

ASSIGNMENT – 4

NAME: SRAVANI

HT NO:2303A510G7

BATCH NO:30

Objective

To design, implement, and deploy a basic ERC-20 compliant token smart contract using Solidity on the Ethereum blockchain.

Output:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows a folder named "BLOCK CHAIN ASSIGNMENTS" containing files "Assignment-4.py" and "BLOCK CHAIN ASSIGNMENT-3-1027.pdf".
- Code Editor:** Displays the content of "Assignment-4.py". The code uses Tkinter to create a window titled "ERC20 Token Generator" with labels for "Token Name", "Token Symbol", and "Total Supply". It includes a "Generate" function that selects a random token from a list and configures the labels.
- Terminal:** Shows the command line output of running the script: "PS E:\User\Jashwanth\desktop\BLOCK CHAIN ASSIGNMENTS> & C:\Users\Jashwanth\AppData\Local\Microsoft\WindowsApps\python3.12.exe "E:\User\Jashwanth\desktop\BLOCK CHAIN ASSIGNMENTS\Assignment-4.py"
- Output Panel:** Shows the output of the Python script execution.
- Bottom Status Bar:** Displays file information (Line 38, Col 16), workspace settings (Spaces: 4, UTRF: 8, Auto: 1), and system information (3.12.10 Microsoft Store).

The screenshot shows a Python code editor with the following details:

- File Explorer:** Shows files like "Assignment-4.py", "BLOCK CHAIN ASSIGNMENT 3-10...", "Block Chain Assignment 1-10G.pdf", and "BLOCKCHAIN ASS-2-10G.pdf".
- Code Editor:** The main pane displays the code for "Assignment-4.py":

```
Assignment-4.py
import tkinter as tk
import random

root = tk.Tk()
root.title("ERC20 Token Generator")
root.geometry("380x250")

root.wm_title("ERC20 Token Generator")
root.wm_iconbitmap("ICO.ico")
root.wm_resizable(0, 0)

root.config(bg="white")

label_name = tk.Label(root, text="Token Name: ", font=("Arial", 14, "bold"))
label_name.grid(row=0, column=0, sticky="E", padx=5, pady=5)

label_symbol = tk.Label(root, text="Token Symbol: ", font=("Arial", 14, "bold"))
label_symbol.grid(row=1, column=0, sticky="E", padx=5, pady=5)

label_supply = tk.Label(root, text="Total Supply: ", font=("Arial", 14, "bold"))
label_supply.grid(row=2, column=0, sticky="E", padx=5, pady=5)

button_generate = tk.Button(root, text="Generate", font=("Arial", 14))
button_generate.grid(row=3, column=0, sticky="E", padx=5, pady=5)

def generate_token():
    token_names = ["MyToken", "AlphaCoin", "BlockCoin", "CryptoToken", "SmartToken"]
    symbols = ["MTK", "ALP", "BLK", "CTC", "SMT"]

    name = random.choice(token_names)
    symbol = random.choice(symbols)
    supply = random.randint(1000, 1000000)

    name_label.config(text=f"(Token Name: {name})")
    symbol_label.config(text=f"(Token Symbol: {symbol})")
    supply_label.config(text=f"(Total Supply: {supply})")

name_label = tk.Label(root, text="Token Name: ", font=("Arial", 11))
symbol_label = tk.Label(root, text="Token Symbol: ", font=("Arial", 11))
supply_label = tk.Label(root, text="Total Supply: ", font=("Arial", 11))

name_label.grid(row=0, column=1, sticky="W", padx=5, pady=5)
symbol_label.grid(row=1, column=1, sticky="W", padx=5, pady=5)
supply_label.grid(row=2, column=1, sticky="W", padx=5, pady=5)

button_generate.config(command=generate_token)

root.mainloop()
```

- Terminal:** The terminal shows command-line output for running the script:

```
PS E:\User\Jashwanth\Desktop\BLOCK CHAIN ASSIGNMENTS & C:\Users\Jashwanth\AppData\Local\Microsoft\WindowsApps\python3.12.exe e:/User/Jashwanth/Desktop/BLOCK CHAIN ASSIGNMENTS\assignment-4.py
PS E:\User\Jashwanth\Desktop\BLOCK CHAIN ASSIGNMENTS & C:\Users\Jashwanth\AppData\Local\Microsoft\WindowsApps\python3.12.exe e:/User/Jashwanth/Desktop/BLOCK CHAIN ASSIGNMENTS\assignment-4.py"
```

- Build with Agent:** A sidebar on the right provides options to onboard AI into the codebase.