

CHAPTER-1 – ABOUT THE INSTITUTION

1.1 HKBK COLLEGE OF ENGINEERING PROFILE

Engineers possess a rare combination - a seamless blend of the practical and the creative. It's a profession that tackles society's challenges and make the world a better place to live. If you have a burning desire to know how things work and want to make an impact on the world around you, engineering is your path. HKBK College of Engineering a place where you'll experience worldclass engineering facilities in a supportive learning environment.

Engineering is shaping the future all around us. An Engineering degree from HKBK equips you with the analytical, technical and professional skills needed to solve some of the biggest challenges we face in the world today. Our renowned faculties help you to develop some of the key solutions to major global challenges.

1.2 VISION AND MISSION

VISION OF THE COLLEGE

To empower students through wholesome education and enable the students to develop into highly qualified and trained professionals with ethics and emerge as responsible citizen with broad outlook to build a vibrant nation

MISSION OF THE COLLEGE

To achieve academic excellence through in-depth knowledge in science, engineering and technology through dedication to duty, innovation in teaching and faith in human values. To enable our students to develop into outstanding professionals with high ethical standards to face the challenges of the 21st century. To provide educational opportunities to the deprived and weaker section of the society, to uplift their socio-economic status

1.3 ENGINEERING STREAMS

- ❖ COMPUTER SCIENCE AND ENGINEERING
- ❖ INFORMATION SCIENCE AND ENGINEERING
- ❖ MECHANICAL ENGINEERING
- ❖ ELECTRONICS AND COMMUNICATION ENGINEERING
- ❖ ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

CHAPTER-2 – ABOUT THE DEPARTMENT

2.1 DEPARTMENT PROFILE

HKBK libraries have subscribed to a number of eBook packages, giving us access to hundreds of thousands of books online. All allow reading the eBook through the browser, and many allow downloading for offline reading or transfer to a mobile device or dedicated eBook reader. Every library has got wireless internet access, places to study and facilities to print, copy or Scan. Boys Hostel Off-Campus Hostel, Girls Hostel Off-Campus Hostel

Available Facilities:

- Basketball Court Cricket Ground
- Civil Engineering Lab, Chemistry Lab, Computer Lab, Electronics Lab, Language Mechanical Lab, Physics Lab
- Cafeteria, Gym, Hospital/Medical Facility, Wi-Fi Campus, Auditorium, Research Seminar Hall, Smart Classrooms, Fine Art Club.

2.2 VISION AND MISSION

VISION

To shape the students as disciplined humane engineers who can build a strong, peaceful and vibrant country and focus on mutual respect, tolerance and professional ethics.

MISSION

To provide the best possible educational experience through excellence in teaching and research activities for today's students and professionals of tomorrow.

To hone young minds and train them to be conscientious individuals who will serve the society as competent professionals in the field of Electronics and Communication Engineering.

CHAPTER 3

Company Profile

COMPANY NAME: M/S ZF India SOLUTIONS



Fig 3.1 Company Profile Photo

ZF is a global technology company supplying advanced mobility products and systems for passenger cars, commercial vehicles and industrial technology. Its comprehensive product range is primarily aimed at vehicle manufacturers, mobility providers and start-up companies in the fields of transportation and mobility. ZF electrifies a wide range of vehicle types. With its products, the company contributes to reducing emissions, protecting the climate as well as enhancing safe mobility. Alongside the automotive sector – passenger cars and commercial vehicles – ZF also serves market segments such as construction and agricultural machinery, wind power, marine propulsion, rail drives and test systems.to provide the complete solution for your automation needs. Quality is our priority.

Website - <https://www.zf.com/india.html>

Industry – ZF INDIA

Headquarters - Tal-Khed, Pune 410501, India

Founded - 1915

Specialties - In India, ZF is one of the leading suppliers of technology solutions and services that are shaping mobility trends in the country.

CHAPTER – 4 - INTERNSHIP ACTIVITY

(Incubation in Innovation and Entrepreneurship)

4.1 ACTIVITY OBJECTIVES:

Course Learning Objectives

- To incubate interns with innovative ideas in engineering streams
- To indoctrinate interns with invaluable research procedures and practices
- To train and nurture aspiring students to kindle their latent talent to develop their own startups
- To guide students to venture into areas compatible to their persona and attain individual gratification.
- To appraise students about the prospects and challenges of liberalization, privatization and globalization.
- To develop intrapreneurship qualities to modernize corporate management

4.2 ACTIVITY OUTCOMES:

Course Outcomes: At the end of the program, students will be able to:

- Take speedy and spot-on decisions
- Inculcate risk taking abilities.
- Select green and pasteurized sectors and areas of their core competence
- Realize the techno economic, environmental and social benefits of entrepreneurship

4.3 ACTIVITY PROGRAM

Program Education Outcomes:

PEO-1: Our graduates will possess good knowledge in engineering fundamentals.

PEO-2: Our graduates will be capable of analyzing and designing systems.

PEO-3: Our graduates will be capable of creating innovative products in multidisciplinary areas.

PEO-4: The graduates will be ethically strong personnel with good communication and interpersonal skills with high moral values.

CHAPTER-5 – INTERNSHIP TASKS PERFORMED

5.1 Introduction to Python

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently. There are two major Python versions: Python 2 and Python 3. Both are quite different.

What is Python?

Python is a programming language that is widely used in web applications, software development, data science, and machine learning (ML). Developers use Python because it is efficient and easy to learn and can run on many different platforms. Python software is free to download, integrates well with all types of systems, and increases development speed.

What are the benefits of Python?

Benefits of Python include: Developers can easily read and understand a Python program because it has basic, English-like syntax. Python makes developers more productive because they can write a Python program using fewer lines of code compared to many other languages. Python has a large standard library that contains reusable codes for almost any task. As a result, developers do not have to write code from scratch. Developers can easily use Python with other popular programming languages such as Java, C, and C++.

The active Python community includes millions of supportive developers around the globe. If you face an issue, you can get quick support from the community. Plenty of helpful resources are available on the internet if you want to learn Python. For example, you can easily find videos, tutorials, documentation, and developer guides. Python is portable across different computer operating systems such as Windows, macOS, Linux, and Unix.

APPLICATION OF PYTHON

Web and Internet Development

- Frameworks such as Django and Pyramid.
- Micro-frameworks such as Flask and Bottle.
- Advanced content management systems such as Plone and django CMS.
- Python's standard library supports many Internet protocols:

- HTML and XML, E-mail processing
- Support for FTP, IMAP, and other Internet protocols.
- Easy-to-use socket interface
- Requests, a powerful HTTP client library.
- BeautifulSoup, an HTML parser that can handle all sorts of oddball HTML.
- Feedparser for parsing RSS/Atom feeds.
- Paramiko, implementing the SSH2 protocol.
- Twisted Python, a framework for asynchronous network programming.
- Scientific and Numeric: Python is widely used in scientific and numeric computing:
- SciPy is a collection of packages for mathematics, science, and engineering.
- Pandas is a data analysis and modeling library.
- IPython is a powerful interactive shell that features easy editing and recording of a work session, and supports visualizations and parallel computing.
- The Software Carpentry Course teaches basic skills for scientific computing, running bootcamps and providing open-access teaching

5.2 FEATURES OF PYTHON

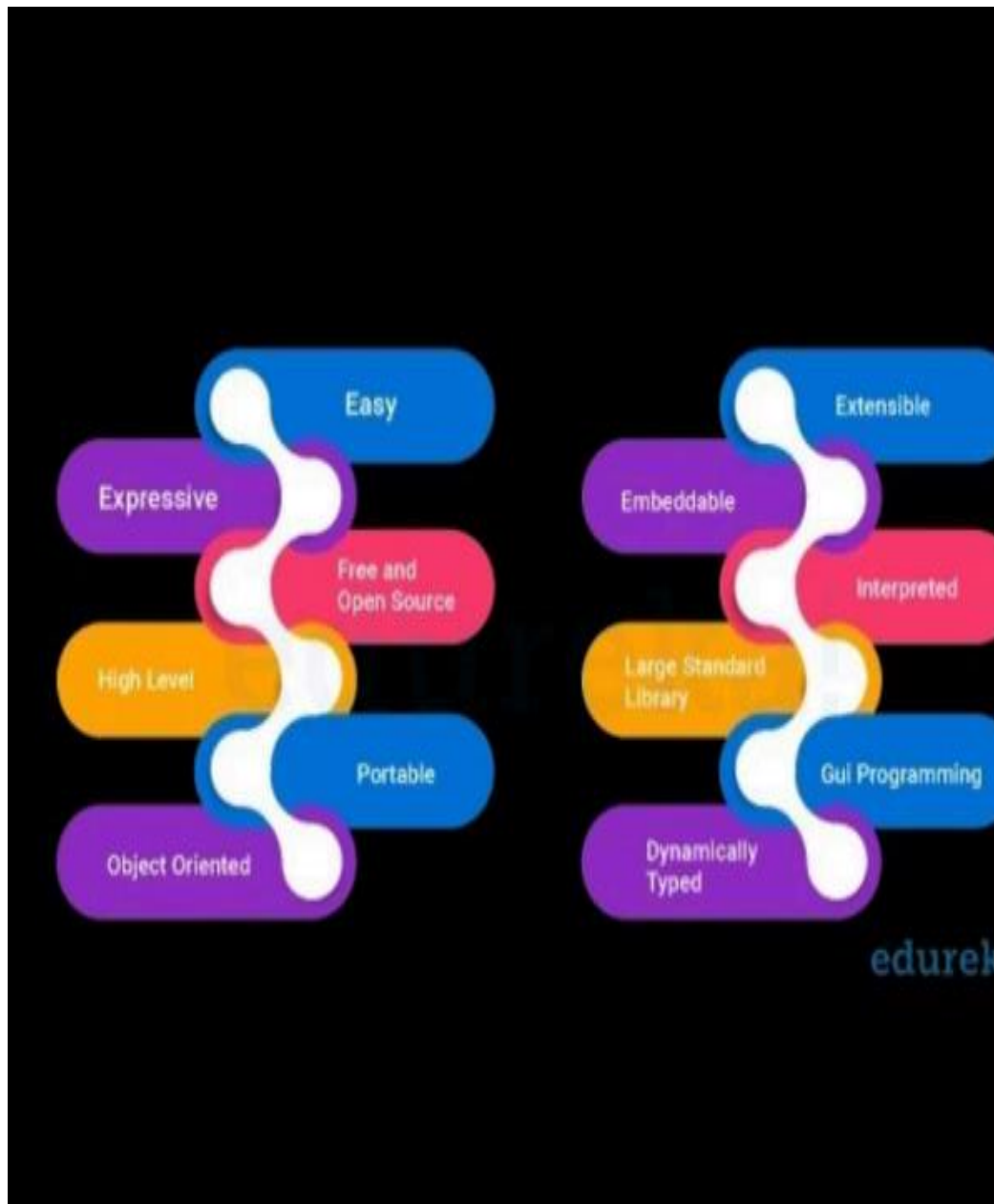


Fig 5.2 Features of python

As a programming language, the features of Python brought to the table are many. Some of the most significant features of Python are:

Easy to Code: Python is a very developer-friendly language which means that anyone and everyone can learn to code it in a couple of hours or days. As compared to other object-oriented programming languages like Java, C, C++, and C#, Python is one of the easiest to learn.

Open Source and Free: Python is an open-source programming language which means that anyone can create and contribute to its development. Python has an online forum where

thousands of coders gather daily to improve this language further. Along with this Python is free to download and use in any operating system, be it Windows, Mac or Linux.

Support for GUI: GUI or Graphical User Interface is one of the key aspects of any programming language because it has the ability to add flair to code and make the results more visual. Python has support for a wide array of GUIs which can easily be imported to the interpreter, thus making this one of the most favorite languages for developers.

Object-Oriented Approach: One of the key aspects of Python is its object-oriented approach. This basically means that Python recognizes the concept of class and object encapsulation thus allowing programs to be efficient in the long run.

High-Level Language: Python has been designed to be a high-level programming language, which means that when you code in Python you don't need to be aware of the coding structure, architecture as well as memory management.

Integrated by Nature: Python is an integrated language by nature. This means that the python interpreter executes codes one line at a time. Unlike other object-oriented programming languages, we don't need to compile Python code thus making the debugging process much easier and efficient. Another advantage of this is, that upon execution the Python code is immediately converted into an intermediate form also known as byte-code which makes it easier to execute and also saves runtime in the long run.

Highly Portable: Suppose you are running Python on Windows and you need to shift the same to either a Mac or a Linux system, then you can easily achieve the same in Python without having to worry about changing the code. This is not possible in other programming languages, thus making Python

5.3 PYTHON COMPILATION

Compilation Process in Python

Understanding Python Compilation

During the compilation process, the Python interpreter analyzes the code, checks for syntax errors, and generates an optimized representation of the code known as bytecode. Bytecode is a low-level representation of the code that is specific to the Python virtual machine (PVM). The PVM then executes the bytecode to perform the desired operations.



Fig 5.3 Compilation Process in Python

Differences between interpreted and compiled languages

Python is often referred to as an interpreted language because the Python interpreter executes the code line by line, interpreting and executing each statement as it encounters them. This interpretive nature of Python offers advantages like easy prototyping, dynamic typing, and runtime flexibility. However, it also introduces some performance overhead. In contrast, compiled languages, such as C++ or Java, go through a separate compilation step before execution. The source code is compiled into machine code specific to the target platform, resulting in faster execution. The compiled program can be directly executed by the computer without the need for an interpreter.

Advantages of compiling Python code

Improved Performance

Early Error Detection

Code Protection

Integration with Other Languages

5.4 SPRINT PLAN MEET



Introduction and Objectives

- Welcome everyone and provide an overview of the meeting's purpose.
- Outline the goals and objectives of the upcoming sprint.



Review of Previous Sprint

- Discuss the outcomes of the previous sprint.
- Review any unfinished tasks or issues that need to be carried over.



Product Backlog Review

- Review the product backlog items that are prioritized and ready for sprint planning.
- Discuss any changes in priorities or new backlog items.



Team Capacity and Velocity

- Review the team's capacity for the upcoming sprint (based on availability, skills, etc.)
- Discuss the team's velocity and past sprint performance metrics if applicable.



Selection of Sprint Goal

- Define a clear sprint goal that aligns with the overall project objectives.
- Ensure that the sprint goal is specific, achievable, and measurable. •



Task Breakdown and Estimation

- Break down the selected backlog items into specific tasks.
- Assign tasks to team members based on their skills and expertise.
- Estimate the effort (story points, hours, etc.) required for each task.



Definition of Done

- Review and confirm the Definition of Done (DoD) for the sprint.
- Ensure everyone understands the criteria that must be met for each task to be considered complete.



Dependencies and Risks

- Identify any dependencies between tasks or with external factors.
- Discuss potential risks that could impact the sprint's success.
- Develop mitigation plans for high-priority risks.



Sprint Timeline and Schedule

- Define the sprint duration (e.g., 1-4 weeks) and specific start/end dates.
- Confirm any planned milestones, checkpoints, or meetings within the sprint.



Tools and Communication

- Confirm the tools and systems (e.g., JIRA, Trello) to be used for task tracking and communication.
- Clarify how progress will be communicated (daily stand-ups, status reports, etc.).

5.5 BASIC CONCEPTS OF PYTHON

1. Variables and Data Types

Python uses dynamic typing, which means you don't need to declare variables explicitly. Here are examples of basic data types:

Integer

```
age = 25
```

Float height = 1.75

String

```
name = "Alice"
```

Boolean

```
is_student = True
```

2. Lists

Lists are ordered collections of items. They can contain elements of different types and are mutable (can be changed).

List of integers

```
numbers = [1, 2, 3, 4, 5]
```

List of strings

```
fruits = ["apple", "banana", "cherry"]
```

Mixed list

```
mixed_list = [10, "hello", True]
```

3. Control Flow Statements

If-Else Statements

```
x = 10
if x > 0:
    print("Positive number")
elif x < 0:
    print("Negative number")
else:
    print("Zero")
```

For Loop

```
# Iterating over a list
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

While Loop

```
# Looping until a condition is met
count = 0
while count < 5:
    print(count)
    count += 1
```

4. Functions

Functions in Python are defined using the def keyword. They can take parameters and return values.

```
# Function to calculate the square of a number
def square(x):
    return x * x
# Using the function
result = square(5)
print("Square of 5 is:", result)
```

5. Dictionaries

Dictionaries are unordered collections of key-value pairs.

```
# Creating a dictionary
person = {
    "name": "Alice",
    "age": 30,
    "city": "New York"
}

# Accessing values using keys
print("Name:", person["name"])
print("Age:", person["age"])
```

6. Classes and Objects

Python supports object-oriented programming. Here's a simple example of defining a class and creating objects:

```
# Class definition
class Car:
    def __init__(self, make, model):
        self.make = make
        self.model = model
    def display_info(self):
        print(f'Car: {self.make} {self.model}')
# Creating objects (instances) of the class Car
car1 = Car("Toyota", "Camry")
car2 = Car("Honda", "Accord")
```

Accessing object attributes and methods

```
car1.display_info()
```

```
car2.display_info()
```

Example Program: Calculating Factorial

Here's a complete example that calculates the factorial of a number using recursion:

```
def factorial(n):
```

```
    if n == 0:
```

```
        return 1
```

```
    else:
```

```
        return n * factorial(n - 1)
```

```
# Example usage
```

```
number = 5
```

```
fact = factorial(number)
```

```
print(f'Factorial of {number} is:', fact)
```

Explanation:

- **Variables and Data Types:** Shows how to declare variables and different data types (integer, float, string, boolean).
- **Lists:** Examples of creating lists and accessing list elements.
- **Control Flow Statements:** Includes if-else statements and loops (for and while).
- **Functions:** Defines a function to calculate the square of a number and demonstrates its usage.
- **Dictionaries:** Shows how to create dictionaries and access values using keys.
- **Classes and Objects:** Defines a Car class with attributes and a method, and creates instances of the class.
- **Example Program:** Demonstrates a factorial calculation function using recursion.

5.6 TASK ASSIGNED TO ME

Pc App for sending command to Device

This App will be useful when we want to send the commands to device remotely.

Steps to followed

- IMEI has to be entered
- Command has to be entered
- The Command will be sent to device remotely by this app.

Security is also taken care for this which has key and Aws certificates to send the commands.

UI View: -

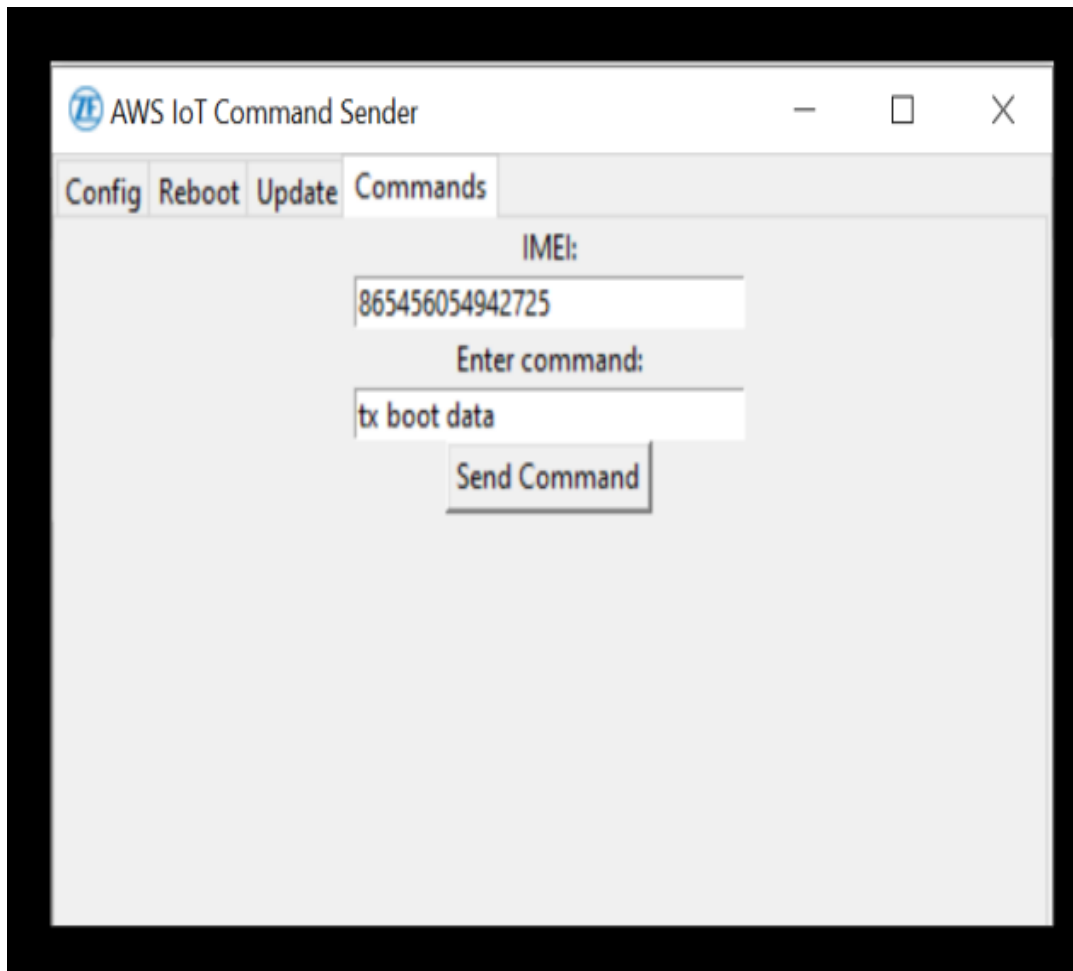


Fig 5.6 UI View

5.7 PROGRAMMING OF APPLICATION

The Version has to specify for the development of any application or program so it is necessary to maintain the new features to added with incremental version and bug fixes, etc...

The Packages and Modules list are as below to be used to achieve the goal are

```
AWSIoTMQTTClient // for the device IOT aonnectiivity
from PIL import Image, ImageTk // for handling the Image
from tkinter import ttk // to create the Window of UI in table view or required titles
#####
#ver 240417.1.1.0
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
from tkinter import *
from PIL import Image, ImageTk
import tkinter as tk
from tkinter import ttk
# Replace these values with your specific configurations
ca_file = 'AmazonRootCA1.pem'
cert_file = 'f3d819d7d3a363726369988a46e20d08f184b60a6b7f5cb7419aa6842ae598b7-
certificate.pem.crt'
key_file = 'f3d819d7d3a363726369988a46e20d08f184b60a6b7f5cb7419aa6842ae598b7-
private.pem.key'
mqtt_host = 'a9tto1qihmqxh-ats.iot.eu-west-1.amazonaws.com'
# Initialize MQTT client
mqtt_client = AWSIoTMQTTClient("MyClientID")
mqtt_client.configureEndpoint(mqtt_host,8883)
mqtt_client.configureCredentials(ca_file, key_file, cert_file)
# Connect to AWS IoT
mqtt_client.connect()
#To get the commad from the user to send the data to aws server subcribered to mentioned
topic
def send_command():
command = command_entry.get() # Get command from entry field
imei = imei_entry.get()
mqtt_topic=f'incoming/$/{imei}/$/'
```

```
payload = command
print("imei is:", imei)
print("command entered is:", command)
mqtt_client.publish(mqtt_topic, payload, 1)
# Create the Tkinter window
root = tk.Tk()
root.title("AWS IoT Command Sender")
# Open and resize the image
image = Image.open("ZF_logo.png")
image = image.resize((16, 16)) # Resize the image to fit as an icon
# Convert the image for Tkinter
tk_image = ImageTk.PhotoImage(image)
# Set the image as the window icon
root.iconphoto(True, tk_image)
notebook = ttk.Notebook(root)
notebook.pack(fill='both', expand=True)
# Tool tab (similar structure as Config)
tool_tab = Frame(notebook)
notebook.add(tool_tab, text='Config')
# Tool tab (similar structure as Reboot)
tool_tab = Frame(notebook)
notebook.add(tool_tab, text='Reboot')
# Tool tab (similar structure as Update)
tool_tab = Frame(notebook)
notebook.add(tool_tab, text='Update')
# Command tab
command_tab = Frame(notebook)
notebook.add(command_tab, text='Commands')
# UI Tab with 30mm width of entry box of the IMEI
command_label = Label(command_tab, text="IMEI:")
imei_entry = Entry(command_tab)
command_label.pack()
imei_entry = Entry(command_tab, width=30)
imei_entry.pack()
```

```
# UI Tab with 30mm width of entry box of the Command
command_label = Label(command_tab, text="Enter command:")
command_entry = Entry(command_tab)
command_label.pack()
command_entry = Entry(command_tab, width=30)
command_entry.pack()
# UI Button which will send the entry.
send_button = Button(command_tab, text="Send Command", command=send_command)
send_button.pack()
#To run in a loop
root.mainloop()
```

```
awsmqtt_subscribed(const sysData &sys){
    char buffer[SIZE_100];
    char data[SIZE_2040];
    snprintf(data, sizeof(data), "mosquitto_sub --cafile /userdata/apps/Electrace/%s --cert
    /userdata/apps/Electrace/%s --key /userdata/apps/Electrace/%s -h
    a9tto1qihmqxh-ats.iot.eu-west-1.amazonaws.com -t incoming/$/%s/$/%s"
    , sys.awsRootCA, sys.awsCert, sys.awsKey, sys.productImei);
    while(true){
        FILE *pipe_fp = NULL;
        // Open mosquitto_sub command as a pipe
        pipe_fp = popen(data, "r");
        if (pipe_fp == NULL) {
            perror("popen");
        }

        // Read messages from the pipe and print them
        while (pipe_fp != NULL && fgets(buffer, sizeof(buffer), pipe_fp) != NULL) {
            printf("Received message: %s", buffer); // Print or process the message as
            required
            system(buffer);
        }
        // Close the pipe
        if (pipe_fp != NULL)
            pclose(pipe_fp);
    }
}
```

Fig 5.7 Root main loop

5.8 Testing and deployment

The certificate and Topics should be in device end must be placed and then take the IMEI number,

The Device must be SUBSCRIBE to given topic then only it will receive the Data sent From the App.

Trail 1

INPUT from the APP:-

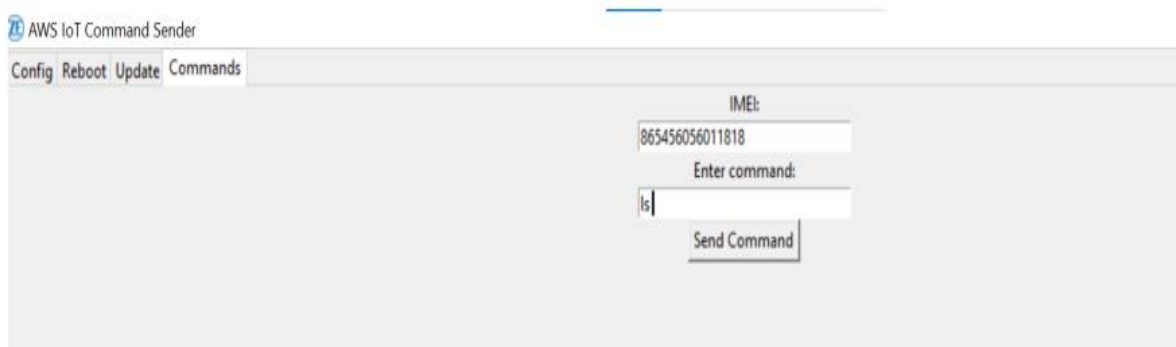


Fig 5.8.1 Input from APP

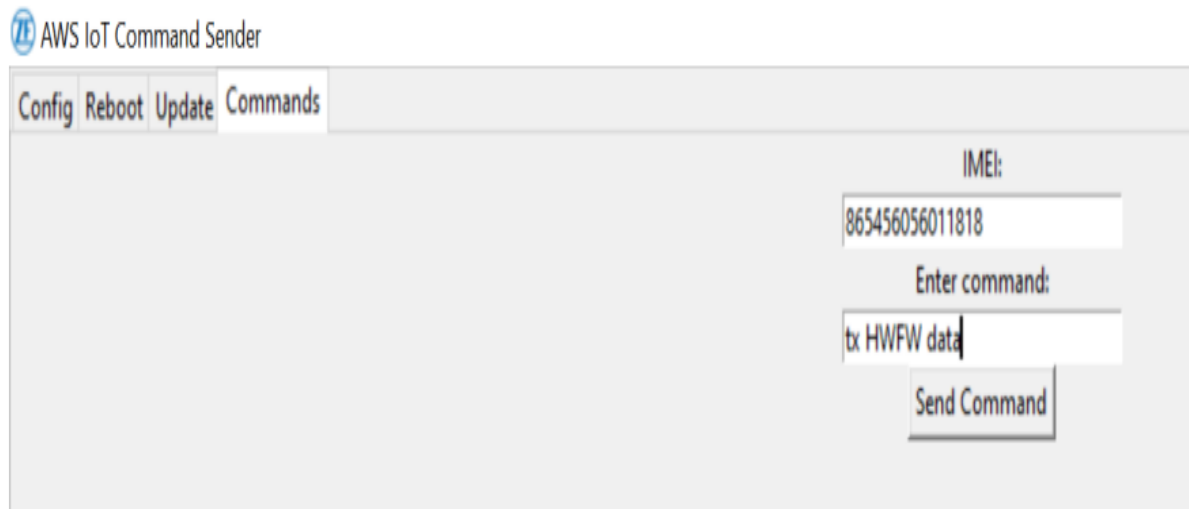
OUTPUT IN THE DEVICE END:



Fig 5.8.2 Output in device end

Trail 2

INPUT: -

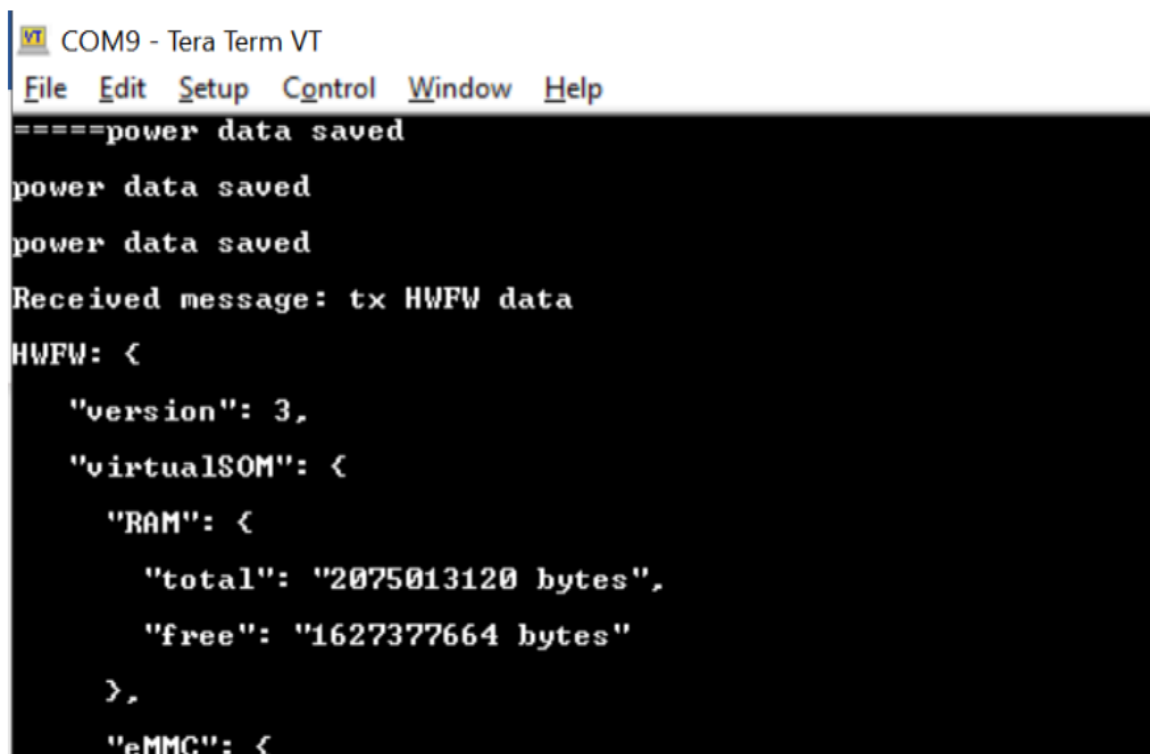


The screenshot shows the 'AWS IoT Command Sender' application window. It has a menu bar with 'Config', 'Reboot', 'Update', and 'Commands'. The 'Commands' tab is active. On the right side, there is a form with the following fields and buttons:

- IMEI: A text box containing the value '865456056011818'.
- Enter command: A text box containing the value 'tx HFWF data'.
- Send Command: A button located below the 'Enter command' text box.

Fig 5.8.3 Input from APP

OUTPUT:



The screenshot shows a Tera Term VT terminal window titled 'COM9 - Tera Term VT'. The terminal displays the following output:

```
====power data saved
power data saved
power data saved
Received message: tx HFWF data
HFWF: <
  "version": 3,
  "virtualSOM": <
    "RAM": <
      "total": "2075013120 bytes",
      "free": "1627377664 bytes"
    },
  "eMMC": <
```

Fig 5.8.4 Output

CHAPTER 6

SPECIFIC OUTCOMES (REFLECTION NOTES)

6.1 Introduction to Python

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently. There are two major Python versions: Python 2 and Python 3. Both are quite different.

Python is a programming language that is widely used in web applications, software development, data science, and machine learning (ML). Developers use Python because it is efficient and easy to learn and can run on many different platforms. Python software is free to download, integrates well with all types of systems, and increases development speed.

6.2 APPLICATION OF PYTHON

Web and Internet Development

- Frameworks such as Django and Pyramid.
- Micro-frameworks such as Flask and Bottle.
- Advanced content management systems such as Plone and django CMS.
- Python's standard library supports many Internet protocols:
- HTML and XML, E-mail processing
- Support for FTP, IMAP, and other Internet protocols.
- Easy-to-use socket interface
- Requests, a powerful HTTP client library.
- BeautifulSoup, an HTML parser that can handle all sorts of oddball HTML.
- Feedparser for parsing RSS/Atom feeds.

6.3 BASIC CONCEPTS OF PYTHON

Variables and Data Types

Lists

Control Flow Statements

If-Else Statements

For Loop

While Loop

Functions

Dictionaries

Classes and Object

CONCLUSION

Python Development:

Python has revolutionized the world of programming with its simplicity, flexibility, and versatility. Its vast range of libraries and frameworks makes it an ideal choice for various applications, including web development, data analysis, machine learning, and automation. Python's large community and extensive resources ensure that developers can easily find support and stay updated with the latest trends. As technology advances, Python's adaptability and scalability will continue to make it a top choice for developers.

Backend Development:

Backend development forms the backbone of any web application, providing the server-side logic, database integration, and API connectivity. With the rise of web technologies, backend development has become increasingly important, requiring expertise in programming languages, frameworks, and databases. Python, with its popular frameworks like Django and Flask, has become a leading choice for backend development, offering a perfect blend of simplicity, efficiency, and scalability. As web applications continue to evolve, backend development will remain a critical component, driving innovation and powering digital experiences.

REFERENCE

Python Development

1. A. Sweigart, *Automate the Boring Stuff with Python*. San Francisco: No Starch Press, 2015.
2. University of Michigan, “Python for Everybody,” Coursera, 2023. [Online]. Available: <https://www.coursera.org/specializations/python>. [Accessed: 31-Jul-2024].
3. University of Michigan, “Python 3 Programming Specialization,” Coursera, 2023. [Online]. Available: <https://www.coursera.org/specializations/python-3-programming>. [Accessed: 31-Jul-2024].
4. J. Portilla, “Complete Python Bootcamp: Go from zero to hero in Python 3,” Udemy, 2023. [Online]. Available: <https://www.udemy.com/course/complete-python-bootcamp/>. [Accessed: 31-Jul-2024].
5. Python Software Foundation, “Python Documentation,” Python.org, 2023. [Online]. Available: <https://docs.python.org/3/>. [Accessed: 31-Jul-2024].
6. Real Python, “Real Python Tutorials,” RealPython.com, 2023. [Online]. Available: <https://realpython.com/>. [Accessed: 31-Jul-2024].

Backend Development

1. M. Kleppmann, *Designing Data-Intensive Applications*. Sebastopol: O’Reilly Media, 2017.
2. University of Michigan, “Django for Everybody,” Coursera, 2023. [Online]. Available: <https://www.coursera.org/specializations/django-for-everybody>. [Accessed: 31-Jul-2024].
3. A. Shaw, “Web Development with Node.js and Express,” Udemy, 2023. [Online]. Available: <https://www.udemy.com/course/the-complete-web-development-bootcamp/>. [Accessed: 31-Jul-2024].
4. J. Portilla, “Backend Development with Python & Django,” Udemy, 2023. [Online]. Available: <https://www.udemy.com/course/python-and-django-full-stack-web-developer-bootcamp/>. [Accessed: 31-Jul-2024].
5. Django Software Foundation, “Django Documentation,” Djangoproject.com, 2023. [Online]. Available: <https://docs.djangoproject.com/en/stable/>. [Accessed: 31-Jul-2024].