SCP3C06 – Machine Learning: Image Classification

|  |  |
| --- | --- |
| **Product**  SAP oud Platform  **Level**  Undergraduate/Graduate  Beginner  **Focus**  Machine Learning  **Author** Ross Hightower | MOTIVATION  This case illustrates the SAP machine learning image classification API  **PREREQUISITES**  SCP3C01 |
|  | |

# Machine Learning – Image Classification

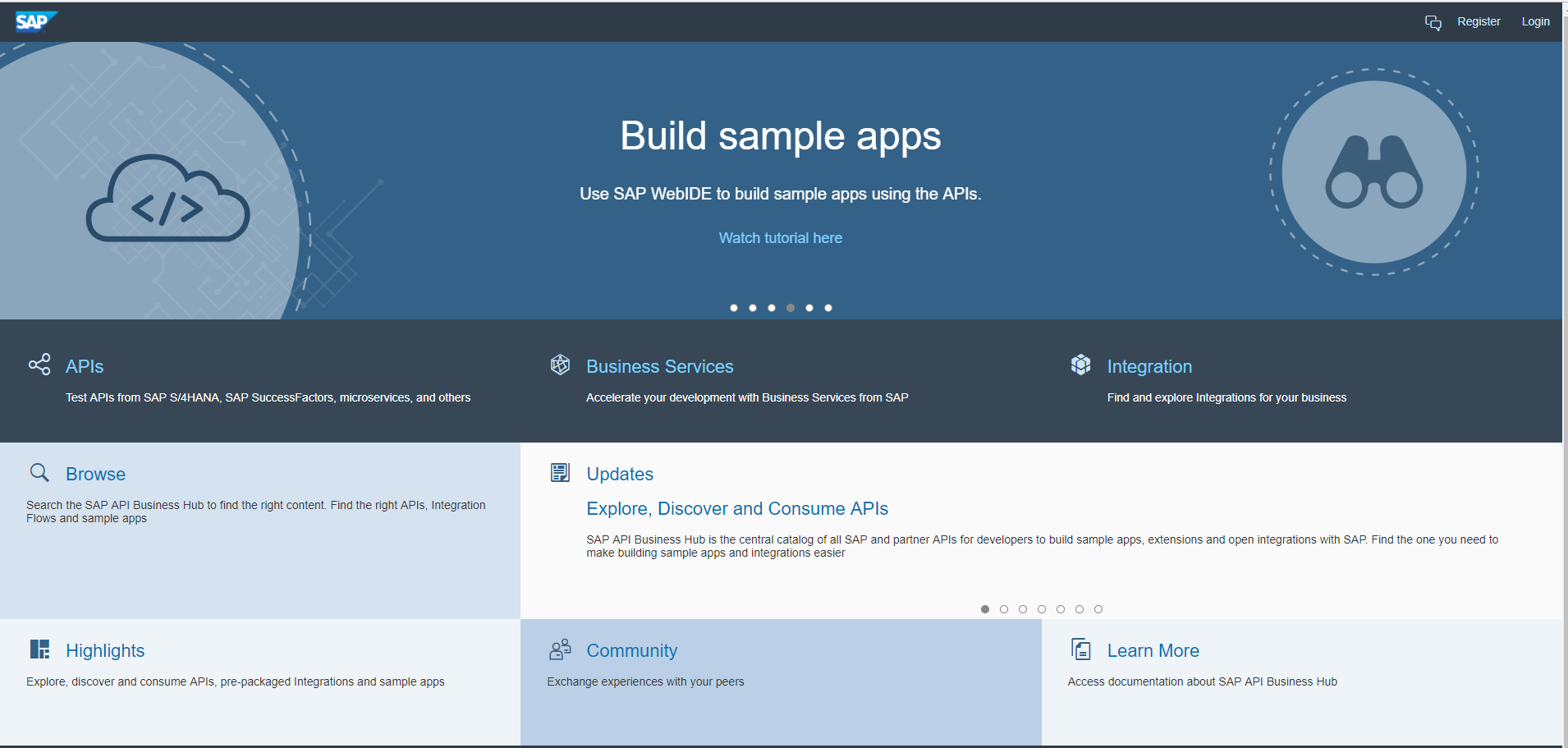
In this case you will implement an SAPUI5 application on the SAP Cloud Platform which uses SAP’s image classification service to classify images.

## Get the Classification Service API Key

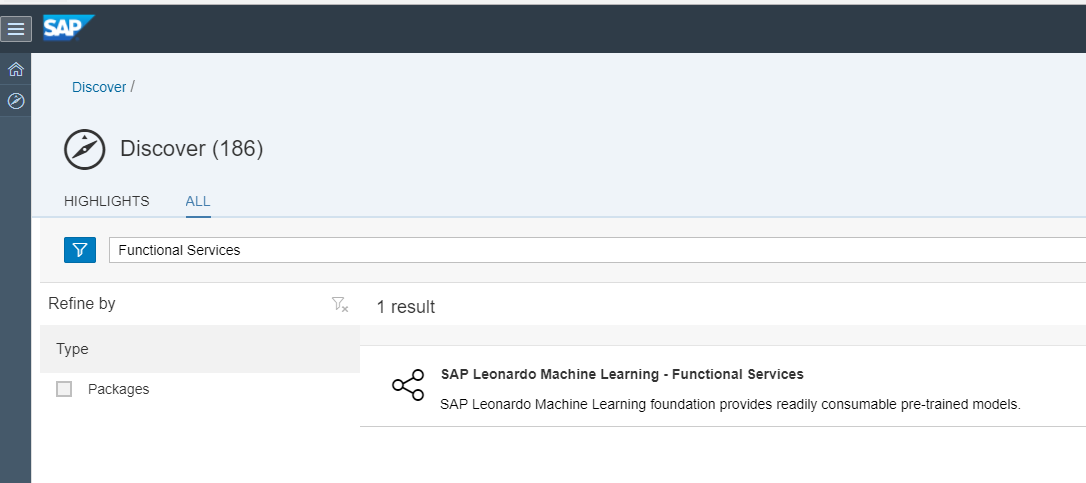
Navigate to <https://api.sap.com/> and click **Login** on the top right. Log in with your SAP Cloud ID credentials.



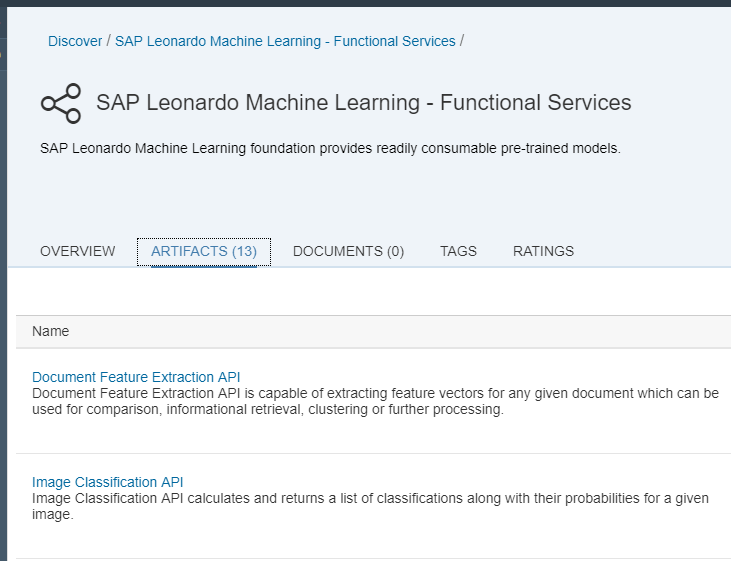
Click on the Browse tile.

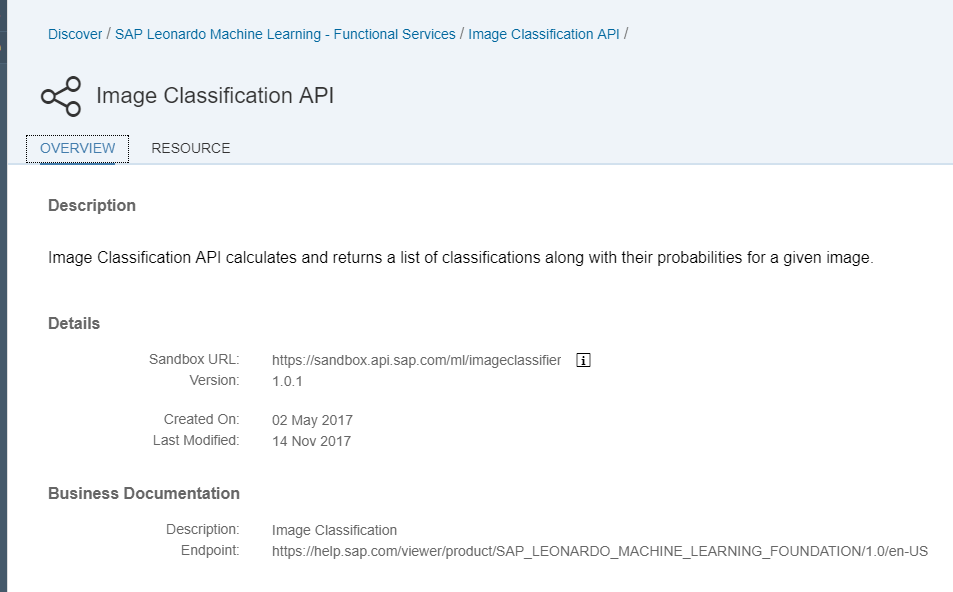


Use the search box to search for **Functional Services. Click the SAP Leonardo Machine Learning – Functional Services tile.**



Then click **Artifacts** and the **Image Classification API** tile.

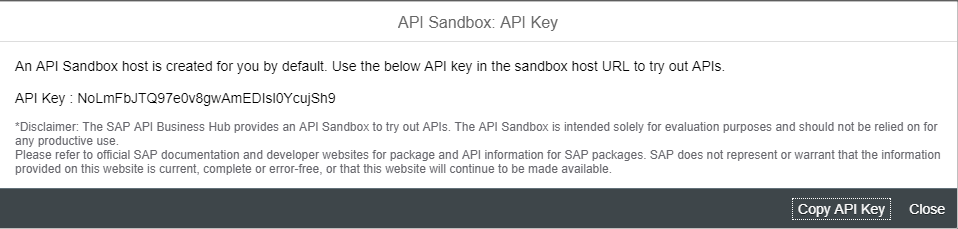




In order to access the service, you need an API Key. Click the **Get API Key** link on the top right.



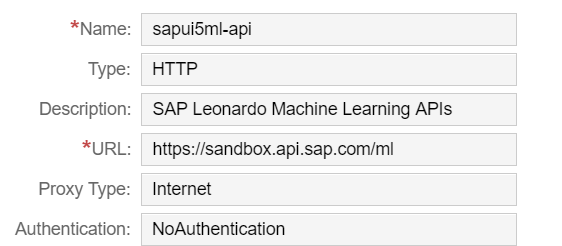
Click **Copy API Key** and paste it somewhere (like a Notepad file).



## Create the Destination

Logon to the SAP Cloud Platform Cockpit and navigate to Connectivity → Destinations using the menu on the left. Click **New Destination** then enter the following information.

|  |  |
| --- | --- |
| Name | Sapui5ml-api |
| Type | HTTP |
| Description | SAP Leonardo Machine Learning APIs |
| URL | https://sandbox.api.sap.com/ml |
| Proxy Type | Internet |
| Authentication | NoAuthentication |



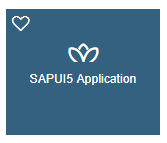
Before clicking save use the **New Property** button to create the following property.



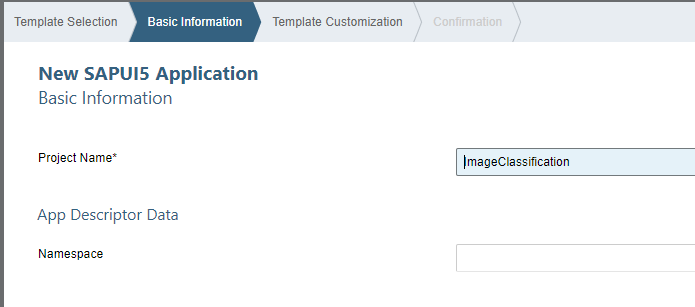
Click **Save.**

## Create the Application

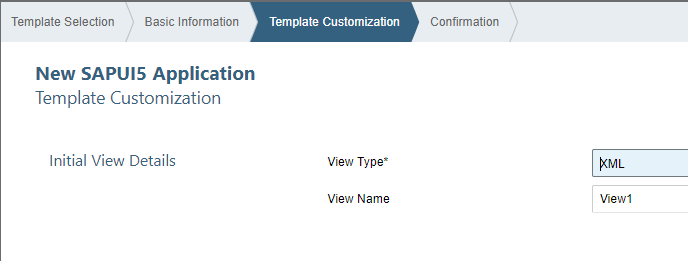
Logon to the Web IDE and create a new project from a template. Select the SAPUI5 Application template and click **Next.**



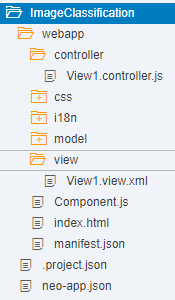
Enter a Project Name and Title and click **Next**.



Leave the View Name and View Type unchanged and click **Finish**.



The project is created.



### Index.html

Add the highlighted piece of code to the index.html file as shown in the image. This identifies the location of the application files.

|  |
| --- |
| <!DOCTYPE HTML>  <html>  <head>  <meta http-equiv="X-UA-Compatible" content="IE=edge" />  <meta charset="UTF-8">  <title>ImageClassifierWorkshop</title>  <script id="sap-ui-bootstrap"  src="../../resources/sap-ui-core.js"  data-sap-ui-libs="sap.m"  data-sap-ui-theme="sap\_belize"  data-sap-ui-compatVersion="edge"  data-sap-ui-resourceroots='{"ImageClassifierWorkshop": ""  **,"sapui5ml":"./"**}'>  </script>  <link rel="stylesheet" type="text/css" href="css/style.css">  <script>  sap.ui.getCore().attachInit(function() {  new sap.m.Shell({  app: new sap.ui.core.ComponentContainer({  height : "100%",  name : "ImageClassifierWorkshop"  })  }).placeAt("content");  });  </script>  </head>  <body class="sapUiBody" id="content">  </body>  </html> |

Listing

### neo-app.json

Next, we have to configure the application to use the destination we created above. Open the neo-app.json file located in the project root folder. Add the highlighted code to the routes array as shown in the image (you can replace the entire contents with this code if you wish). Note the comma between the test-resources route and the new route.

|  |
| --- |
| {  "welcomeFile": "/webapp/index.html",  "routes": [{  "path": "/resources",  "target": {  "type": "service",  "name": "sapui5",  "entryPath": "/resources"  },  "description": "SAPUI5 Resources"  }, {  "path": "/test-resources",  "target": {  "type": "service",  "name": "sapui5",  "entryPath": "/test-resources"  },  "description": "SAPUI5 Test Resources"  }**, {**  **"path": "/ml",**  **"target": {**  **"type": "destination",**  **"name": "sapui5ml-api"**  **},**  **"description": "ML API destination"**  **}**  ],  "sendWelcomeFileRedirect": true**,**  **"headerWhiteList": [**  **"APIKey"**  **]**  } |

Listing 2



### Image-api.json

Create a new file in the model folder called **image-api.json** and paste the code below into it. Replace <<<<< COPY YOUR API KEY >>>>> with the API Key you saved above. The url value is added to the end of the url from the destination to construct the url for the image classification service.

|  |
| --- |
| {  "url" : "/ml/imageclassifier/inference\_sync",  "APIKey":"<<<<< COPY YOUR API KEY >>>>>"  } |

Listing 3

### manifest.json

The manifest.json file contains meta data for the application including data sources for the application models. Open the manifest.json file located in the webapp folder. If necessary, click the Code Editor tab at the bottom of the editor. Find the models section and add the following code. Use the image to see how to insert the code. Note the comma between the new code and the existing code.

|  |
| --- |
| "images": {  "type": "sap.ui.model.json.JSONModel",  "preload": true,  "uri": "model/image-api.json"  } |

Listing 4



### View1.view.xml

Open the View1.view.xml file located in the view folder and replace the code with the code shown below.

|  |
| --- |
| See the code in the accompanying text file |

Listing 5

### View1.controller.js

Replace the code in the View1.controller.js file, located in the controller folder, with the code in Listing 5.

|  |
| --- |
| See the code in the accompanying text file |

Listing 6

Now run the app by selecting the project folder and clicking the Run icon (). Once the app starts, replace extended\_runnable.file in the url with index and hit enter.



Use the button to select a file. There are limits to the size of the file you can use. The machine learning algorithm will attempt to classify the image.

