**Displaying an SVG in APL with python**

SVG images aren’t supported by Alexa APL, only jpg and png are.

We’ll download an SVG image, convert the SVG to PNG, save this in an S3 bucket (the image source for APL has to be https), and display the PNG. We’ll get the SVG from <https://www.yr.no/en> a weather station prediction for London from;

<https://www.yr.no/en/content/2-2643743/meteogram.svg>

**Converting SVG to PNG**

This can be done using cairosvg (<https://cairosvg.org/>)

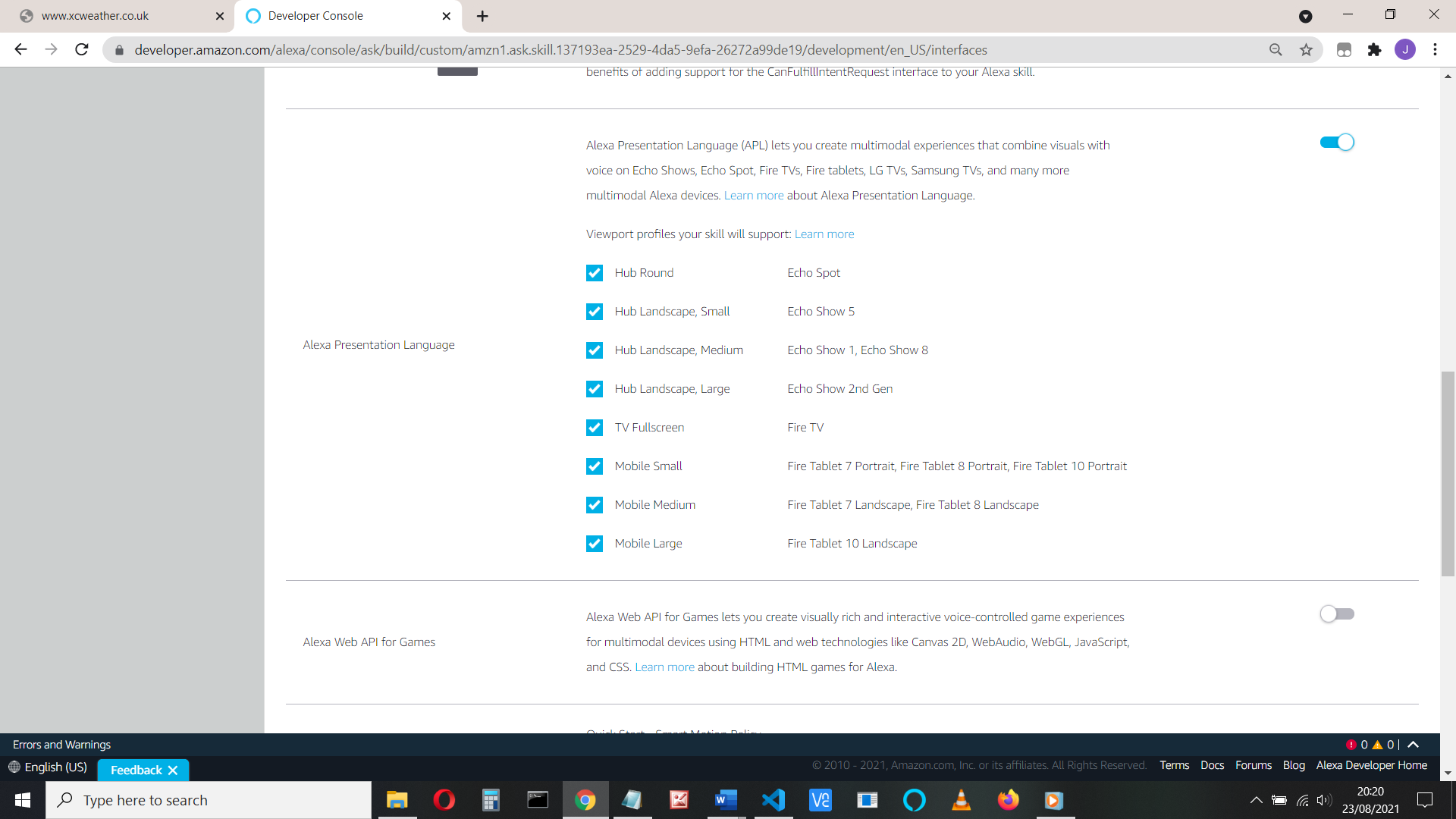
**Instructions**

Start by creating an Alexa Hosted skill, I’ve called it **‘Display Vector Graphics’**, use custom model, Alexa Hosted (Python) and your region

Select Start from Scratch and Create the skill

Check the invocation name, change to **‘Display Vector Graphics’**

Click Interfaces and select the APL:



Save the model and build it.

Open the requirements.txt file, add these lines to and then save it:

**CairoSVG==2.5.2**

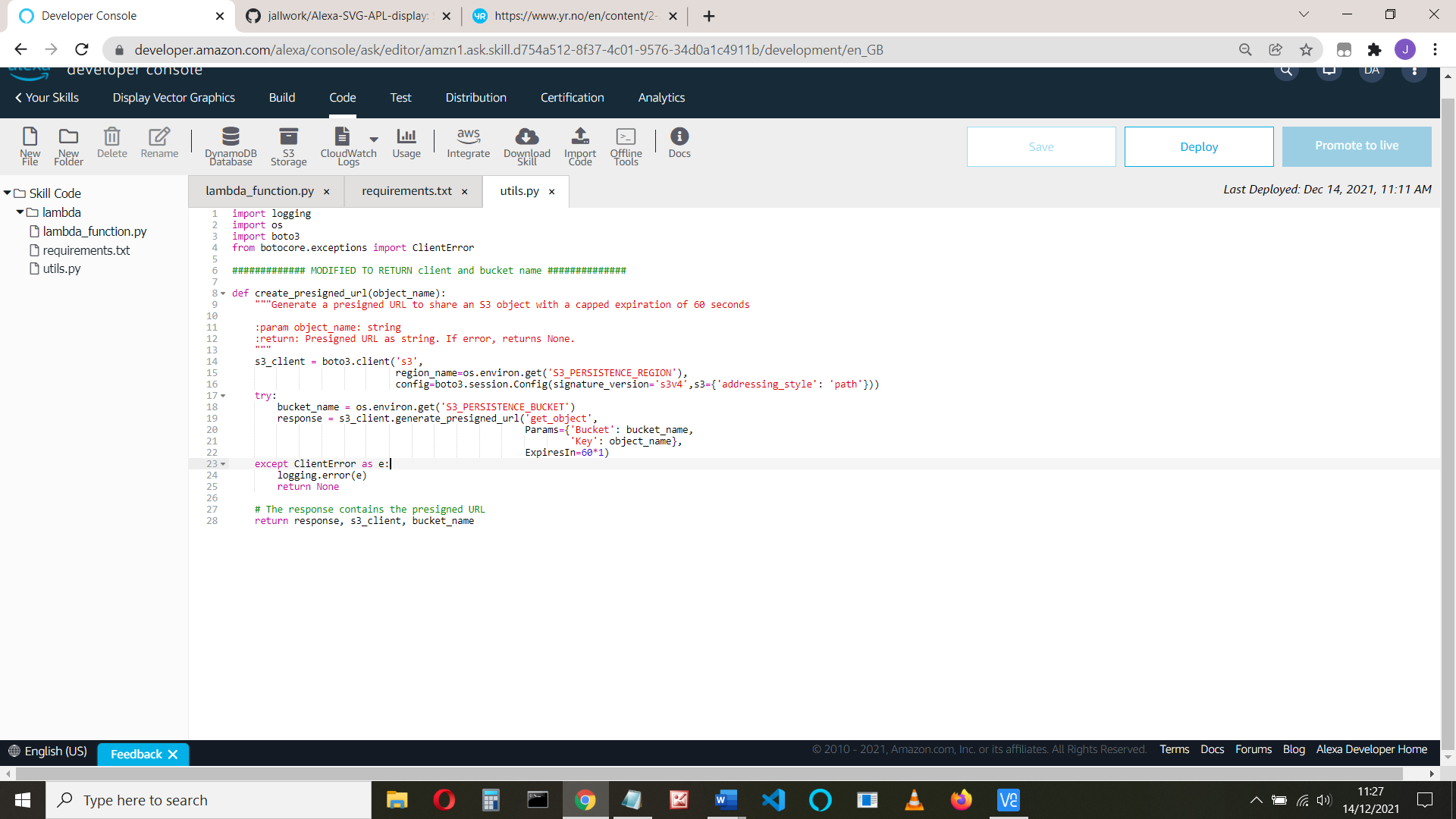
**boto3 ==1.20.24**

We’ll use this to convert our SVG to PNG

To make life easier I modify create\_presigned\_url in utils.py:

Click utils.py tab and change the return line of code to:

**return response, s3\_client, bucket\_name**



This returns the bucket name and s3\_client information that we need later

Save the utils.py file

Open the lambda\_function.py code.

Add the following imports and JSON code at the top of the code:

from ask\_sdk\_model.interfaces.alexa.presentation.apl import (

**RenderDocumentDirective**, ExecuteCommandsDirective, SpeakItemCommand, AutoPageCommand, HighlightMode)

**import urllib.request**

**import cairosvg**

from utils import create\_presigned\_url

from urllib.request import urlopen

mainjson = {

"type": "APL",

"version": "1.7",

"license": "Copyright 2021 Amazon.com, Inc. or its affiliates blah blah",

"settings": {},

"theme": "dark",

"import": [],

"resources": [],

"styles": {},

"onMount": [],

"graphics": {},

"commands": {},

"layouts": {},

"mainTemplate": {

"parameters": [

"payload"

],

"items": [

{

"type": "Container",

"items": [

{

"source": "imageurl",

"type": "Image",

"width": "100vw",

"height": "100vh"

}

]

}

]

}

}

The JSON file is for the APL display and just displays a single image.

We’ll change the url for the image using:

mainjson["mainTemplate"]["items"][0]["items"][0]["source"] = image\_url

We’ll use urlopen to read our SVG file from a web site and save it in out temporary folder (/tmp), then convert it to PNG and save that in an S3 bucket.

in the Launch request code add

# Retrieve SVG file and saving locally

urllib.request.urlretrieve("https://www.yr.no/en/content/2-2643743/meteogram.svg","/tmp/meteogram.svg")

# Convert file

PNGfilename = "/tmp/meteogram.png"

cairosvg.svg2png(url="/tmp/meteogram.svg", write\_to= PNGfilename)

# save in S3 bucket

# now get presigned url and save #### create\_presigned\_url HAS BEEN MODIFIED

object\_name = "S3meteogram.png"

meteo\_url, s3\_client, bucket\_name = create\_presigned\_url(object\_name)

# and presigned url returns s3\_client and bucket\_name used here:

s3\_client.upload\_file(Filename= PNGfilename, Bucket=bucket\_name, Key=object\_name)

#Now it’s been uploaded to S3, we can use that as our url for the APL

mainjson["mainTemplate"]["items"][0]["items"][0]["source"] = meteo\_url

# and change the return so that it uses APL:

return (

handler\_input.response\_builder

.speak(speak\_output)

.add\_directive(

RenderDocumentDirective(

token="pagerToken",

document=mainjson

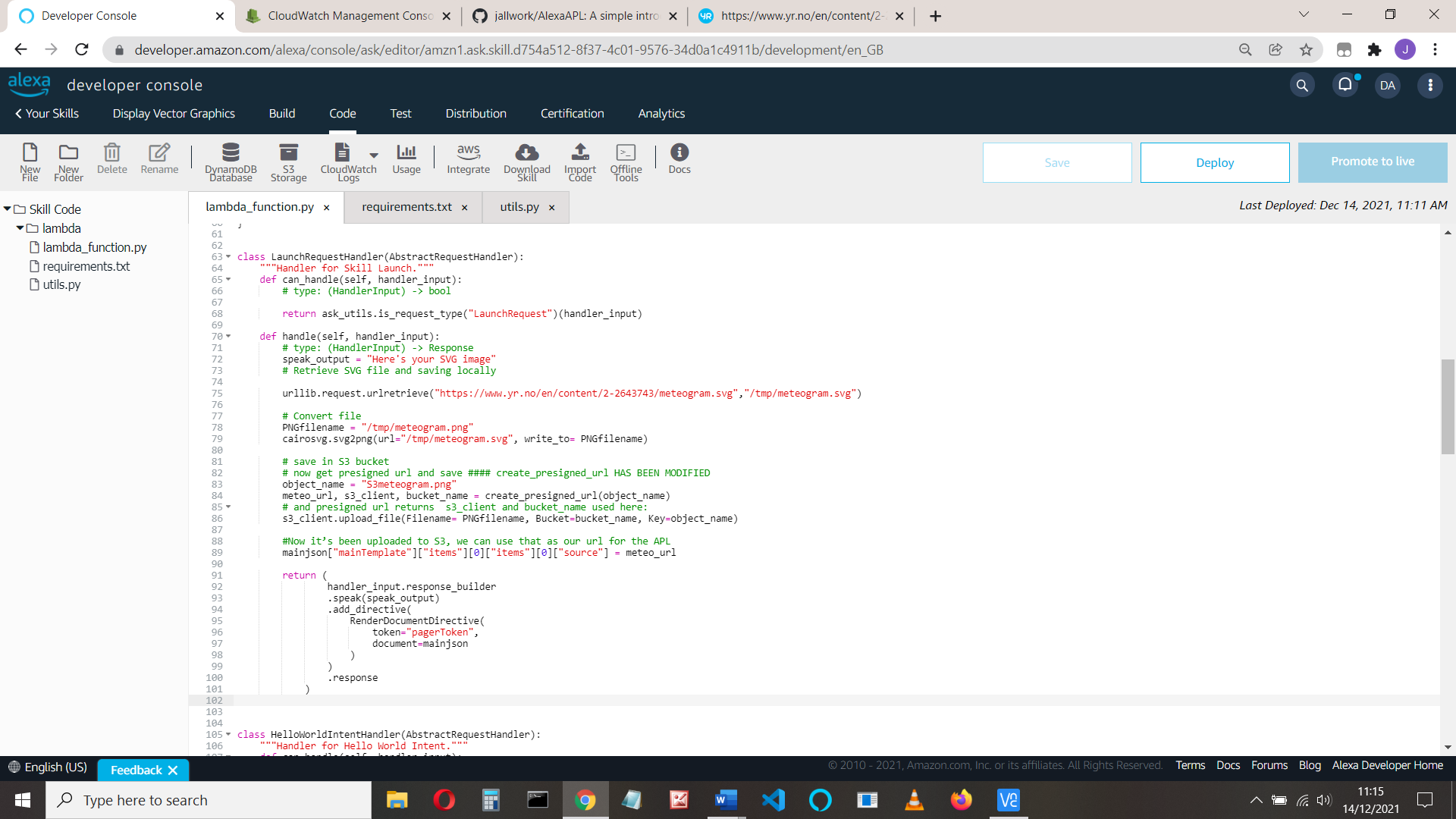
)

)

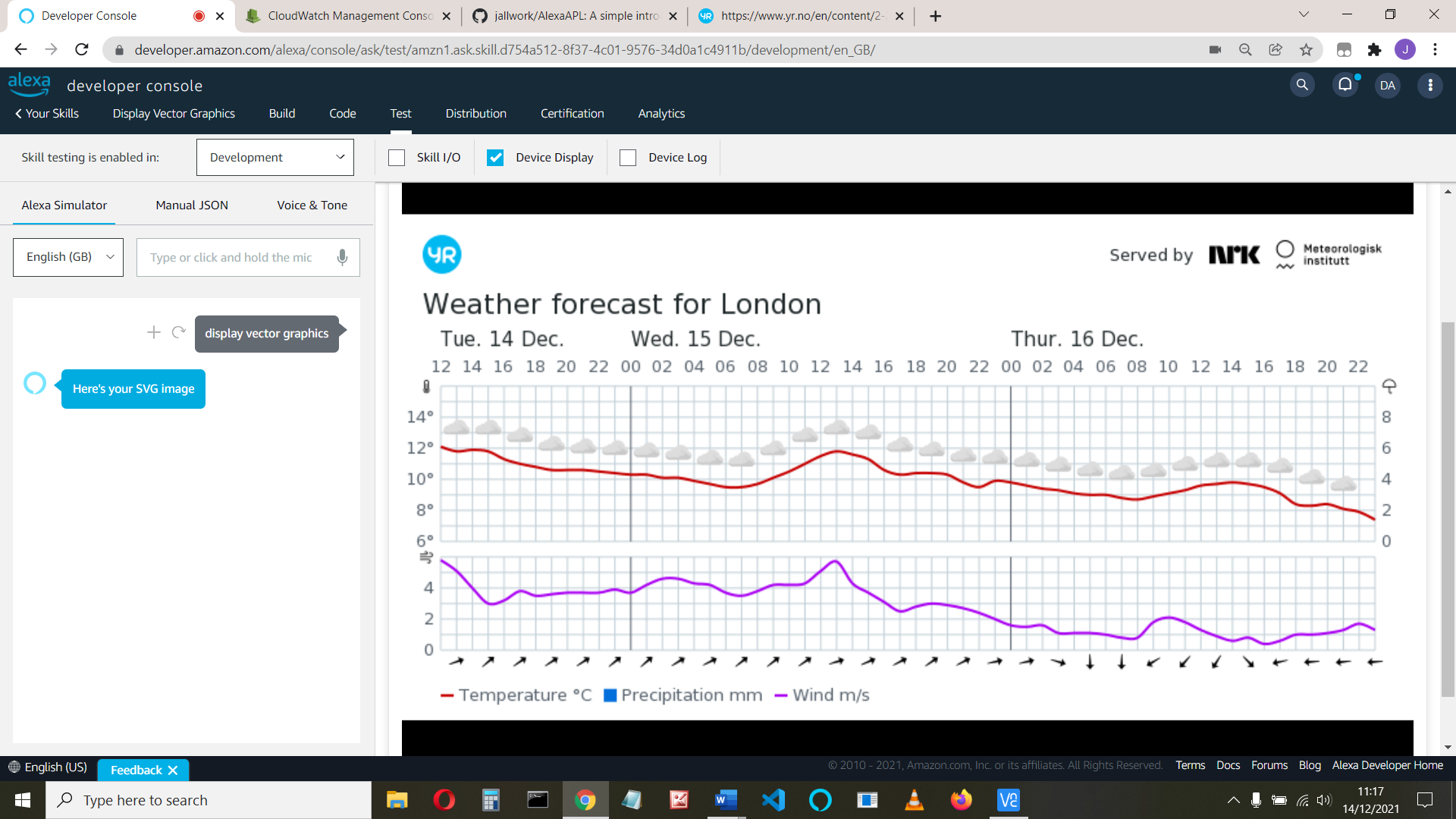
.response

)

The full Launch request is now:



Save, deploy, enable testing and test your code: enable development and invoke your skill:



References:

https://stackoverflow.com/questions/67308958/convert-a-svg-to-png-using-python