SONY

Vision and Sensing Application SDK AI Model and PPL Import Functional Specifications

Version 0.2.0

2023 - 1 - 30

TOC

1. Change history

Date	What/Why
2023/01/30	Initial draft

2. Terms/Abbreviations

Terms/Abbreviations	Meaning
PPL	A module that processes the output of the AI model(Output Tensor) of edge AI devices
SAS	Shared Access Signatures URI that grant limited access to Azure storage resources

3. Reference materials

- Reference/Related documents
  - API Reference
    - https://developer.aitrios.sony-semicon.com/development-guides/reference/api-references/
  - Console Access Library Functional Specifications
    - https://developer.aitrios.sony-semicon.com/development-guides/documents/specifications/

4. Expected use case

- You want to import AI models created in your environment into Console for AITRIOS
- You want to import created PPL into Console for AITRIOS
- You want to check the import status of the AI model or PPL
- You want to convert an AI model imported into Console for AITRIOS into a format that can be deployed to edge AI devices
- You want to check AI model conversion state
- You want to delete an AI model or PPL that has been imported into Console for AITRIOS

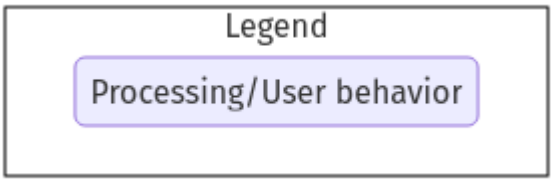
5. Functional overview/Algorithm

Functional overview

- Users can use Console Access Library in the SDK's Dev Container (Local PC or Codespaces)
    - Users can do the following through the Console Access Library:
        - Import AI models into Console for AITRIOS
            - Use AI models supported by Console for AITRIOS. See here.
            - Must store AI models to Azure Blob Storage to import into Console for AITRIOS
        - Convert AI models imported into Console for AITRIOS
        - Import PPL into Console for AITRIOS
            - Import the following PPL:
                - SDK supports ".wasm"(File not yet AOT compiled)

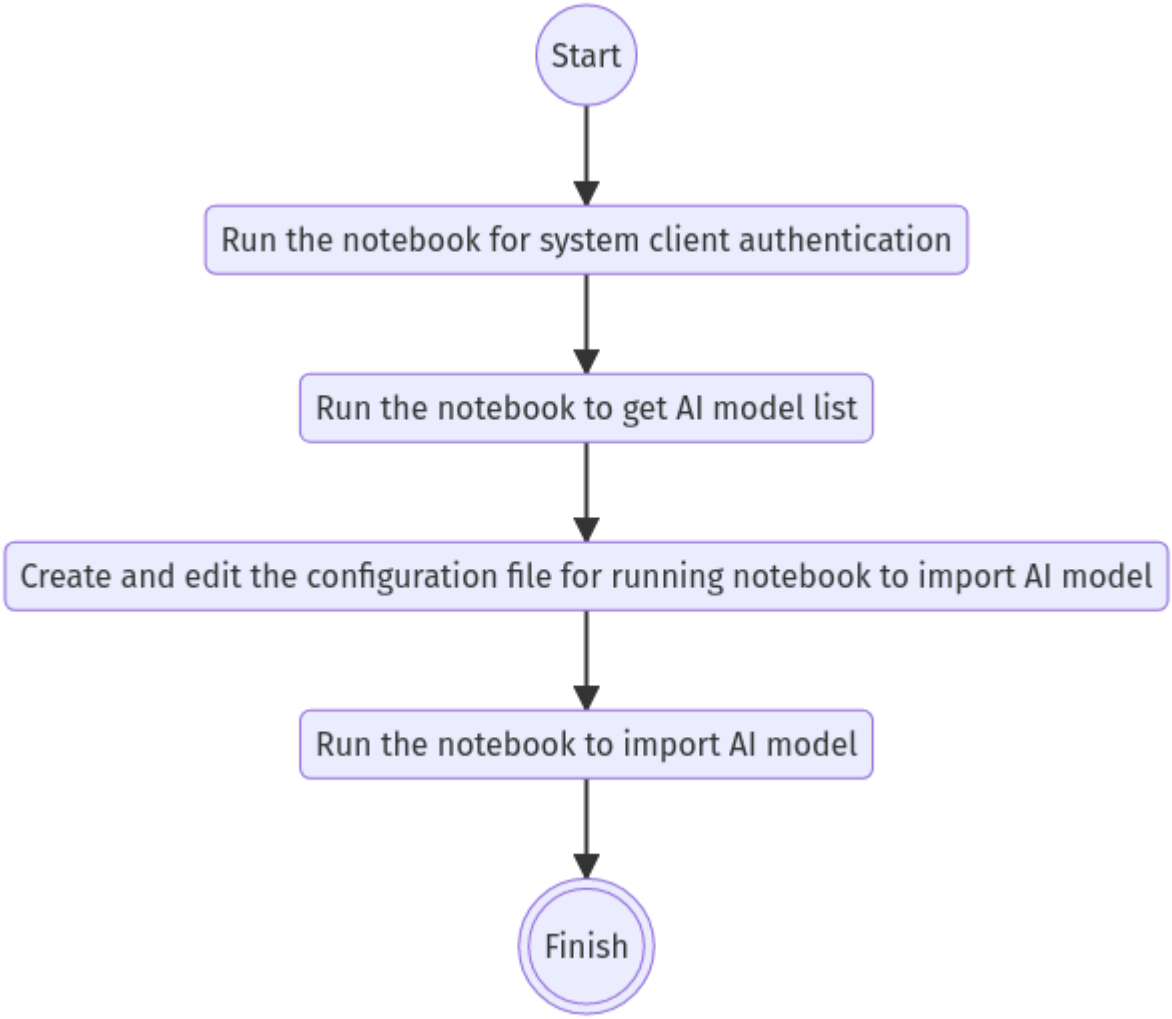
Extensions for PPL files that can be imported	SDK support
.wasm(File not yet AOT compiled)	Yes
.aot(AOT compiled file)	No

Flow



Import AI model

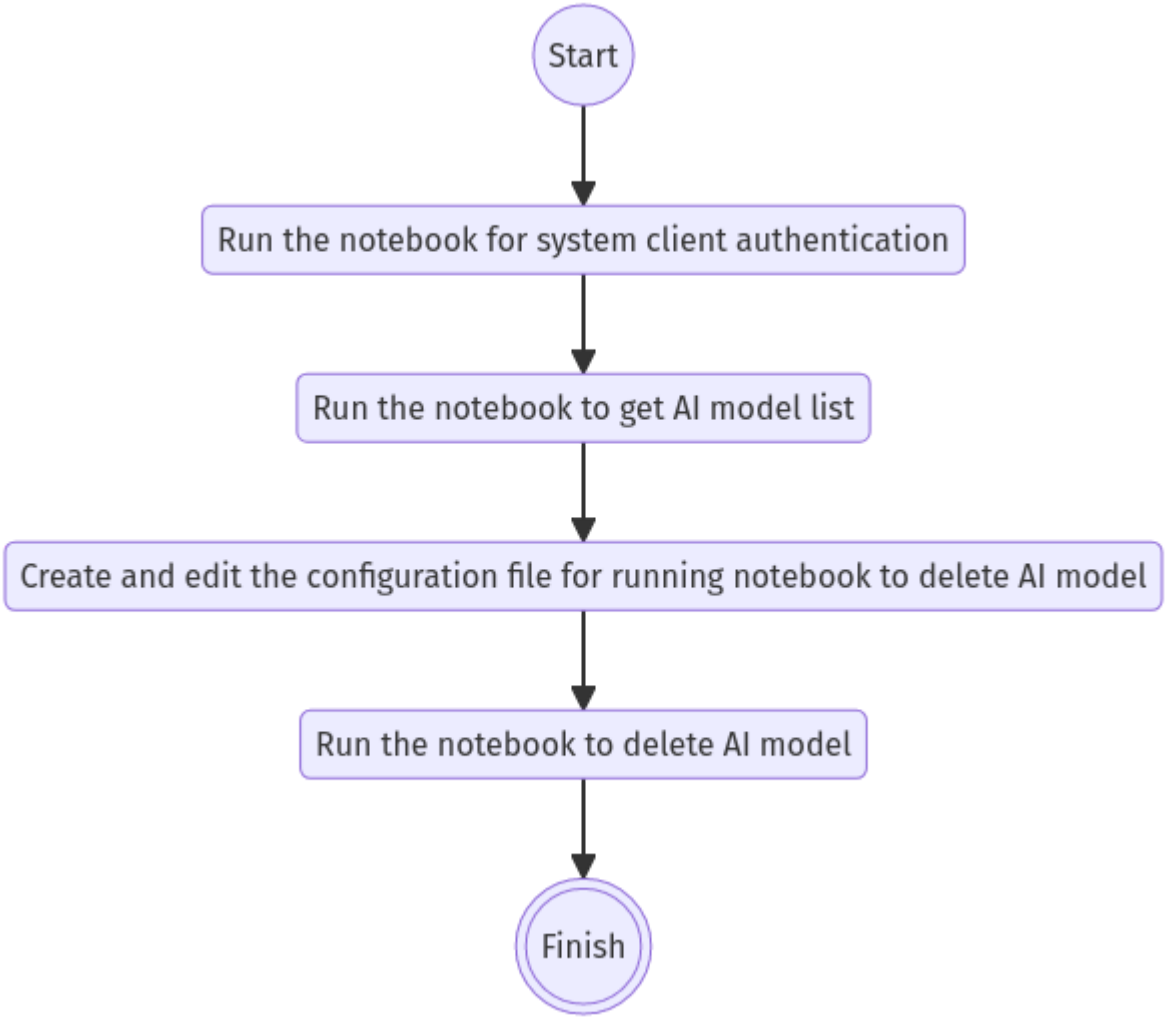
- Flow



- Flow details
  1. Run the notebook for system client authentication
  2. Run the notebook to get AI model list
     - Run the notebook to get a list of AI models that have been imported into Console for AITRIOS, and get settings in the configuration file, **`model_id`**.
       - Assume the following case
         - You want to upgrade an AI model that has already been imported into Console for AITRIOS
         - You want to check the AI model import status of Console for AITRIOS
         - You want to check the conversion status of the AI model in Console for AITRIOS
  3. Create and edit the configuration file for running notebook to import AI model
     - Create and edit the configuration file *configuration.json* to configure notebook runtime settings
  4. Run the notebook to import AI model
     - Run the notebook with the following features:
       - Imports AI models into Console for AITRIOS
       - Checks the AI model import status of Console for AITRIOS
       - Converts an AI model imported into Console for AITRIOS
       - Checks AI model conversion state.

Delete AI model

- Flow

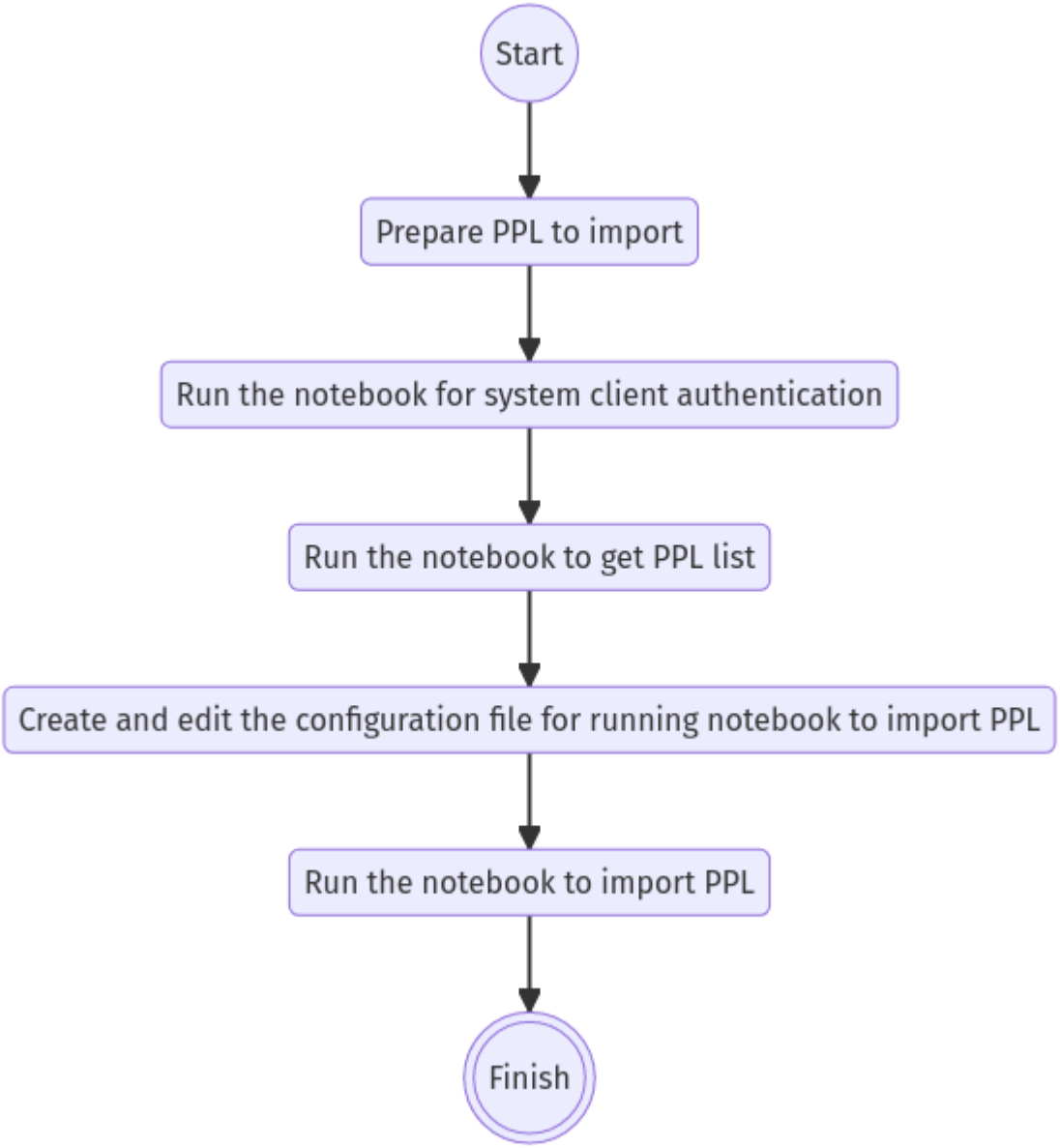


- Flow details

  1. Run the notebook for system client authentication
  2. Run the notebook to get AI model list
     - Run the notebook to get a list of AI models that have already been imported into Console for AITRIOS, and get settings in the configuration file, **`model_id`**.
  3. Create and edit the configuration file for running notebook to delete AI model
     - Create and edit the configuration file *configuration.json* to configure notebook runtime settings
  4. Run the notebook to delete AI model
     - Run the notebook to delete the AI model from Console for AITRIOS

Import PPL

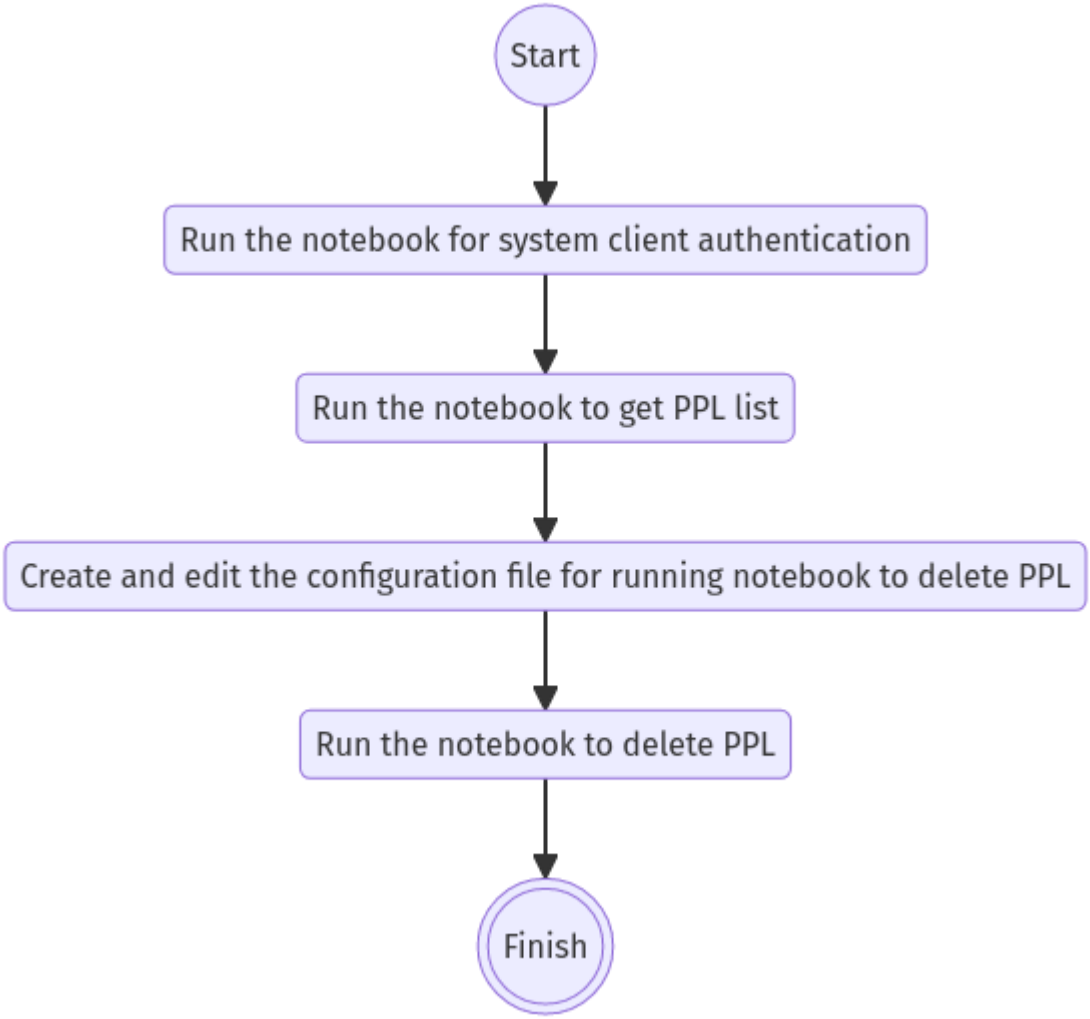
- Flow



- Flow details
  1. Prepare PPL to import
     - Store the PPL to import into the SDK runtime environment
  2. Run the notebook for system client authentication
  3. Run the notebook to get PPL list
     - Run the notebook to get a list of PPL that have already been imported into Console for AITRIOS, and get settings in the configuration file, **`app_name`** and **`version_number`**.
       - Assume the following case
         - You want to check the PPL import status of Console for AITRIOS
  4. Create and edit the configuration file for running notebook to import PPL
     - Create and edit the configuration file *configuration.json* to configure notebook runtime settings
  5. Run the notebook to import PPL
     - Run the notebook with the following features:
       - Encodes PPL in Base64 format
       - Imports PPL into Console for AITRIOS
       - Checks the PPL import status of Console for AITRIOS

Delete PPL

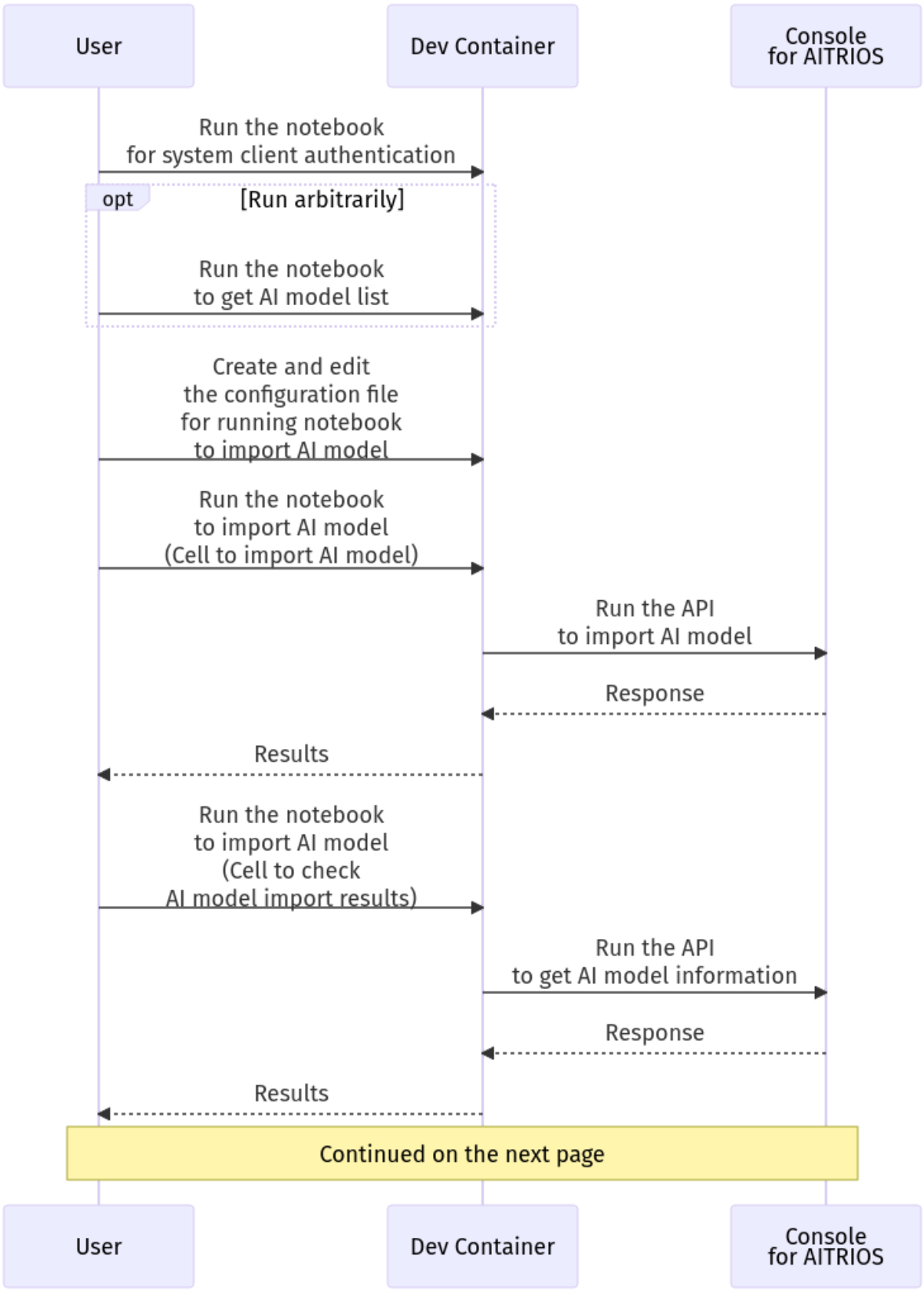
- Flow

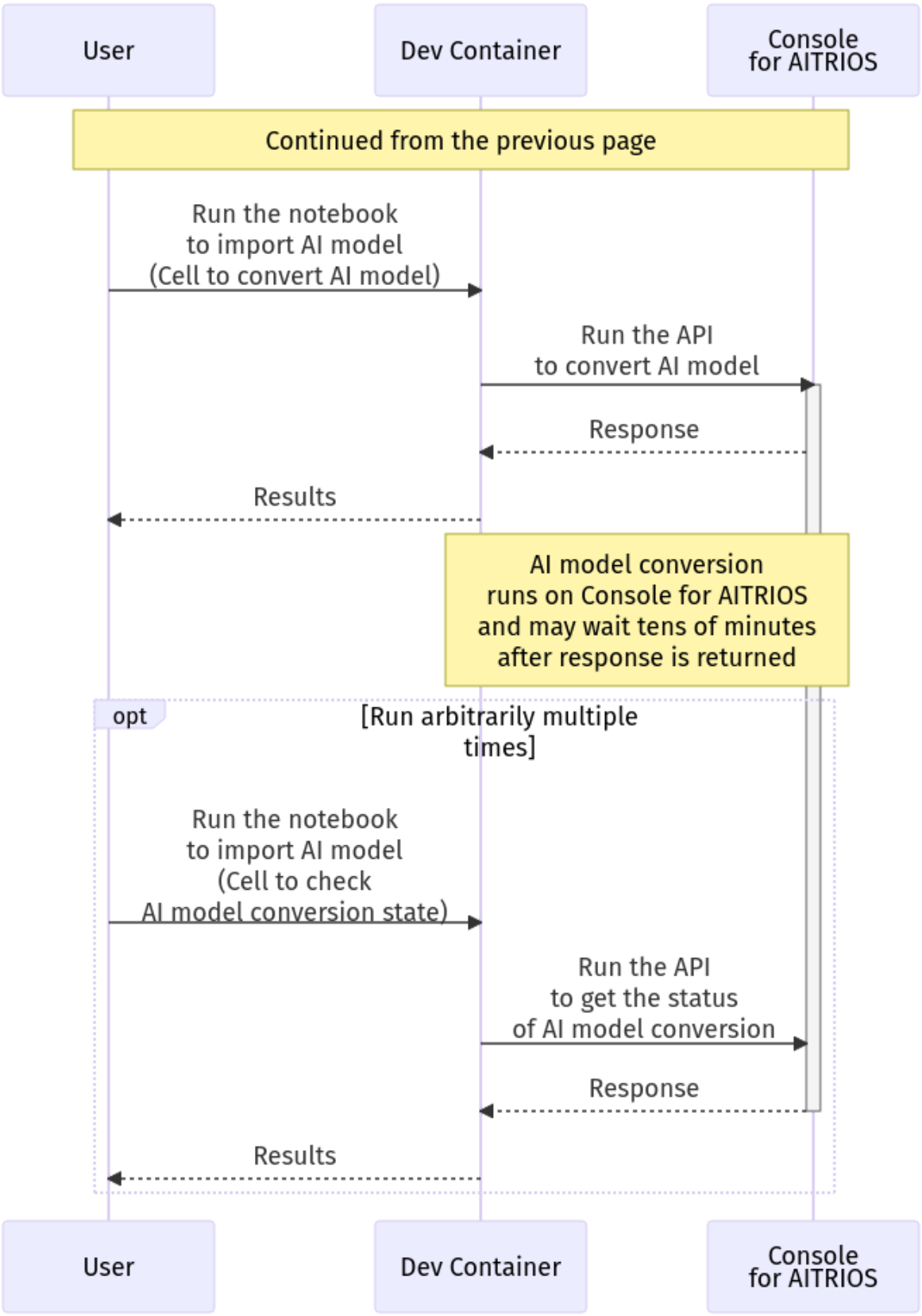


- Flow details
  1. Run the notebook for system client authentication
  2. Run the notebook to get PPL list
     - Run the notebook to get a list of PPL that have already been imported into Console for AITRIOS, and get settings in the configuration file, **`app_name`** and **`version_number`**.
  3. Create and edit the configuration file for running notebook to delete PPL
     - Create and edit the configuration file *configuration.json* to configure notebook runtime settings
  4. Run the notebook to delete PPL
     - Run the notebook to delete the PPL from Console for AITRIOS

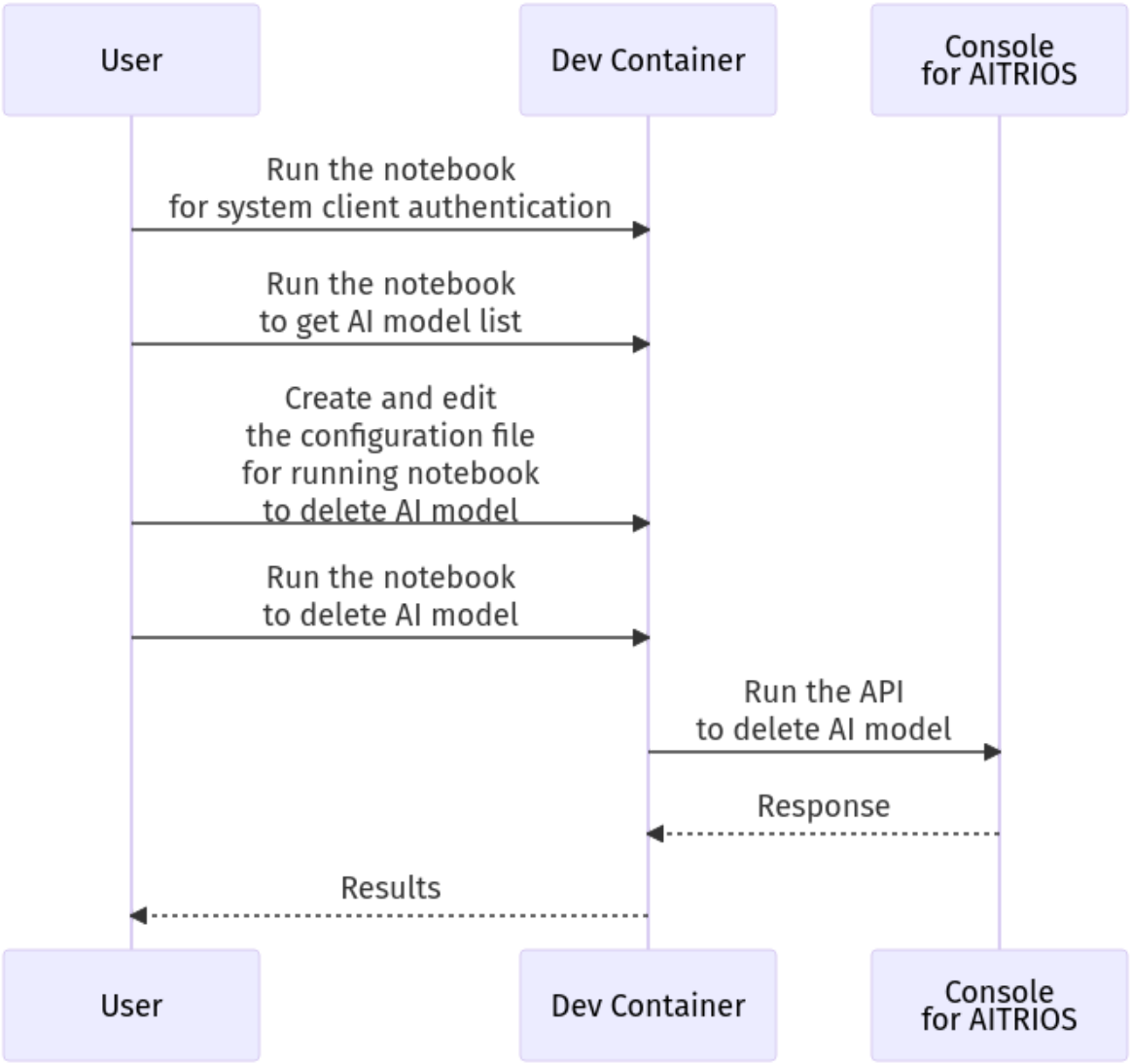
Sequence

Import AI model

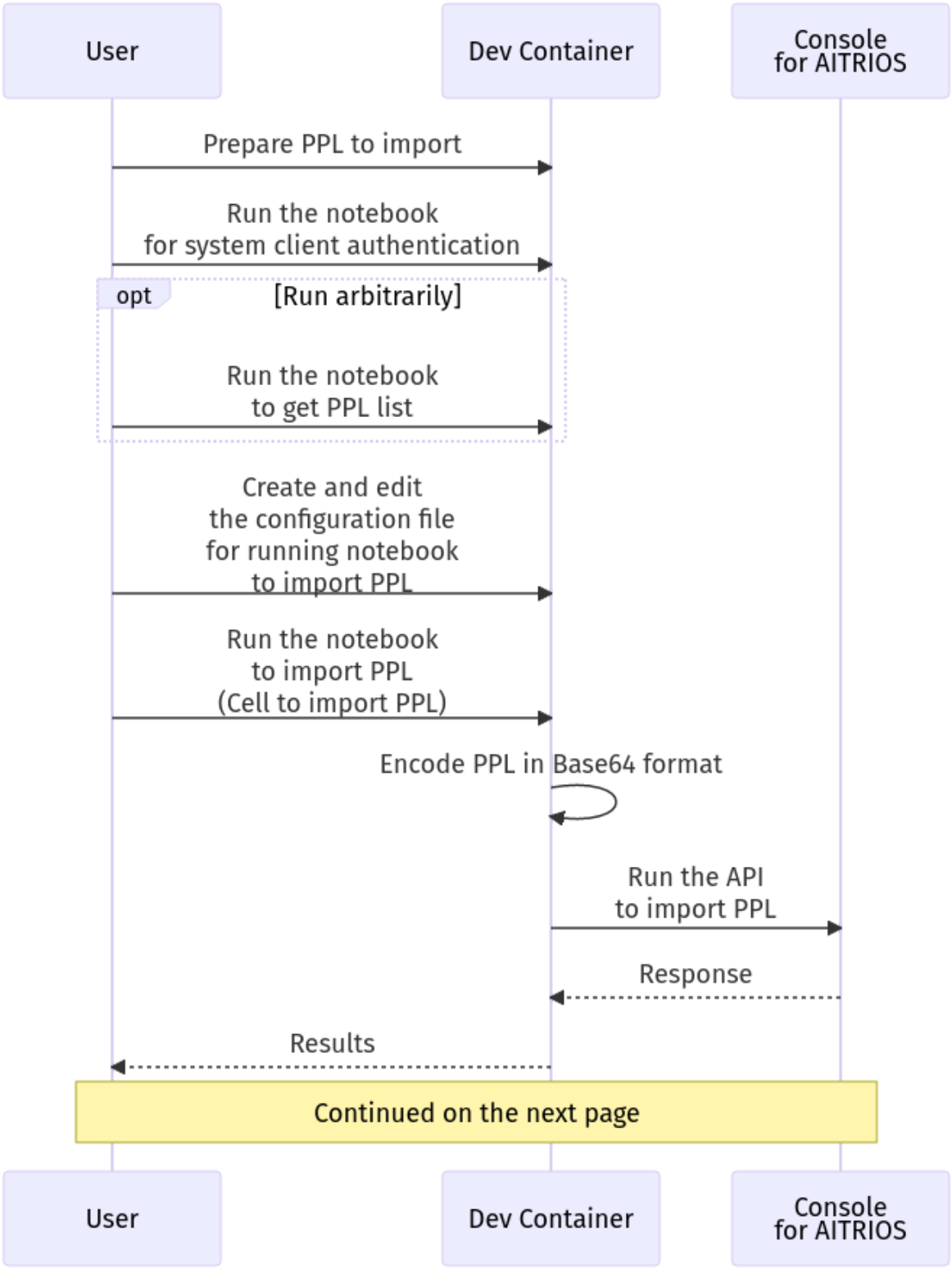


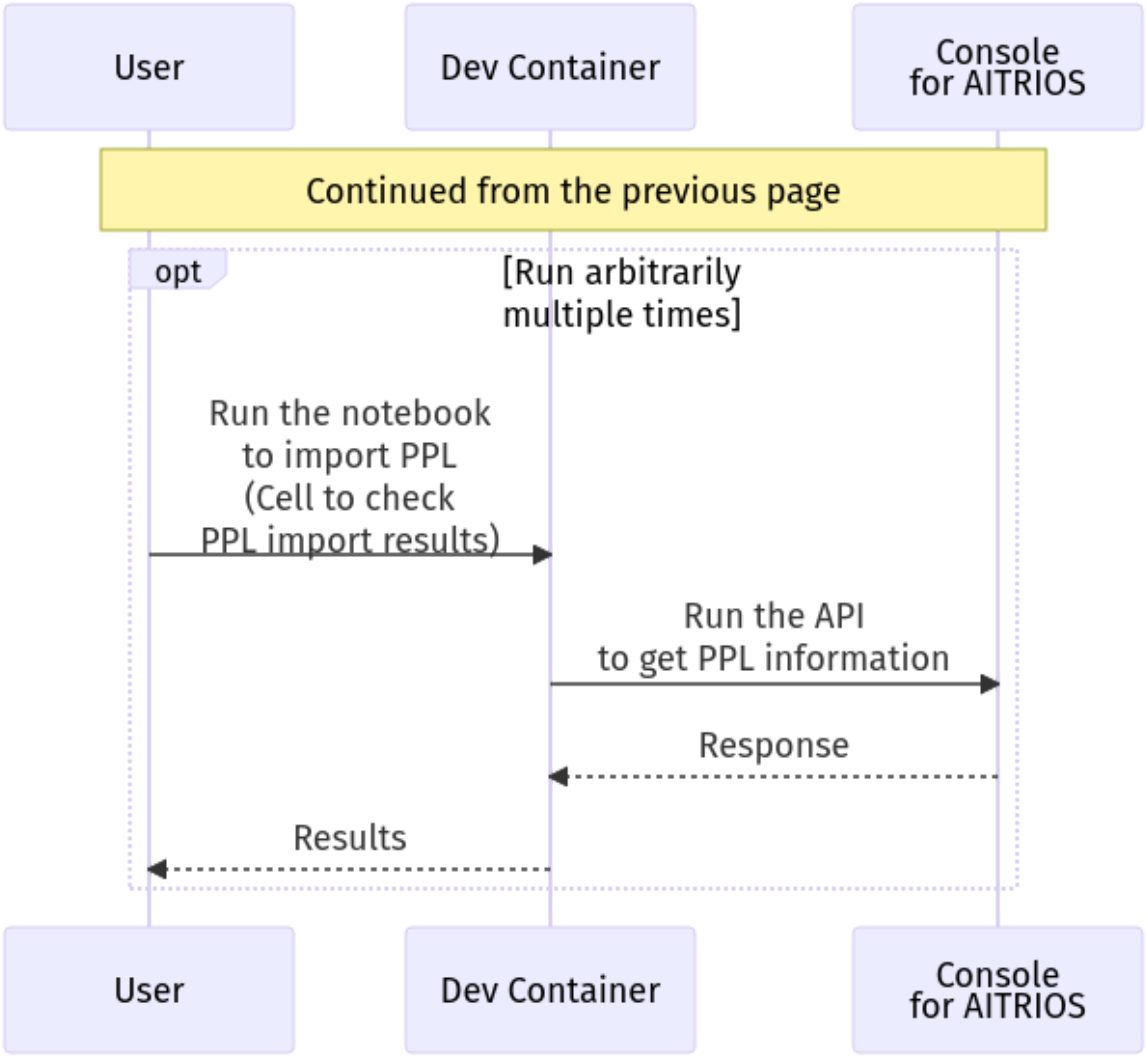

User
Dev Container
Console for AITRIOS
Continued from the previous page
Run the notebook to import AI model (Cell to convert AI model)
Run the API to convert AI model
Response
Results
AI model conversion runs on Console for AITRIOS and may wait tens of minutes after response is returned
opt
[Run arbitrarily multiple times]
Run the notebook to import AI model (Cell to check AI model conversion state)
Run the API to get the status of AI model conversion
Response
Results
User
Dev Container
Console for AITRIOS

Delete AI model

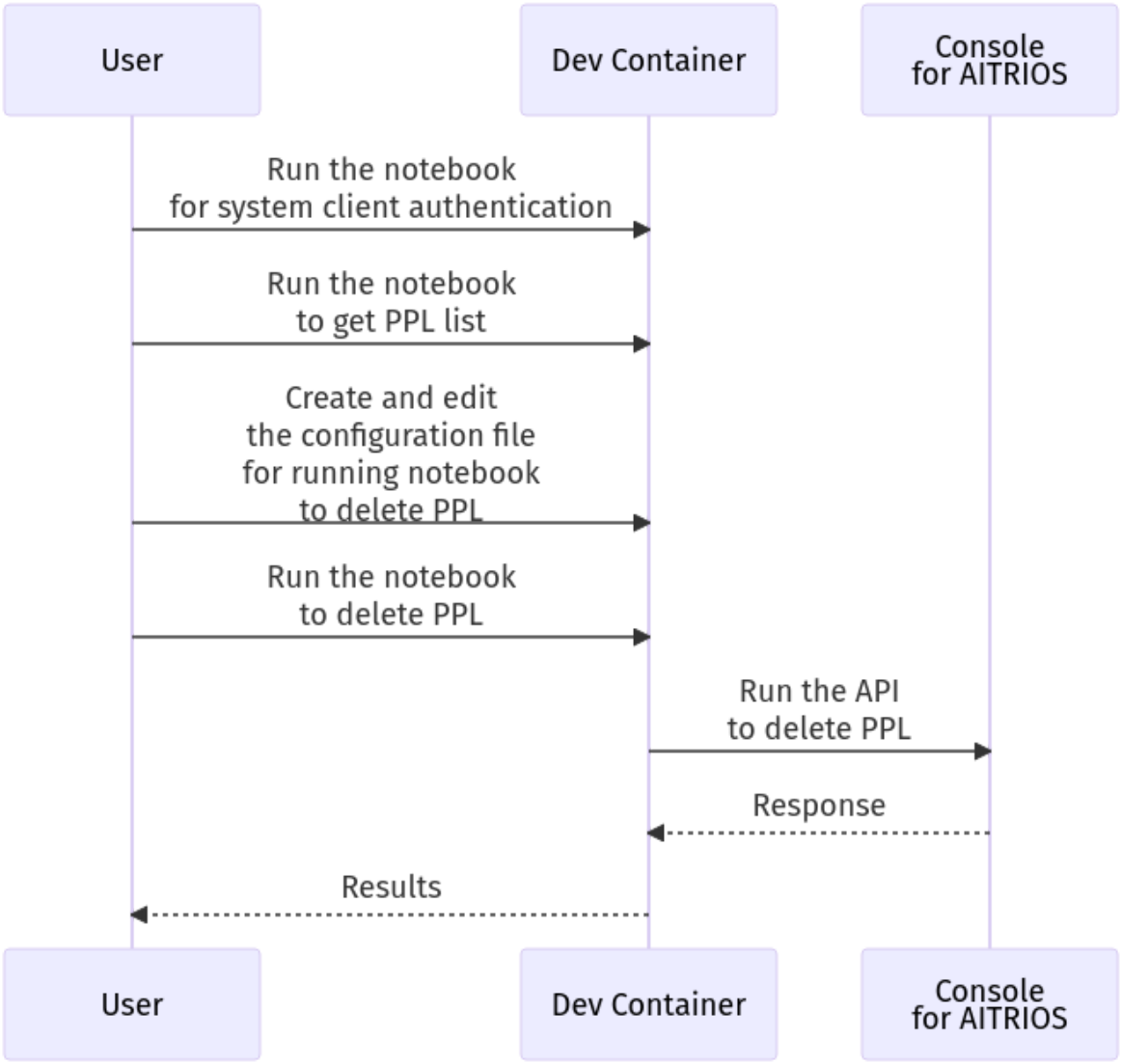


Import PPL


User
Dev Container
Console for AITRIOS
Prepare PPL to import
Run the notebook for system client authentication
opt
[Run arbitrarily]
Run the notebook to get PPL list
Create and edit the configuration file for running notebook to import PPL
Run the notebook to import PPL (Cell to import PPL)
Encode PPL in Base64 format
Run the API to import PPL
Response
Results
Continued on the next page
User
Dev Container
Console for AITRIOS


User
Dev Container
Console for AITRIOS
Continued from the previous page
opt
[Run arbitrarily multiple times]
Run the notebook to import PPL (Cell to check PPL import results)
Run the API to get PPL information
Response
Results
User
Dev Container
Console for AITRIOS

Delete PPL


User
Dev Container
Console for AITRIOS
Run the notebook for system client authentication
Run the notebook to get PPL list
Create and edit the configuration file for running notebook to delete PPL
Run the notebook to delete PPL
Run the API to delete PPL
Response
Results
User
Dev Container
Console for AITRIOS

6. User interface specifications(Import AI model)

Prerequisite

- You have registered as a user through Portal for AITRIOS and participated in the AITRIOS project
- You have prepared an AI model
- You have uploaded an AI model to Azure Blob Storage and gotten its SAS URI

How to start each function

1. Launch the SDK environment and preview the **`README.md`** in the top directory
2. Jump to the **`README.md`** in the **`tutorials`** directory from the hyperlink in the SDK environment top directory
3. Jump to the **`README.md`** in the **`3_prepare_model`** directory from the hyperlink in the **`README.md`** in the **`tutorials`** directory
4. Jump to the **`README.md`** in the **`develop_on_sdk`** directory from the hyperlink in the **`README.md`** in the **`3_prepare_model`** directory
5. Jump to the **`README.md`** in the **`3_import_to_console`** directory from the hyperlink in the **`README.md`** in the **`develop_on_sdk`** directory
6. Jump to each feature from each file in the **`3_import_to_console`** directory

Run the notebook for system client authentication

1. Jump to the **`README.md`** in the **`set_up_console_client`** directory from the hyperlink in the **`README.md`** in the **`3_import_to_console`** directory
2. Open the notebook for system client authentication, **.ipynb*, in the **`set_up_console_client`** directory, and run the python scripts in it

Run the notebook to get AI model list

1. Jump to the **`README.md`** in the **`get_model_list`** directory from the hyperlink in the **`README.md`** in the **`3_import_to_console`** directory
2. Open the notebook to get AI model list, **.ipynb*, in the **`get_model_list`** directory, and run the python scripts in it

Create and edit the configuration file for running notebook to import AI model

All parameters are required, unless otherwise indicated.

The parameters passed to the Console Access Library API are as specified in the *Console Access Library API*.

1. Create and edit the configuration file, **`configuration.json`**, in the execution directory.

Configuration	Meaning	Range	Remarks
`model_id`	ID of AI model to import If it is a new ID, it is newly registered. Upgrade if it is a registered ID.	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`ai_model.ai_model.AIModel.import_base_model`** ・**`ai_model.ai_model.AIModel.get_base_model_status`** ・**`ai_model.ai_model.AIModel.publish_model`**
`model`	SAS URI for AI model to import	SAS URI format Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`ai_model.ai_model.AIModel.import_base_model`**

Configuration	Meaning	Range	Remarks
converted	Option to indicate converted	true or false Details follow the Console Access Library API specification.	Optional If omitted, specify false Used for the following Console Access Library API ・**ai_model.ai_model.AIModel.import_base_model**
vendor_name	Vendor name (specify for new registration)	String Details follow the Console Access Library API specification.	Optional If omitted, no vendor name Used for the following Console Access Library API ・**ai_model.ai_model.AIModel.import_base_model**
comment	AI model and version description AI model and version description for new registrations, Set as description of version when upgrading	String Details follow the Console Access Library API specification.	Optional If omitted, no description Used for the following Console Access Library API ・**ai_model.ai_model.AIModel.import_base_model**

Configuration	Meaning	Range	Remarks
network_type	Network type	String Details follow the Console Access Library API specification.	Optional Specify only for new registration If omitted, specify "1" Used for the following Console Access Library API ・**ai_model.ai_model.AIModel.import_base_model**
labels	Label name For Custom Vision, set the contents of the label.txt file that comes with the AI model file	["label01","label02","label03"] Details follow the Console Access Library API specification.	Optional Used for the following Console Access Library API ・**ai_model.ai_model.AIModel.import_base_model**

Run the notebook to import AI model

1. Open the notebook, **`import_to_console.ipynb`**, in the **`3_import_to_console`** directory, and run the python scripts in it
    - The script does the following:
        - Checks that *configuration.json* exists in the **`3_import_to_console`** directory
            - If an error occurs, the error description is displayed and running is interrupted.
        - Checks the contents of *configuration.json*
            - If an error occurs, the error description is displayed and running is interrupted.
        - Runs the API to import AI model
            - If the import is successful, **`import_to_console.ipynb`** displays a successful message
        - Runs the API to check AI model import results
            - If the AI model information is successfully gotten, **`import_to_console.ipynb`** displays a successful message and the gotten status
        - Runs the API to convert AI model
            - If the API execution is successful, **`import_to_console.ipynb`** displays a successful message
            - It takes several tens of minutes to complete conversion of the AI model, so checks AI model conversion state
        - Runs the API to check AI model conversion state
            - If the conversion status of the AI model information is successfully gotten, **`import_to_console.ipynb`** displays a successful message and the gotten status
    - If an error occurs, the error description is displayed in the **`import_to_console.ipynb`** and running is interrupted.
        - See Cloud SDK Console Access Library(Python) Functional Specifications for details on errors and response times

7. User interface specifications(Delete AI model)

Prerequisite

- You have registered as a user through Portal for AITRIOS and participated in the AITRIOS project
- You have imported AI model into Console for AITRIOS

How to start each function

1. Launch the SDK environment and preview the **`README.md`** in the top directory
2. Jump to the **`README.md`** in the **`tutorials`** directory from the hyperlink in the SDK environment top directory
3. Jump to the **`README.md`** in the **`3_prepare_model`** directory from the hyperlink in the **`README.md`** in the **`tutorials`** directory
4. Jump to the **`README.md`** in the **`develop_on_sdk`** directory from the hyperlink in the **`README.md`** in the **`3_prepare_model`** directory
5. Jump to the **`README.md`** in the **`delete_model_on_console`** directory from the hyperlink in the **`README.md`** in the **`develop_on_sdk`** directory
6. Jump to each feature from each file in the **`delete_model_on_console`** directory

Run the notebook for system client authentication

1. Jump to the **`README.md`** in the **`set_up_console_client`** directory from the hyperlink in the **`README.md`** in the **`delete_model_on_console`** directory
2. Open the notebook for system client authentication, **.ipynb*, in the **`set_up_console_client`** directory, and run the python scripts in it

Run the notebook to get AI model list

1. Jump to the **`README.md`** in the **`get_model_list`** directory from the hyperlink in the **`README.md`** in the **`delete_model_on_console`** directory
2. Open the notebook to get AI model list, **.ipynb*, in the **`get_model_list`** directory, and run the python scripts in it

Create and edit the configuration file for running notebook to delete AI model

All parameters are required, unless otherwise indicated.

The parameters passed to the Console Access Library API are as specified in the *Console Access Library API*.

1. Create and edit the configuration file, **`configuration.json`** , in the execution directory.

Configuration	Meaning	Range	Remarks
`model_id`	ID of AI model to delete	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`ai_model.ai_model.AIModel.delete_model`**

Run the notebook to delete AI model

1. Open the notebook, **`delete_model_on_console.ipynb`**, in the **`delete_model_on_console`** directory, and run the python scripts in it
    - The script does the following:
        - Checks that *configuration.json* exists in the **`delete_model_on_console`** directory
            - If an error occurs, the error description is displayed and running is interrupted.
        - Checks the contents of *configuration.json*
            - If an error occurs, the error description is displayed and running is interrupted.
        - Runs the API to delete AI model
            - If the deletion is successful, **`delete_model_on_console.ipynb`** displays a successful message
    - If an error occurs, the error description is displayed in the **`delete_model_on_console.ipynb`** and running is interrupted.
        - See Cloud SDK Console Access Library(Python) Functional Specifications for details on errors and response times

8. User interface specifications(Import PPL)

Prerequisite

- You have registered as a user through Portal for AITRIOS and participated in the AITRIOS project
- You have prepared PPL

How to start each function

1. Launch the SDK environment and preview the **`README.md`** in the top directory
2. Jump to the **`README.md`** in the **`tutorials`** directory from the hyperlink in the SDK environment top directory
3. Jump to the **`4_prepare_application`** directory from the hyperlink in the **`README.md`** in the **`tutorials`** directory
4. Jump to the **`README.md`** in the **`2_import_to_console`** directory from the hyperlink in the **`README.md`** in the **`4_prepare_application`** directory
5. Jump to each feature from each file in the **`2_import_to_console`** directory

Prepare PPL to import

1. Prepare a PPL to import and store it in any directory

Run the notebook for system client authentication

1. Jump to the **`README.md`** in the **`set_up_console_client`** directory from the hyperlink in the **`README.md`** in the **`2_import_to_console`** directory
2. Open the notebook for system client authentication, **.ipynb*, in the **`set_up_console_client`** directory, and run the python scripts in it

Run the notebook to get PPL list

1. Jump to the **`README.md`** in the **`get_application_list`** directory from the hyperlink in the **`README.md`** in the **`2_import_to_console`** directory
2. Open the notebook to get PPL list, **.ipynb*, in the **`get_application_list`** directory, and run the python scripts in it

Create and edit the configuration file for running notebook to import PPL

All parameters are required, unless otherwise indicated.

Do not use symbolic links to files and directories.

The parameters passed to the Console Access Library API are as specified in the *Console Access Library API*.

1. Create and edit the configuration file, **`configuration.json`**, in the execution directory.

Configuration	Meaning	Range	Remarks
`app_name`	PPL name	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`deployment.deployment.Deployment.import_device_app`**
`version_number`	PPL version	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`deployment.deployment.Deployment.import_device_app`**
`ppl_file`	PPL file path	Absolute path or relative to notebook (*.ipynb)	Don't abbreviate

Configuration	Meaning	Range	Remarks
`comment`	PPL description	String Details follow the Console Access Library API specification.	Optional If omitted, no comment Used for the following Console Access Library API ・ **`deployment.deployment.Deployment.import_device_app`**

Run the notebook to import PPL

1. Open the notebook, **`import_to_console.ipynb`**, in the **`2_import_to_console`** directory, and run the python scripts in it
    - The script does the following:
        - Checks that *configuration.json* exists in the **`2_import_to_console`** directory
            - If an error occurs, the error description is displayed and running is interrupted.
        - Checks the contents of *configuration.json*
            - If an error occurs, the error description is displayed and running is interrupted.
        - Encodes PPL in Base64 format
            - If an error occurs, the error description is displayed and running is interrupted.
        - Runs the API to import PPL
            - If the import is successful, **`import_to_console.ipynb`** displays a successful message
        - Runs the API to check PPL import results
            - If the PPL information is successfully gotten, **`import_to_console.ipynb`** displays a successful message and the gotten status
    - If an error occurs, the error description is displayed in the **`import_to_console.ipynb`** and running is interrupted.
        - See Cloud SDK Console Access Library(Python) Functional Specifications for details on errors and response times

9. User interface specifications(Delete PPL)

Prerequisite

- You have registered as a user through Portal for AITRIOS and participated in the AITRIOS project
- You have imported PPL into Console for AITRIOS

How to start each function

1. Launch the SDK environment and preview the **`README.md`** in the top directory
2. Jump to the **`README.md`** in the **`tutorials`** directory from the hyperlink in the SDK environment top directory
3. Jump to the **`4_prepare_application`** directory from the hyperlink in the **`README.md`** in the **`tutorials`** directory
4. Jump to the **`README.md`** in the **`delete_application_on_console`** directory from the hyperlink in the **`README.md`** in the **`4_prepare_application`** directory
5. Jump to each feature from each file in the **`delete_application_on_console`** directory

Run the notebook for system client authentication

1. Jump to the **`README.md`** in the **`set_up_console_client`** directory from the hyperlink in the **`README.md`** in the **`delete_application_on_console`** directory
2. Open the notebook for system client authentication, **.ipynb*, in the **`set_up_console_client`** directory, and run the python scripts in it

Run the notebook to get PPL list

1. Jump to the **`README.md`** in the **`get_application_list`** directory from the hyperlink in the **`README.md`** in the **`delete_application_on_console`** directory
2. Open the notebook to get PPL list, **.ipynb*, in the **`get_application_list`** directory, and run the python scripts in it

Create and edit the configuration file for running notebook to delete PPL

All parameters are required, unless otherwise indicated.

The parameters passed to the Console Access Library API are as specified in the *Console Access Library API*.

1. Create and edit the configuration file, **`configuration.json`**, in the execution directory.

Configuration	Meaning	Range	Remarks
`app_name`	PPL name	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`deployment.deployment.Deployment.delete_device_app`**
`version_number`	PPL version	String Details follow the Console Access Library API specification.	Don't abbreviate Used for the following Console Access Library API ・**`deployment.deployment.Deployment.delete_device_app`**

Run the notebook to delete PPL

1. Open the notebook, **`delete_application_on_console.ipynb`**, in the **`delete_application_on_console`** directory, and run the python scripts in it
    - The script does the following:
        - Checks that *configuration.json* exists in the **`delete_application_on_console`** directory
            - If an error occurs, the error description is displayed and running is interrupted.
        - Checks the contents of *configuration.json*
            - If an error occurs, the error description is displayed and running is interrupted.
        - Runs the API to delete PPL
            - If the deletion is successful, **`delete_application_on_console.ipynb`** displays a successful message
    - If an error occurs, the error description is displayed in the **`delete_application_on_console.ipynb`** and running is interrupted.
        - See Cloud SDK Console Access Library(Python) Functional Specifications for details on errors and response times

10. Target performances/Impact on performances

- Usability
    - When the SDK environment is built, AI models and PPL can be imported into Console for AITRIOS without any additional installation steps
- UI response time of 1.2 seconds or less
- If processing takes more than 5 seconds, then the display during processing can be updated sequentially

11. Assumption/Restriction

- If you cancel and restart an encoding or import process, start each process from the beginning instead of resuming in the middle

12. Remarks

- None

13. Unconfirmed items

- None