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 : Talk to the World – Backend and Oracle Integration

Objective/Aim:

To understand and implement backend communication between off-chain servers and smart contracts using Oracles, enabling blockchain applications to interact with real-world data.

Apparatus/Software Used:

- **Solidity (v0.8.x)** – Smart contract programming language.
- **Remix IDE / Hardhat / Truffle** – for writing and deploying contracts.
- **Node.js & Express.js** – Backend service integration.
- **Chainlink Oracle / API Consumer contract** – for off-chain data fetching.
- **MetaMask** – for blockchain wallet and transactions.
- **Ethereum Test Network (Sepolia / Goerli)** – for deployment and testing.

Theory:

What are Oracles?

Oracles act as bridges between blockchain and the outside world.
They fetch external data (e.g., weather, prices, randomness) and feed it into smart contracts through secure transactions.

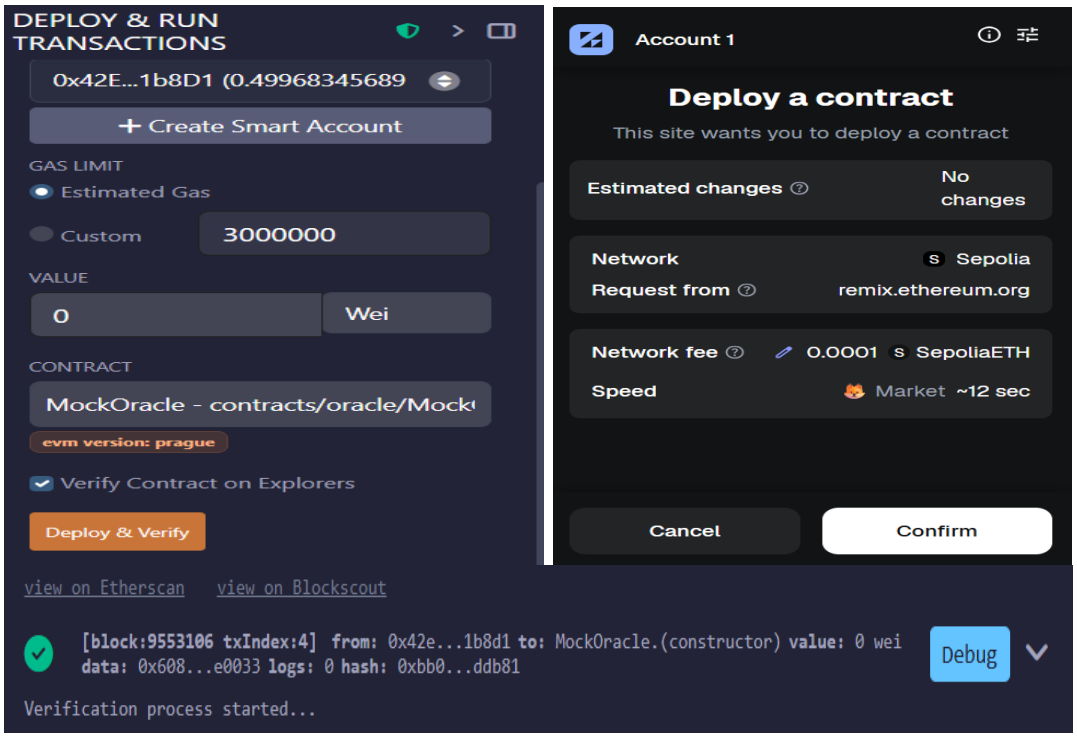
Types of Oracles:

Type	Function
Inbound Oracles	Fetch real-world data into blockchain (e.g., price feeds).
Outbound Oracles	Send blockchain data to external systems.
Compute Oracles	Perform off-chain computation for smart contracts.
Cross-chain Oracles	Enable data transfer between different blockchains.

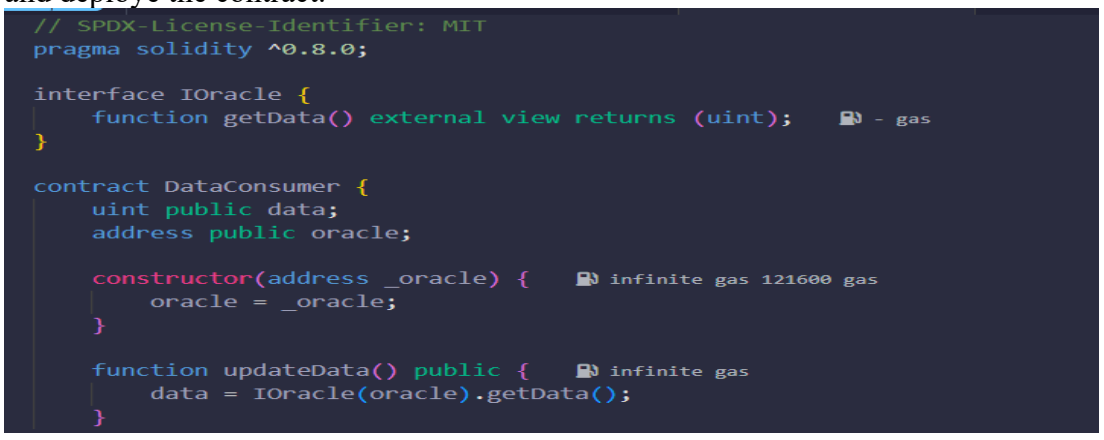
Procedure:

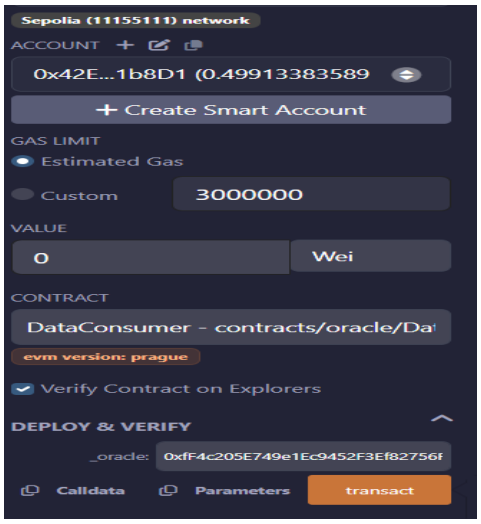
1. Open **Remix IDE** and create a new file MockOracle.sol.

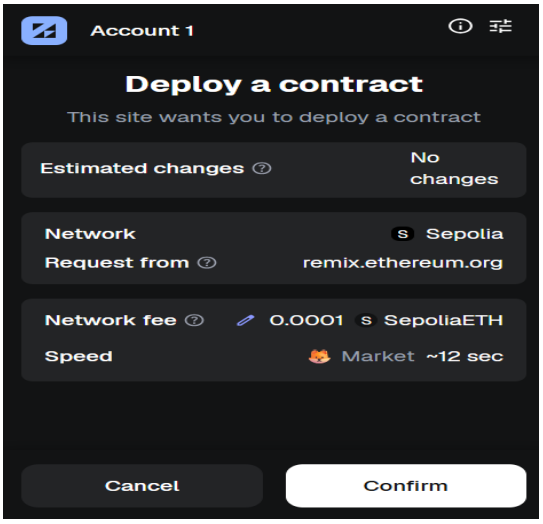
2. Deploy MockOracle first → copy its address .



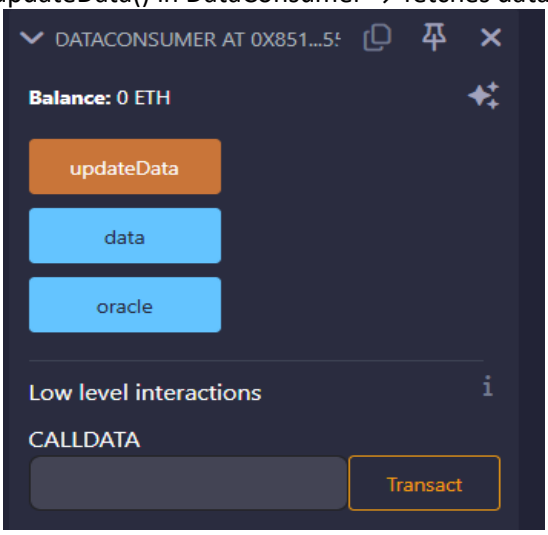
3. Now paste the contract address of the MockOracle in the deploye section of the DataConsumer and deploye the contract.

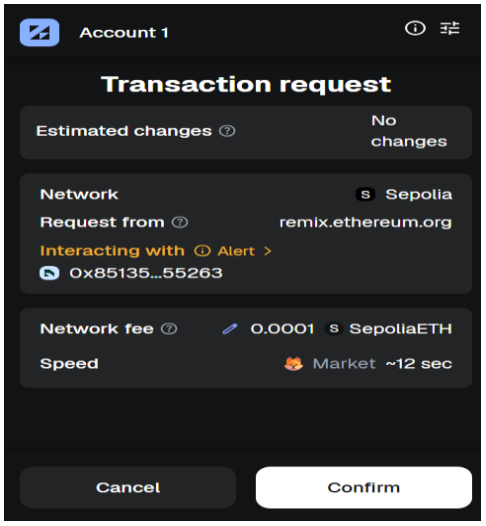






4. Now Call updateData() in DataConsumer → fetches data from MockOracle.





Backend Integration (Node.js):

1. Create a folder backend-oracle/.
Initialize a Node.js project:

```
npm init -y
npm install express ethers dotenv
```

2. Create .env file for storing credentials:

```
.env
RPC_URL="https://mainnet.infura.io/v3/143bcff100834977a566b0991bb67475"
PRIVATE_KEY="3273174e57cfec3e11f6da6f5895d9d742f17e3d29652d1837ee47faf3e21ef0"
CONTRACT_ADDRESS="0x851351b777155f3DAa6C08cE5EBDFA4f3FA55263"
```

3. Create server.js file:

```
const express = require('express');
const { ethers } = require('ethers');
require('dotenv').config();

const app = express();
const PORT = 3000;

const provider = new ethers.JsonRpcProvider(process.env.RPC_URL);
const wallet = new ethers.Wallet(process.env.PRIVATE_KEY, provider);

const abi = [
    "function data() public view returns (uint)",
    "function updateData() public"
];
const contract = new ethers.Contract(process.env.CONTRACT_ADDRESS, abi, wallet);

app.get('/update', async (req, res) => {
    const tx = await contract.updateData();
    await tx.wait();
    res.send("Data updated on blockchain");
});
```

```
app.get('/read', async (req, res) => {
    const currentData = await contract.data();
    res.send(`Current Data: ${currentData}`);
});

app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

4. Run the server:

```
found 0 vulnerabilities
PS C:\Users\shrut\OneDrive\Desktop\backend-oracle> node server.js
[dotenv@17.2.3] injecting env (3) from .env -- tip: sync secrets across teammates & ma
ps
Server running on port 3000
```

5. Now open browser and test:
    - http://localhost:3000/update → updates blockchain data
    - http://localhost:3000/read → reads blockchain data

Observation:

From this experiment, we conclude that:

- Oracles serve as critical bridges between blockchain and the real world.
- Using backend servers (Node.js + Ethers.js), developers can automate off-chain data fetching.
- Chainlink provides a decentralized and secure way to bring external APIs on-chain.
- This integration expands blockchain's potential beyond isolated ledgers — enabling real-world use cases like DeFi price feeds, weather insurance, and supply chain tracking.

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 :

Signature of the Faculty:

Regn. No. :

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