



Publisher homepage: www.universepg.com, ISSN: 2663-7804 (Online) & 2663-7790 (Print)

<https://doi.org/10.34104/ajeit.020.054065>

Australian Journal of Engineering and Innovative Technology

Journal homepage: www.universepg.com/journal/ajeit



Development of Smart Librarian with the Virtual Assistant (PRIMO)

Al Mahmud Al Mamun^{1*}, Tahmina Islam², Md. Masrur Sobhan Siam¹, and Md. Enamul Kabir³

¹Dept. of Computer Science and Engineering, Prime University, Bangladesh; ²Dept. of Electrical & Electronics Engineering, World University of Bangladesh, Dhaka, Bangladesh; and ³Dept. of Engineering, Confidence Power Rongpur Limited, Rongpur, Bangladesh.

*Correspondence: almamun.1mail@gmail.com (Al Mahmud Al Mamun, Dept. of Computer Science and Engineering, Prime University, Dhaka, Bangladesh).

ABSTRACT

In modern life, with the ability to perform tasks, the virtual assistant (VA) can make our lives easier and smart. The virtual assistant can perform as a librarian, very smartly, and effectively. We build our VA with Raspberry Pi and Alexa Voice Service. As a result, few discussions that occur in library environments such as find books, short review books, university notice are accurately performed. The common way of communication used by people in day to day life is through speech. If the assistant system can heard to the customer for the handle of the day to day affairs, then grant the right reply, it will be much simple for customers to transmit with their assistant system, and the assistant will be much better "Smart" as a personal assistant. We heard a very old story "Ali Baba and the Forty Thieves", where the mouth of a treasure cave secured by magic. It unrolls on the words "unroll sesame" and seals itself on the words "near sesame". The magic is a VA in the modern world. The VA system built on artificial intelligence (AI), machine learning, natural language processing, and voice recognition technology.

Keywords: Artificial intelligence, Machine learning, Voice recognition, Raspberry Pi, and Alexa voice service.

INTRODUCTION:

An intelligent virtual assistant (IVA) is a program agent that can execute tasks or assistances for a single based on rebuke commands. Few VAs are expert to interpret personal speech and quickly respond via unify voices. Customers can ask their assistants' questions, regulate home automation tools, media recount via voice, and oversee other key tasks such as email, to-do lists, calendars (Hoy MB, 2018) and many other things with verbal commands (Hoy, Matthew, 2018). An alike concept, however with differences, lays under the dialogue systems (Klüwer, Tina, 2011). In the library, this is difficult and time consuming for the human librarian to tell the specific location of any

UniversePG | www.universepg.com

books and short review of any book. But a librarian using a VA system can perform those tasks very smartly, accurately and effectively within less time. With the rise of highly autonomous systems concerns in the field of human factors are focusing on designing appropriate tools to address this new class of technology (Alam and Alam, 2020; Hancock, 2017).

The smart librarian should help the users to find any book location and short review of any book in the library. When users ask for a book location, the librarian will provide the bookshelf number, and column. When users ask for a book review, the librarian will provide the author name and a short

review of the book. This project is based on the VA system built on AI, machine learning, NLP, and voice recognition technology (Suykens, 2014). This assistant system help and service to assist users by its listening, thinking and speaking ability for any users questions or voice command with specific tasks, including- find a book location, short review of a book, university notice, class routine, examination schedule, Semester final result, and others effective information. In Maciel *et al.* (2014) was showed the design, development and assessment of an animated personal assistant with synthetic voice to support the online learning, integrated with LMS Moodle, called Avatar Education Plugin.

The modern definition of AI is "the analysis and design of intelligent components" where an intelligent component perceives its environment and grabs actions which overlay chances of success. ML is a practice of AI that handovers systems the capacity to automatically learn and develop from experience without being exactly programmed. ML shows on the expansion of computer courses that can examine data and use it to learn for themselves. Natural Language Processing (NLP) is an area of AI that shows with the interaction in the side of computers and personals using the natural language. The ultimate key objective of NLP is to read, decipher, understand, and create sense of the personal languages in a manner that is significant. Most NLP approaches rely on ML to obtain meaning from personal languages.

Voice recognition (speaker recognition) is a technique of computing technology that creates specialized software and systems for voice recognition, stand-alone and authentication of individual speakers. IVA or Intelligent Personal Assistant (IPA) is a software agent that can perform tasks or services for a person based on verbal orders. Some VAs is able to interpret personal speech and respond through synthetic voices. Users can ask questions of their assistants, regulate home automation tools and media playback via voice, and manage other basic tasks, such as calendars with emails, to-do lists, and verbal commands.

Literature Review:

The VA system built on AI, Machine Learning, Natural Language Processing, and Voice Recognition UniversePG | www.universepg.com

technology. NLP is an area of AI that deals with the interaction between computers and personal using natural language. The ultimate goal of the NLP is to read, decipher, understand and comprehend human languages in a valuable way. Most NLP strategies rely on ML to extract meaning from human language. Voice recognition evaluates the voice biometrics of an individual, such as the frequency and flow of their voice and their natural accent.

In 1769, Wolfgang von Kempelen began development his speaking machine is a manually operated speech synthesizer. Radio Rex was the first voice-activated toy released in 1911. In 1952, Three Bell Labs researchers, Stephen Balashek, R. Biddulph, and K. H. Davis built a system called "Audrey" for single speaker digit recognition. Gunnar Fant advanced and published the origin-filter model of speech generation in 1960. Another early tool that enabled digital speech recognition was the IBM Show box, which was unveiled to the general public at the launch of the market at the early 1917 Satel World Fair. This primary computer was developed in 1981, about 20 years before the first IBM personal introduction. The computer was able to detect 16 spoken words and numbers from 0 to 9.

Speech recognition in 2015 claims to have jumped 49% of dramatic performance through CTC-trained LSTM (Long short-term memory, a recurrent neural network). The magic is a modern virtual assistant. The next breakthrough in the outcome of voice recognition technology was achieved in the 1970s at Carnegie Mellon University (CMU) in Pittsburgh, Pennsylvania, with considerable support from the United States Dept. of Defense and its DARPA agency. Their tool "Harpy" mastered about 1000 words, a three-year-old vocabulary. Digital speech recognition technology became a feature of personal computers for consumers in the 1990s, along with Microsoft, IBM, Philips and Learnout and Hussey Fight. AT&T deployed the Voice Recognition Call Processing service in 1992 to route telephone calls without the use of a human operator. The launch of the first smartphone, the IBM Simon Market, in 1994, laid the groundwork for smart VAs as we know them today. The first modern digital VA to be installed on a smartphone was the Siri, which was

introduced on October 4, 2011 as a feature of the iPhone 4S. The common way of communication used by people in daily life is through speech (Islam and Hossain, 2019).

Aim and Purpose of the study

The purpose of the project is to develop a Smart assistant system that provides an intelligent VA with the functionalities as finding any book, listening short review of a book, to know university notice, class routine, examination schedule, semester final result and others effective information. This study is focusing on the VA system development with Raspberry Pi and Alexa voice service. As all these functionalities and services of the project are explained, the basic structure and constructions of the project are basically illustrated with its objectives.

METHODOLOGY:

This project mainly concerns the work on VA system development. The methods of this project are really simple, just connect requirements and setting up an effective voice service on an effective single board computer system. Apart from the project itself, there is also some investigation works on the existed products in this area and the tendency of voice products for personal assistant developing. This is really a difficult issue in this project. I investigated before began implementation and also at the time of implementation. At the investigation, I was mainly focused on the selection of effective single board computer and voice service system. And finally, I select Raspberry Pi 3 Model B as a single board computer and Alexa Voice Service (Herrera, Sebastian, 2019).

Research work organization - The project work is organized based on the actual task for the pre-planning, implementation, test, and optimization. As it has been primarily planned of the developer to work and study 7 days in a week and daily 10 hours; at first 45 days only for study, thinking and pre-planning; then 30 days for resources collections and implementation; after that 15 day for testing and summarizing the work, totally it is 90 days' work. Apart from the pre-planning, implementation, test, and optimization, the developer also defined the work plan every time before

the implementation and improves the project after the accomplishment of each individual section. Developer communicates for any kind of help through Facebook, mobile phone and Gmail for sharing the ideas and discusses the project.

Information Retrieval - As this project includes the functions and services as finding book location, listening short review of a book, provides the university notice, class routine, examination schedule, semester final result, and others effective information.

Finding book location: The Primo should allow the users to find any book located in the library of our university. When users ask for a book location, this system will provide the bookshelf number, and column.

Example:

Users: Where is AI Books?

Primo: You will find the books at library bookshelf-5, row-5.

There are 5 books based on Artificial Intelligence.

1. Artificial Intelligence: A Modern Approach.
2. The Emotion Machine.
3. Introduction to Artificial Intelligence.
4. Machine Learning.
5. Human plus Machine

Listen to a short review of any book: The Primo should allow the users to listen to a short review of any book in the library. When users tell a book name, this system will provide the author name and short review on the book.

Example:

Users: Introduction to AI !

Primo: This book is authored by Philip C Jackson.

Introduction to AI shows a launch to the science of augmenting processes in computers, and the study methods and results. You'll find an automated understanding of natural languages, robot systems, heuristic scene analysis, and specific AI accomplishments. The book will find at bookshelf-5, row-5, and column- 3. We build my system with Raspberry Pi and Alexa Voice Service. After done all installation the system start with data collections and skill development.

1. System Overview

A general overview of the system is presented in the block drawing of the system as follows-

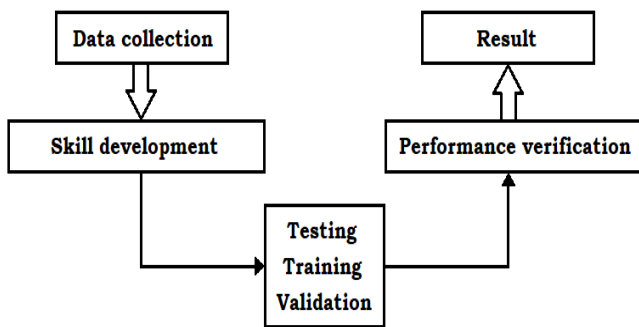


Fig 1: A general overview of the system is presented in the block drawing of the system.

2. Data collection

The dataset contains books name, author’s names of books, a short review of books, locations of the book in the library. Some of those data extracted from the internet and some are calmed from the library at the Prime University. The data for book names, author’s names and location of the books are calmed from the library. The data for the short review of books are calmed from the internet.

3. Skill development

The skill builds by using BLUEPRINT’s frame. The skill is cut up into two fields; those are the location of books and the short review of books. Both fields are divided into two parts those question topics and answers.

Table 1: For the location of books.

Questions Topics	Answers	
	Location	Book name
Programming Language	You will find the books at library bookshelf 5, raw 2 and 3.	There are 3 types of programming books. 1. Python. 2. Java. 3. C.

Table 2: For the short review of books.

Questions Topics	Answers	
	Author Name	Review
Machine Learning	This book is authored by Tom M. Mitchell.	This book covers the area of machine learning, the research of algorithms that allow computer programs to automatically develop through confront. The book is intended to assist high level undergraduate and early graduate courses in machine learning. You will find the books at library bookshelf 5, raw 5 and column 2.

4. Training:

At the training session based on detection, the trainer asks by question topics in different ways to see the detection of topic and book names. After trained 3/4 or more times the system could detect the topic and book names properly. Some of book and topics names trained 10/15 or more times to depend on pronunciations of the specific word.

5. Testing

The performance at the testing session based on three fields, those are detection, accuracy, and response. The testers ask the system the questions based on topics and book names, the system answer to the tester

location of books and a short review of the books. The system reaches expected performance at the testing session.

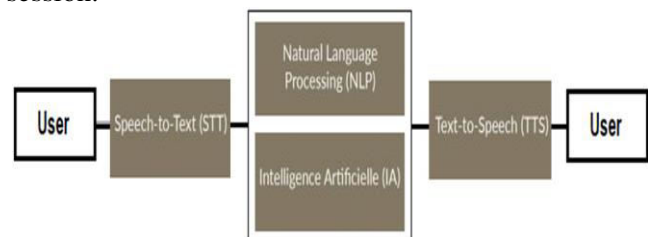


Fig 2: Working principle of Virtual Assistant System.

The voice assistant system performs with a few simple steps. At first, user voice commands convert to text data by using Speech-to-Text (STT), then the text data

processes by NLP and IA, finally the processed data convert to the speech by using Text-to-Speech (TTS).

Realization

To realization, our project divides into three parts -

1. Create a product profile
2. Prototype with Raspberry Pi
3. Build Alexa Voice Service skill

3.1 Create a product profile

To Create product profile we done two steps -

1. Register a Product
2. Activate Security Profile

3.1.1 Register a Product

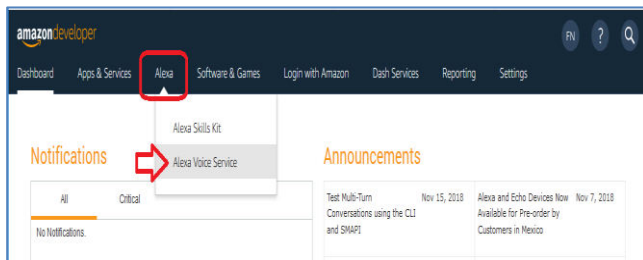
To register a product we made an Amazon Developer Profile, register our prototype or commercial tool and set up our key profile. When done, we have the necessary credentials to examine the Alexa Voice Service (AVS).

Build an Amazon developer account:

We create our Amazon developer account with no charge from the link <https://developer.amazon.com/>, same to others online account creation.

Register our prototype and build a security profile:

After we've built an Amazon developer key, we need to build a product and key profile. This will enable our software customer to join to AVS. Log in to <https://developer.amazon.com/>, we enter in the Dashboard by default click the **ALEXA VOICE SERVICE** button in the universal navigation to start creating products with Alexa built-in.



This is our first time using AVS; we have seen a welcome screen. Then Click on the **GET STARTED** button, and after that click on the **CREATE PRODUCT** button. If we're a returning developer, click the blue **CREATE PRODUCT** button at the top right side of the screen.

Fill in product information:

Note: These instructions are for developers' prototyping on Raspberry Pi. If we are registering a commercial product profile, we will want to use our own custom information here.

1. Product Name: Use **AVS Tutorials Project** (or any other).
2. Product ID: Use **Prototype Pi** (or any other). No zones are allowed for the Product ID field.
3. Select **Tool with Alexa built-in** for Please Select Our Product Type. Select **No** for Will our tool use a companion app?
4. Choose **Other** for Product Category and write **Prototype** in the (please specify) and Brief product description field.
5. Select **Hands-free** for "How will users interact with our product?"
6. Skip the Upload an image step (not required for prototyping).
7. Select **No** for "Do we intend to distribute product commercially?"
8. Select **No** for "Will our tool be used for Alexa for Business?"
9. Select **No** for "Is this children's good or is it or else directed to child younger than 13 years old?"

Click **NEXT** to continue.

Set up our security profile:

Here, we create our security profile as given below, we generate an ID and after generation, we download the JSON file. The JSON file will use to set up Alexa on Raspberry Pi.

1. Click **CREATE NEW PROFILE**
2. Enter our own custom **Security Profile Name** and **Security Profile Description** for the following fields - or use the below example names:
 1. Security Profile Name: **AVS Project** (or any other).
 2. Security Profile Description: **AVS Tutorials** (or any other).

Click **NEXT**.

Security Profile ID will be generated for us.

3. Select **other tools and platforms** from “Web-Android/Kindle - iOS - Other tools and platforms” in the **Platform Information**.
4. Write a name for our Customer ID here - we can just use **Prototype**.
5. Click "Generate ID". We should get a Client ID and an option to download it.
6. We're creating this good profile on our Raspberry Pi; click **Download** to get our credentials onto our AVS prototype. Save the config.json file to our /home/pi directory.
7. Check the box beside “I agree to the AVS consensus and the AVS Program Requirements”.

Click **FINISH**.

We now have entry to the Alexa Voice Service APIs. Click **OK** on evoke to continue. Our tool now is listed on our AVS dashboard.

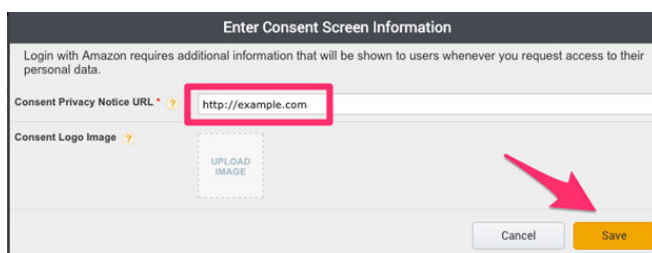
Note: If we're building a commercial product, we'll need to provide our customers with a link to our privacy policy.

3.1.2 Activate Security Profile

In this step, we need to enable our security profile. We can skip this step because we are prototyping on Raspberry Pi.

Enable our security profile:

1. Open a web browser, and visit - <https://developer.amazon.com/lwa/sp/overview.html>.
2. Near the top of the page, select the commercial tool security profile we created earlier from the drop-down menu and click the **CONFIRM** button.
3. Enter our privacy policy URL beginning with <http://> or <https://>.
4. We may upload an image. The image will be shown on the Login with Amazon consent page to give our users context.
5. Click the **SAVE** button.



Now that we've created our product profile, it's time to get our hardware and start building.

3.2 Prototype with Raspberry Pi

Prototype with Raspberry Pi is done by six steps that are- Required Hardware, Set up Raspberry Pi Input AVS Credentials, Build the AVS Tool SDK, Get a Refresh Token, and Talk with Alexa.

3.2.1 Required Hardware

To build our system we need some hardware tools and also some additional hardware tools.

Required hardware-

1. A Raspberry Pi 3
2. Charger DC 5v and minimum 2 Amp
3. 8/16/32GB micro SD Card
4. Card reader
5. 3.5mm compatible speaker
6. Mini-microphone
7. Computer Display
8. Keyboard
9. Mouse.

Additional hardware-

1. HDMI to VGA Adapter
2. USB Sound Adapter

Now that we've got everything we need in hand, let's set up our Raspberry Pi and start building!

3.2.2 Set up Raspberry Pi

To set up our Raspberry Pi we download Raspbian OS and photo our SD card, assemble our Raspberry Pi, and Startup our Raspberry Pi.

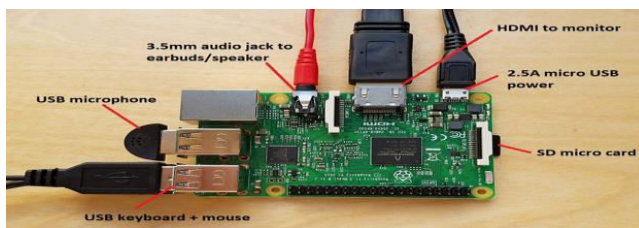
Download Raspbian OS and photo our SD card:

Before we can start our Raspberry Pi, we'll need to use our Linux, MacOS or Windows PC to load an Operating System onto an SD card to boot the Pi from. On our PC, download the NOOBS_v2_9_0.zip file from the given link http://director.downloads.raspberrypi.org/NOOBS/images/NOOBS-2018-10-11/NOOBS_v2_9_0.zip, and then unzip it locally on our PC.

After unzipping the download, open the **NOOBS_v2_9_0** folder and drag and drop the entire folder's contents onto an empty 32 GB microSD card.

Assemble our Raspberry Pi:

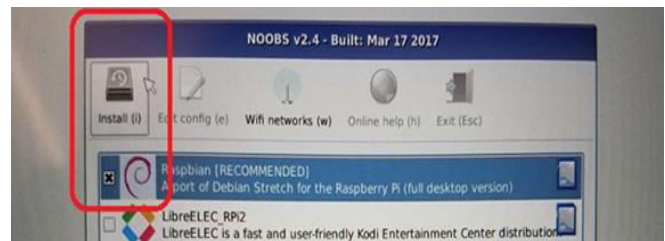
1. Check that our micro SD card is inserted into the micro SD card slot on our Pi. **The SD card contacts should face up.**
2. Plug in the USB microphone and install our ear buds or speaker into the **3.5mm audio jack** on our Pi.
3. Connect the keyboard and mouse via the USB ports. Make sure we don't cover the USB microphone.
4. Connect our monitor using the HDMI port.
5. Connect the Ethernet Cable (if not using Wi-Fi)



Start up our Raspberry Pi:

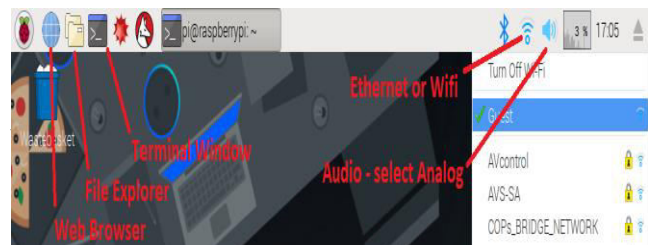
Plug in the power supply to the **micro USB** connector on the Pi. We should see a loading screen go through some startup steps before booting to desktop. If we run into any errors upon startup, try booting from a different pre-imaged micro SD card. If nothing happens after that, ensure our microSD card is inserted upside-down (contacts facing up). Upon successful startup, we'll be prompted to select an operating system and locale. Check the box next to **Raspbian [RECOMMENDED]** and select our language and keyboard preferences at the bottom of the screen. Once we've selected our preferences, click "Install" in the upper left side of the pop-up box.

Note: if we don't have the correct locale/keyboard selected, we might get password errors. Make sure to use the correct keyboard layout for our keyboard - we can test it when selecting to ensure any special characters we type to show up correctly.



It should take around 15 minutes to fully install. Once it's done, we'll see a success message - click OK and we'll be taken to a desktop. Select our locale/language/keyboard but feel free to close the window when it asks us to change our password or do any other setup steps. If we missed the option, we can always check our Keyboard Configuration by clicking on the Raspberry icon in the top right and selecting **Mouse** and **Keyboard settings** from the **Preferences** menu. Click **Keyboard Layout** from the **Keyboard** tab and select the appropriate configuration. Make sure whatever we pick, we can write characters such as "quotes" and @ symbols.

If we're not using cabled Ethernet, activate **2.4 GHz Wi-Fi** in our Raspberry Pi by snick on the connection symbol in the **top-right side** of the toolbar and selecting our SSID. Note that the Raspberry Pi 3B5 GHz does not work with Wi-Fi. To make confirm our sound will be output via a 3.5mm audio pluck, right-click on the speaker graphic at the top-right of our pie and select "**Analog**". Verify connectivity by opening a web browser - click on the **globe icon** in the top left toolbar.



3.2.3 Input AVS Credentials

At this time we have to download the AVS Tool SDK and our credentials:

Download the AVS tool SDK: Now that our tool has internet connectivity, unlock an extreme by clicking on the black window symbol near the top left corner of our toolbar. Our extreme should come up in /home/pidirectory. Let's start by upgrading apt-get to confirm we have examined to the required dependencies.

Note that the AVS tool SDK v1.10.0 requires PulseAudio to be installed and BlueAlsa must be disabled. Copy and paste the following order into our extreme window and hit return to modernize apt-get.

```
sudo apt-get upgrade
```

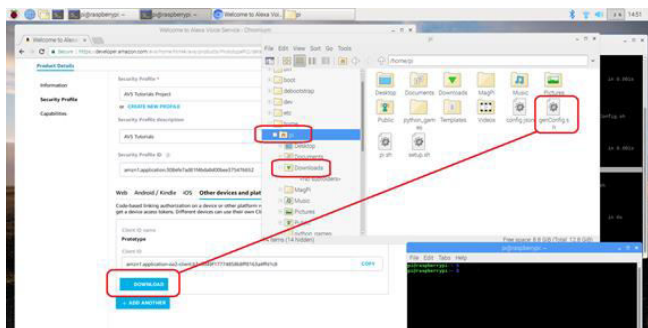
Now, let's get the SDK installation and configuration scripts. Copy and paste the following into our terminal and hit return:

```
wget https://raw.githubusercontent.com/alexa/avs-device-sdk/master/tools/Install/setup.sh \  
wget https://raw.githubusercontent.com/alexa/avs-device-sdk/master/tools/Install/genConfig.sh \  
wget https://raw.githubusercontent.com/alexa/avs-device-sdk/master/tools/Install/pi.sh
```

Download our credentials:

If we didn't already store it to our Pi when building our good profile, it's time to get our **config.json** file onto our customer tool. Start by launch a browser and logging into our AVS dashboard. Click on our good Name, it should be **AVS Tutorials Project** or whatever we named it when building the good profile. This will take us to a good menu - on the left side we should see Good Details. Select key Profile below that and choose **other tools and platforms** from the Web - Android/Kindle - iOS - Other tools and platforms menu.

When we click the **Download** button on our key Profile in our web browser, we'll see a **config.json** file appear in our home/pi/ downloads folder. In the file store, copy this file from the /downloads folder and place it in our home/pifolder as shown in the photo below.



Now that our Raspberry Pi has our own **unique credentials** loaded on it, let's build the AVS Tool SDK to voice-enable our prototype.

3.2.4 Build the AVS Tool SDK

We are now ready to run the install script. It will install all dependencies from Sensory to Wake Word Engine (WW). WW compares incoming audio to an online model of a wake word ("Alexa") and will trigger audio transmission in the cloud when triggered. Note that these WWEs are provided for prototyping purposes only and need to be licensed for commercial tools. The AVS tool SDK is modular and flexible. When we're ready to build our product, we can choose any WWE we prefer. Note that for AVS products, the word space must be Alexa so that our users do not get confused about how they will interact with our tool. To run the install script, open a terminal (or just use our existing terminal window) by clicking on the window console in the toolbar at the top-left corner of the screen. Pulls the certificates from our config.json file to run the installed script. To start the setup script, copy and paste the following command into our terminal window and hit return.

```
cd /home/pi/  
sudo bash setup.sh config.json [-s 1234]
```

Note that the field in double brackets is the **Tool Serial Number** which will be distinctive to each instance of the SDK. In this case, it's pre-populated with 1234. Type "AGREE" when it prompts us to accept the licensing terms from our third-party libraries. Unless, of course, we disagree! This will kick off the installation process which could take over 20 minutes. Note that about 15 minutes into the install, it's going to pause and ask us to accept Sensory Wake Word's terms and conditions. Once we've finished compiling, we should see a success screen similar to the one shown here. If our tool freezes - don't worry, just plug in our power cord and restart. When we return to our desktop, run the above setup.sh command again to finish our installation. Now all we need to do is launch the sample application and take a refresh token from AVS so that our tool can authenticate with the cloud via Login with Amazon (LWA).

3.2.5 Get a Refresh Token

Our Raspberry Pi now has the AVS tool SDK installed and our credentials loaded, but our tool still needs a **refresh token** to enable our client to maintain a connection to the Alexa Voice Service in the cloud.

We using the **startsample.sh** script to launch the Sample App and request a refresh token for our prototype tool. In a terminal window, navigate to the /home/pi_directory and runthe **startsample.sh-**

```
cd /home/pi/  
sudo bash startsample.sh
```

Once the App starts, startup debug messages will start scrolling through the window rapidly, and we'll see a notification that our tool is checking for Authorization. Look for the box with the URL and the code we need to authenticate our tool:

Using any internet-connected tool, go to amazon.com/us/code to log in with our Amazon developer credentials and input the code provided by our sample app. It might take up to 30 seconds for CBL Auth Delegate to successfully get a refresh token from Login With Amazon (LWA). We'll get a success message, and our sample app should be ready to go! Our hard work has paid off, and now it's time to Talk with Alexa!

3.2.6 Talk with Alexa

Congratulations on creating our first prototype with Alexa built-in! If our sample app is still running, we should see ASCII art indicating that Alexa's status is IDLE, meaning the client is waiting for us to initiate a conversation. When we say **Alexa**, we should see a bunch of messages scroll in our terminal window. One of those will show the status changing to Listening, indicating the wake word has been recognized. Then say **"Tell me a joke."** If Alexa responds with Thinking..., then Speaking, we have a working prototype... and probably, a very bad joke. If we don't hear anything over our earbuds, check that our audio output is set to **"Analog"** by right-clicking on the speaker icon in the top-right corner of our Pi's screen. Our audio output might be set to HDMI by default. Also, ensure our speaker/earbuds are turned on and plugged into our Raspberry Pi's **3.5mm audio jack**. And take a refresh.

```
cd /home/pi/  
sudo bash startsample.sh
```

If Alexa isn't responding click on keyboard "t + ENTER", then ask ALEXA. **Try a multi-turn**

interaction, says "Alexa", and then asks **"Set an alarm"**. When she asks what time, just say a number, like **"8"**. She'll want to know if that's **AM** or **PM**. It's a more natural method of communication because we can continue speaking without starting every phrase with "Alexa." (Islam et al., 2020)

3.3 Build Alexa Voice Service skills

We have two ways to build our skill,

1. Build skill with Alexa Developer Console
2. Build skill with Skill Blueprints.

We mainly use "Skill Blueprints" to develop our skills. In this platform, this is very easy to build any useful skill.

Build skill with SKILL BLUEPRINTS

Amazon Alexa is one of the most capable voice assistants on the market. Amazon has a big install base for smart speakers (Daniel B. Kline, 2017). That's largely due to the skills that let Amazon's gadgets do far more than the abilities they come with out of the box. Since Amazon unveiled its Alexa Blueprints portal, which allows us to make our own skills for Alexa, this means we can further customize Alexa to our liking. We can even publish our customized skills and share them with other Alexa users. We just need to follow one of the templates Amazon has available. To create an Alexa skill of our own-

1. LOGIN TO BLUEPRINTS

Start by going to <https://blueprints.amazon.com/>. Once we're there, click on sign in at the top of the screen. We'll then have to log in with our Amazon account details.

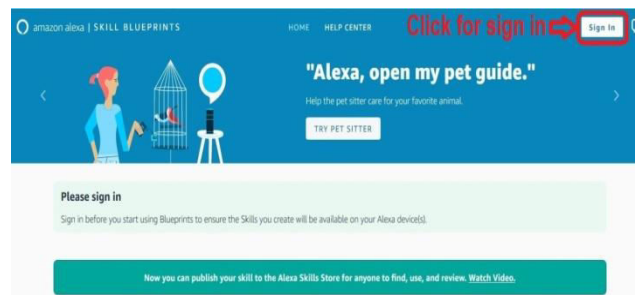


Fig 3: Login to blue prints.

If we don't have our Amazon account, we just need to create an account by clicking the "Create your Amazon account" with the required information.

2. CHOOSE YOUR TEMPLATE

Start by selecting a blueprint. Alexa thinks of blueprints as a collection of templates. There are dozens of options in six different categories: greetings and events, agencies and busy-nesses, fun and games, learning and knowledge, at home and storytellers. After finding an Alexa Blueprint template that strikes our fancy, click on it.

We select “Onboard Guide” templates for our requirements. Notice the Play button and progress bar, too. It lets us hear a sample of a skill created with that blueprint that can inspire and provide helpful ideas before we dive into the rest of the process. Click on “MAKE YOUR OWN” to make a skill for us.

3. POPULATE THE BLUEPRINT’S FIELDS

Here, three types of fields, “Where to find important resources”, “How to do things” and “Contact info”. We have to fulfill these areas with our requirements. If we want to add more, we can add by clicking “ADD ITEM” “ADD HOW TO” and “ADD CONTACT”. After complete adding all of the required we will be a need for our assistant, we have to click “NEXT: EXPERIENCE”. Now we need to customize the experience with welcome and goodbye message. We can use multiple messages to welcome our users and to goodbye to them. After complete the Intro and Exit message, we have to click “NEXT: NAME”.

4. NAME YOUR SKILL

Choosing a name for our skill is one of the last steps to go through. Picking one might seem very straight forward, but we should keep Alexa’s limitations in mind. Now, click “NEXT CREATE SKILL” to complete our skill creation. Finally, “Primo Smart Assistant” ready to perform. If we notice Alexa can’t understand us when we ask to use the skill, come back and pick a

name that is a bit simpler. We can do that by “Skills You’ve Made” at the top of the Alexa Blueprints homepage.

Click a skill to select it, then the Edit button associated with it. Then we will see all of the skills we have built. We have to click the skill need to change. Now we can change the name by clicking the particular are near skill name. We all can edit our skills by clicking at the “EDIT” option. We also can Delete, Share and Publishing to skills store our created skill by clicking “DELETE”, “SHARE WITH OTHERS” and “PUBLISH TO SKILLS STORE”.

RESULTS AND DISCUSSION:

The system tested on a subset of the dataset in the developer environment with performance more than 95%. In the library environment with real users the performance pretty good. The system detects questions with 90% correctly and answers the questions with 100% accurately. The response of the system was expected, but sometimes the system stopped before the full answer. Overall performances of the system in the real library environment with the real user are nearly 85% or more.

Heuristic evaluation report - To test the performance of any system heuristic evaluation is one of the best processes. In this process, users can perform directly to determine the performance and effectiveness of any system. This is very much helpful to the future development of the system.

To recognize the performance of the Primo Smart Assistant system we carried out a heuristic evaluation based on Speech detection, accuracy, response, and effectiveness. The questionnaires and rating are given below -

1. Detection

#	Review Checklist	Rating
1.1	Is this assistant detecting the skill name properly?	4.25
1.2	Is this assistant detecting Books name correctly?	4.25
1.3	Is this assistant detecting Student ID correctly?	4.50
1.4	Is this assistant detecting specific notice correctly?	4.50
1.5	Is this assistant detecting batch correctly?	4.50

2. Accuracy

#	Review Checklist	Rating
2.1	Is this assistant informing the location of books correctly?	4.75
2.2	Is this assistant review books correctly?	4.50
2.3	Is this assistant informing student result correctly?	4.75
2.4	Is this assistant informing specific notice correctly?	4.75
2.5	Is this assistant informing the class routine correctly?	4.75

3. Response

#	Review Checklist	Rating
3.1	Is this assistant responding quickly?	4.00
3.2	Is this assistant responding about books quickly?	4.25
3.3	Is this assistant responding to student result quickly?	4.25
3.4	Is this assistant responding notice quickly?	4.25
3.5	Is this assistant responding class routine quickly?	4.25

4. Effectiveness

#	Review Checklist	Rating
4.1	Is this system effective as a department assistant?	4.50
4.2	Is this system effective as a library assistant?	4.50
4.3	Is this system effective as a smart assistant?	4.75
4.4	Is this system effective for librarian robot?	5.00
4.5	Is this system effective for smart robots?	4.75

Where overall rating we get 4.55 out of 5.00.
 The overall rating on detection is 4.4 out of 5.00.
 The overall rating on accuracy is 4.7 out of 5.00.
 The overall rating on response is 4.4 out of 5.00.
 The overall rating on effectiveness is 4.7 out of 5.00.

Rating ranges:

Very poor	Poor	Neutral	Good	Very good
1	2	3	4	5

CONCLUSION:

This system built on Raspberry Pi and Alexa Voice Service platform. I was very new for those areas but finally, the system has met the objectives in the university environment with reliability. This system is really good as a library assistant, as a department assistant and it also smartly usable as a personal assistant. Confidently I can say, the system is valuable and usable to all kinds of organizations. However, I believe it's just beginning. In the future, we will build a more effective system that shows our intellectual power and efficiency. Our VA system help and service
 UniversePG | www.universepg.com

to assist users by its listening, thinking and speaking ability for any users questions or voice command with specific tasks, including- find a book location, short review of a book, university notice, class routine, examination schedule, Semester final result, and others effective information. This project is focusing on the VA system development with Raspberry Pi and Alexa voice service. This system is really good as a library assistant, as a department assistant and it also smartly usable as a personal assistant. Confidently I can say, the system is valuable and usable to all kinds of organizations. The following recommendations for future work -

1. Complete VA System.
2. Artificially Intelligent Robot.
3. Home Assistant Robot.

ACKNOWLEDGEMENTS:

First of all, I must thank the Almighty upon who rests the supreme authority. I would like to express my heartfelt gratitude to my honorable teacher Masruh Sobhan Siam and my dear friend Enamul Kabir for the guidance, inspiration and constructive suggestion that

helped me in the preparation and accomplish. Finally, appreciations are placed for responsible parents, honorable teachers, fellow classmates, and friends for sharing their knowledge and ideas.

CONFLICT OF INTERESTS:

The author (s) declared no potential conflicts of the interest with respect to the present research work.

REFERENCES:

1. Alam N., and Alam M. (2020). The trend of different parameters for designing integrated circuits from 1973 to 2019 and linked to Moore's law, *Aust. J. Eng. Innov. Technol.*, **2**(2), 16-23.
<https://doi.org/10.34104/ajeit.020.0160>
2. Daniel B. Kline (30 January 2017). "Alexa, How Big Is Amazon's Echo?". The Motley Fool.
3. Hancock, P.A. (2017). Imposing limits on autonomous systems. *Ergonomics*, **60**(2), 284-291.
4. Herrera, Sebastian. (2019). "Amazon Extends Alexa's Reach In to Wearables". *WSJ*. Retrieved 26 September.
5. Klüwer, Tina. (2011). "From chatbots to dialog systems." *Conversational agents and natural language interaction: Techniques and Effective Practices*. IGI Global., 1-22.
6. Hoy, Matthew B. (2018). "Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants". *Medical Reference Services Quarterly*. **37**(1): 81-88.
<https://doi.org/10.1080/02763869.2018.1404391>
7. Hoy MB. (2018). Alexa, siri, cortana, and more: An introduction to voice assistants. *Med Ref Serv Q*, **37**:81-8.
8. <https://developer.amazon.com/docs/alexa-voice-service/set-up-raspberry-pi.html>
9. <https://www.raspberrypi.org/documentation/>
10. <https://Blueprints.amazon.com>
11. https://en.wikipedia.org/wiki/Raspberry_Pi
12. https://en.wikipedia.org/wiki/Amazon_Alexa
13. https://en.wikipedia.org/wiki/Virtual_assistant
14. https://en.wikipedia.org/wiki/Artificial_intelligence
15. https://en.wikipedia.org/wiki/Machine_learning
16. https://en.wikipedia.org/wiki/Natural_language_processing
17. https://en.wikipedia.org/wiki/Speaker_recognition
18. https://www.youtube.com/watch?v=7iQgBS_kp2M&t=67s
19. Islam MT and Hossain MS. (2019). Hybridization of vigenere technique with the collaboration of RSA for secure communication, *Aust. J. Eng. Innov. Technol.*, **1**(6), 6-13.
<https://doi.org/10.34104/ajeit.019.06013>
20. Islam S, Islam MS, and Mandal S. (2020). One dimensional heat transfer through a uniform plane wall by using finite volume method, *Aust. J. Eng. Innov. Technol.*, **2**(2), 24-30.
<https://doi.org/10.34104/ajeit.020.0240>
21. Maciel, A. M. A., Rodrigues, R. L., Carvalho, E. C. B. (2014). Desenvolvimento de um Assistente Virtual Integrado ao Moodle para Suporte a Aprendizagem Online. In *Proceedings of Simpósio Brasileiro de Informática na Educação*. Dourados, MS, Brazil, November, 06-10, 2014.
22. Suykens, J.A.K. (2014). *Introduction to Machine Learning*. MIT Press, 765-773.

Citation: Mamun AMA, Islam T, Siam MMS, and Kabir ME. (2020). Development of smart librarian with the virtual assistant (PRIMO), *Aust. J. Eng. Innov. Technol.*, **2**(4), 54-65.

<https://doi.org/10.34104/ajeit.020.054065>

