Centurion University
Shaping Lives... Empowering Communities...

School: ..... Campus: .....

Academic Year: ..... Subject Name: ..... Subject Code: .....

Semester: ..... Program: ..... Branch: ..... Specialization: .....

Date: .....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement : Contract QA – Testing Smart Contracts

Objective/Aim:

To understand and identify common vulnerabilities in Ethereum smart contracts, perform basic security auditing using Remix IDE, and demonstrate how to prevent these issues with secure coding practices.

Apparatus/Software Used:

1. **Remix IDE**
2. **MetaMask Wallet**
3. **Solidity Compiler (0.8.x)**
4. **Ethereum Testnet (Sepolia**

Theory/Concept:

A **Smart Contract Audit** is a process of examining code to identify **security flaws, logic errors, and potential vulnerabilities** before deployment on the blockchain.

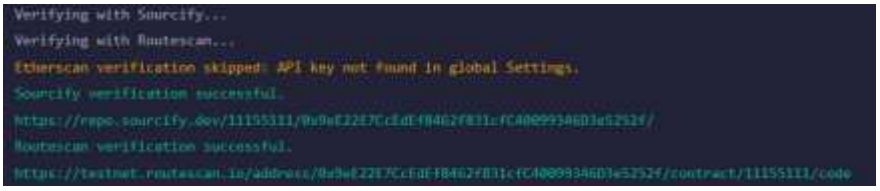
Since smart contracts are **immutable**, even a small bug can lead to major losses.

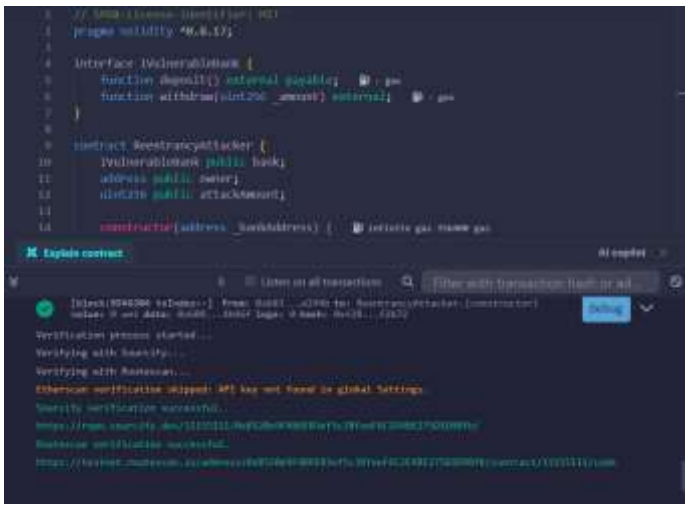
Common **smart contract vulnerabilities** include:

Vulnerability	Description	Example Issue
Reentrancy Attack	Exploiting repeated calls before state updates	DAO Hack (2016)
Integer Overflow/Underflow	Arithmetic beyond uint range	Incorrect token balance
Unchecked External Calls	Calling untrusted contracts without validation	Malicious contract exploit
Timestamp Dependency	Using block.timestamp to control logic	Predictable mining manipulation
Front Running	Miners exploit pending transaction data	Priority gas auctions
Unrestricted Access Control	Missing onlyOwner modifiers	Anyone can perform admin actions

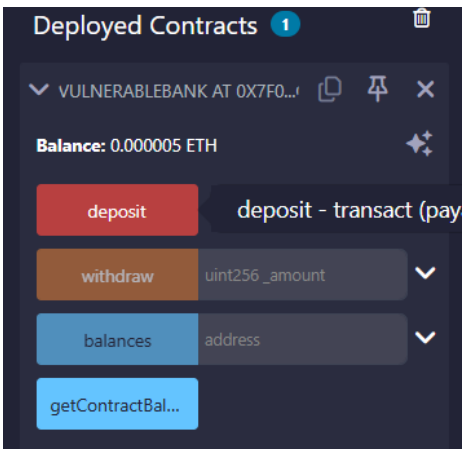
procedure

- Setup Environment
  - Open Remix IDE → Create new file VulnerableContract.sol.
- Write a Vulnerable Contract (Example – Reentrancy):





- **Deploy and Test on Remix:**
  - Deploy the contract.
  - Deposit some ETH from one account.
  - Call withdraw() multiple times from a malicious contract that re-enters the function.



- **Observe Unexpected Behavior:**
  - Balance drained due to reentrancy before state update.
- **Recompile and Redeploy.**
  - Reentrancy issue resolved.

Observation Table:

Step	Action	Result
1	Deployed vulnerable contract	Worked as expected
2	Performed reentrancy attack	Funds drained unexpectedly
3	Modified code with safety pattern	Reentrancy prevented
4	Verified balances	Secure and stable behavior

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty: