



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 Experiment :

### \* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Algorithm:

1. Write a smart contract that imports Chainlink's price feed interface.
2. Initialize a reference to the desired data feed (ETH/USD).
3. Deploy the contract on the Sepolia testnet using Remix + MetaMask.
4. Call the smart contract function to get the latest ETH/USD price from the Chainlink oracle.
5. Observe and verify the live data is fetched from off-chain and stored on-chain.

### \* Softwares used

1. Remix IDE
2. MetaMask Wallet (Sepolia testnet)
3. Chainlink Data Feeds
4. Solidity 0.8.x.

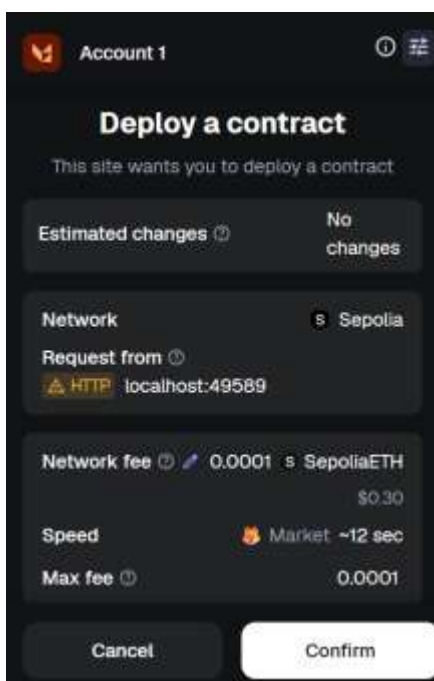
## \* Testing Phase: Compilation of Code (error detection)

1. Smart contract was compiled and deployed on Sepolia Testnet using Remix and MetaMask.
2. Chainlink price feed address for ETH/USD was correctly linked.
3. Contract function `getLatestPrice()` was called.
4. The returned integer value represented the current market price of Ethereum in USD.
5. Tool logs and Transaction Hash confirmed live oracle data interaction.

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.7;
3
4 // Chainlink Price Feed Interface
5 import "@chainlink/contracts/src/v0.8/interfaces/AggregatorV3Interface.sol";
6
7 contract PriceConsumer {
8
9     AggregatorV3Interface internal priceFeed;
10
11
12     constructor() {
13         priceFeed = AggregatorV3Interface(
14             0x694AA1769357215DE4FAC081bf1f309aDC325306
15         );
16     }
17
18
19     function getLatestPrice() public view returns (int) {
20         (
21             ,
22             int price,
23             ,
24             ,
25         ) = priceFeed.latestRoundData();
26         return price / 10**8; // Price with 8 decimals
27     }
28 }

```



## \* Implementation Phase: Final Output (no error)

Applied and Action Learning

1. Deployment successful on Sepolia testnet.
2. getLatestPrice() returned real-time ETH/USD price (e.g., 3310 USD).
3. Transaction details visible on-chain (Etherscan) and in Remix logs.
4. This proved data was fetched from the real world via Chainlink oracle to the smart contract.

## \* Observations

1. Smart contracts cannot access external data directly but can use oracles like Chainlink for real-time information.
2. Chainlink Data Feeds provide secure, tamper-proof prices for blockchain applications.
3. External data fetched becomes part of blockchain history, making it immutable once stored on-chain.
4. Oracle integration enables powerful decentralized applications in areas like DeFi, insurance, and gaming.

## 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		
<b>Total</b>	<b>50</b>		

**Signature of the Student:**

**Name :**

**Regn. No. :**

**Signature of the Faculty:**

Page No.....

*\* As applicable according to the experiment.  
Two sheets per experiment (10-20) to be used.*