SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE			DEPARTMENT OF COMPUTER SCIENCE ENGINEERING		
Program Name: B. Tech		**Assignment Type:** Lab		**Academic Year:** 2025-26	
Course Coordinator Name		Dr. Jagjit Singh Dhatterwal			
Instructor(s) Name		Dr. Jagjit Singh Dhatterwal			
Course Code	23CS201PE401	**Course Title**	Blockchain Engineering		
Year/Sem	III/II	**Regulation**	R25		
Date and Day of Assignment	19-01-2026	**Time(s)**	9:00AM to 11:00AM		
Duration	2 Hours	**Applicable to Batches**	(23CSBTB19, 23CSBTB20 23CSBTB21, 23CSBTB22 23CSBTB23, 23CSBTB24 23CSBTB25, 23CSBTB26)		
Assignment Number: 03/12					

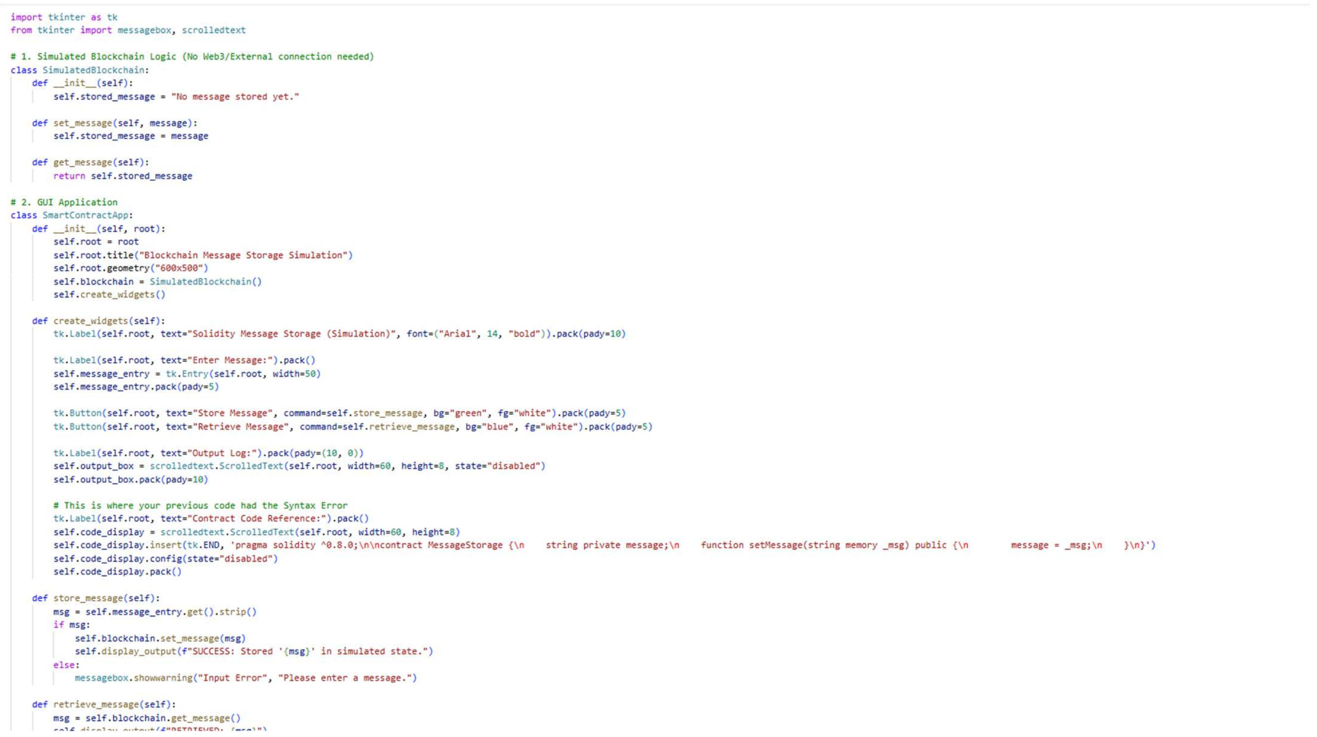
Q. No.	Question	Expected Time to complete
1	**Objective** To understand Solidity smart contract fundamentals by building a simple Ethereum contract that stores and retrieves user-provided messages securely on the blockchain. **Problem Statement** Develop a basic Solidity smart contract that allows users to: • Store a message on the blockchain • Update the message • Retrieve the stored message This practical helps understand **state variables, functions, constructors, and data types** in Solidity. **Requirements** **Software Requirements** • VS Code • Solidity Extension (by Juan Blanco) • Remix IDE (Online: https://remix.ethereum.org) • Web Browser (Chrome/Edge) **Practical Implementation** **Step 1: Set Up Solidity Environment**	

1. Open **VS Code**
2. Install **Solidity Extension**
3. Open **Remix IDE**
4. Create a new file named:
5. `MessageStorage.sol`

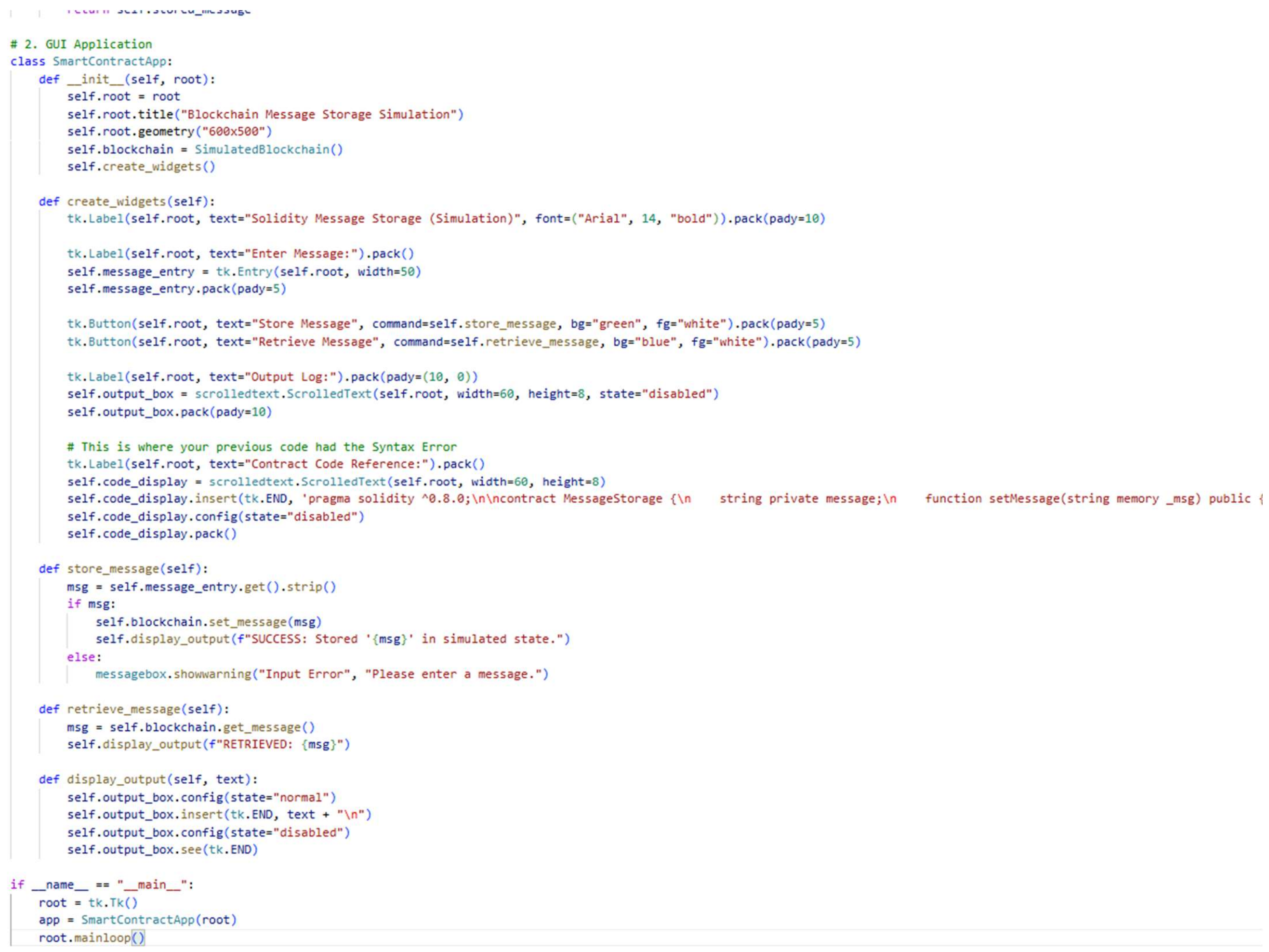
Step 2: Write the Solidity Smart Contract

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

/*
    Contract Name: MessageStorage
    Purpose: Store and retrieve messages on
blockchain
*/

contract MessageStorage {

    // State variable to store message
    string private message;

    // Constructor to set initial message
    constructor(string memory _message) {
        message = _message;
    }

    // Function to update message
    function setMessage(string memory
_newMessage) public {
        message = _newMessage;
    }

    // Function to retrieve message
    function getMessage() public view
returns (string memory) {
        return message;
    }
}
```

Code:



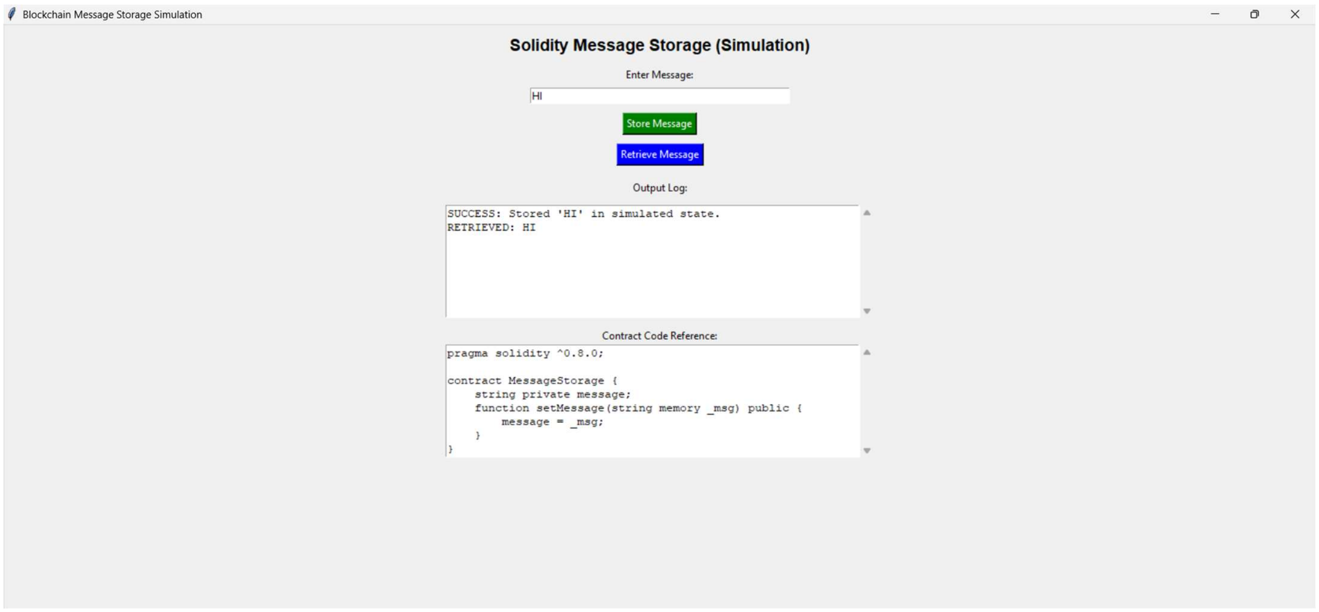
```python
import tkinter as tk
from tkinter import messagebox, scrolledtext

# 1. Simulated Blockchain Logic (No Web3/External connection needed)
class SimulatedBlockchain:
    def __init__(self):
        self.stored_message = "No message stored yet."

    def set_message(self, message):
        self.stored_message = message

    def get_message(self):
        return self.stored_message

# 2. GUI Application
class SmartContractApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Blockchain Message Storage Simulation")
        self.root.geometry("600x500")
        self.blockchain = SimulatedBlockchain()
        self.create_widgets()

    def create_widgets(self):
        tk.Label(self.root, text="Solidity Message Storage (Simulation)", font=("Arial", 14, "bold")).pack(pady=10)

        tk.Label(self.root, text="Enter Message:").pack()
        self.message_entry = tk.Entry(self.root, width=50)
        self.message_entry.pack(pady=5)

        tk.Button(self.root, text="Store Message", command=self.store_message, bg="green", fg="white").pack(pady=5)
        tk.Button(self.root, text="Retrieve Message", command=self.retrieve_message, bg="blue", fg="white").pack(pady=5)

        tk.Label(self.root, text="Output Log:").pack(pady=(10, 0))
        self.output_box = scrolledtext.ScrolledText(self.root, width=60, height=8, state="disabled")
        self.output_box.pack(pady=10)

        # This is where your previous code had the Syntax Error
        tk.Label(self.root, text="Contract Code Reference:").pack()
        self.code_display = scrolledtext.ScrolledText(self.root, width=60, height=8)
        self.code_display.insert(tk.END, 'pragma solidity ^0.8.0;\n\ncontract MessageStorage {\n    string private message;\n    function setMessage(string memory _msg) public {\n        message = _msg;\n    }\n}')
        self.code_display.config(state="disabled")
        self.code_display.pack()

    def store_message(self):
        msg = self.message_entry.get().strip()
        if msg:
            self.blockchain.set_message(msg)
            self.display_output(f"SUCCESS: Stored '{msg}' in simulated state.")
        else:
            messagebox.showwarning("Input Error", "Please enter a message.")

    def retrieve_message(self):
        msg = self.blockchain.get_message()
        self.display_output(f"RETRIEVED: {msg}")
```



```python
        return self.stored_message

# 2. GUI Application
class SmartContractApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Blockchain Message Storage Simulation")
        self.root.geometry("600x500")
        self.blockchain = SimulatedBlockchain()
        self.create_widgets()

    def create_widgets(self):
        tk.Label(self.root, text="Solidity Message Storage (Simulation)", font=("Arial", 14, "bold")).pack(pady=10)

        tk.Label(self.root, text="Enter Message:").pack()
        self.message_entry = tk.Entry(self.root, width=50)
        self.message_entry.pack(pady=5)

        tk.Button(self.root, text="Store Message", command=self.store_message, bg="green", fg="white").pack(pady=5)
        tk.Button(self.root, text="Retrieve Message", command=self.retrieve_message, bg="blue", fg="white").pack(pady=5)

        tk.Label(self.root, text="Output Log:").pack(pady=(10, 0))
        self.output_box = scrolledtext.ScrolledText(self.root, width=60, height=8, state="disabled")
        self.output_box.pack(pady=10)

        # This is where your previous code had the Syntax Error
        tk.Label(self.root, text="Contract Code Reference:").pack()
        self.code_display = scrolledtext.ScrolledText(self.root, width=60, height=8)
        self.code_display.insert(tk.END, 'pragma solidity ^0.8.0;\n\ncontract MessageStorage {\n    string private message;\n    function setMessage(string memory _msg) public {
        self.code_display.config(state="disabled")
        self.code_display.pack()

    def store_message(self):
        msg = self.message_entry.get().strip()
        if msg:
            self.blockchain.set_message(msg)
            self.display_output(f"SUCCESS: Stored '{msg}' in simulated state.")
        else:
            messagebox.showwarning("Input Error", "Please enter a message.")

    def retrieve_message(self):
        msg = self.blockchain.get_message()
        self.display_output(f"RETRIEVED: {msg}")
```

```python
    def display_output(self, text):
        self.output_box.config(state="normal")
        self.output_box.insert(tk.END, text + "\n")
        self.output_box.config(state="disabled")
        self.output_box.see(tk.END)

if __name__ == "__main__":
    root = tk.Tk()
    app = SmartContractApp(root)
    root.mainloop()
```

Output:



```
SUCCESS: Stored 'HI' in simulated state.
RETRIEVED: HI
```

```
pragma solidity ^0.8.0;

contract MessageStorage {
    string private message;
    function setMessage(string memory _msg) public {
        message = _msg;
    }
}
```