**Tutorial : Creating Custom Agents and Tools with Watsonx Orchestrate**

[**sumkbose@sg.ibm.com**](mailto:sumkbose@sg.ibm.com) **and** [**david.gunawan@ibm.com**](mailto:david.gunawan@ibm.com)

**The following sections uses the Flask framework (I have another alternate framework below with the FastAPI that is simpler)**

**Pre-Requisites: Install** the latest version of python, pip

**Step-1** : Create a file watsonxHello2.py and the following lines of code: **This is your core python code** (let’s suppose this function would have the business logic later)

def hello\_world():

return ( "Message": "app up and running successfully")

**Step-2 : Convert this python function (encapsulating your business logic) into web service** by creating a web application. To create the web application use the Flask framework.

To do this add the following lines (highlighted in green) and change the previous code (highlighted in yellow) to :

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

@app.route("/")

def hello\_world():

return jsonify({

"Message": "app up and running successfully"

})

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

* From the command prompt execute python watsonxHello2.py
* Open any Browser and type <http://127.0.0.1:5000/> to view the message. If you see this application and the corresponding services are running on the localhost.

**Step-3 : Add the swagger / OpenAPI code for service specification**

Create aswagger.jsonfile and place it in a new folder called/staticunder the root.

{

    "swagger": "2.0",

    "info": {

      "title": "Access API",

      "version": "1.0.0"

    },

    "paths": {

      "/": {

        "get": {

          "description": "Returns message showing app is up",

          "produces": [

            "application/json"

          ],

          "responses": {

            "200": {

              "description": "Successful operation"

            }

          }

        }

      },

      "/access": {

        "post": {

          "description": "Grants access to an user",

          "consumes": [

            "application/json"

          ],

          "produces": [

            "application/json"

          ],

          "parameters": [

            {

              "name": "access data",

              "in": "body",

              "description": "JSON data for the api",

              "required": true,

              "schema": {

                "$ref": "#/definitions/AccessData"

              }

            }

          ],

          "responses": {

            "200": {

              "description": "User granted access"

            },

            "400": {

              "description": "Invalid request data"

            }

          }

        }

      }

    },

    "definitions": {

      "AccessData": {

        "type": "object",

        "properties": {

          "name": {

            "type": "string"

          },

          "server": {

            "type": "string"

          }

        }

      }

    }

  }

Then add the following lines of code in the watsonxHello2.py (highlighted in green)

from flasgger import Swagger, LazyString, LazyJSONEncoder

from flasgger import swag\_from

from flask\_swagger\_ui import get\_swaggerui\_blueprint

app = Flask(\_\_name\_\_)

app.json\_encoder = LazyJSONEncoder

SWAGGER\_URL="/swagger"

API\_URL="/static/swagger.json"

swagger\_ui\_blueprint = get\_swaggerui\_blueprint(

SWAGGER\_URL,

API\_URL,

config={

'app\_name': 'Access API'

}

)

app.register\_blueprint(swagger\_ui\_blueprint, url\_prefix=SWAGGER\_URL)

@app.route("/")

def hello\_world():

return jsonify({

"Message": "app up and running successfully"

})

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

* From the command prompt execute python watsonxHello2.py
* Open any Browser and type http://127.0.0.1:5000/ to view the message. If you see this application and the corresponding services are running on the localhost.
* Open any Browser and type http://127.0.0.1:5000/swagger/ to view the swagger UI and service specification.
* You see that the swagger.json has two service definitions : 1) “/” and 2) “/access” . While we implemented our logic for the first definition but we did not implement our logic for the second definition.
* Test this using http://127.0.0.1:5000/ and http://127.0.0.1:5000/access
* Since no corresponding implementation in the watsonxHello2.py exists for the /access service, the http://127.0.0.1:5000/access fails.
* So what you need to understand is that the .json file lists the API definitions. The implementation for these corresponding definitions are in the python file’s functions.

**Step-3 :** All the services are now running under localhost. So these services cannot be accessed by the external world. In order to make these services accessible to the external world do the following :

* From the command prompt execute ipconfig. Note down the ip-address of your machine.

As an example, for my machine :

A screenshot of a computer

AI-generated content may be incorrect.

Change the following code to (marked in yellow) to :

if \_\_name\_\_ == '\_\_main\_\_':

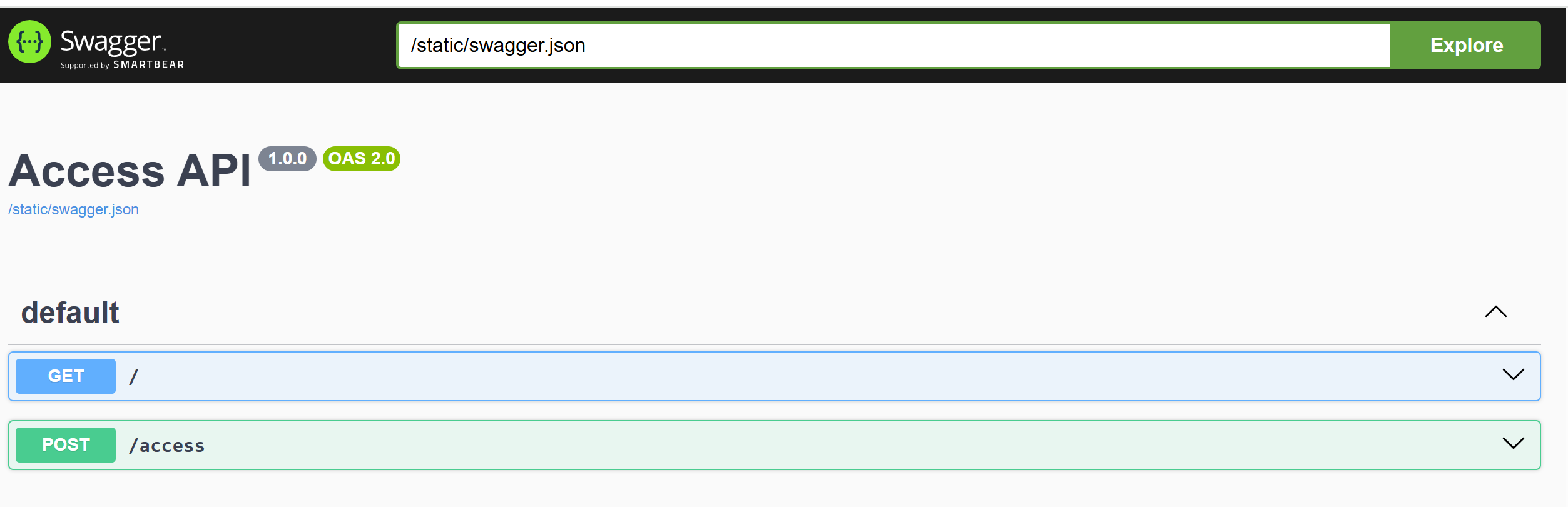
app.run(host="192.168.10.105",port=5000)

* Test this using
  + http://192.168.10.105:5000/

(Should generate the message “app up and running successfully”)

* + http://192.168.10.105:5000/swagger

(Should generate the following: )

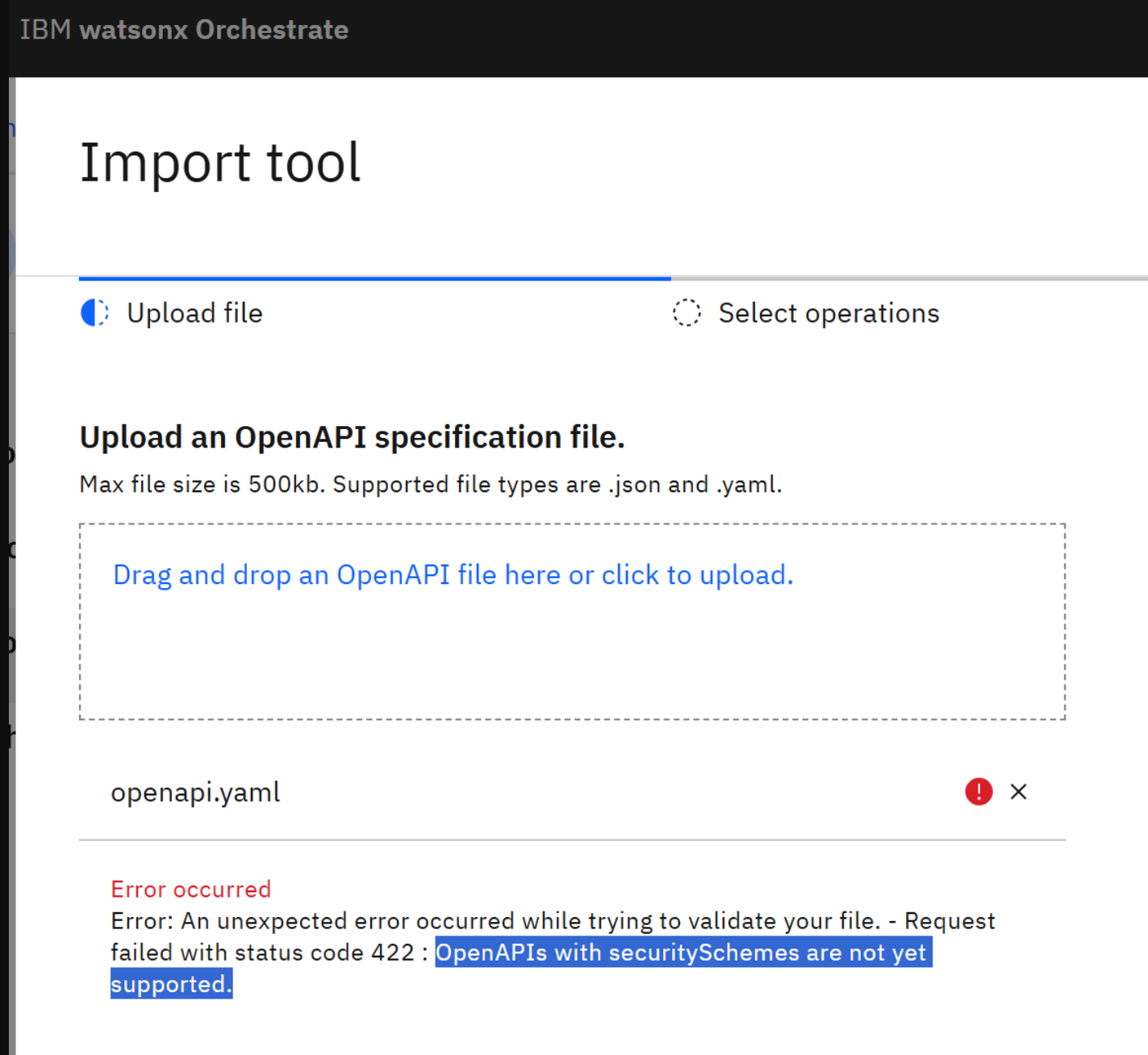


* + <http://192.168.10.105:5000/access>

(Should fail)

**This means that the service can be accessible by other computers on the network.**

**Step-4 :** Import this swagger.json (this is the OpenAPI file) in the watsonx orchestrate tool while building your agent.



**Remark** : OpenAPI file for making reservations for booking.com can be found at : <https://developers.booking.com/_spec/metasearch/connect-api/open-api.yaml?download#:~:text=items:%20type:%20integer%20description:,be%20a%20multiple%20of%20100>

**The following sections uses the FastAPI framework to the above. This is a lot simpler**