

ASSIGNMENT-2

NAME-AKSHITHA

ROLL_NO:2303A51360

BATCH-29

QUESTION:

A smart contract is a self-executing program stored on the blockchain.

The Simple Storage contract is a beginner-level Solidity contract

that allows users to:

- Store a value on the blockchain
- Retrieve the stored value

-interface should contain:

- 1.money send
- 2.previous Hash
- 3.Current Hash
- 4.Transaction output

CODE:

```
import tkinter as tk
```

```
import hashlib
```

```
import time
```

```
# Blockchain-style variables
```

```
money_received = 0
```

```
previous_hash = "GENESIS_HASH"
```

```
current_hash = "GENESIS_HASH"

def generate_hash(amount, timestamp):
    data = f"{amount}{timestamp}{previous_hash}"
    return hashlib.sha256(data.encode()).hexdigest()

def send_money():
    global money_received, previous_hash, current_hash

    try:
        amount = float(entry_amount.get())
    except ValueError:
        label_status.config(text="Enter a valid number!")
        return

    money_received += amount

    previous_hash = current_hash
    current_hash = generate_hash(amount, time.time())

    label_received.config(text=f"{money_received} ETH")
```

```
label_prev_hash.config(text=previous_hash)

label_curr_hash.config(text=current_hash)

label_status.config(text="Transaction Successful ✓ ")

entry_amount.delete(0, tk.END)

# GUI Window

window = tk.Tk()

window.title("Simple Storage Blockchain App")

window.geometry("450x500")

# Heading

tk.Label(window, text="SMART STORAGE BLOCKCHAIN APP",
        font=("Arial", 14, "bold")).pack(pady=10)

# Money input

tk.Label(window, text="Money to Send").pack()

entry_amount = tk.Entry(window)

entry_amount.pack(pady=5)

# Button
```

```
tk.Button(window, text="Send Money",
command=send_money).pack(pady=10)

# Display fields

tk.Label(window, text="Received Money").pack()

label_received = tk.Label(window, text="0 ETH")

label_received.pack(pady=5)

tk.Label(window, text="Previous Hash").pack()

label_prev_hash = tk.Label(window, text=previous_hash,
wraplength=400)

label_prev_hash.pack(pady=5)

tk.Label(window, text="Current Hash").pack()

label_curr_hash = tk.Label(window, text=current_hash,
wraplength=400)

label_curr_hash.pack(pady=5)

# Status

label_status = tk.Label(window, text="")

label_status.pack(pady=10)

window.mainloop()
```

OUTPUT:

The screenshot shows the Visual Studio Code interface with the following details:

- Left Sidebar (Explorer):** Shows the project structure with files like `Simple Storage smart.py`, `.venv`, `BLOCKCHAIN`, and `pyvenv.cfg`.
- Open Editors:** The main editor shows the Python code for the "Simple Storage smart" application. The code uses Tkinter to create a GUI window titled "SMART STORAGE BLOCKCHAIN APP". It includes fields for "Money to Send", a "Send Money" button, and labels for "Received Money", "Previous Hash", "Current Hash", and "Status". The status label displays the text "GENESIS_HASH".
- Bottom Status Bar:** Shows the file name "Simple Storage smart.py", line count "Ln 12, Col 49", and other development information.
- Bottom Right Panel:** Displays a "Build with Agent" section with AI-related instructions.
- Top Right Panel:** Shows recent sessions and a PowerShell pip command history.

```
# GUI Window
window = tk.Tk()
window.title("Simple Storage Blockchain App")
window.geometry("450x500")

# Heading
tk.Label(window, text="SMART STORAGE BLOCKCHAIN APP",
        font=("Arial", 14, "bold")).pack(pady=10)

# Money input
tk.Label(window, text="Money to Send").pack()
entry_amount = tk.Entry(window)
entry_amount.pack(pady=5)

# Button
tk.Button(window, text="Send Money", command=send_money).pack(pady=10)

# Display fields
tk.Label(window, text="Received Money").pack()
label_received = tk.Label(window, text="0 ETH")
label_received.pack(pady=5)

tk.Label(window, text="Previous Hash").pack()
label_prev_hash = tk.Label(window, text=previous_hash, wraplength=400)
label_prev_hash.pack(pady=5)

tk.Label(window, text="Current Hash").pack()
label_curr_hash = tk.Label(window, text=current_hash, wraplength=400)
label_curr_hash.pack(pady=5)

# status
label_status = tk.Label(window, text="")
label_status.pack(pady=10)

window.mainloop()
```