# SKILL 8

## [CS601] – Cryptography and Blockchain

*Date* – 11/04/2023 | *By* Aishwarya Suryakant Waghmare, PRN – 2001106059
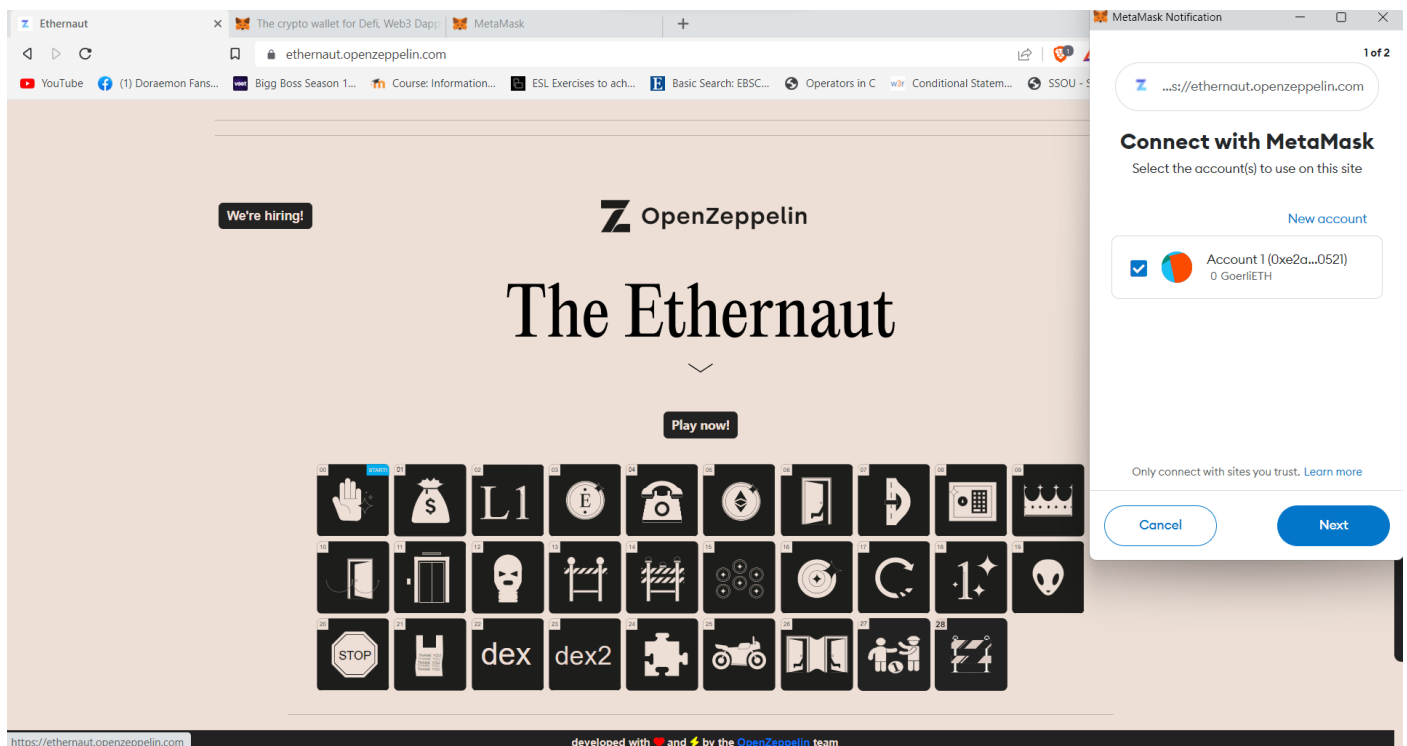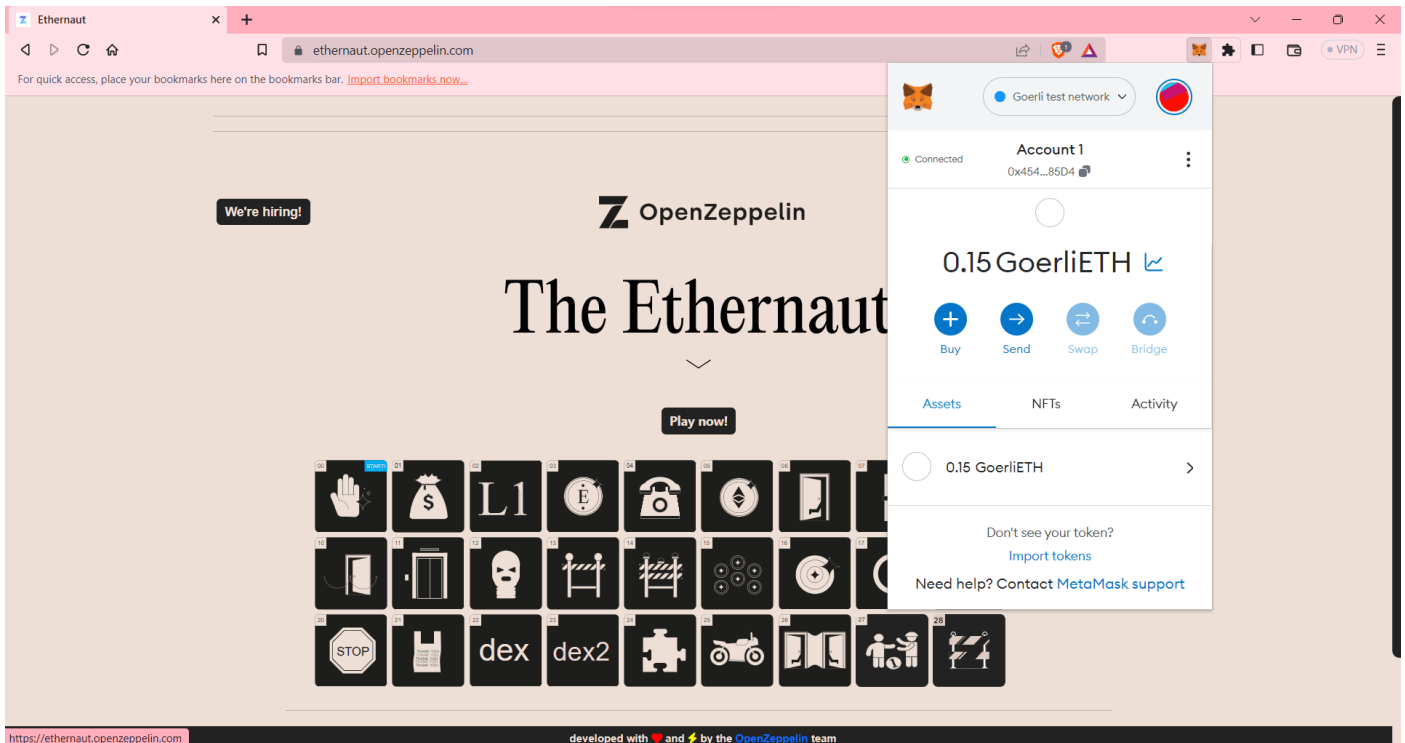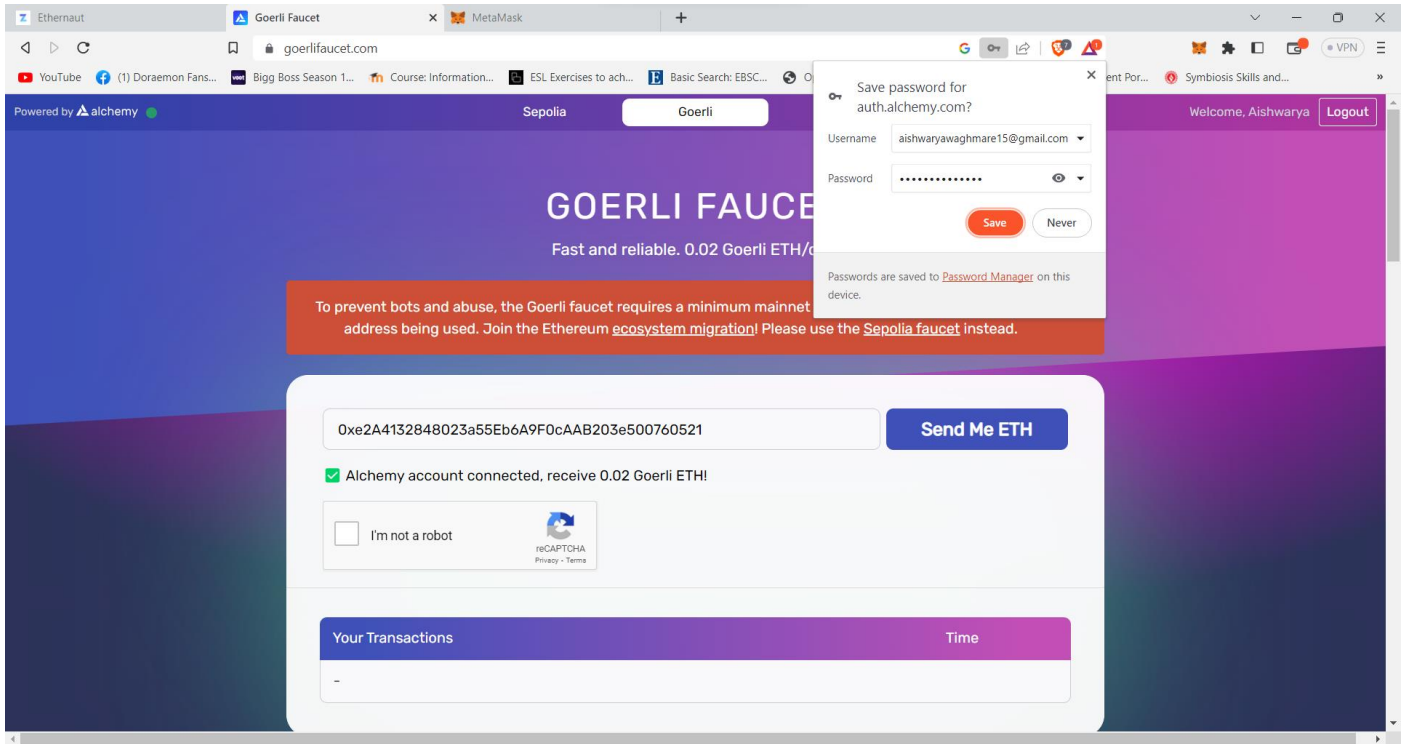
## Title/Aim of the practical :

Vulnerabilities in Solidity – Smart Contracts – by making use of the Metamask and Ethernaut CTF.
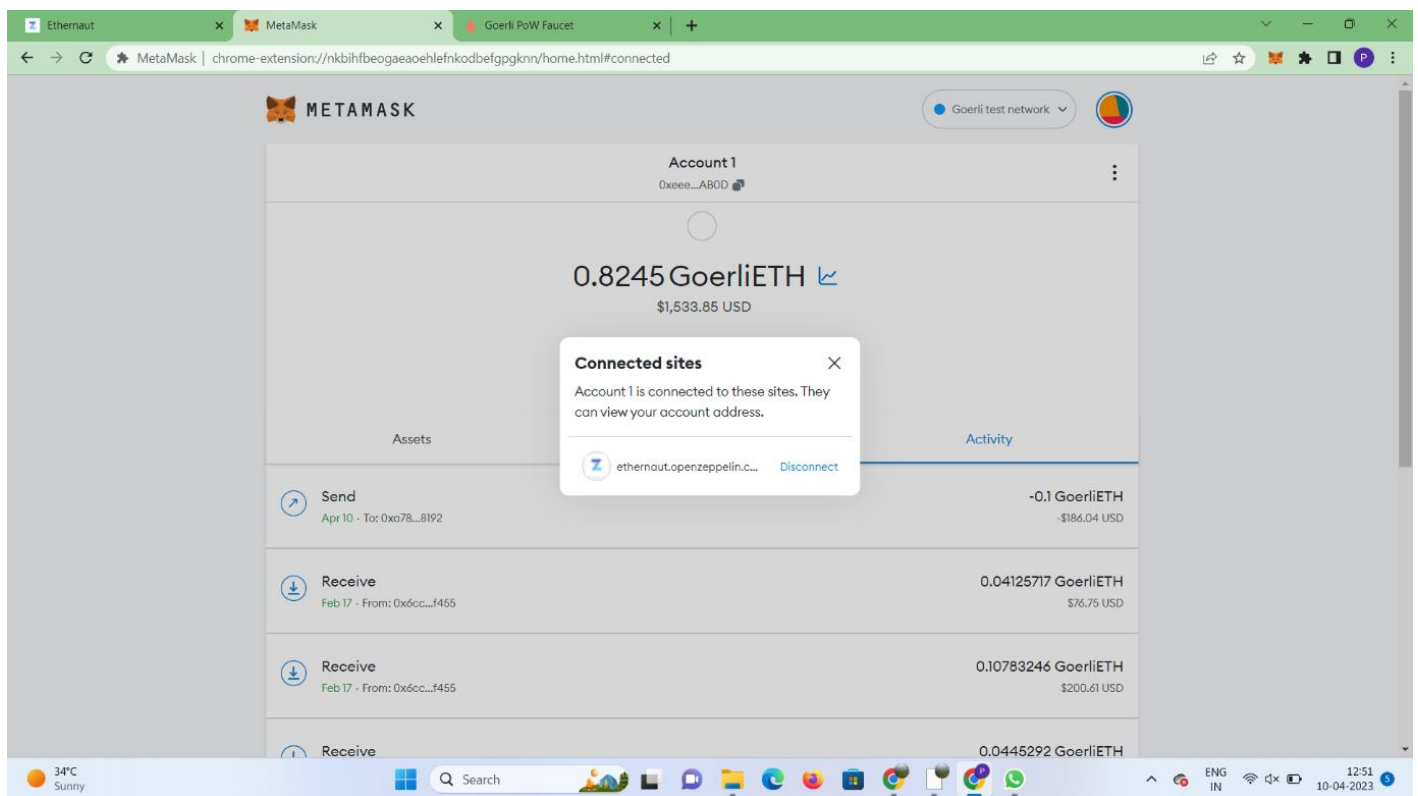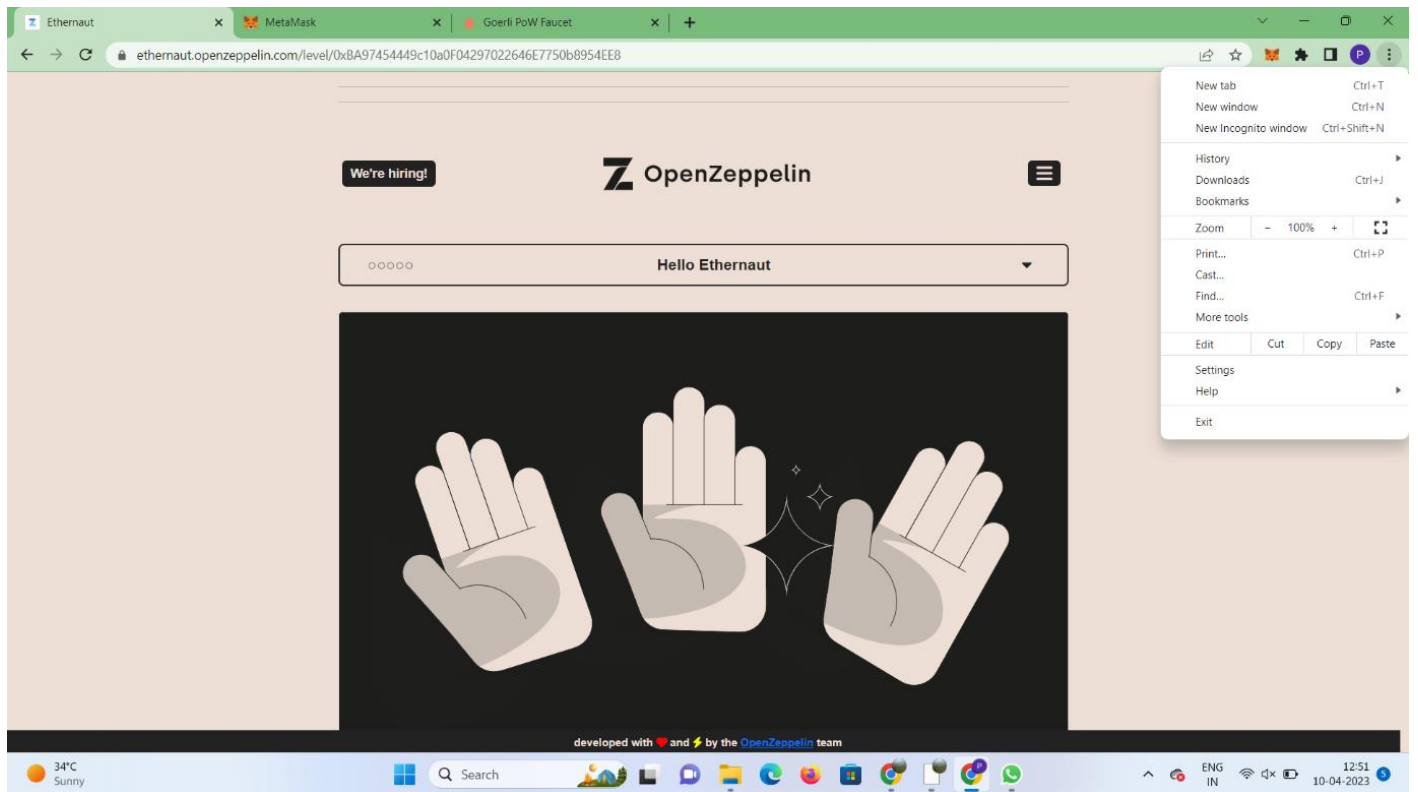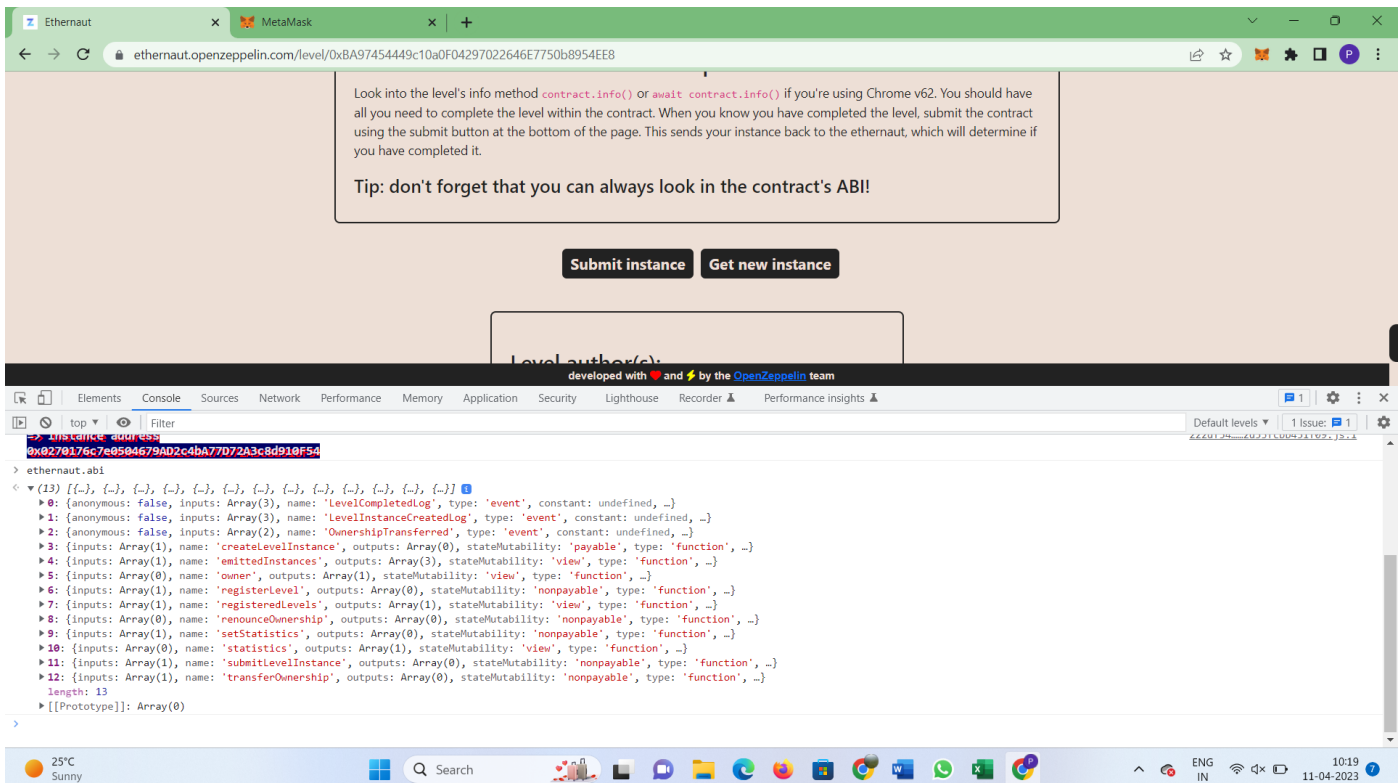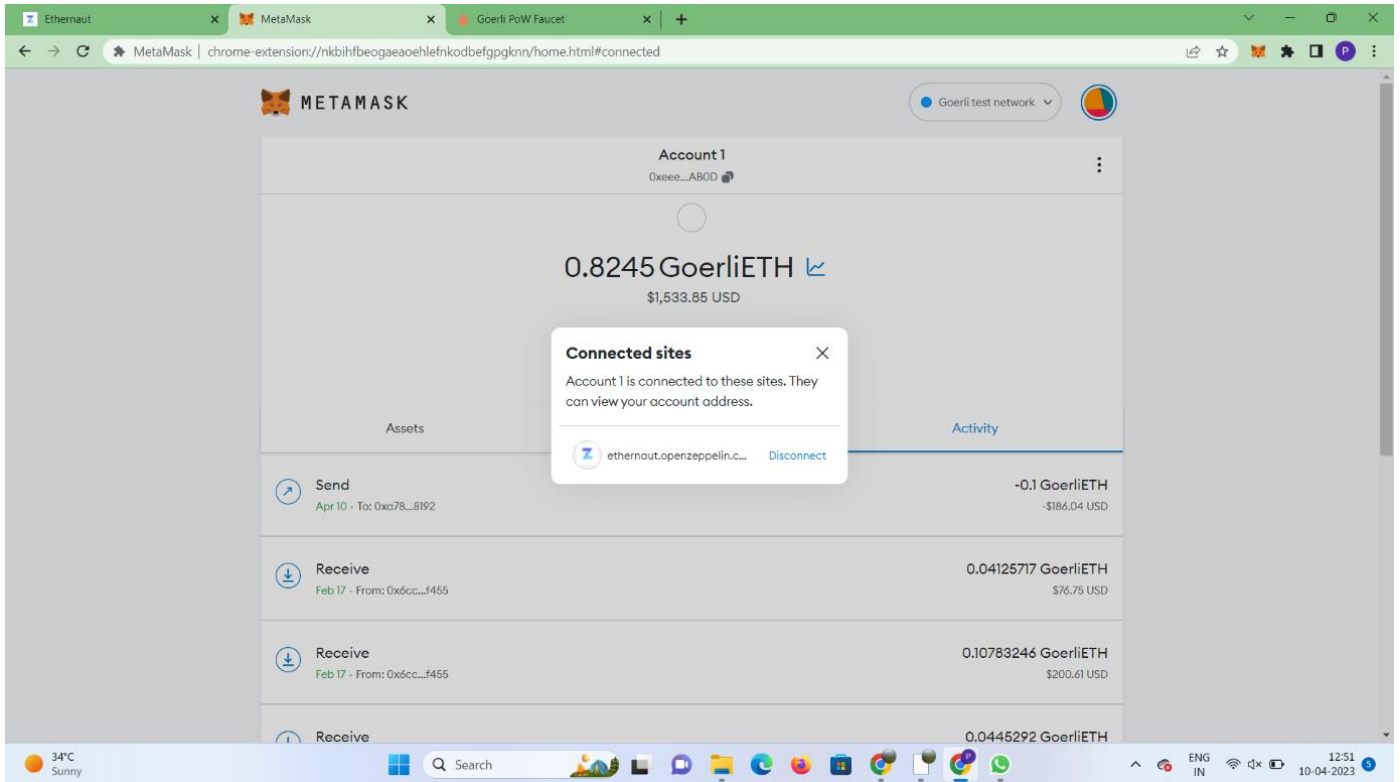
## Apparatus/Tools/ Resources used :

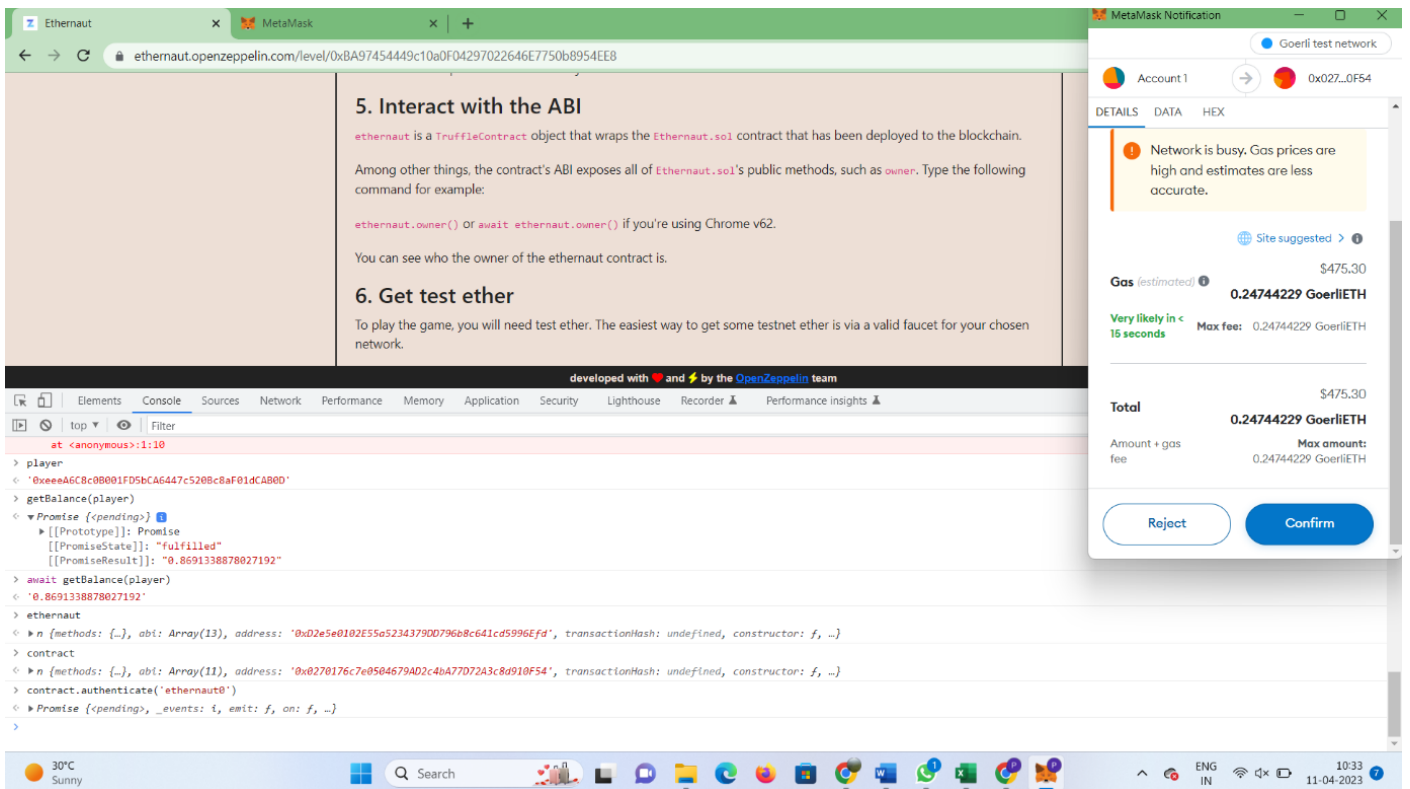- Lecture Notes
- E-Resources
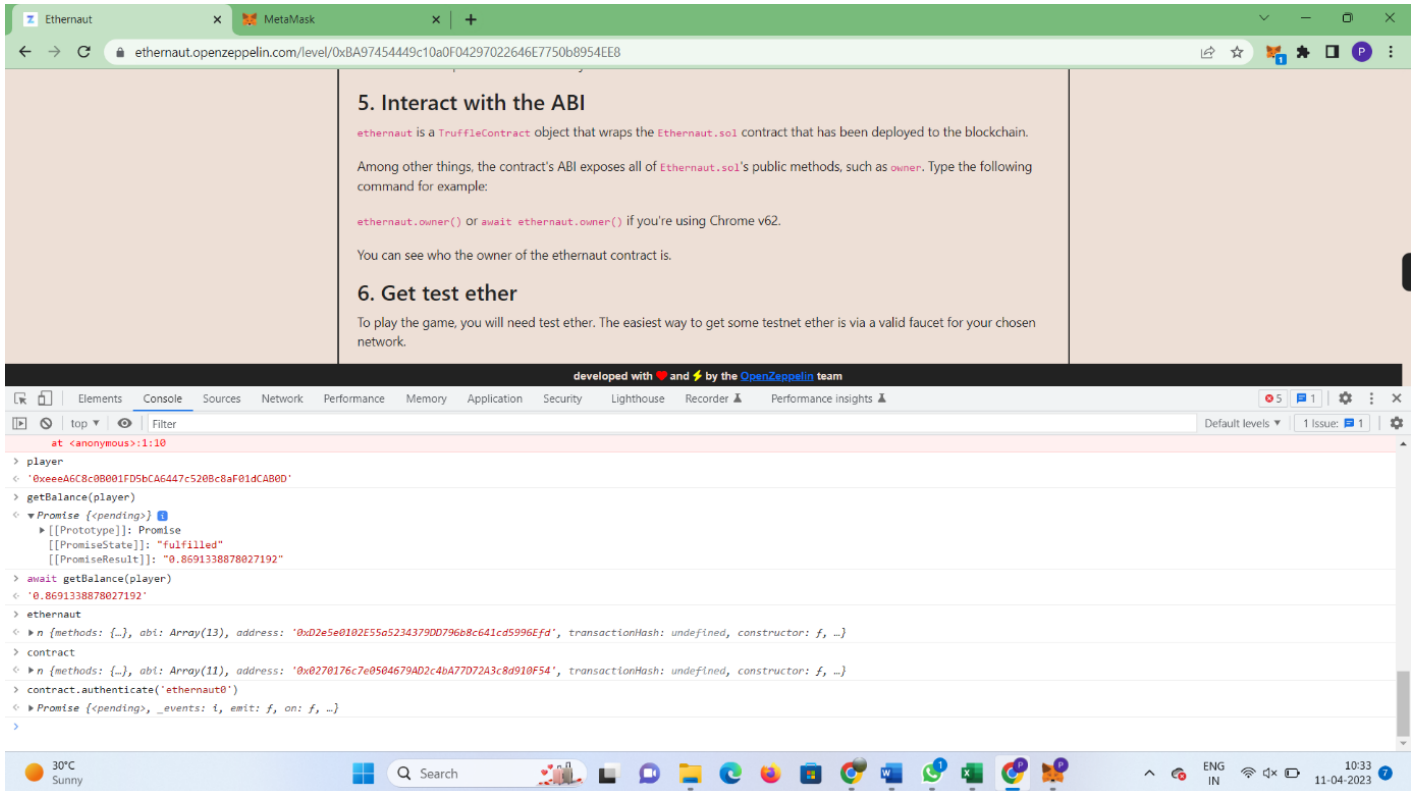- E-Book
- Laptop
- Remix IDE

## Steps performed in the activity :

## 5. Interact with the ABI

ethernaut is a TruffleContract object that wraps the Ethernaut.sol contract that has been deployed to the blockchain.

Among other things, the contract's ABI exposes all of Ethernaut.sol's public methods, such as owner. Type the following command for example:

ethernaut.owner() or await ethernaut.owner() if you're using Chrome v62.

You can see who the owner of the ethernaut contract is.

## 6. Get test ether

To play the game, you will need test ether. The easiest way to get some testnet ether is via a valid faucet for your chosen network.

developed with ♥ and ⚡ by the OpenZeppelin team

```
> player
< '0xeeeA6C8c0B001FD5bCA6447c520Bc8aF01dCAB0D'
> getBalance(player)
< ▼ Promise {<pending>} ⓘ
    ▶ [[Prototype]]: Promise
      [[PromiseState]]: "fulfilled"
      [[PromiseResult]]: "0.8691338878027192"
> await getBalance(player)
< '0.8691338878027192'
> ethernaut
< ▶ n {methods: {…}, abi: Array(13), address: '0xD2e5e0102E55a5234379DD796b8c641cd5996Efd', transactionHash: undefined, constructor: f, …}
> contract
< ▶ n {methods: {…}, abi: Array(11), address: '0x0270176c7e0504679AD2c4bA77D72A3c8d910F54', transactionHash: undefined, constructor: f, …}
> contract.authenticate('ethernaut0')
< ▶ Promise {<pending>, _events: i, emit: f, on: f, …}
>
```



## 5. Interact with the ABI

ethernaut is a TruffleContract object that wraps the Ethernaut.sol contract that has been deployed to the blockchain.

Among other things, the contract's ABI exposes all of Ethernaut.sol's public methods, such as owner. Type the following command for example:

ethernaut.owner() or await ethernaut.owner() if you're using Chrome v62.

You can see who the owner of the ethernaut contract is.

## 6. Get test ether

To play the game, you will need test ether. The easiest way to get some testnet ether is via a valid faucet for your chosen network.

developed with ♥ and ⚡ by the OpenZeppelin team

**MetaMask Notification** — Goerli test network

Account 1 → 0x027...0F54

DETAILS | DATA | HEX

⚠ Network is busy. Gas prices are high and estimates are less accurate.

🌐 Site suggested ⟩ ⓘ

| | $475.30 |
| Gas (estimated) ⓘ | 0.24744229 GoerliETH |
| Very likely in < 15 seconds | Max fee: 0.24744229 GoerliETH |

| | $475.30 |
| Total | 0.24744229 GoerliETH |
| Amount + gas fee | Max amount: 0.24744229 GoerliETH |

Reject | Confirm

```
> player
< '0xeeeA6C8c0B001FD5bCA6447c520Bc8aF01dCAB0D'
> getBalance(player)
< ▼ Promise {<pending>} ⓘ
    ▶ [[Prototype]]: Promise
      [[PromiseState]]: "fulfilled"
      [[PromiseResult]]: "0.8691338878027192"
> await getBalance(player)
< '0.8691338878027192'
> ethernaut
< ▶ n {methods: {…}, abi: Array(13), address: '0xD2e5e0102E55a5234379DD796b8c641cd5996Efd', transactionHash: undefined, constructor: f, …}
> contract
< ▶ n {methods: {…}, abi: Array(11), address: '0x0270176c7e0504679AD2c4bA77D72A3c8d910F54', transactionHash: undefined, constructor: f, …}
> contract.authenticate('ethernaut0')
< ▶ Promise {<pending>, _events: i, emit: f, on: f, …}
>
```
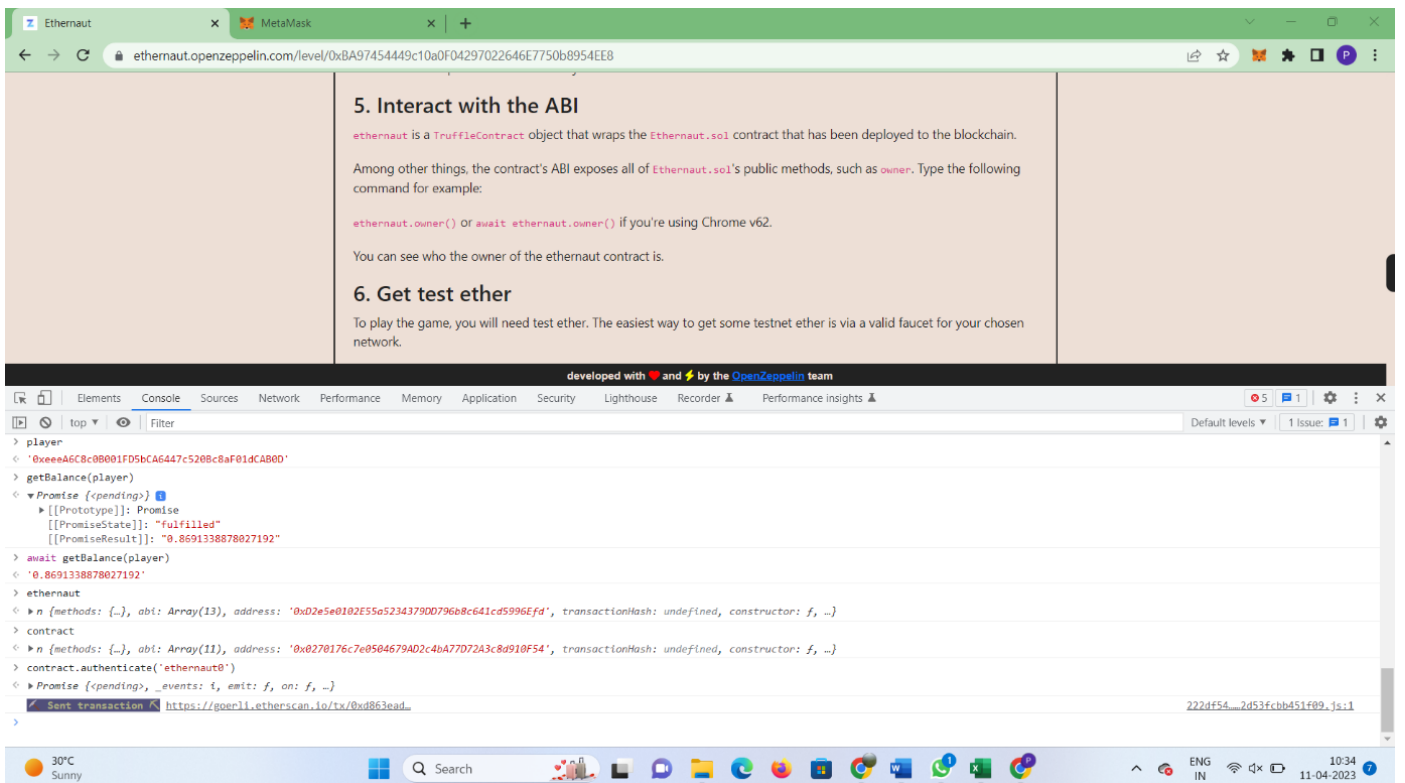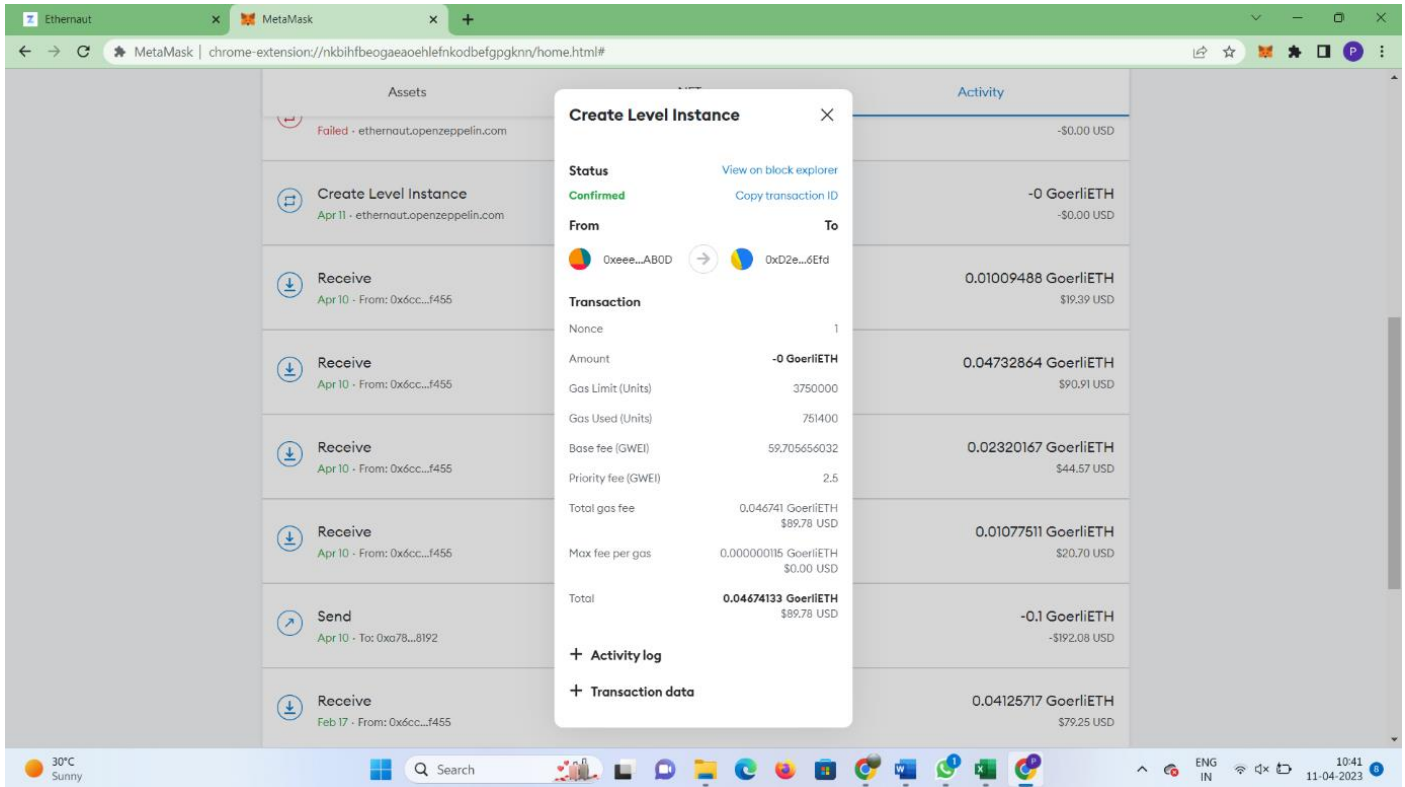
## Parameters achieved/ Conclusion :

Therefore, understood and implemented Vulnerabilities in Solidity – (Smart Contracts ) – by making use of the Metamask and Ethernaut CTF.