CSE 157 Lab Session 1

Write-up Assignment

Airi Kokuryo

Student ID: 2086695

1. Pi Setup

(a)

Initial Setup

- Install the Raspberry Pi Imager onto your personal computer
- Install the newest version of the Raspberry Pi OS(32-bit) into the Micro SD card, using the USB-A Micro SD Card Reader
- Put the SD card back to the Raspberry Pi
- Connect the Raspberry Pi to the monitor with the HDMI-DVI Adapter and power it using the Power Supply

Ways to connect to the Wi-fi

(1) Raspberry Pi Imager

The Raspberry Pi could be connected to the Wi-Fi by changing the setting on "Advanced Options" on the Raspberry Pi Imager.

(2) Welcome Wizard

The Wi-Fi could also be configured through the Wi-Fi setup menu that comes up with the first boot-up. Despite this, the "ResWifi" that was being used for the lab will not appear in this wizard. However, any Wi-Fi connections that would not require a two-way handshake could be connected via this Welcome Wizard.

(3) Using the Shell Script

- Set up the Wi-Fi using the Shell Script file "wifi.sh", which allows the connection with the Wi-Fi requiring the two-way handshake
- Make the Shell Script executable with the command:

```
ch mod +x ./wifi.sh
```

- Run the Shell Script

```
./wifi.sh
```

- Use the Gold ID and password to connect to the ResWifi.

Ways to change the password:

(1) Using the passwd command

- On the Command Prompt, use the command to change the password;

```
passwd
```

- Type the current password and the new password

(2) Using the menu

- On the Command Prompt, use the command to open the menu;

```
sudo raspi-config
```

- From the menu, go to "System Options" → "S3 Password"
- Type the current password and the new password

(b)

Process of remotely logging into Pi (Using Windows)

- Find the IP address of the Raspberry Pi with the command;

```
if config
```

- OR on find the Raspberry Pi IP address through Windows PC by the command;

```
ping raspberrypi
```

- Go to the menu→Interface Options→enable the SSH connections

```
sudo raspi -config
```

- On your laptop, type the command;

```
ssh username@IP address
```

- Type the Pi account password

I do not see myself doing this often, because the Raspberry Pi has its own OS and GUI which allowed me to do the setups without any problem. I did not use the SSH connections for Lab 1 as it did not require the use of ad-hoc mode to wirelessly connect with other Raspberry Pis. If it was in ad-hoc mode, I would have used it more since it would have been inconvenient to write and run the Python scripts that require an internet connection.

(c)

Network Information

The network information can be found using the command;

```
ifconfig
```

The network information generally tells the current configuration for the interfaces, such as interface names and IP addresses as well as flags for interface status. It gives details about the network environment and its connections.

Interfaces

The interfaces represent the connection point between a device and a network. In this lab, we have created two interfaces that would be mainly used, so the interfaces will represent whether the Raspberry Pi is in the Wi-Fi or Ad-hoc modes.

(1) The wifi-interface

This interface provides network information about the Wi-Fi connection, including IP address and SSID. This allows the Raspberry Pi to have a standard Wi-Fi connection, typically connecting to a wireless network as a client.

(2) The adhoc-interface

In this mode, the adhoc-interface provides network information for an ad-hoc Wi-Fi connection. The SSID here functions as an identifier that allows the Raspberry Pi to act as a peer-to-peer client, enabling direct communication with other devices in a decentralized, ad-hoc network.

2. Pi for development

(a)

<u>Process for setting up the Pi for development</u>

- Check the updates for the Raspberry Pi OS using the command

```
sudo apt-get update
```

- Install the latest version of Python 3 using the command

```
sudo apt-get install python3-dev
```

- Install IDEs (Integrated Development Environments) using the command

```
sudo apt install <IDE name>
```

(b)

<u>IDE installation</u>

I have installed Thonny as the IDE, as this has a simple and easy user interface. Blue J also had a good user interface, however, it was more lenient towards Java users. While the lab will only be using Python, I thought Thonny would fill the needs for lab purposes without a problem.

The installation process is as follows;

Install IDEs (Integrated Development Environments) using the command

```
sudo apt install <IDE name>
```

(example) `sudo apt install thonny`

3. Creating a Weather Report Program

(a)

The program will use the requests.get command to retrieve the parameter information from the url of the open-meteo api. The response will be stored in "data" instance as a JSON. Time and date will be taken from the system using the datetime.now. In the program, the important parameters for the HTTP request are written with the URL, therefore the params method is empty. The script checks whether the response status code is 200 and handles the case where the request fails. The output of "report.txt" will require all the data to be in String, and therefore is using strftime for time, and json.dumps() for data collected by JSON to convert them to String.

(b)

The code includes "requests" modules, which enable Python scripts to send HTTP requests to web services or APIs. The API URL contains information specifying what kind of request and data will be retrieved. The Python script, using the "requests" module, sends an HTTP request to the API URL. The API then processes the request and returns a response that the script can work with.

4. Creating a Bootup Script to Run the Program at Startup

(a)

- Open .bashrc file using nano (or vim);

```
sudo nano /home/pi/.bashrc
```

- On the bottom of the script, add the lines to run the Python script at boot;

```
echo Running at boot
sudo python /home/pi/generate_report.py
```

- Reboot the Raspberry Pi

```
sudo reboot
```

- After rebooting, open the command prompt so it will run the commands

(b)

It might be useful to include programs in the boot sequence when each program can not be operated by hand every time it reboots. For example, if the IoT device using the Raspberry Pi needs to be turned off and on depending on their topology, these devices must be running the scripts at boot. On the other hand, this would not be useful when the script needs to be updated or requires control. Also during the testing process, it is better to have the scripts running by hand so the bugs are fixed quicker without having to check it at boot every time.

5. Conclusion

During the lab, I learned the ways to initially set up the Raspberry Pi. Also, I have learned several ways to connect to the Wi-Fi, and how it could be changed to the Ad-hoc mode for connecting to other devices. I have also learned the SSH connection with the laptop, which allowed me to control the Raspberry Pi using my laptop. Getting information from the open-meteo API was one procedure where I learned how to collect current data using Python as well. I also learned how this python script could be run at boot by adding command lines in the file which runs at start.