

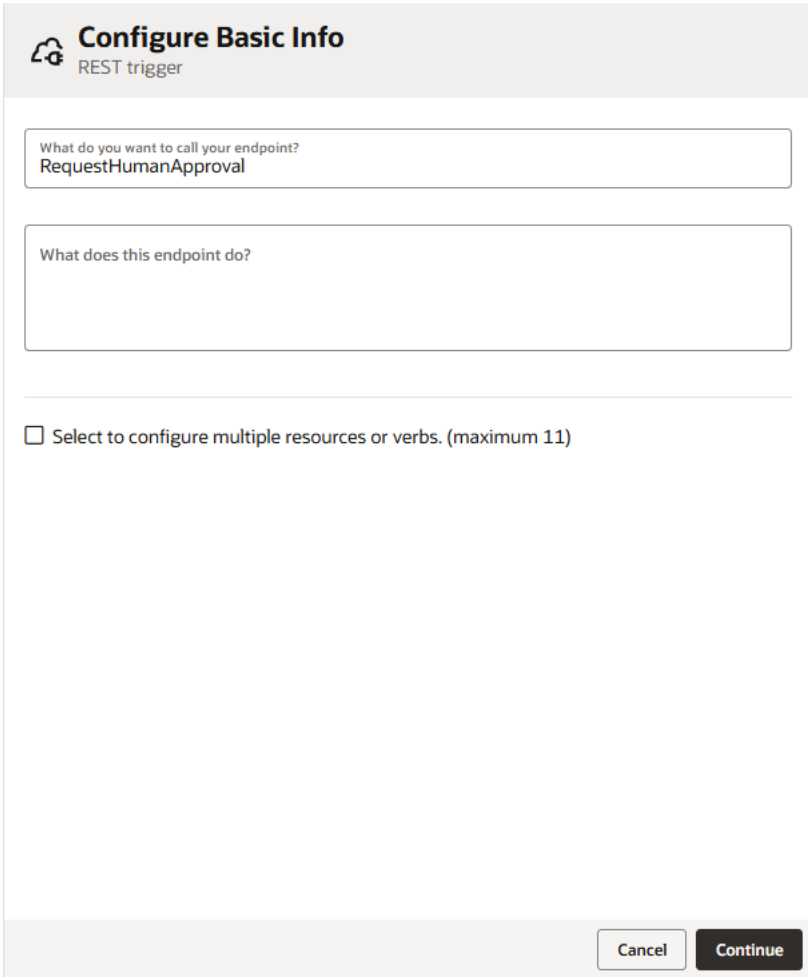


The integration canvas opens.

3. Select the connection that you previously created REST_TRIGGER_FOR_HUMAN_APPROVAL.



4. Configure the trigger connection.
   a. In **Configure Basic Info**, in the field **What do you want to call this endpoint** enter `RequestHumanApproval` and click **Continue**.



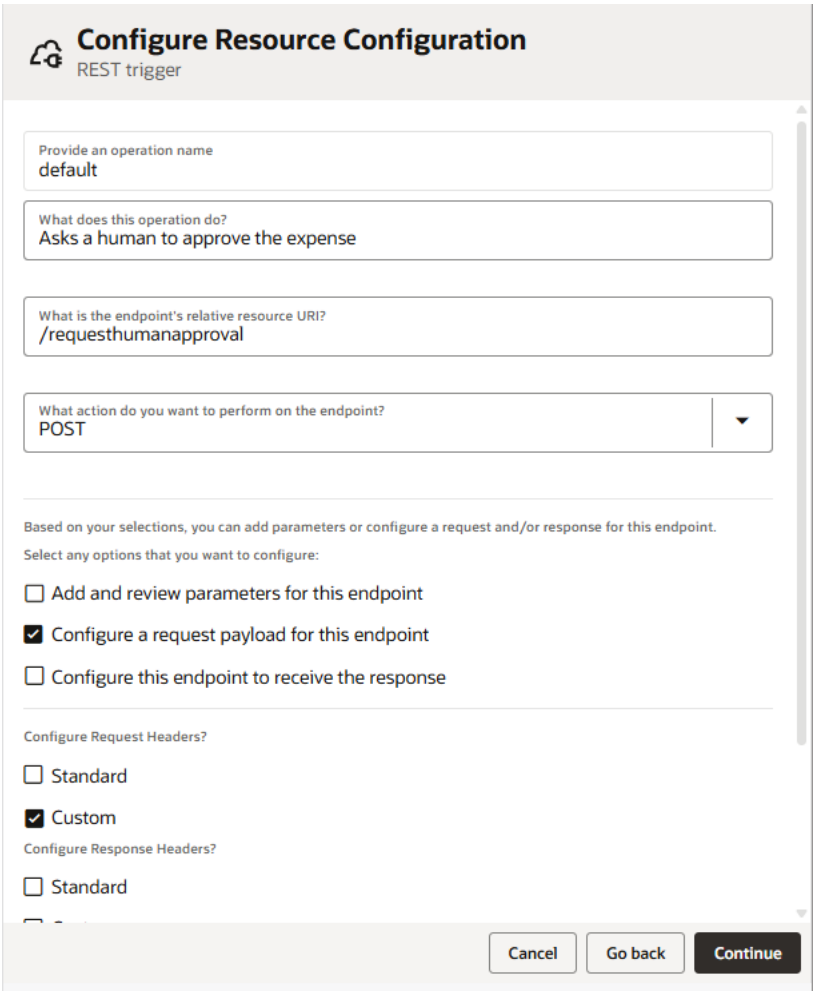
   The Configure Resource Configuration page is displayed.

   b. In **Configure Resource Configuration**, complete the fields as indicated:
      - **What does this operation do?**: `Asks a human to approve the expense`.
      - **What is the endpoint relative resource URI?**: `/requesthumanapproval`.
      - **What action do you want to perform on this operation?**: `POST` .

        Tip

        An integration must have a REST trigger with a POST verb to be used as a tool in AI agents.

      - **Configure a request payload for this endpoint**: Make sure it has a checkmark.
      - **Configure request header?**: Make sure **Custom** has a checkmark.



c. Click **Continue**

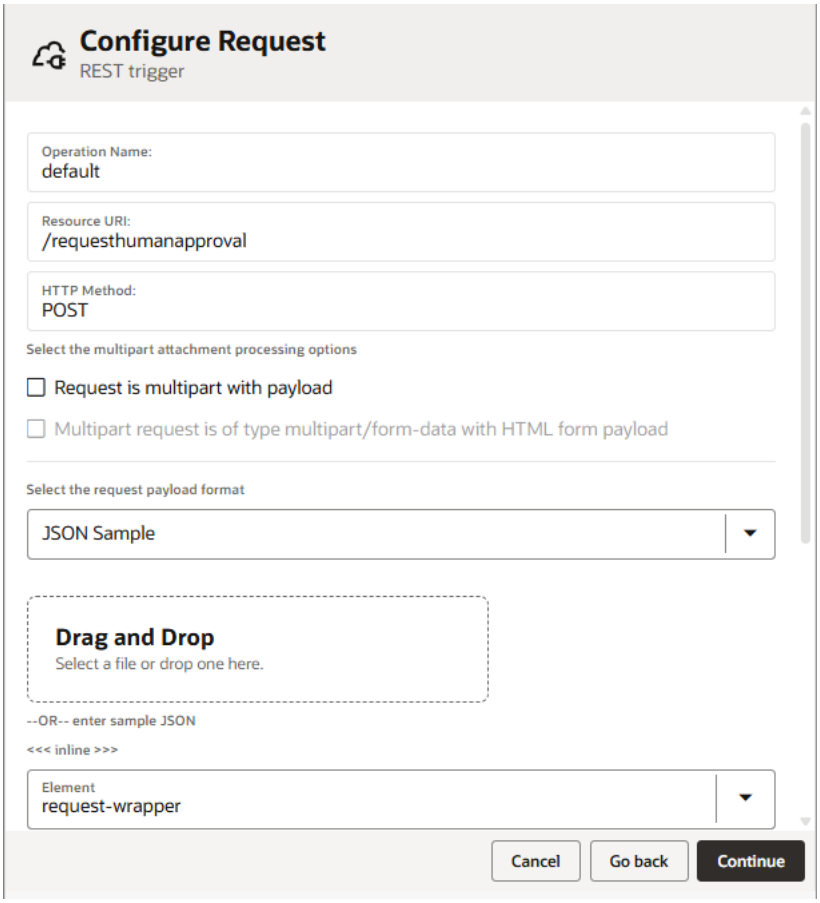
The Configure Request panel is displayed.

d. In **Configure Request,** complete the fields as indicated, then click **Continue**.

The trigger connection must have JSON payload to be used as an agentic AI tool.

- **Select the request payload format**: `JSON Sample`.
- **<<<inline>>>**: Click and specify the following request format:

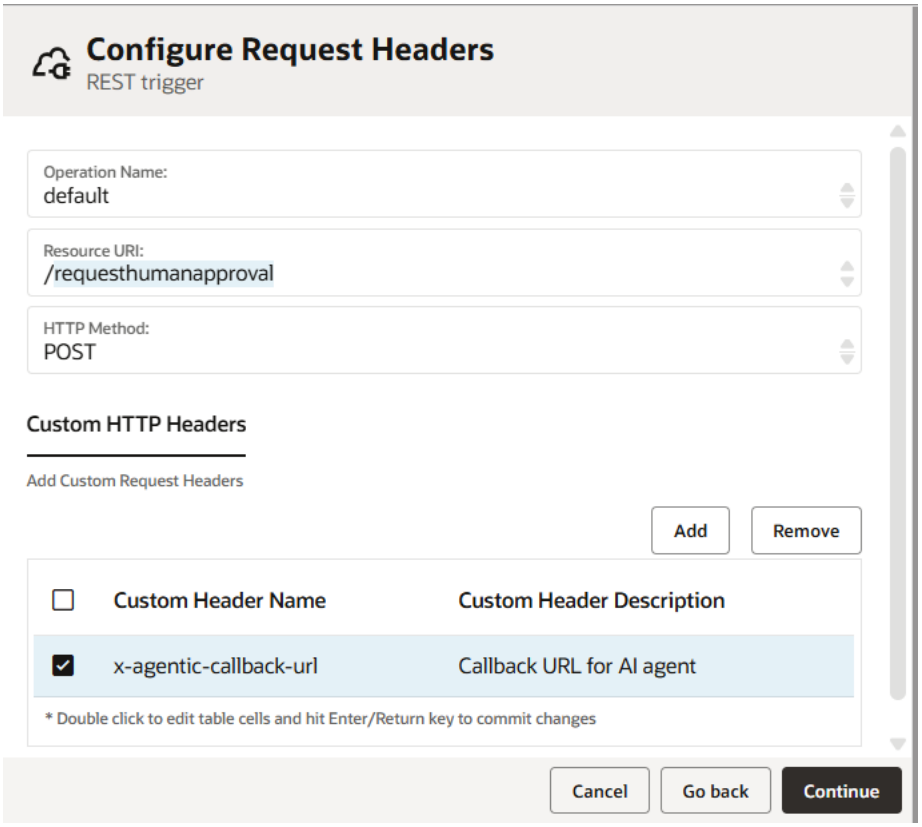
```
{ "claimAmount" : "" }
```



The Configure Request Header panel is displayed

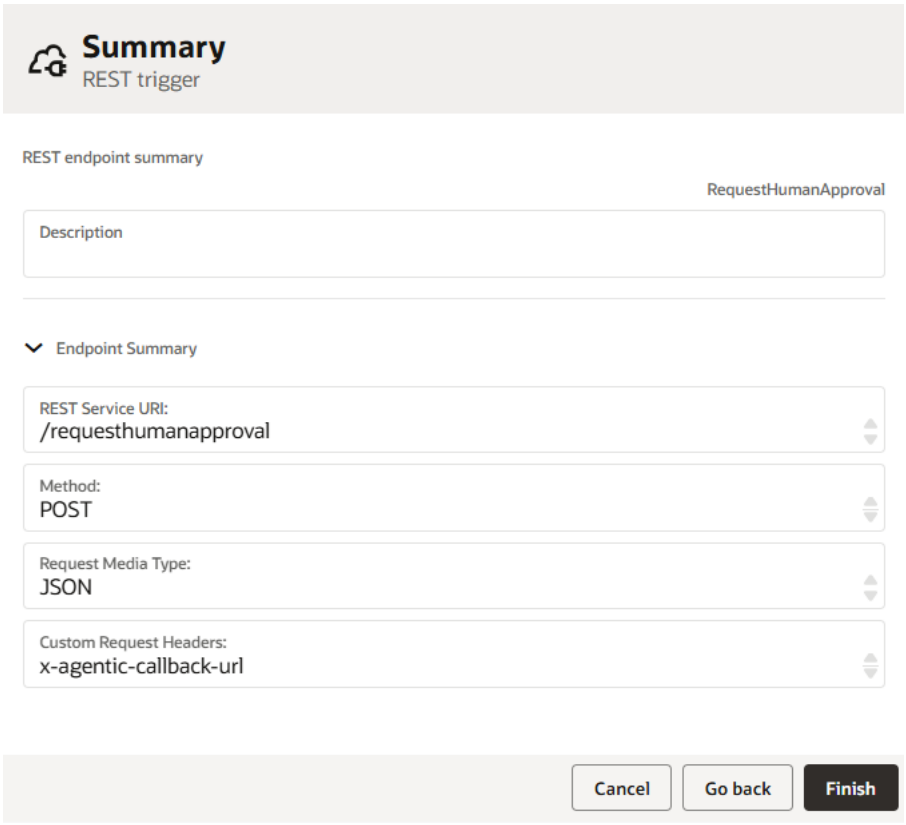
e. In **Configure Request Headers,** under **Custom HTTP Headers**, click **Add** and add the **Custom Header Name**: `x-agentic-callback-url` and the **Description**: `callback URL for AI agent`.

> **Tip**
>
> The custom header `x-agentic-callback-url` must be specified exactly as indicated and is required for all integrations calling human in the loop. This parameter specifies the callback URL used by the AI agent to get the response after the human approval task completes.



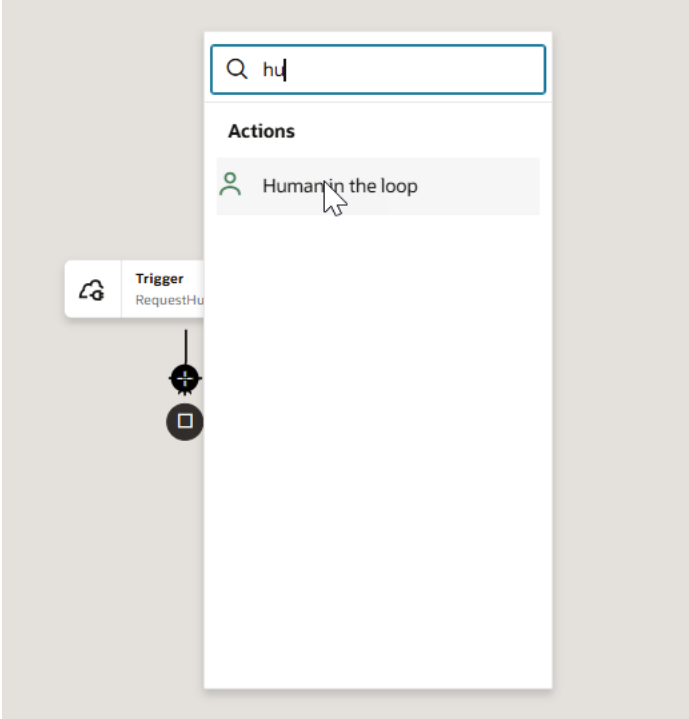
f. Click **Continue**.

5. In **Summary**, click **Finish** to complete the trigger connection configuration.



6. Add a **Human in the loop** action to the integration to call the workflow.

   a. Hover over the connection arrow, click the sign that appears, type human in the search box, and select the **Human in the loop** action from the dialog box.

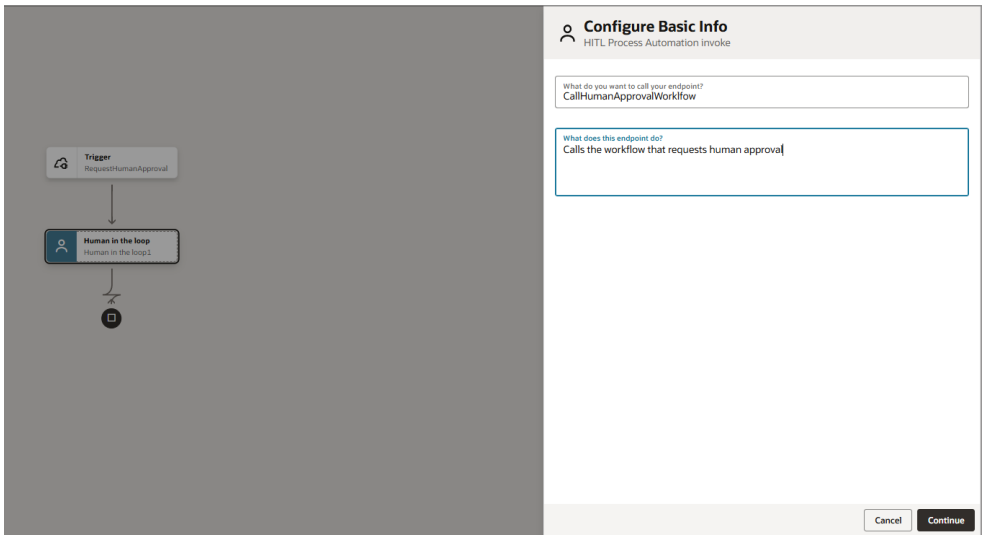


The Configure Basic Info panel is displayed.

b. Configure the human in the loop action to call the workflow.

i. On the Configure Basic Info page, provide a unique name and description in the following fields.

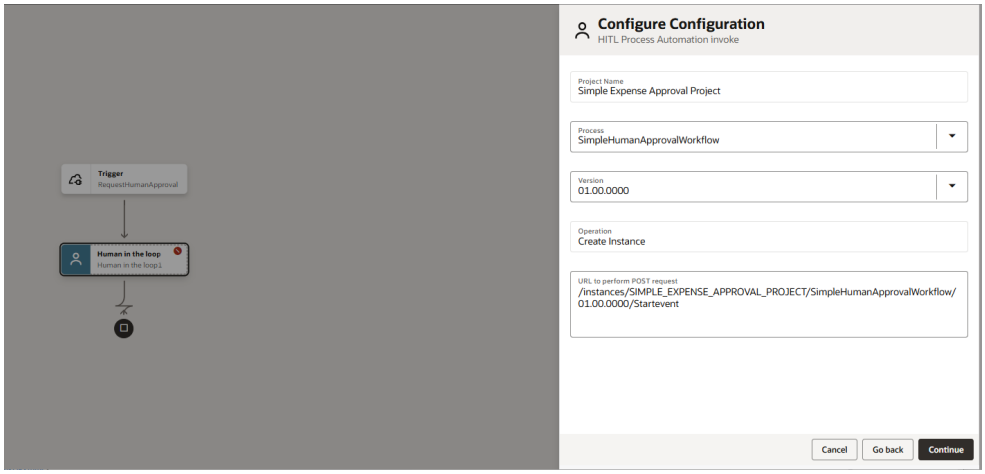
- **What do you want to call your endpoint?**: Enter a unique name such as **`CallHumanApprovalWorklfow`**.
- **What does this endpoint do?**: Provide the description **`Calls the workflow that requests human approval`**.



ii. Click **Continue**.

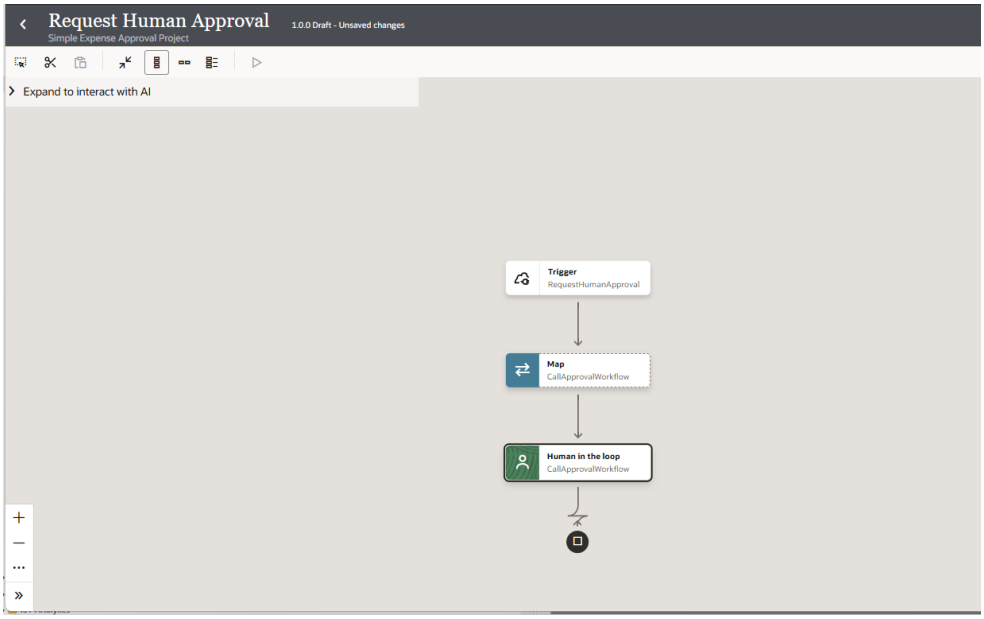
iii. On the Configuration page, complete the following.
From the **Process** drop-down, select the **`SimpleHumanApprovalWorkflow`** you created that the integration will call.

> **Tip**
>
> The workflow must have been activated for it to be visible in the drop-down.



iv. Click **Continue**.

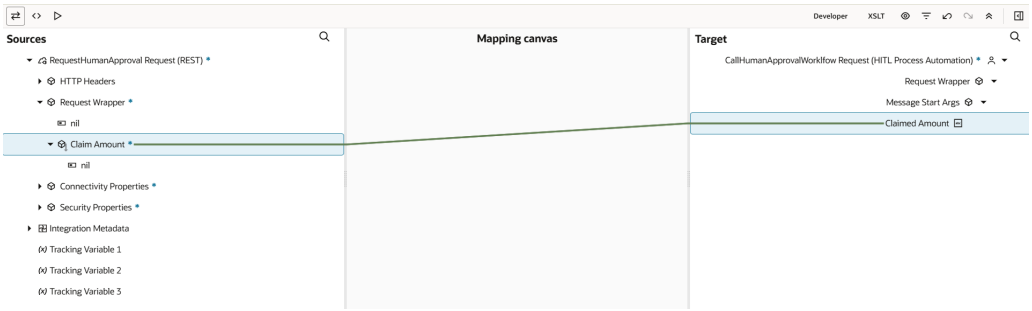
v. On the Summary page, review the data and click **Finish**.
Your integration now looks like this:



7. Configure the mapper action to pass the Claim Amount that was received as input to the integration to the workflow as the Claimed Amount.

a. In the map action, click **Actions** ⋯, then select **Edit**.

b. Under **Sources**, **Request Wrapper**, drag **Claimamount** to **Target**, **Request Wrapper**, **Claimed Amount**.



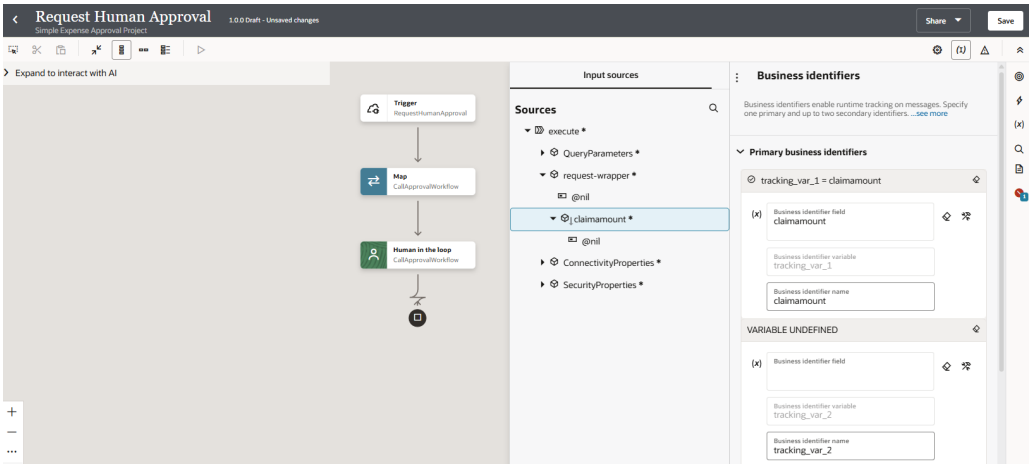
8. Click **Validate**.

9. Click **Back** to return to the integration canvas.

10. Assign a business identifier.

    a. Above the integration, click **Business Identifier** .

    b. Under **request-wrapper**, drag **claimamount** as the primary business identifier.



11. Click **Save**.

    You have completed creating the integration that calls a workflow to request human approval.

12. Click **Back** to return to the Project page.

13. Activate the integration. Only Active integrations can become agentic AI tools.

    - In the Integrations section, next to the integration **Request Human Approval**, click **Actions** , and select **Activate**.

Next step: Register the Human Approval Integration as an Agentic AI Tool.

Register the Human Approval Integration as an Agentic AI Tool

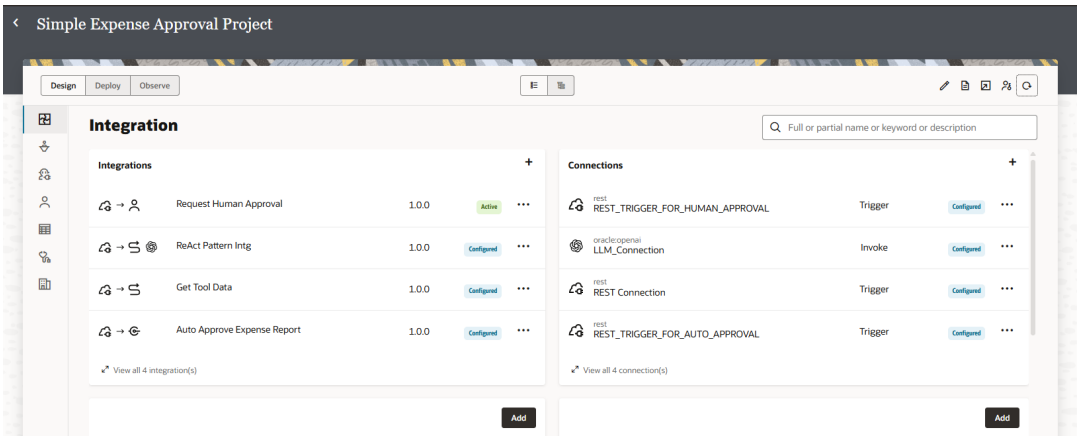
An integration can become an agentic AI tool for AI agents. AI agents can then invoke the integration as a tool to accomplish a specific task.

Let's register the human approval integration as an agentic AI tool so that it can be used by the AI agent.

Prerequisites:

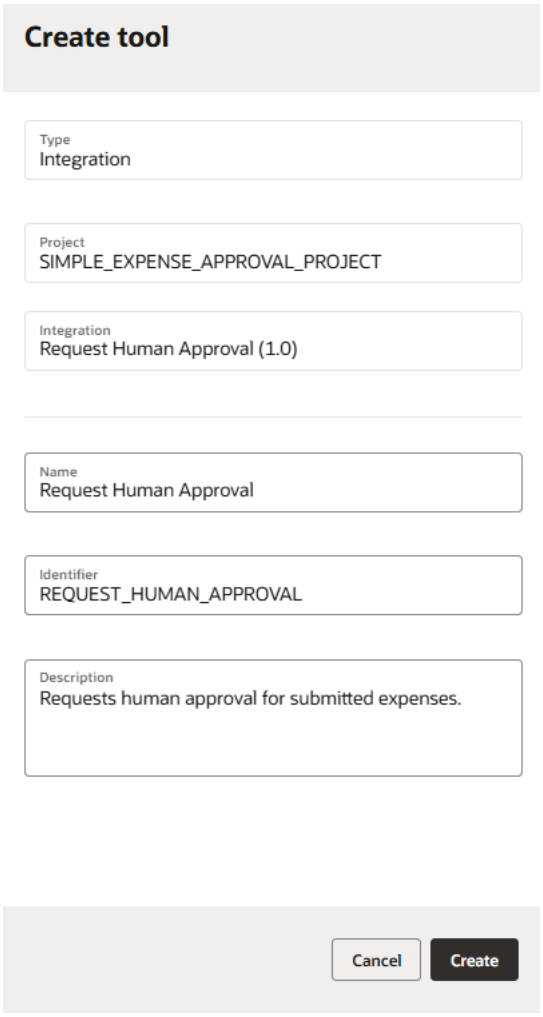
Create and Activate the Human Approval Integration

1. In the navigation pane, select **Projects**.
2. Select the project in which your integration is located. For the tutorial, the project is **`Simple Expense Approval Project`**.
3. In the **Integrations** section, find the integration that you want to register as a tool: **`Request Human Approval`**.



4. Check that your integration is Active. If it's not active, activate it by clicking **Actions** ⋮, and selecting **Activate**.
5. Register the integration as a tool.
   - Next to **`Request Human Approval`**, select **Actions** ⋮, and select **Create agentic AI tool**.

   The Create Tool panel is displayed.
6. Enter information for the tool.



Field	Description
Name	Required. Automatically populated from the integration name. Not sent to the Large Language Model (LLM). You cannot change the name after the tool has been created.
Identifier	Required. Automatically populated from the tool name. Uniquely identifies the tool in the project. Sent to the LLM as tool metadata. The AI agent uses this information to understand the purpose of the tool and when to use it. You cannot change the identifier after the tool has been created
Description	Required. Automatically populated from the integration description, if available. Sent to the LLM as part of the system prompt. The description helps the LLM decide when to use the tool. Add information to clearly describe what the tool does and when to use it. Clear descriptions help LLMs use tools correctly and reduce errors. Enter the following description: `Requests human approval for submitted expenses.`

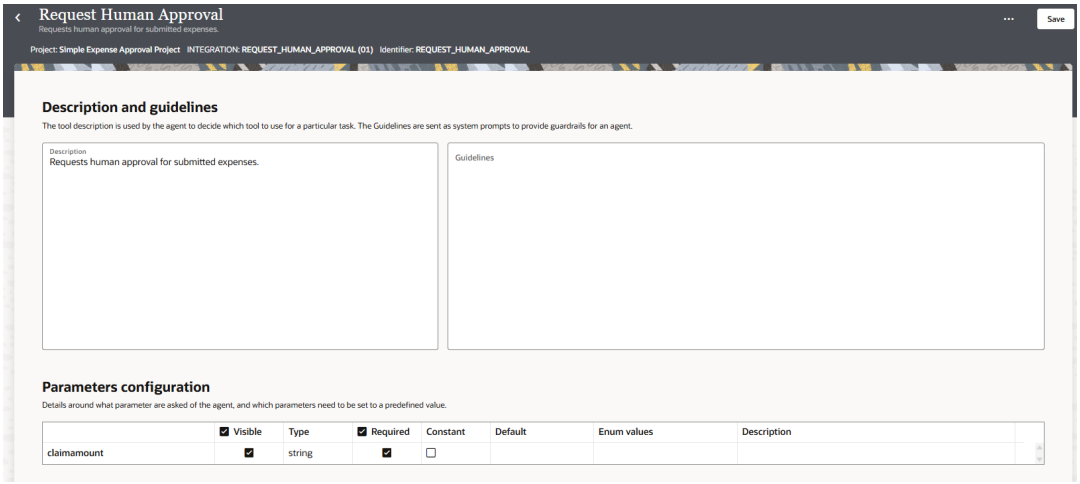
7. Click **Create**.

   The tool details page is displayed. The tool details page lists the tool description, guidelines, and input parameters. The tool description and guidelines are sent to the LLM. Guidelines are constraints to limit tool behavior and respect corporate policies. Guidelines influence the LLM decision process.

8. Review tool parameters.

   When we take a look at the Parameters configuration, we see integration input parameters. The **`claimamount`** parameter is listed and it is sent to the LLM because it's marked as Visible. We don't need to make any changes.

   If you were working with an integration that had many parameters, you could make visible to the LLM only parameters relevant to the tool.



9. Click **Save** to save your changes.

10. Click **Back** to return to the Project page.

    The integration can now be used as a tool in an AI agent.

Next step: Since we already created an AI agent in the first tutorial Tutorial: Build and Run Your First AI Agent in the step Build the AI Agent, let's add the human approval tool to the AI agent. Go to Add the Human Approval Tool to the AI Agent.

Add the Human Approval Tool to the AI Agent

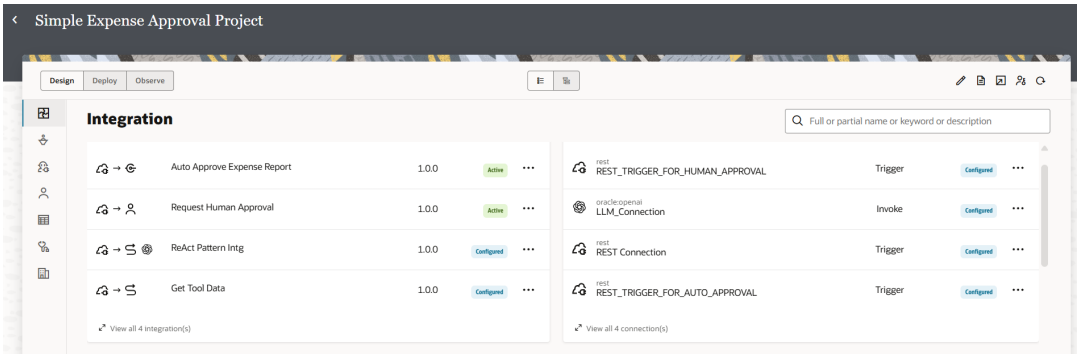
We now have all that's needed to add the human approval tool to the AI agent: the human approval form, human approval workflow, the human approval integration, and agentic AI tool. We can now add the tool to our AI agent.

Prerequisites:

- Build the human approval tool. See Build the Human Approval Tool.
- Register the human approval integration as an agentic AI tool. See Register the Human Approval Integration as an Agentic AI Tool.

1. In the navigation pane, select **Projects**.
2. Select the project in which you created your AI agent. For this tutorial, select the **Simple Expense Approval Project** project.

The project details page is displayed. You should see created connections, and the integrations you created Auto Approve Expense Report as well as Request Human Approval. You also see the system-generated integrations ReAct Pattern Intg and Get Tool Data.



3. Check that the Request Human Approval integration is Active.

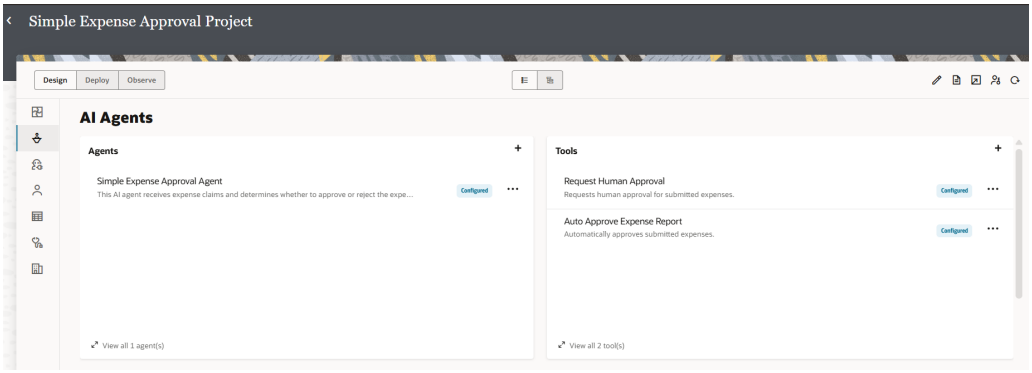
   The integration must be activated to be used as a tool in an AI agent. If the Request Human Approval integration isn't Active, activate it:

   - Next to the **Request Human Approval** integration, click **Actions** ⋮, and select **Activate**.

4. Add the human approval tool to your AI agent.

   a. In the left navigation pane, select **AI Agents** .

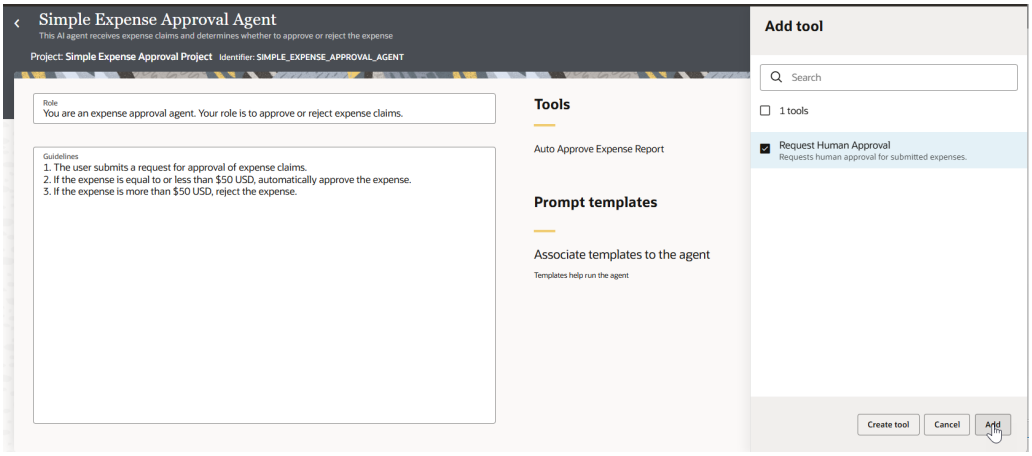
      You should see the **Simple Expense Approval** agent, and the tools **Auto Approve Expense Report** and **Request Human Approval**.



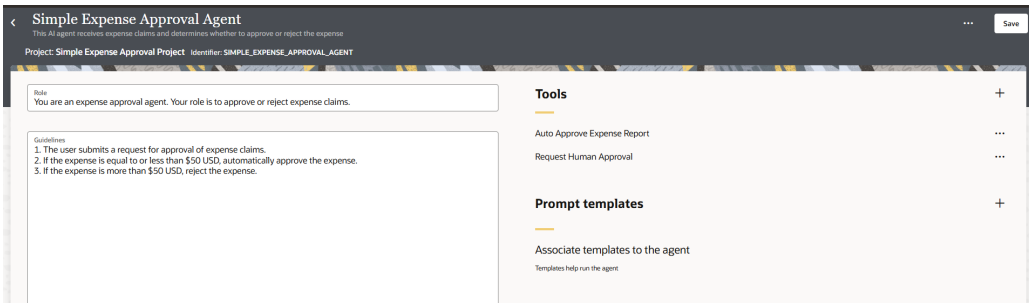
   b. In the **Agents** section, select the AI agent **Simple Expense Approval Agent**.

      The agent details page is displayed.

   c. In **Tools**, click **+** to add a new tool, select the **Request Human Approval** tool you previously created and click **Add**.



You should now see the Human Approval tool added to the AI agent.



d. Click **Save**.

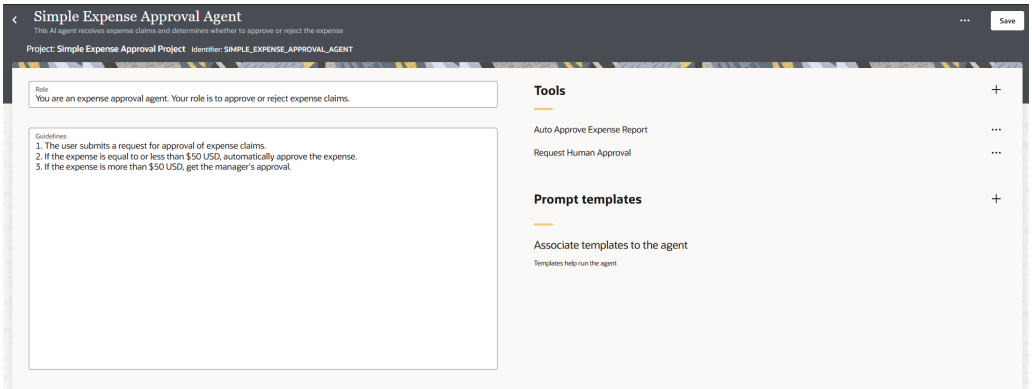
5. Change the AI agent guidelines to take into account the human approval tool.

a. Under **Guidelines**, update step 3 to indicate that manager's approval is required for expenses more than $50.

Replace: `3. If the expense is more than $50 USD, reject the expense.`

with:

```
3. If the expense is more than $50 USD, get the manager's approval.
```



b. Click **Save**.

Your agent is now ready to run.

6. Click **Back** to return to the AI Agents page.
7. Activate the AI agent.
   a. In the **Agents** section, find the **Simple Expense Approval Agent**.
   b. Click **Actions**, and select **Activate**.

Next step: Run and Test Human Approval in the AI Agent

Run and Test Human Approval in the AI Agent

Now that we have added the human approval tool to the AI agent, we can run and test the agent with the new tool.

Prerequisites: Add the Human Approval Tool to the AI Agent

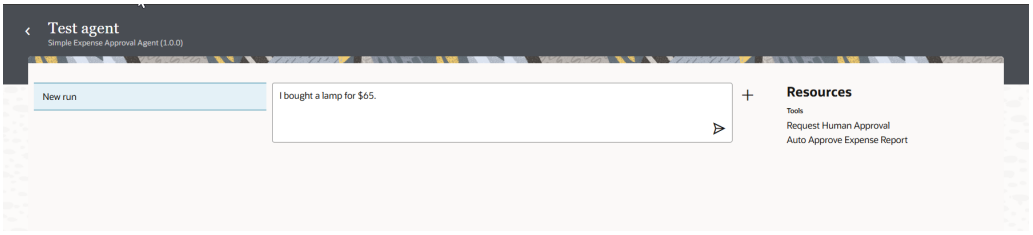
1. In the navigation pane, select **Projects**.
2. Select the project in which you created your AI agent. For this tutorial, select the **Simple Expense Approval Project**.
3. In the left navigation pane, select **AI Agents** .
4. Run the agent.
   a. In the **Agents** section, find the AI agent to run: **Simple Expense Approval Agent**.
   b. Click **Actions**, and select **Run**.

   The Test Agent page is displayed.
5. Test the agent.
   a. In the Test Agent page, enter a natural language prompt to test the AI agent.

   Our AI agent is an expense approval agent and we're testing the agent requesting human approval. Enter an expense for more than $50 USD and click **Run** .

   For example:

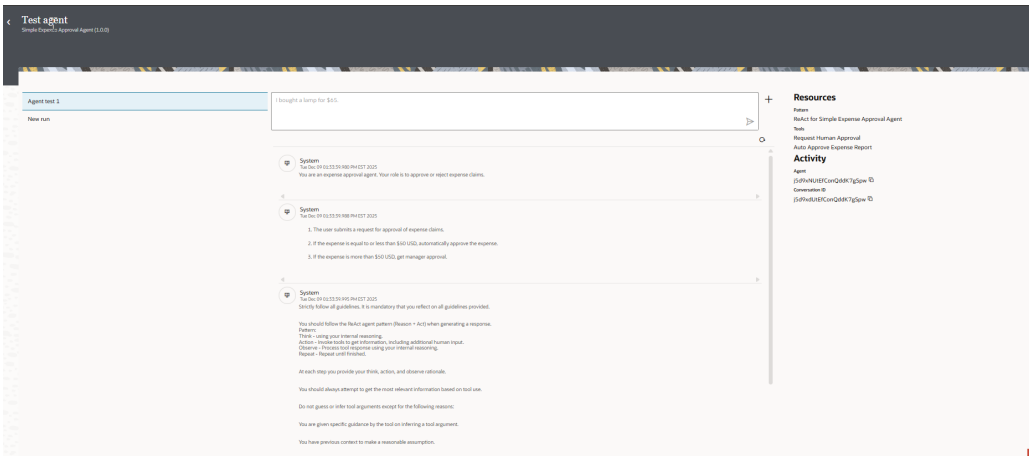
   ```
   I bought a lamp for $65.
   ```



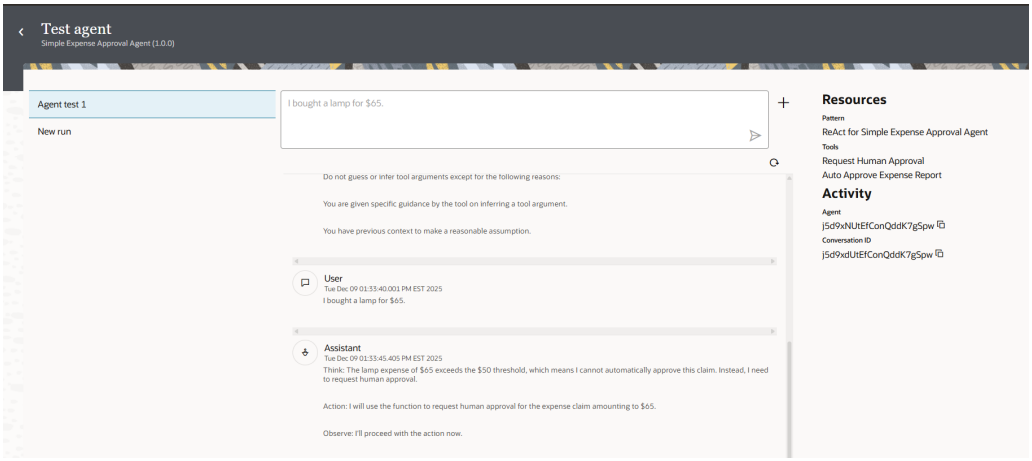
   The progress bar displays and then you see agent actions.
   b. Look at the agent actions.
      - **System**: indicates what you sent to the LLM in the system prompt. This includes the agent role, the agent guidelines, the AI agent pattern guidelines.

- **User**: user indicates what you entered to send to the LLM.
- **Assistant**: shows the agent reasoning. You should see in the reasoning that the agent is going to request human approval.

System is what you sent to the LLM:



Assistant displays that the AI agent is going to invoke the human approval tool. The human approval tool is the integration that runs the approval workflow. The AI agent cannot go any further. It needs to wait for human approval. You need to manually approve or reject the expense.



6. Approve the expense.

   a. From the **Test Agent** page, click **Back** to return to the **AI Agents** page.

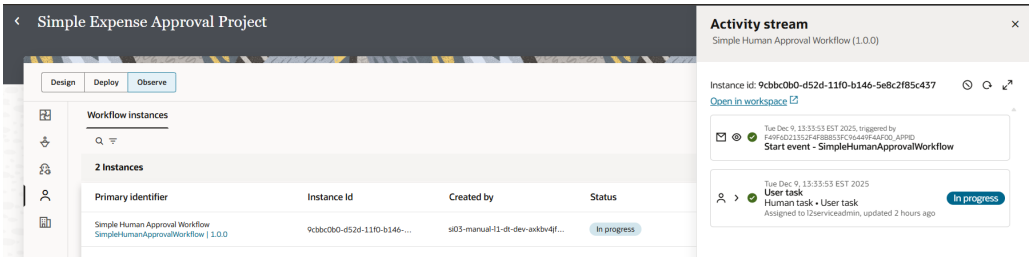
   b. Select the **Observe** tab.

   c. In the left toolbar of your project, click **Human in the Loop** .

   You'll see workflow instances.

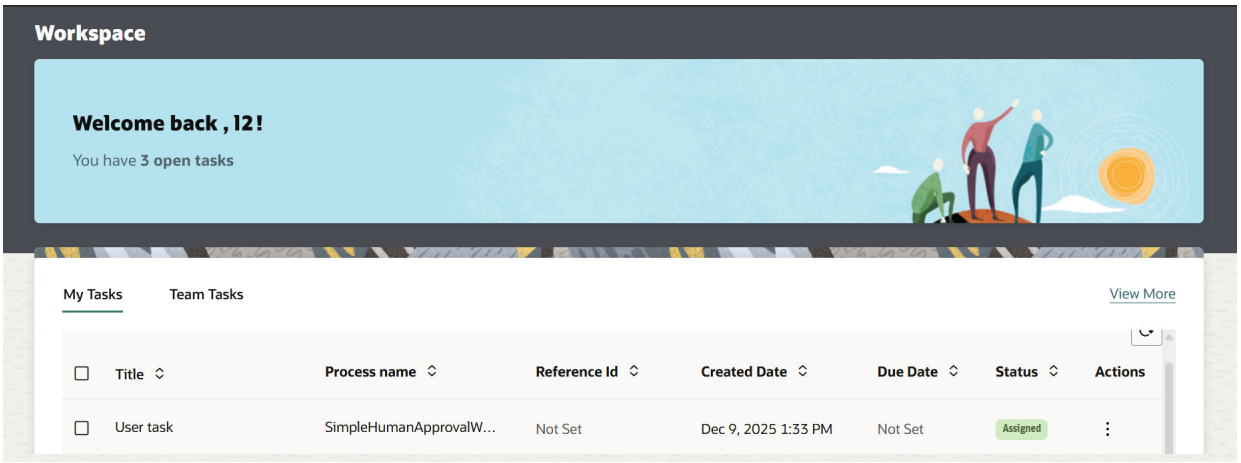
   d. Hover over the Simple Human Approval Workflow that was invoked by the AI agent and click **View audit** .

The Activity stream panel opens. The start event indicates that the workflow was started. The user task's status shows In progress indicating that the user hasn't completed the task.



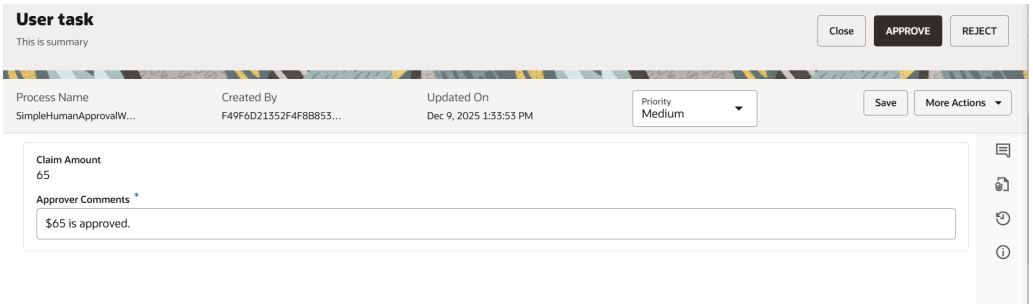
e. Click the **Open in workspace** link.

The Process Automation Workspace opens in another browser tab.



f. Click on the task to view the form.

The task form is displayed.



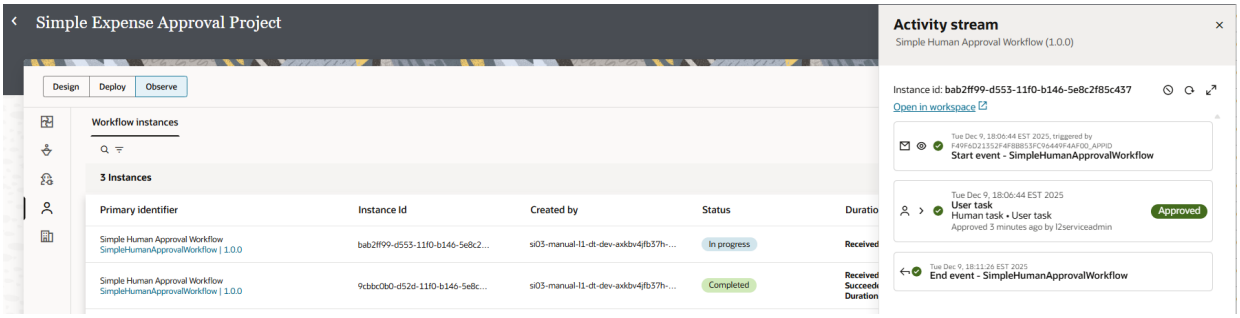
g. In Approver comments, add any comment you want such as **$65 is approved** and click **Approve**.

The task is approved, and you get a confirmation message. You are back in the Workspace main page.

h. Close the Workspace Tasks tab in the browser.

i. In your browser, navigate back to the **Observe** tab and refresh the Activity stream tab.

You should see the end event in your workflow. This indicates the workflow was completed.



7. Check the end of the AI agent run.

a. In the left navigation pane, select **AI Agents** .

b. Select the **Observe** tab.

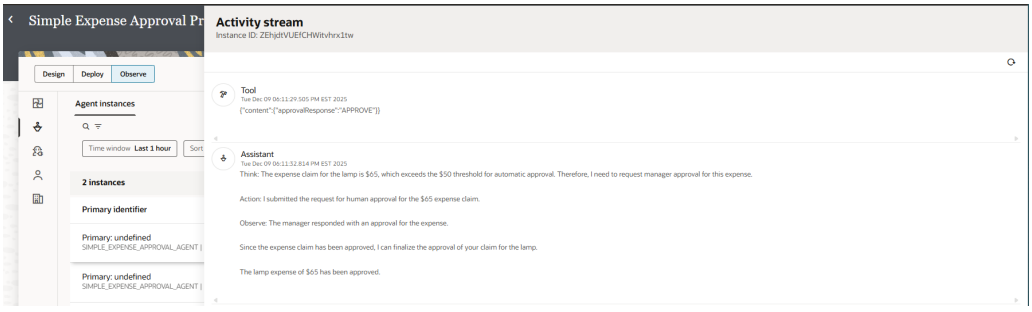
The Agent instances page is displayed.

c. Hover over the agent instance that you just ran SIMPLE_EXPENSE_APPROVAL_AGENT and click **View details** .

The Activity stream panel opens.

Notice:

- **Tool**: this is the response from the workflow to the AI agent, which is APPROVE.
- **Assistant**: You can see the reasoning of the AI agent for Think, Action, Observe. You then can see the conclusion. The AI agent approves the claim because it was approved by the manager.



Congratulations!

You've created a human approval tool and run it in your AI agent. You learned how to build a human approval tool, how to approve an assigned user task, and how to track the human approval tool in the AI agent.

Getting Started with Knowledge Bases

Use a knowledge base to create a built-in RAG (Retrieval Augmented Generation) engine to enhance the capabilities of agentic AI.

Fulfill Your Business Goals

Goal	Details	How knowledge bases help
Give agentic AI access to corporate documents	To do their jobs effectively, AI agents need access to corporate documents. RAG engines are the most effective way to collect this information. However, creating your own RAG infrastructure is complex and difficult.	If you have a RAG engine already, the AI agents in Oracle Integration can work with it. However, if you don't have a RAG engine, not to worry. You can create one easily in Oracle Integration using a knowledge base. A knowledge base offers a simplified experience with a RAG engine, so you can get started quickly without having to learn all the ins and outs of a RAG engine, including how to chunk the data.
Control the versions of documents that agentic AI uses	Business requirements change, and so do corporate documents. You need a way to keep your knowledge base up to date with the latest documentation so that your AI agents can work effectively.	A knowledge base helps you easily identify the versions of each uploaded documents, so you can update and delete documents as needed.

Knowledge Base Essentials

Definition	A **knowledge base** provides agentic AI with a built-in RAG (Retrieval-Augmented Generation) engine, which is often a document database. Create a knowledge base if your organization doesn't yet have a RAG framework and you need to provide agentic AI with a document database. You can create a knowledge base to work with agentic AI that you build in Oracle Integration or elsewhere. A knowledge base offers an easy way to create a RAG framework without having to understand all of its complex details, such as embeddings and chunking. Knowledge bases have robust and configurable back-ends, along with default values that help you make decisions quickly and start uploading your documents through a drag-and-drop interface. If your organization already has a RAG engine, such as a vector database into which you've ingested documents, agentic AI in Oracle Integration can connect to it and search against it. For example, you might have uploaded all your expense policies to the database. In such cases, you probably don't need to create a knowledge base.  Knowledge base

Usage	Working with a knowledge base involves two phases: • **Ingestion:** Add your documents to the vector database. Oracle Integration provides a predefined integration for ingestion, so you can easily drag and drop documents into your knowledge base. Most people convert PDFs to Markdown for this task. Oracle handles the chunking for you. After uploading the documents, run test queries to confirm success. • **Retrieval:** Ask an agent a question, and allow the agent to retrieve an answer from the knowledge base. You can even specify the number of results to receive. The output includes the search results, a matching score, the document version, and a link to the source document.
AI-powered capabilities	You can easily create tools for agentic AI using the recipes provided by Oracle Integration. To explore all AI features, see AI Innovation and Oracle Integration in *Using Integrations in Oracle Integration 3*.
Availability	Knowledge bases are available in select Oracle Integration editions. See Oracle Integration Editions in *Provisioning and Administering Oracle Integration 3*. Additionally, you must configure OpenSearch to use knowledge bases.

3

Use Integrations as Tools in an MCP Server

You can invoke integrations from any Model Context Protocol(MCP) client by registering the integration as an agentic AI tool, then enabling MCP for the project.

Topics:

FAQs for Projects as MCP Servers

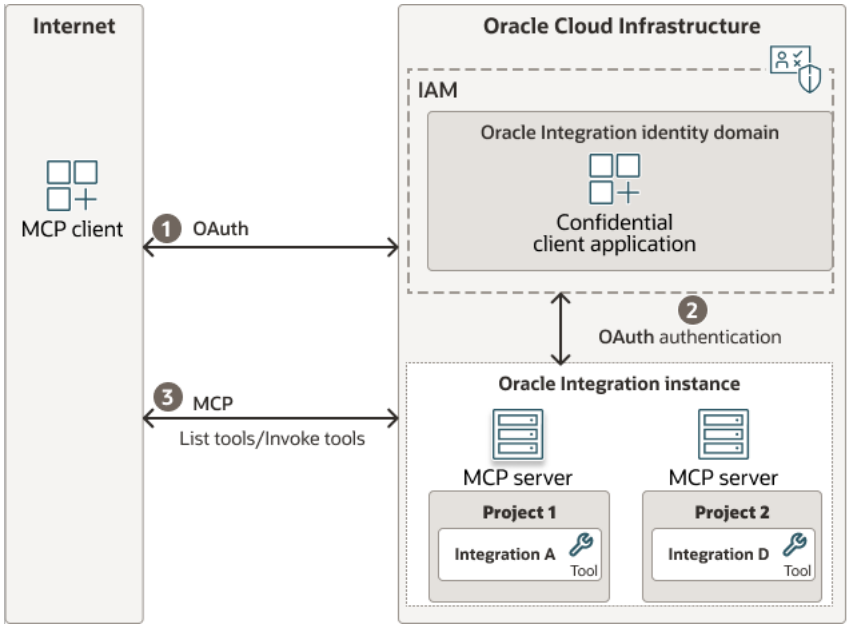
Projects can be configured to act as Model Context Protocol(MCP) servers for agentic AI tools.

How do projects become MCP servers?

You can invoke integrations from any MCP client by registering the integration as an agentic AI tool, then enabling MCP for the project. Each project becomes an MCP server and has its own MCP server URL.

Only integrations registered as agentic AI tools are discoverable through MCP.

Security is provided through OAuth.



How many MCP severs are there?

There is one MCP server per Oracle Integration project.

Supported operations are `tools/list` and `tools/call`.

Each project has its own MCP server URL.

For example:

```
https://myinstance.integration.eu-frankfurt-1.ocp.oraclecloud.com/mcp-
server/v1/projects/PROJECTA/mcp
https://myinstance.integration.eu-frankfurt-1.ocp.oraclecloud.com/mcp-
server/v1/projects/PROJECTB/mcp
```

What kind of security is there?

You must use OAuth 2.0 to expose integrations as tools with MCP. You need to create a confidential client application, assign scopes and roles, and activate it. See Complete Prerequisites: Create and Activate the Client Application.

- You must be the OCI tenant and domain administrator to configure the confidential client application.
- You need at least one confidential client application per Oracle Integration instance.
- The confidential client application must be assigned the ServiceInvoker role. For a detailed description of user roles, see What Users Can Do in the Integrations Design Section by Role.

Can any integration become a tool?

Any integration can be used as an Agentic AI tool, but the integration must meet the following criteria:

- The integration must be part of a project.

- The integration must be Active.
- The first connection in your integration must be a REST trigger connection with the REST Adapter.
- The REST trigger connection in your integration must have:
  - Authentication type OAuth.
  - JSON payload
  - POST verb

Is MCP available in all regions?

Yes, MCP is available in all Oracle Integration regions. .

Where do I find the MCP server URL?

You can find the MCP server URL in the project details page. If the project has MCP server enabled, you'll see the MCP server URL. For instructions, see Get the MCP Server URL.

Workflow to Use an Integration as a Tool with MCP

Here's a summary of the steps to follow to use an integration as an agentic AI tool and discover it through Model Context Protocol (MCP).

Step	Task
Prerequisites	Complete prerequisites: 1. Find out more about how projects can be MCP servers. See FAQs for Projects as MCP Servers . 2. Create a confidential application. Before you can discover an integration as a tool through MCP, you need to configure a confidential application to connect with OAuth. For instructions, see Complete Prerequisites: Create and Activate the Client Application. Step summary: a. Create a confidential application. b. Assign scopes to the confidential application. c. Assign the ServiceInvoker role to the confidential application. d. Activate the confidential application. e. Get the confidential application client ID and secret. f. Get the confidential application access token.
1	Register an Integration as an Agentic AI Tool
2	Enable MCP for the Project
3	Get the MCP Server URL
4	Discover Integrations as Tools from MCP Clients

Complete Prerequisites: Create and Activate the Client Application

Configure and activate the confidential client application, then take note of the client ID and secret. You'll need that information to connect from third-party applications.

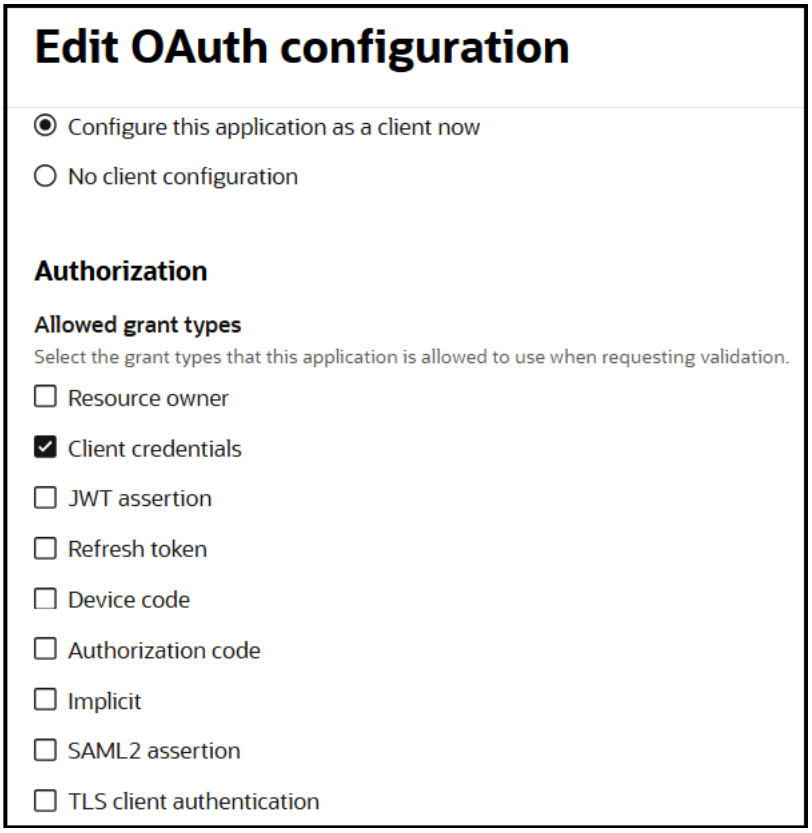
When you configure the confidential client application, you specify the grant type and assign scopes and roles. The confidential client application requires the ServiceInvoker role.

1. Access the identity domain.
   a. Log in to the Oracle Cloud Infrastructure Console with your identity domain administrator credentials.
   b. In the navigation pane, click **Identity & Security**.
   c. Click **Domains**.
   d. Select your compartment.
   e. Select the identity domain.
   f. In the menu bar, click **Integrated applications**.

      This is the location at which you create the client application for your grant type.

2. Create and configure the client application.
   a. Click **Add application**.
   b. Select **Confidential Application**, then click **Launch workflow**.
   c. Enter a **name**.

      The remaining fields on this page are optional and can be ignored.
   d. Click **Submit**.
   e. Click the **OAuth configuration** tab, then the **Edit OAuth configuration** subtab.
   f. In the **Client configuration** panel, select **Configure this application as a client now**.
   g. For client credentials, select **Client credentials** in the **Allowed grant types** section



h. Leave the **Redirect URL**, **Post-logout redirect URL**, and **Logout URL** fields blank.

i. For **Client type**, ensure that **Confidential** is selected.

j. Bypass several fields and scroll down to the **Token issuance policy** section.

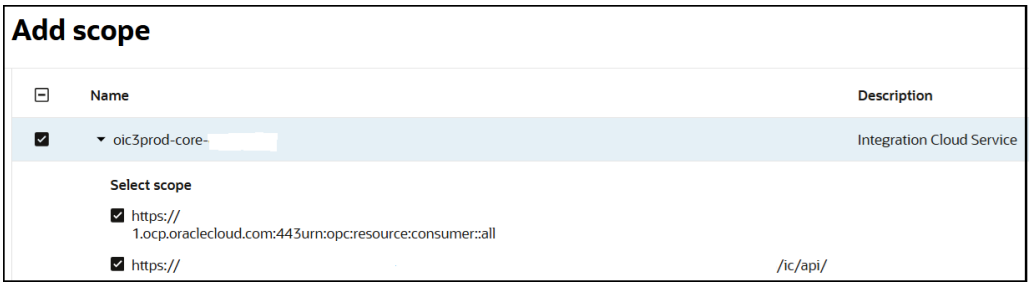
k. Select **Confidential** in the **Authorized resources** section.

l. Click the **Add Resources** toggle.

m. Click **Add scope**.

n. Find and expand the Oracle Integration application for your instance.

o. Select the two scopes appended with the following details: **urn:opc:resource:consumer::all** and **ic/api/**.



p. Click **Add**.

The scopes are displayed in the **Resources** section.

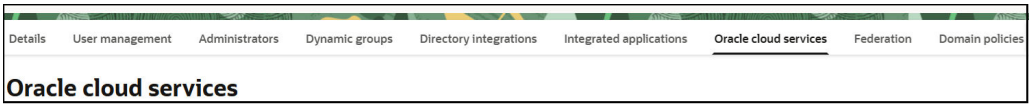
q. Ignore the **Add app roles** check box. This selection is not required.

r. Click **Submit**.

The details page for the client application is displayed.

3. Add the **ServiceInvoker** role to the client application.

a. In the menu bar, click **Oracle cloud services**.

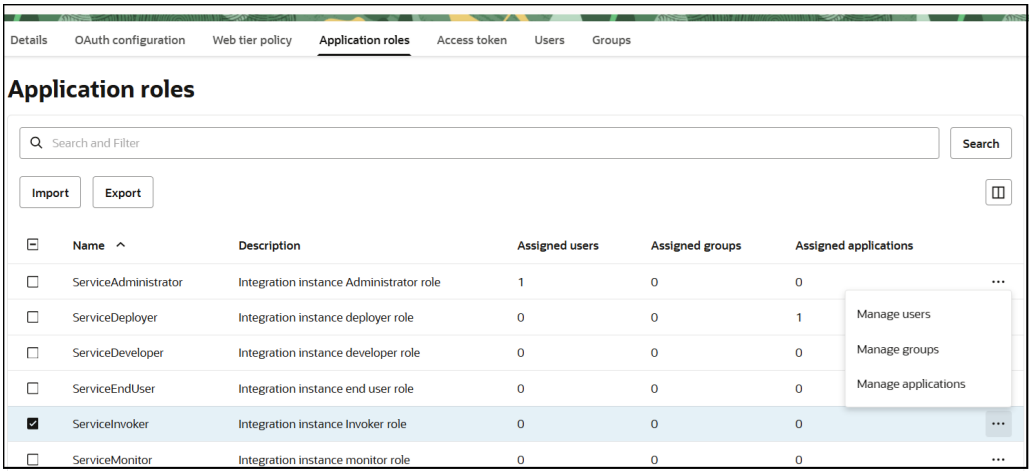


b. Click the specific application corresponding to the Oracle Integration instance.

c. In the menu bar, click **Application roles**.

d. For client credentials, expand **ServiceInvoker**, then click **Actions** ... next to **Assigned applications**.

Select to assign users, groups, and applications to the instance application.

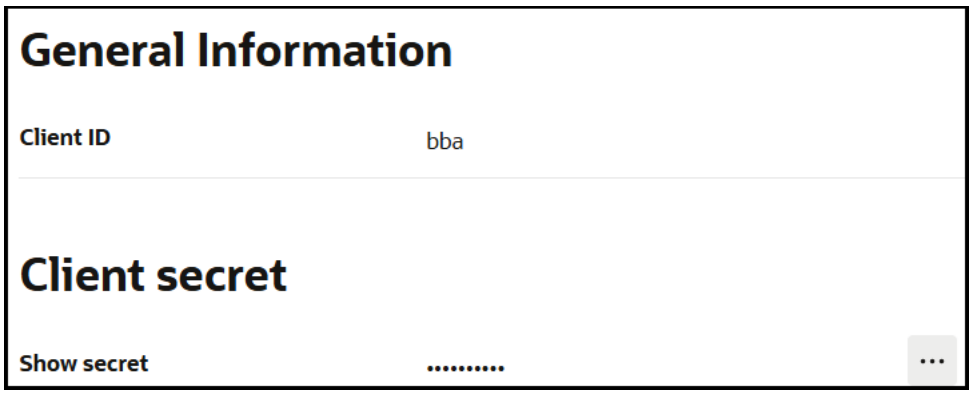


4. Activate the confidential application.

- From the **Actions** menu at the top, select **Activate**, and then **Activate application** to activate the client application for use.

5. Get the confidential client application client ID and secret.

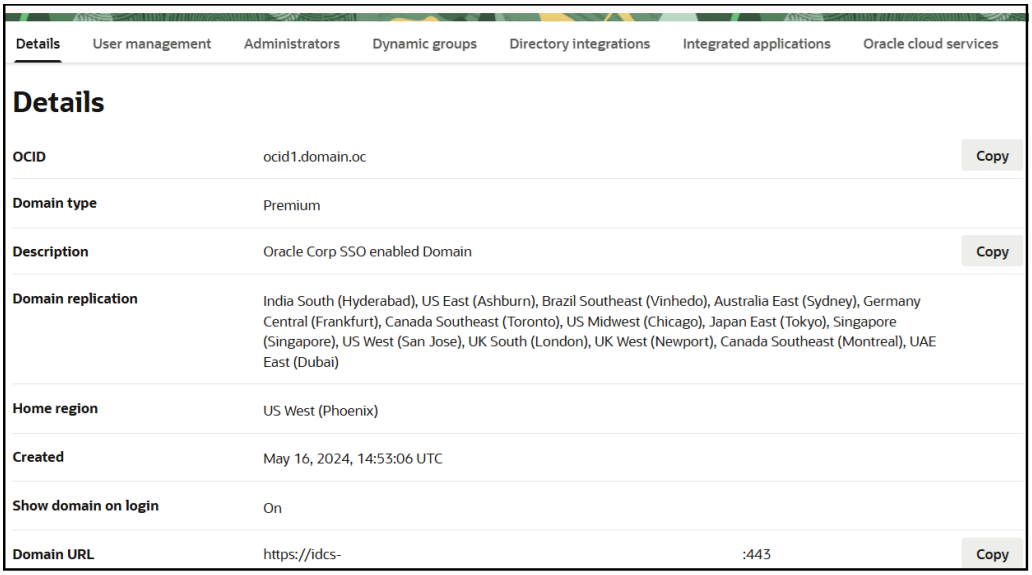
In the **General Information** section, note the client ID and client secret values. These values are required for the third-party application that is communicating with the identity domain.



6. Get the access token for the client credentials grant type.
   a. Fetch the access client to make an access token request with the client credentials.

```
##Syntax
curl -i -H 'Authorization: Basic <base64Encoded clientid:secret>' -H
'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --
request POST https://
<Identity_Domain_Service_Instance>.identity.oraclecloud.com/oauth2/v1/
token -d 'grant_type=client_credentials&scope=<app scope>'
###where
#### <base64-clientid-secret> - Base 64 encode clientId:ClientSecret
#### <app scope> - Scope added while creating application in client
configuration section (Ends with urn:opc:resource:consumer::all)

##Example
curl -i -H 'Authorization: Basic OGQyM...ZDA0Mjcz' -H 'Content-Type:
application/x-www-form-urlencoded;charset=UTF-8' --request POST https://
<identity_domain_host>/oauth2/v1/token -d
'grant_type=client_credentials&scope=https://<Resource APP
Audience>urn:opc:resource:consumer::all'
```

   Where `Identity_Domain_Service_Instance` is the value in the **Domain URL** field of the **Details** tab of the instance application.



b. Capture the `access_token` from the response to use for authorization.

```
{
    "access_token": "eyJ4NXQjG...dfsdfsFgets2ed",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Next Step: Register an Integration as an Agentic AI Tool

Register an Integration as an Agentic AI Tool

An integration can become an agentic AI tool for AI agents. AI agents can then invoke the integration as a tool to accomplish a specific task.

Prerequisites:

Any integration can be used as an Agentic AI tool, but the integration must meet the following criteria:

- The integration must be part of a project.
- The integration must be Active.
- The first connection in your integration must be a REST trigger connection with the REST Adapter.
- The REST trigger connection in your integration must have:
  - Authentication type OAuth.
  - JSON payload
  - POST verb

1. In the navigation pane, select **Projects**.
2. Select the project in which your integration is located.
3. In the **Integrations** section, find the integration that you want register as a tool.

4. Check that your integration is Active. If it's not active, activate it by clicking **Actions** ⋮, and selecting **Activate**.
5. Register the integration as a tool. Click **Actions** ⋮, and select **Create agentic AI tool**

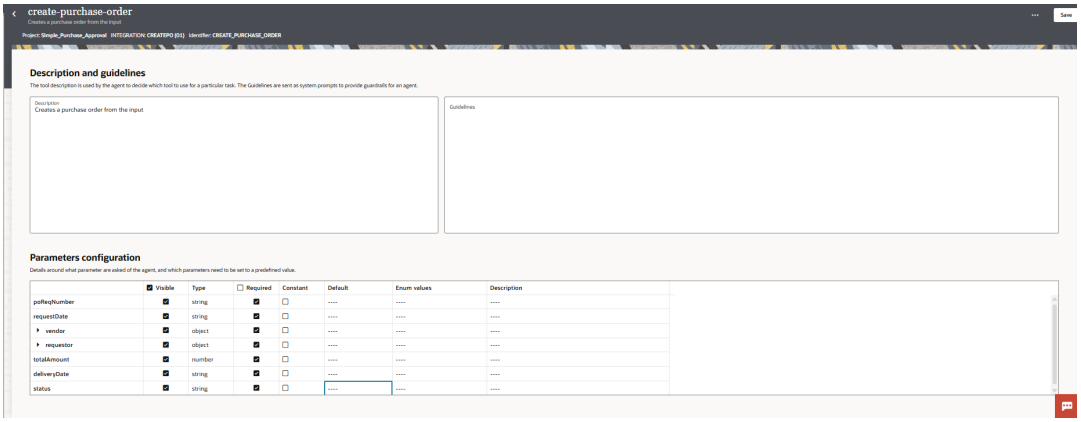
   The Create Tool panel is displayed.
6. Enter information for the tool.

Field	Description
Name	Required. Not sent to the Large Language Model (LLM). Automatically populated from the integration name. Example: `validate-invoice-data`. You cannot change the name after the tool has been created.
Identifier	Required. Uniquely identifies the tool in the project. By default, the identifier is automatically populated from the tool name. Sent to the LLM as tool metadata. The AI agent uses this information to understand the purpose of the tool and when to use it. Specify a descriptive tool identifier so that the AI agent knows exactly what the tool does. Example: `validate-invoice-data`. You cannot change the identifier after the tool has been created
Description	Required. By default, automatically populated from the integration description. Sent to the LLM as part of the system prompt. The description helps the LLM decide when to use the tool. Add information to clearly describe what the tool does and when to use it. Clear descriptions help LLMs use tools correctly and reduce errors. Example: `Validates invoice data against business rules and vendor database. Usage: Use when processing invoices to ensure data accuracy before approval.`

7. Click **Create**.

   The Tool details page is displayed. The tool lists the tool description, guidelines, and input parameters.
8. Enter additional information for the tool and identify which parameters are sent to the LLM.

   You already defined the description.



Field	Description
Guidelines	Optional. Sent to the LLM as part of the system prompt. Guidelines are constraints to limit tool behavior and respect corporate policies. Guidelines influence the LLM decision process. Specify constraints such as when the tool should be used and when it should not be used. For example, if you had a weather tool and you wanted to limit queries to only cities in the United States, you could specify as a guideline: `only use the tool for cities in U.S.`
Parameters configuration	Lists all input parameters for the integration. These are automatically populated. Configure parameter information for the tool. • **Visible:** Add a checkmark to the fields that will be sent to the LLM. Fields that do not have a checkmark are not sent to the LLM. – Expose only essential parameters. – Do not send to the LLM technical or internal parameters that are not relevant to the purpose of the tool. • **Type:** Automatically populated from the integration. • **Required:** When you indicate that a field is required, you are indicating to the LLM that it must assign a value to that parameter. Specifying a parameter as required ensures that there will always be a value for the parameter when the AI agent calls the agentic AI tool. • **Constant:** Enter any specific value that you want to assign to the parameter and you do not want the LLM to decide on. • **Default Values:** Specify default values for parameters when you know there's a safe value for the parameter if no value is assigned. This reduces the chance of LLM hallucination. • **Enum Values:** Specify comma-separated values when there are several options that could be sent to the LLM. For example, for temperature, you would specify: `celsius,fahrenheit`. • **Description:** Required. Describe all parameters clearly. The LLM depends on the descriptions to understand the tool. Clear descriptions reduce LLM errors.

9. Click **Save** to save your changes.

Next step: Enable MCP for the Project

Enable MCP for the Project

When you enable MCP for a project, the project becomes an MCP server. Any integrations registered as agentic AI tools are discoverable through the MCP server URL, and AI agent frameworks that support MCP can invoke the integrations. Each project has its own MCP server URL.

For additional details on how MCP works with projects, see FAQs for Projects as MCP Servers .

Prerequisites: Register an Integration as an Agentic AI Tool

1. In the navigation pane, select **Projects**.
2. Select the project for which you want to enable MCP.
3. In the upper right corner, click  to display the Project details.
4. Click **Enable MCP server**.
5. Click **Save changes**.

   The MCP server URL is created when you save the project.
6. Click  to display the Project details again.

   The MCP server URL is listed. You can now use the MCP server URL to invoke integrations from AI agent frameworks and MCP clients.

Next Step: Discover Integrations as Tools from MCP Clients

Get the MCP Server URL

You can find the MCP server URL in the project details page. If the project has MCP server enabled, you will see the MCP server URL.

For additional details on how MCP works with projects, see FAQs for Projects as MCP Servers .

1. In the navigation pane, click **Projects**.
2. Select your project.
3. In the upper right corner, click  to display the Project details.

   The MCP server URL is listed.

Next Step: Discover Integrations as Tools from MCP Clients

Discover Integrations as Tools from MCP Clients

To discover integrations as agentic AI tools from MCP clients or AI agent frameworks that support MCP, you need to specify the MCP server URL and use the transport mechanism streamable HTTP.

Prerequisites:

You need to create a confidential client application, register an integration as an agentic AI tool, enable MCP for the project, and get the MCP server URL. The confidential client application requires at a minimum, the ServiceInvoker role.

1. Complete Prerequisites: Create and Activate the Client Application
2. Register an Integration as an Agentic AI Tool
3. Enable MCP for the Project
4. Get the MCP Server URL

Information You Need to Connect to a Project's MCP server

MCP server URL:

- How to get it: see Get the MCP Server URL.
- Use the runtime URL. Format:

  ```
  https://<NameOfServiceInstance>.integration.<region>.ocp.oraclecloud.com/
  mcp-server/v1/projects/<ProjectName>/mcp
  ```

  Example:

  ```
  https://myinstance.integration.eu-frankfurt-1.ocp.oraclecloud.com/mcp-
  server/v1/projects/PROJECTA/mcp
  ```

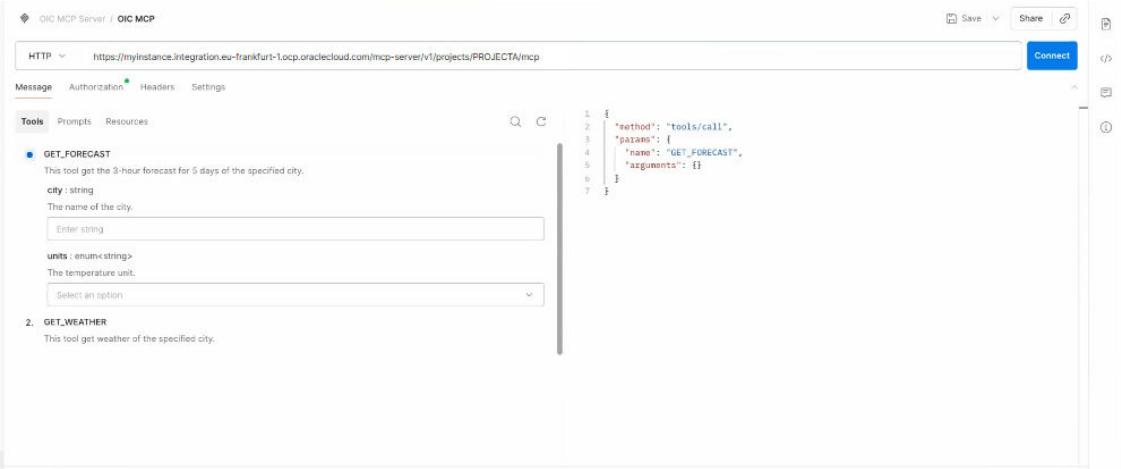
Access token for the confidential client application. See Complete Prerequisites: Create and Activate the Client Application to get the information.

Transport mechanism:

- streamable HTTP

Connect to the Project MCP Server with an MCP Client

Here's an example of connecting to a project's MCP server with Postman. You configure OAuth Authorization in Postman and add the MCP Server URL.



4

Design and Run AI Agents

Send data back to AI agents with conversation ID or pass dynamic data to AI agents with prompt templates.

Topics:

Run an AI Agent with Knowledge of Previous Runs

When an AI agent runs, a conversation ID is automatically generated by default, unless you specify the conversation ID. You can specify the conversation ID in different runs of an AI agent to associate multiple requests with a single conversation. A conversation includes all interactions with the AI agent such as prompts, data, and decisions made. Use the conversation ID when sending data back to an AI agent or when you want the AI agent to have knowledge of prompts, data, and decisions made in other runs.

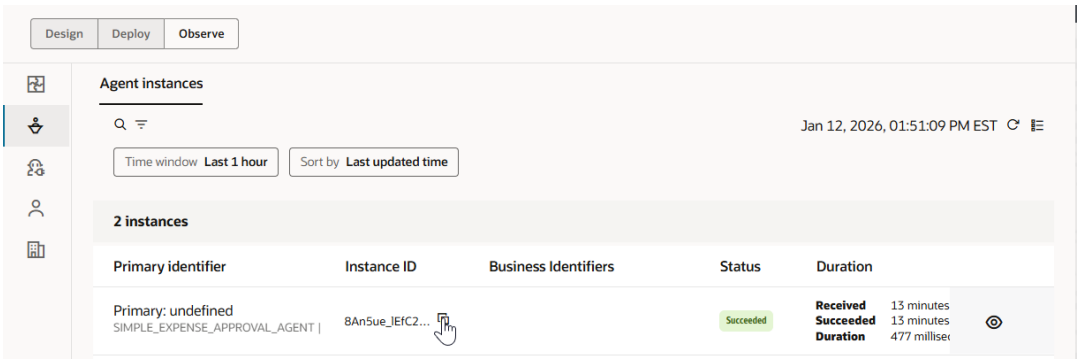
Data Retention for AI Agents

AI agent data is retained according to the data retention period of the Oracle Integration service instance. For configuration details, see Edit the Data Retention Period for an Instance.

Prerequisite: Get the Conversation ID

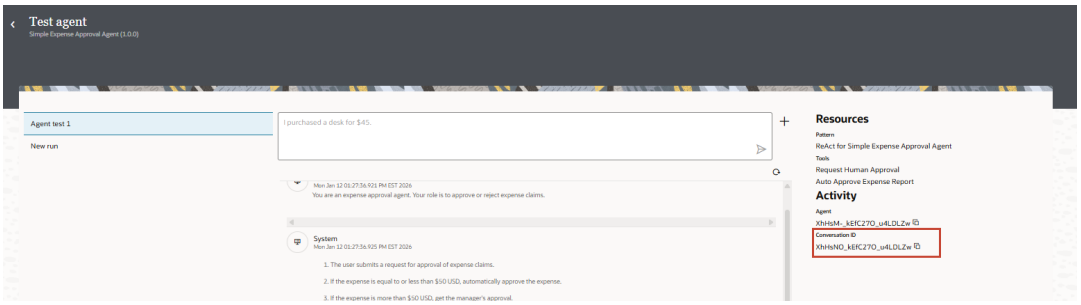
If the AI agent has already run, you can find the conversation ID in the Agent instances page:

1. In your project, click the **Observe** tab.
2. In the left navigation pane, select **AI Agents** .
   The Agent instances page is displayed.
3. Copy the conversation ID from the **Instance ID** column.



If the AI agent has not yet run, you can find the conversation ID in the Test Agent page, in the run of the AI agent:

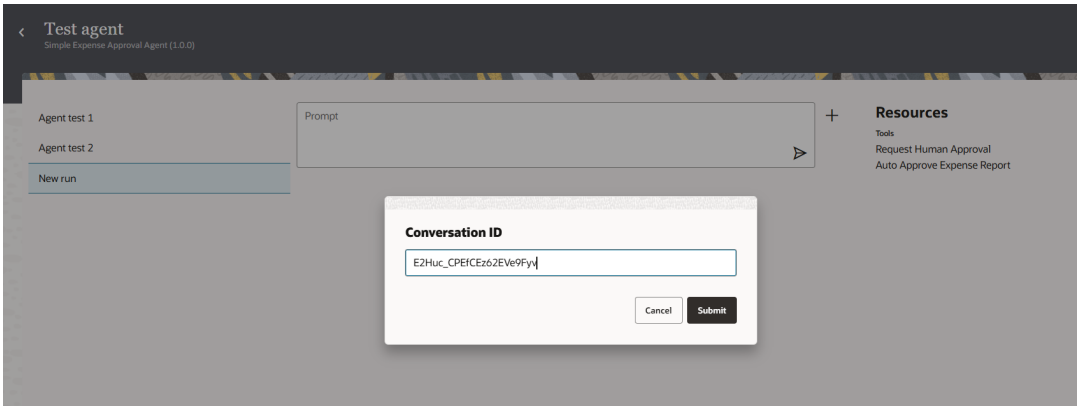
1. In your project, click the **Design** tab.
2. In the left navigation pane, select **AI Agents** .
3. In the **Agents** box, find the AI agent to run.
4. Click **Actions** , and select **Run**.
   The Test Agent page is displayed.
5. Select any test run. You'll see the conversation ID under the **Activity** section.



Run the AI Agent and Specify the Conversation ID

1. In your project, click the **Design** tab.
2. In the left navigation pane, select **AI Agents** .
3. In the **Agents** box, find the AI agent to run.
4. Click **Actions** , and select **Run**.

   The Test Agent page is displayed.
5. Click **+** to display the conversation ID dialog, specify the conversation ID, and click **Submit**.



6. In **Prompt**, enter a natural language prompt for the AI agent and click **Run** .

   Your AI agent can now make decisions knowing the conversation history of all other runs with the same conversation ID.

Send Dynamic Data to the AI Agent with Prompt Templates

Use prompt templates to define prompts that include variable data. When you run the AI agent, you select the prompt template and specify the JSON payload. The prompt template is combined with the payload and is submitted to the Large Language Model (LLM). Prompt templates use Liquid syntax.

Example:

- You configure the prompt template as: The rate is {{rateValue}}
- You specify this JSON payload: {"rateValue":500}
- The prompt sent to the LLM is: "The rate is 500"

Send Dynamic Data to the AI Agent

1. In the navigation pane, select **Projects**.
2. Select the project that contains your AI agent.
3. In the left navigation pane, select **AI Agents** .
4. Create the prompt template.
   a. In the **Prompt templates** box, click **Add** .

      The **Create prompt template** panel is displayed.

   b. Configure required fields.

      - **Name:** Name for your prompt template. Provide a descriptive name so that you know when to use the template. For example, if you had a weather AI agent, you could create a prompt template for cities in diferent countries, like Cities in France.
      - **Identifier:** Oracle Integration generates this value using the Name value.
      - **Description:** Provide additional information about the prompt template so anyone can know when to use it.

   c. Click **Create**.

      The Prompt Template details page is displayed.

   d. In **Prompt**, specify the prompt that you want to send to the LLM with any variables as placeholders for data, then click **Save**.

      Use Liquid syntax. For example:

      ```
      Get the weather for {{ city }}, France
      ```

   e. Click **Back** to return to the AI Agents page.

      You can now associate the prompt template with an AI agent.
5. Associate the prompt template with an AI agent.
   a. Find the AI agent with which you want to associate the prompt template.
   b. Deactivate the AI agent if it's Active.
   c. Select the AI agent.

      The Agent details page is displayed.

d. In the **Prompt Templates** section, click **+**, select the prompt template you just created, then click **Add**.

e. Click **Save** to save your changes to the AI agent.

f. Click **Back** to return to the AI Agents page.

You can now run the AI agent with the new prompt template.

6. Run the AI agent and specify the prompt template.

a. In the **Agents** box, find the AI agent to run.

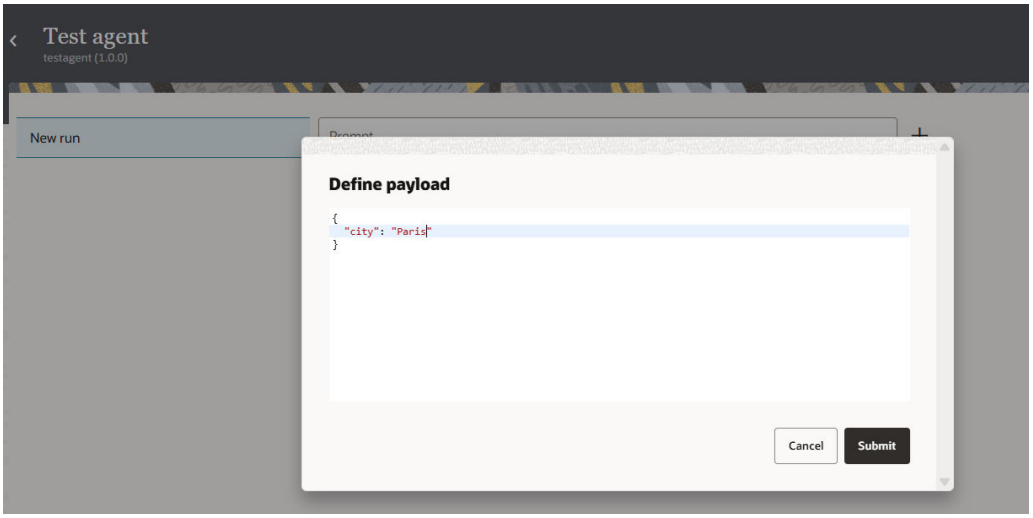
b. Click **Actions**, and select **Run**.

c. Click **New run** to test the agent with the prompt template.

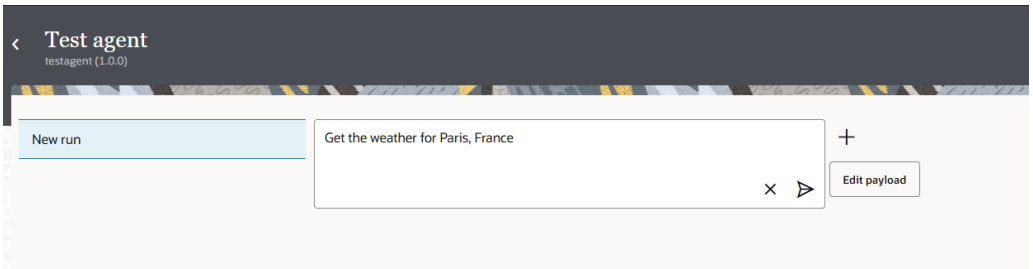
d. Click in the **Prompt** field to display a list of prompt templates, choose the prompt template you created.

The Define payload dialog is displayed.

e. In **Define payload**, enter the values for the variables and click **Submit**.



The values you entered are added to the prompt created from the prompt template.



f. In **Prompt**, click **Run**.