PHASE 4

**Configure Hardware Components**

In Phase 3, we discussed the assembly of the RF circuit and the installation of software dependencies. Now, let's move on to the next steps.

**Provision IBM Cloud Services**

To implement voice controls for the serverless home automation hub, you'll need to provision several IBM Cloud services. These services will help you with natural language processing, IoT communication, and more.

**Watson Assistant:**

This service will be used to analyze natural language and determine user intent based on their input. You can find the Watson Assistant service in the IBM Cloud catalog.

**Speech to Text:**

The Speech to Text service is essential for transcribing audio input from your voice commands into text that can be processed by the system.

**Watson IoT Platform:** The Watson IoT Platform acts as the MQTT messaging broker for communication between devices and your Raspberry Pi.

**Twilio:**

Twilio is a service that enables you to integrate VoIP and SMS capabilities into your platform, allowing you to receive voice and text commands.

You will need an IBM Cloud account to provision these services. After logging in, navigate to each of the links above, and select the "Create Service" button for each service.

**Create Serverless Functions**

IBM Cloud Functions is a serverless framework that allows you to run code automatically in response to various events. In this implementation, IBM Cloud Functions actions will communicate with your Raspberry Pi via MQTT messages.

**Create Actions:**

Start by navigating to the ibm\_cloud\_functions directory.

Use the following commands to create two actions, one for transcribing audio and another for publishing MQTT messages:

cd serverless-home-automation/ibm\_cloud\_functions

wsk action create conversation conversation.js

wsk action create iot-pub iot-pub.py

**Set Default Service Credentials:**

To make it easier to call these actions, you can set default service credentials for each action. This eliminates the need to pass in the service credentials every time you want to call the Watson services.

To obtain the service credentials for the IBM Watson Assistant, Twilio, and IBM IoT Platform services, click each provisioned service in the IBM Cloud dashboard and select the "View credentials" option.

Depending on whether your Watson Assistant service uses username/password or IAM-based credentials, use one of the following sets of commands to update your actions with the appropriate credentials:

**For username/password-based credentials:**

wsk action update conversation -p username ${conversation\_username} -p password ${conversation\_password} -p workspace\_id ${conversation\_workspace\_id}

wsk action update iot-pub -p iot\_org\_id ${iot\_org\_id} -p device\_id ${device\_id} -p api\_token ${api\_token} -p device\_type ${device\_type}

wsk package bind /whisk.system/watson-speechToText myWatsonSpeechToText -p username ${stt\_username} -p password ${stt\_password}

**For IAM-based credentials:**

wsk action update conversation -p iamApiKey ${apikey} -p workspace\_id ${conversation\_workspace\_id}

wsk action update iot-pub -p iot\_org\_id ${iot\_org\_id} -p device\_id ${device\_id} -p api\_token ${api\_token} -p device\_type ${device\_type}

wsk package bind /whisk.system/watson-speechToText myWatsonSpeechToText -p username ${stt\_username} -p password ${stt\_password}

**Create a Sequence:**

You can arrange these actions into a sequence to streamline the process. Use the following command to create a sequence that consists of three actions: transcribing audio, analyzing the text result, and publishing MQTT messages.

wsk action create homeSequence --sequence myWatsonSpeechToText/speechToText,conversation,iot-pub

**Configure Node Server on Raspberry Pi**

A Node.js server on your Raspberry Pi will listen for MQTT messages and control the RF outlets. To set this up, run the following commands on your Raspberry Pi:

sudo cp serverless-home-automation/iot-gateway/node-mqtt.service /etc/systemd/system/

sudo systemctl enable node-mqtt

sudo systemctl start node-mqtt

sudo systemctl status node-mqtt

This will create a systemd service that will automatically start on boot and run the node server, which subscribes to the Watson IoT Platform's MQTT broker and listens for intent entity pairs to control your RF outlets.