

## ASSIGNMENT-5:

2303A51600

BATCH-29

NAME-ARSHA VARDHINI

PROBLEM STMT:

TO DEVELOP AND PERFORM UNIT TESTING OF SMART CONTRACTS USING THE TRUFFLE FRAMEWORK

-PROBLEM STATEMENT (EQUIVALENT TO GIVEN CIPHER PRACTICAL)

DEVELOP A BASIC PERSONAL PORTFOLIO SMART CONTRACT THAT STORES AND RETRIEVES USER PROFILE DATA, AND VALIDATE ITS BEHAVIOR USING TRUFFLE UNIT TESTS

CODE WITH OUTPUT:

The screenshot shows a code editor with several tabs open. The main tab contains Python code for a Tkinter-based application. The code defines a class with methods for storing profiles, getting profiles, sending ether, and checking balances. It uses Tkinter's `Label` and `Entry` widgets to create a simple user interface. To the right of the code editor, a window titled "Blockchain Portfolio Interface" is displayed. This window has fields for Name (Arsha), Email (arsha@gmail.com), Skills (Blockchain, AI, ML), and a Receiver Address (b2C4E3F7D1A9B5C6E4F1234567890ABCDEF). It also has an "Amount (ETH)" input field and a message "Profile stored successfully!". Below the input fields are four buttons: "Store Profile", "Get Profile", "Send Ether", and "Check Balance". At the bottom of the code editor, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The DEBUG CONSOLE tab shows a command-line prompt: PS C:\Users\ARSHA THALLAPALLY\Desktop\Blockchain> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/Blockchain/blockchain\_app.py". The bottom right corner of the screen shows a taskbar with several Python-related icons.

```
tk.Label(app, text="Amount (ETH)").pack()
amount_entry = tk.Entry(app)
amount_entry.pack()

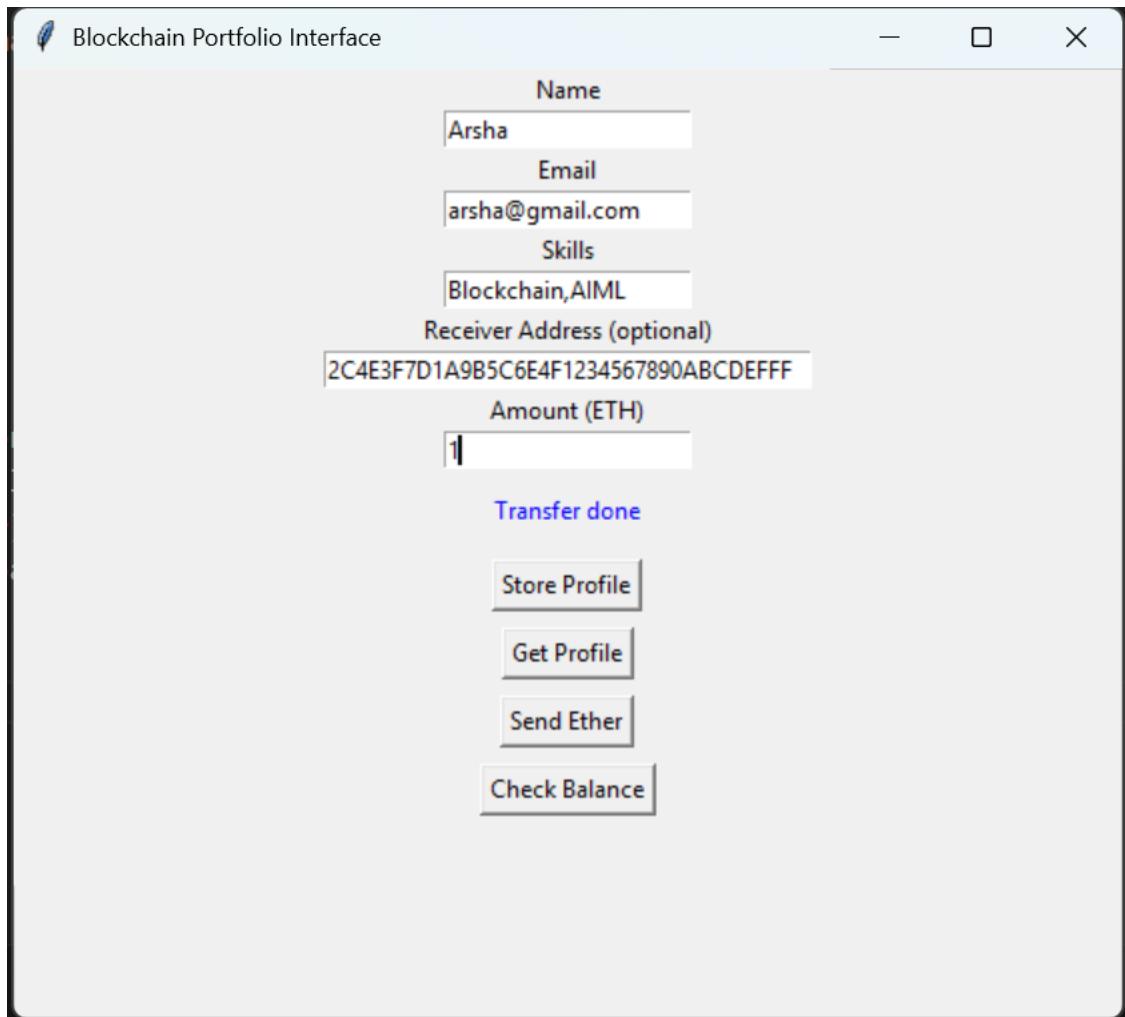
result = tk.Label(app, text="", fg="blue")
result.pack(pady=10)

def store_profile():
    profile["name"] = name_entry.get()
    profile["email"] = email_entry.get()
    profile["skills"] = skills_entry.get()
    result.config(text="Profile stored successfully!")

def get_profile():
    result.config(text=f"Name: {profile['name']}\nEmail: {profile['email']}")

def send_ether():
    if use_blockchain:
        result.config(text="Ether transfer simulated (Ganache required)")
    else:
        result.config(text="Blockchain not connected. Transfer simulated")
```

## GET PROFILE:



## Send Eth:

