BLOCKCHAIN ENGINEERING PRACTICAL-1.

NAME: AKSHITHA

ROLL NO:2303A51360

BATCH NO:29

QUESTION 1:

Objective:

To learn blockchain interaction by creating a cryptocurrency wallet, checking wallet balance, and simulating transactions using Python and Web3

Requirements:

• Install Python 3.x

• Set up VS Code with Python extension

• Install required Python libraries:

• pip install web3

• Use a test blockchain network (Ethereum Sepolia / Ganache local blockchain)

• Basic understanding of blockchain wallets and private keys

Practical Description:

Step 1: Environment Setup

• Install Python and VS Code

• Install Web3.py library

• Create a Python file named wallet_interaction.py

Step 2: Wallet and Blockchain Interaction Script

Create a Python script that:

• Connects to a blockchain network

• Loads a wallet using a private key

• Fetches wallet address

• Checks wallet balance

• Demonstrates transaction preparation (without real funds)

CODE:

```
import tkinter as tk
from tkinter import messagebox
from web3 import Web3

# ---------------- Blockchain Setup ----------------
GANACHE_URL = "http://127.0.0.1:7545"

try:
    web3 = Web3(Web3.HTTPProvider(GANACHE_URL))
    connected = web3.is_connected()
except:
    connected = False

# Replace with Ganache account if online
SAMPLE_ADDRESS =
"0x0000000000000000000000000000000000000000"
```

```
# ---------------- Functions ----------------
def check_balance():
    if connected:
        try:
            wei_balance = web3.eth.get_balance(SAMPLE_ADDRESS)
            balance_eth = web3.from_wei(wei_balance, 'ether')
            formatted_balance = f"{balance_eth:.2f}"

            messagebox.showinfo(
                "Wallet Balance",
                f"Connected to Ganache\n\n"
                f"Wallet Address:\n{SAMPLE_ADDRESS}\n\n"
                f"Balance: {formatted_balance} ETH"
            )
        except Exception as e:
            messagebox.showerror("Error", str(e))
    else:
        # Simulation Mode
        balance_eth = 10
        formatted_balance = f"{balance_eth:.2f}"

        messagebox.showinfo(
```

```
            "Wallet Balance",
            f"Simulation Mode (Offline)\n\n"
            f"Simulated Balance: {formatted_balance} ETH"
        )

def simulate_transaction():
    if connected:
        messagebox.showinfo(
            "Transaction",
            "Connected Mode\n\nTransaction feature can be added here."
        )
    else:
        messagebox.showinfo(
            "Transaction",
            "Simulation Mode\n\nTransaction simulated successfully!"
        )

# ---------------- GUI Setup ----------------
root = tk.Tk()
root.title("Blockchain Wallet Simulator")
root.geometry("350x300")
root.resizable(False, False)
```

```
# Title
title_label = tk.Label(
    root,
    text="Blockchain Wallet (Python + Web3)",
    font=("Arial", 12, "bold")
)
title_label.pack(pady=10)

# Status
status_text = "Connected to Ganache" if connected else "Simulation Mode (Offline)"
status_color = "green" if connected else "red"

status_label = tk.Label(
    root,
    text=status_text,
    fg=status_color,
    font=("Arial", 10, "bold")
)
status_label.pack(pady=5)

# Buttons
btn_balance = tk.Button(
```

```
    root,
    text="Check Wallet Balance",
    width=25,
    command=check_balance
)
btn_balance.pack(pady=10)

btn_transaction = tk.Button(
    root,
    text="Simulate Transaction",
    width=25,
    command=simulate_transaction
)
btn_transaction.pack(pady=10)

btn_exit = tk.Button(
    root,
    text="Exit",
    width=25,
    command=root.quit
)
btn_exit.pack(pady=10)
```

Run app

root.mainloop()

OUTPUT: