**Convert your Raspberry Pi into an Alexa device using Alexa Voice Service SDK**

Back in 2017 , my company was working on a product that embedded Amazon Alexa Voice Service (AVS) on device and my team was involved in the onboarding solution for the device.

In order to better understand the use cases, I installed Amazon Alexa Voice Service SDK on a Raspberry Pi .

Here is a video of a short interaction with Alexa Voice Service installed on Raspberry Pi

<https://youtu.be/bajws_5RN8M>

If you are interested in creating an Alexa device on Raspberry pi please follow the latest instructions using this link

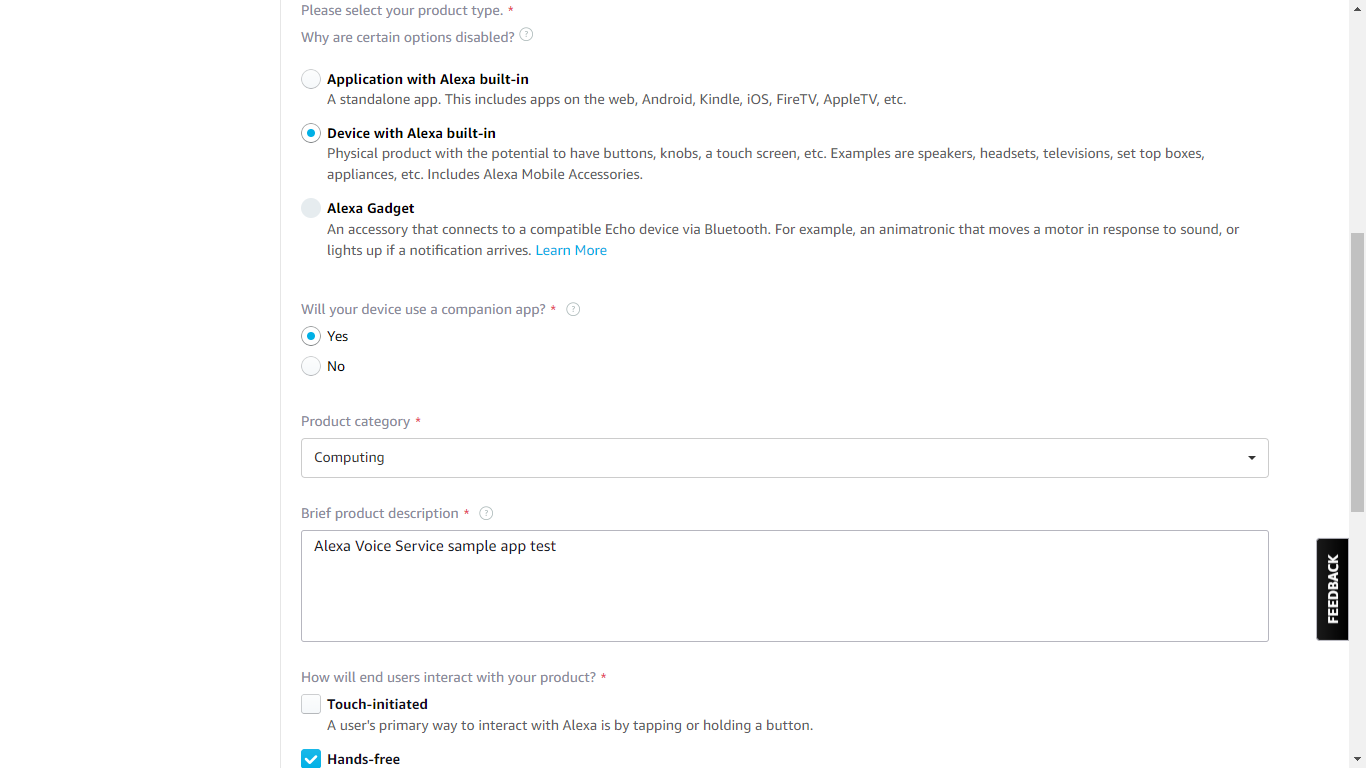
<https://developer.amazon.com/en-US/docs/alexa/avs-device-sdk/raspberry-pi.html>

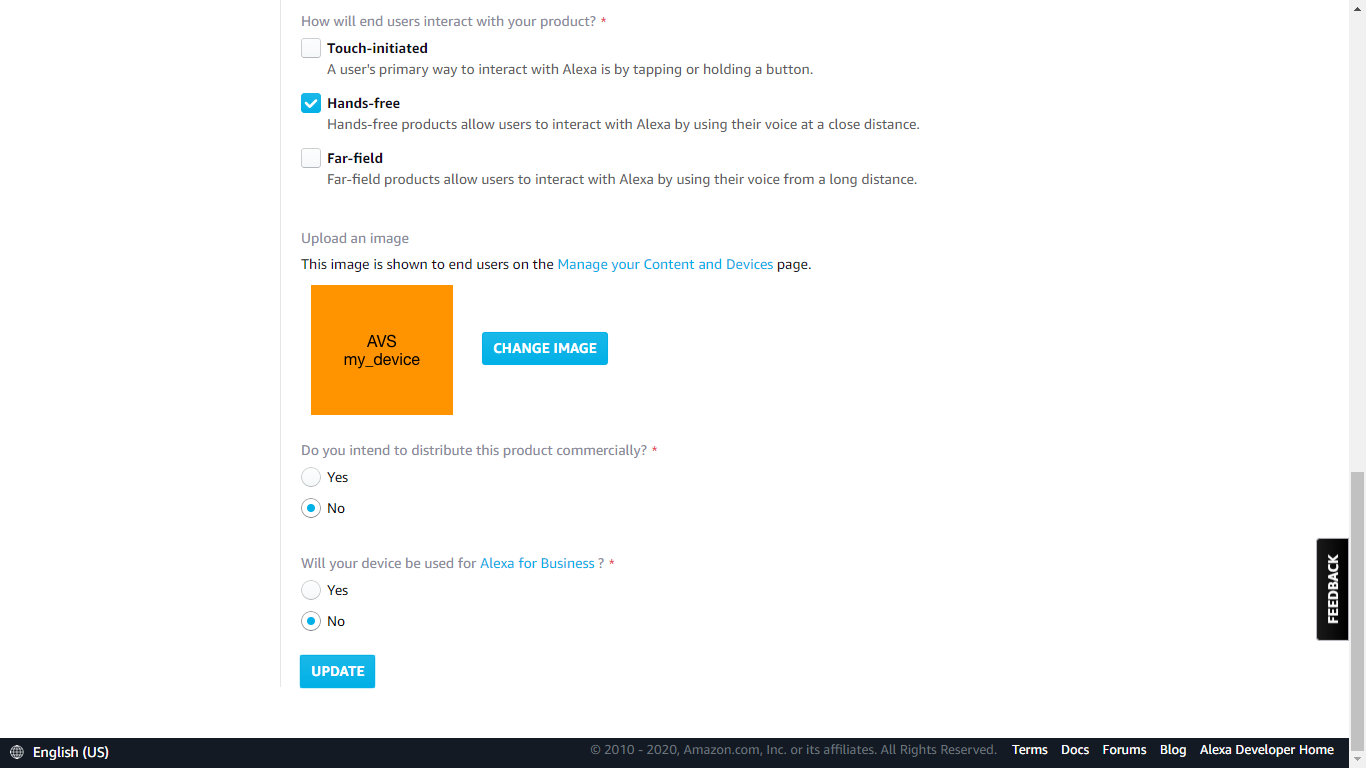
**Provisioning an AVS device under your Amazon Account**

If you are distributing such custom devices, you need to provide a way for the user to onboard Alexa under their Amazon account. I built a POC for this provisioning flow using Nodejs (and also a Java version). Here are the steps involved.

1. Create a developer account under Amazon if you don’t already have one.
2. Browse to <https://developer.amazon.com/alexa/console/avs/products>
3. Click on Add New Product
4. Fill up the information regarding your device. Give it a Product Name and Product ID

Here are the other options I chose for my device

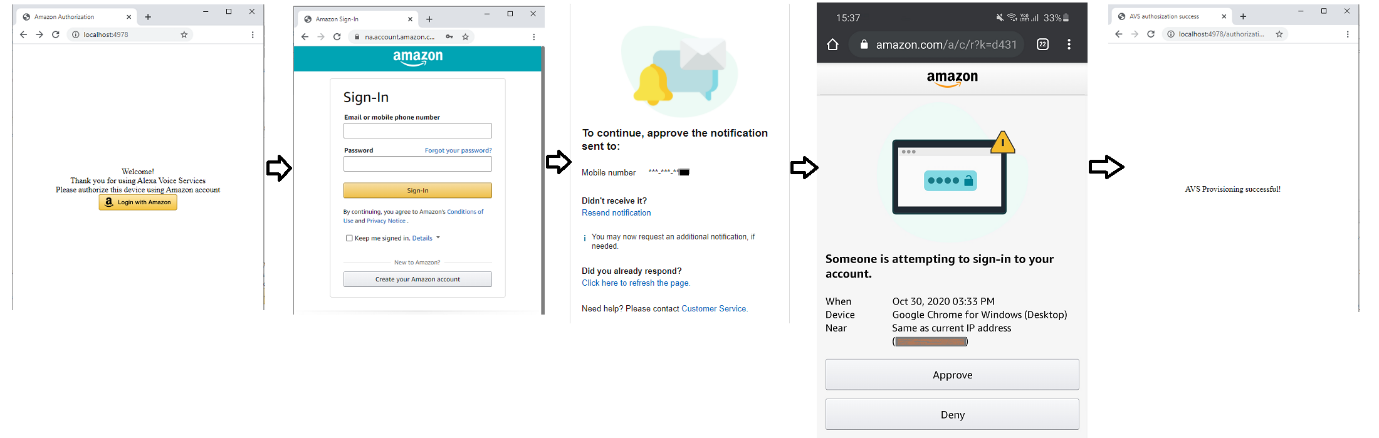




1. Save the product details
2. Choose Security Profile and select create new profile .
3. Fill up the option for security profile name and description and click Next .
4. The rest of the information like Profile ID, Client ID and Client secret are now automatically filled up for you .
5. I kept the options to just Web based flow but you may choose to add Android and iOS app integration (as we did in the final product)
6. Fill up the allowed origin and callback URL details. Callback URl is the address of the page to which Amazon authorization service will redirect to with the result of User authorization . This is the service or page that needs to handle the authorization response.

**AVS Device Provisioning App flow**

This is how the flow looks like



The code for this App flow can be found under this repository <https://github.com/ravindrabharathi/AVS-provisioning-nodejs> .

Although I have not updated the code for recent version of dependency libraries, the code still works as the screenshots above are recent. I’ll follow up with another post explaining the code structure and important parts that handle authorization, provisioning of device, etc