L0TTERY DAPP

A Project Report

Submitted by:

Name : Riya Ram (2241018048)

Name : Aishika Chakraborty(2241003004)

Name : Pallavi Meher(2241013107)

Name :Anwesha Mishra(2241013126)

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Faculty of Engineering and Technology, Institute of Technical Education and Research

SIKSHA 'O' ANUSANDHAN (DEEMED TO BE) UNIVERSITY

Bhubaneswar, Odisha, India

CERTIFICATE

This is to certify that the project report titled “ Lottery Dapp ” being submitted by Pallavi Meher, Anwesha Mishra, Riya Ram and Aishika Chakraborty of section 2241034 to the Institute of Technical Education and Research, Siksha ‘O’ Anusandhan (Deemed to be) University, Bhubaneswar for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering is a record of original confide work carried out by them under my/our supervision and guidance. The project work, in my/our opinion, has reached the requisite standard fulfilling the requirements for the degree of Bachelor of Technology.

The results contained in this project work have not been submitted in part or full to any other University or Institute for the award of any degree or diploma.

(Name and signature of the Project Supervisor)

Department of Computer Science and Engineering

Faculty of Engineering and Technology;
Institute of Technical Education and Research;
Siksha ‘O’ Anusandhan (Deemed to be) University

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project guide and faculty members for their valuable guidance, constant support, and encouragement throughout the development of this project. Their suggestions and feedback played an important role in successfully completing this work.

I am also thankful to my institution for providing the necessary resources and learning environment required for this project. I would like to acknowledge the use of online resources, documentation, and developer communities that helped me understand blockchain concepts, smart contracts, and decentralized application development.

Finally, I would like to thank my friends and family for their motivation and support during the course of this project. Their encouragement helped me stay focused and complete this project successfully.

Place: **Signature of Student**

Date:

DECLARATION

We declare that this written submission represents our ideas in our own words and where other's ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/fact/source in our submission. We understand that any violation of the above will cause for disciplinary action by the University and can also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been taken when needed.

Signature of Student with Registration Number

Date: ——————

REPORT APPROVAL

This project report titled **“LOTTERY -D-APP”** submitted by **Riya Ram, Aishika Chakraborty, Anwesha Mishra, Pallavi Meher,** is approved for the degree of *Bachelor of Technology in Computer Science and Engineering*.

Examiner(s)

Supervisor

Project Coordinator

PREFACE

The rapid advancement of blockchain technology has enabled the development of decentralized applications that eliminate the need for centralized authorities. This project, Lottery DApp, aims to design and implement a transparent and secure lottery system using blockchain technology.

The purpose of this study is to overcome the limitations of traditional lottery systems such as lack of transparency, risk of fraud, and dependency on intermediaries. The methodology used involves developing a smart contract using Solidity, deploying it on a blockchain network, and integrating it with a web-based frontend using Web3 technologies. The frontend of the application is hosted online at https://riyaram05.github.io/Lottery-Dapp, allowing users to interact with the system through a cryptocurrency wallet such as MetaMask.

The major findings of this project indicate that blockchain-based lottery systems provide higher transparency, enhanced security, and automated execution compared to conventional systems. The smart contract ensures fair winner selection and secure fund distribution without manual intervention, making the system reliable and efficient.

INDIVIDUAL CONTRIBUTIONS

Riya Ram	problem formulation and solution design
Aishika Chakraborty	Identification of problem statement
Pallavi Meher	experimentation; result analysis and design
Anwesha Mishra	result validation

TABLE OF CONTENTS

	Title Page	i
	Certificate	ii
	Acknowledgement	iii
	Declaration	iv
	Report Approval	v
	Preface	vi
	Individual Contributions	vii
	Table of Contents	viii
	List of Figures	ix
	List of Tables	x
1.	**INTRODUCTION**	1
	1.1 Project Overview/Specifications 1.2 Motivation(s) 1.3 Uniqueness of the Work	1 1 1
2.	**LITERATURE SURVEY**	**2**
	2.1 Existing System 2.2 Problem Identification	2 3
3.	**MATERIALS AND METHODS**	
	3.1 System Architecture 3.2 Working Model 3.4 Tools and Technologies 3.4 Implementation details	7 8 9 10
4.	**Experimentation and Results** 4.1 System Specification 4.2 Parameters Used (if any) 4.3Experimental Outcomes	11
5.	**CONCLUSIONS**	14
6.	**REFERENCES**	15
7.	**APPENDICES**	17
8.	**REFLECTION OF THE TEAM MEMBERS ON THE PROJECT**	20

LIST OF FIGURES

NO	FIGURE NAME	PAGE NO
1	Solidity code sample	4
2	Homepage of LOTTERY-D-APP	8
3	Wallet connection	9
4	Ticket Purchase	9
5	Transaction confirmation	10
6	Winner Selection/Result Display	11

LIST OF TABLES

NO	TABLE NAME	PAGE NO
1	Tools and technologies table	5
2		
3		
4		
5		
6		
7		
8		
9		

Introduction

With the rapid growth of digital technologies, traditional centralized systems are increasingly being replaced by decentralized solutions that offer higher transparency, security, and efficiency. One such emerging technology is blockchain, which enables the development of decentralized applications (DApps) that operate without the need for a central authority. Blockchain technology ensures immutability, transparency, and trust by maintaining a distributed ledger that records all transactions securely. These characteristics make blockchain an ideal platform for applications involving financial transactions and trust-sensitive operations such as lottery systems.

The Lottery DApp is a blockchain-based decentralized application designed to conduct lottery operations in a secure, fair, and transparent manner. In conventional lottery systems, the entire process is managed by centralized organizations, which may lead to issues such as lack of transparency, delayed prize distribution, and potential manipulation. Participants must trust the authority running the lottery, as there is no direct way to verify the fairness of winner selection. The Lottery DApp addresses these challenges by eliminating third-party involvement and automating the lottery process using smart contracts.

Smart contracts are self-executing programs deployed on a blockchain network that execute predefined rules without manual intervention. In this project, smart contracts are used to handle participant entry, maintain a record of players, randomly select a winner, and automatically transfer the prize amount to the winning participant. Since all operations are executed on the blockchain, the system ensures transparency and prevents unauthorized modifications. Every transaction is recorded permanently on the blockchain, allowing participants to verify the process independently.

The frontend of the Lottery DApp is developed using web technologies, enabling users to interact with the application through a user-friendly interface. Users connect their cryptocurrency wallet, such as MetaMask, to participate in the lottery by sending a fixed amount of cryptocurrency. This integration ensures secure authentication and transaction handling. By combining blockchain technology with a simple and accessible interface, the Lottery DApp provides a reliable alternative to traditional lottery systems.

Overall, the Lottery DApp demonstrates the practical application of blockchain technology in creating transparent and trustless systems. The project highlights how decentralized applications can improve security, reduce fraud, and enhance user trust in financial systems, making it a significant step toward modernizing traditional lottery platforms.

Literature Survey

Existing System:

Lottery systems have been a popular form of gambling and fundraising for decades. Traditionally, these systems are centralized and controlled by governments or private organizations. The conventional lottery process involves participants purchasing tickets through authorized vendors, and winners are determined by a draw conducted by the central authority. While these systems have been widely used, they are often criticized for several limitations, including lack of transparency, delayed prize distribution, and potential manipulation by intermediaries. Participants have to rely solely on the authority running the lottery for fairness, as there is usually no direct way to verify the process.

With the advent of blockchain technology, decentralized applications (DApps) have emerged as a solution to overcome the limitations of centralised systems. Blockchain is a distributed ledger technology that provides immutability, transparency, and security. Every transaction recorded on the blockchain is publicly verifiable and cannot be altered retroactively, making it an ideal platform for applications that require trust and fairness. Several studies and projects have explored blockchain-based lottery systems, emphasizing that smart contracts can automate participant management, winner selection, and prize distribution without manual intervention.

Identification of Problem:

Existing blockchain lottery models demonstrate improved transparency and security compared to traditional systems. For instance, smart contracts eliminate the need for intermediaries, reducing the risk of fraud and errors. Moreover, these systems provide real-time verification of transactions, allowing participants to independently confirm the fairness of each lottery draw. However, many existing DApps still face challenges such as complex user interfaces, limited accessibility, or reliance on pseudo-random number generation mechanisms that may not be fully secure.

The proposed Lottery DApp builds upon the strengths of previous research while addressing their shortcomings. It combines a user-friendly web interface with secure smart contract logic to ensure that participants can easily enter the lottery and verify the process. By leveraging blockchain's decentralized nature, the system guarantees transparency, reduces operational costs, and increases participant trust. This literature survey establishes the need for a secure, automated, and accessible lottery system, providing the foundation for the design and implementation of the Lottery DApp.

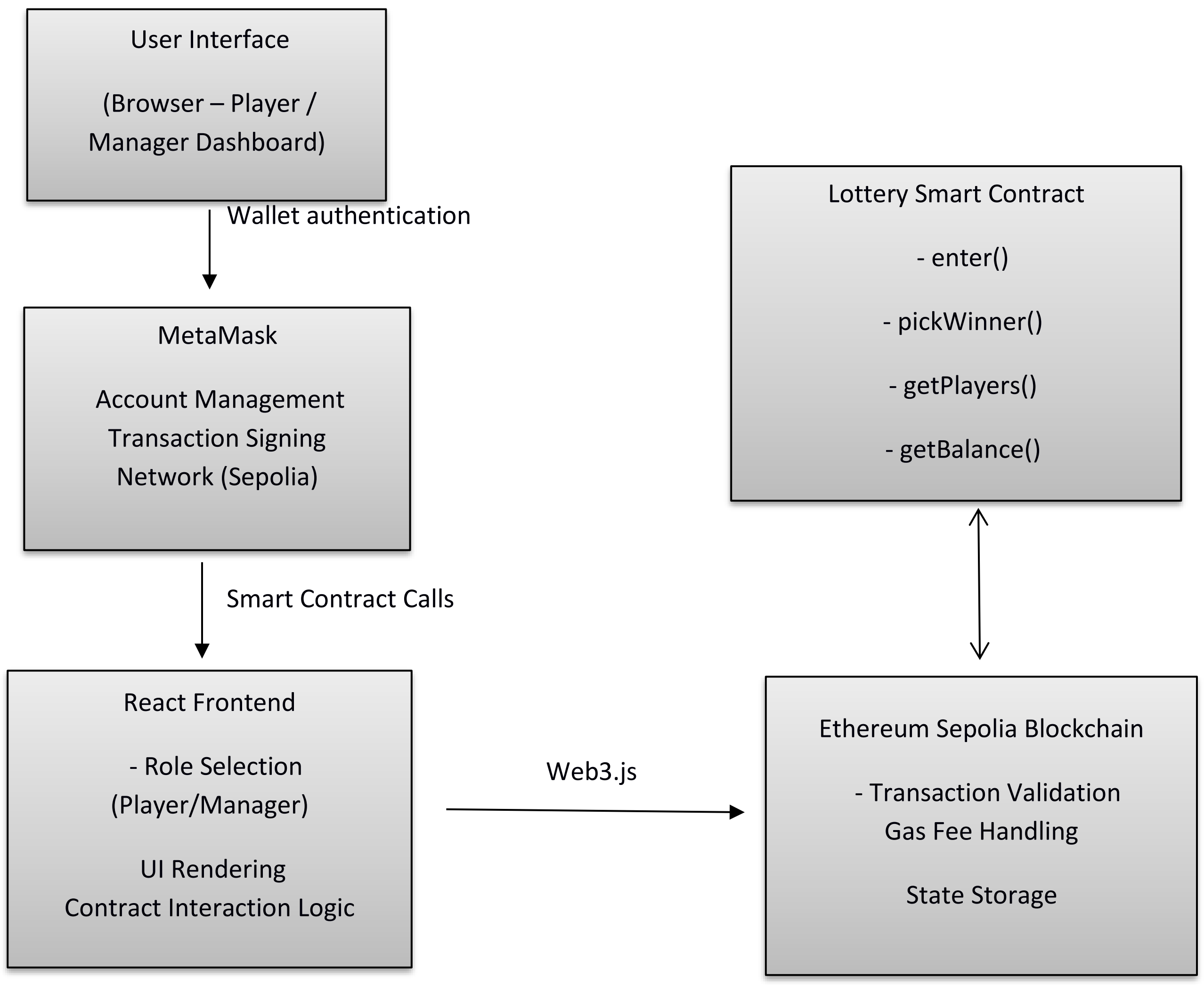
System Architecture

The Lottery DApp is designed as a decentralised application (DApp) to ensure security, transparency, and automation. Unlike traditional lottery systems, it does not require a central authority. The system architecture consists of four main components:

- **Frontend Interface**
    - Built using HTML, CSS, and JavaScript (React.js)
    - Provides a user-friendly interface for participants to enter the lottery
    - Displays lottery status, number of participants, and winner details
    - Connects with the blockchain smart contract via Web3.js / Ethers.js
- **Smart Contract**
    - Written in Solidity and deployed on Ethereum blockchain
    - Handles all lottery operations automatically:
        - Participant registration
        - Fund collection
        - Winner selection
        - Prize distribution
    - Reduces the need for manual intervention, ensuring trustless execution
- **Blockchain Network**
    - Records all transactions permanently and immutably
    - Provides decentralized verification of every lottery event
    - Securely stores participant data and transaction history

- **Wallet Integration**
  - Uses MetaMask or similar cryptocurrency wallets
  - Provides secure authentication and transaction signing
  - Enables automatic prize transfer to the winning participant

Working Model



Tools and Technologies

The Lottery DApp is developed using a combination of blockchain technologies, web development tools, and programming languages to ensure a robust, secure, and user-friendly system.

- **Programming Languages**
  - Solidity: Used for writing smart contracts that automate the lottery logic, including participant registration, random winner selection, and prize distribution. Solidity ensures that the smart contracts are self-executing and trustless.
  - JavaScript: Used for frontend interaction and integration with smart contracts. It enables dynamic updates and communication between the user interface and blockchain.
- **Frontend Development**
  - HTML & CSS: Used to create the structure and styling of the web application.
  - React.js: Provides a dynamic and responsive user interface, allowing real-time updates of lottery status, participants, and winners.

3. Blockchain Platform

- Ethereum / Polygon: Public blockchain networks where the smart contract is deployed. These platforms ensure immutability, security, and transparency of all lottery transactions.

4. Wallet Integration

- MetaMask: A cryptocurrency wallet that allows users to securely connect to the Lottery DApp. It enables authentication, entry payment, and automatic receipt of prize money.

- **IDE and Development Tools**

  - Remix IDE: Online development environment for writing, testing, and deploying Solidity smart contracts.
  - Visual Studio Code (VS Code): Used for frontend development and managing project files.
- **Libraries and APIs**
  - Web3.js / Ethers.js: JavaScript libraries used to connect the frontend with smart contracts on the blockchain. These libraries allow sending transactions, reading contract data, and handling events.
- **Hosting / Deployment**
  - GitHub Pages: Used to host the frontend application online, making it easily accessible to participants.

Tools and Technologies Table

Category	**Tool / Technology**	**Purpose**
Programming Languages	Solidity, JavaScript	Smart contract and frontend logic
Frontend Development	HTML, CSS, React.js	User interface design and responsiveness
Blockchain Platform	Ethereum / Polygon	Deployment and execution of smart contracts
Wallet Integration	MetaMask	Secure authentication and transactions
IDE / Development Tools	Remix IDE, VS Code	Writing, testing, and managing code
Libraries / APIs	Web3.js / Ethers.js	Connect frontend to blockchain smart contract
Hosting / Deployment	GitHub Pages	Make frontend accessible online

Implementation Details

The implementation of the Lottery DApp is divided into three main areas: Smart Contract Development, Frontend Development, and Deployment & Testing. This section provides a step-by-step explanation of how the project was built and executed.

Methodology

The project follows a structured methodology to ensure a fully functional and secure decentralized application:

1. **Requirement Analysis:**
   - Identify limitations in centralized lottery systems such as lack of transparency, manual intervention, and potential fraud.
   - Define project objectives: transparency, automation, security, and user-friendliness.
2. **Design:**
   - Design smart contract logic for participant registration, fund management, random winner selection, and prize distribution.
   - Design the frontend interface for participant interaction and real-time updates.
3. **Smart Contract Development:**
   - Implement the lottery logic in Solidity.
   - Include functions for entering the lottery, generating a random winner, transferring the prize, and resetting the system.
4. **Frontend Development:**
   - Build a responsive and interactive user interface using React.js.
   - Connect the frontend with the smart contract using Web3.js or Ethers.js.
   - Implement wallet integration for secure authentication and transactions.
5. **Integration and Testing:**

   - Connect the frontend and smart contract.
   - Test all functionalities including wallet connection, lottery entry, winner selection, and prize distribution.

6. **Deployment:**
   - Deploy the smart contract on Ethereum/Polygon testnet.
   - Host the frontend online using GitHub Pages for easy accessibility.

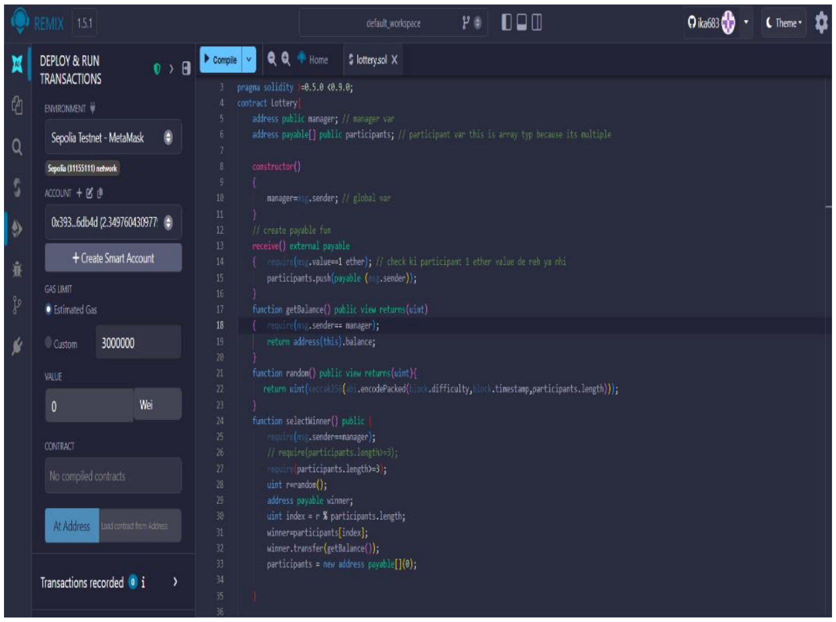
Smart Contract Implementation

The smart contract is the core of the Lottery DApp. It handles all the logic securely and automatically.

Key Features

- Manager Role: Only the contract deployer can select a winner.
- Player Entry: Users can enter the lottery by sending cryptocurrency (e.g., 0.1 ETH).
- Random Winner Selection: Uses blockchain parameters (timestamp + participants) to generate a pseudo-random number.
- Prize Distribution: Transfers all funds automatically to the selected winner.
- Reset Mechanism: Resets the participant array for the next lottery round.

Solidity Code Sample



ENVIRONMENT: Sepolia Testnet - MetaMask
Sepolia (11155111) network
ACCOUNT: 0x393...6db4d (2.349760430977
GAS LIMIT: Estimated Gas / Custom 3000000
VALUE: 0 Wei
CONTRACT: No compiled contracts

```
pragma solidity >=0.5.0 <0.9.0;
contract Lottery{
    address public manager; // manager var
    address payable[] public participants; // participant var this is array typ because its multiple

    constructor()
    {
        manager=msg.sender; // global var
    }
    // create payable fun
    receive() external payable
    {   require(msg.value==1 ether); // check ki participant 1 ether value de reh ya nhi
        participants.push(payable (msg.sender));
    }
    function getBalance() public view returns(uint)
    {   require(msg.sender== manager);
        return address(this).balance;
    }
    function random() public view returns(uint){
       return uint(keccak256(abi.encodePacked(block.difficulty,block.timestamp,participants.length)));
    }
    function selectWinner() public {
        require(msg.sender==manager);
        // require(participants.length>=3);
        require(participants.length>=3);
        uint r=random();
        address payable winner;
        uint index = r % participants.length;
        winner=participants[index];
        winner.transfer(getBalance());
        participants = new address payable[](0);

    }
```

Smart Contract Explanation:

The Lottery Smart Contract is a decentralized application written in Solidity and deployed on the Ethereum blockchain. The purpose of this contract is to conduct a fair and transparent lottery system without the involvement of any central authority.

Contract Overview

The contract allows multiple participants to enter a lottery by sending exactly 1 Ether. All the collected Ether is stored securely in the smart contract. Once a minimum number of participants is reached, the contract manager can randomly select a winner, who receives the entire prize pool.

Key Components of the Contract

1. Manager

- The manager is the account that deploys the contract.
- The manager has special privileges such as viewing the contract balance and selecting the winner.
- The manager's address is stored using the variable manager.

2. Participants
    - Participants are users who enter the lottery by sending exactly 1 Ether.
    - Their wallet addresses are stored in a dynamic array called participants.

Functions Used in the Contract

1. Constructor
    - The constructor is executed only once during contract deployment.
    - It assigns the deployer's address as the manager of the lottery.
2. receive() Function
    - This is a payable function that allows the contract to receive Ether.
    - It ensures that only 1 Ether is accepted per entry.
    - The sender's address is added to the participants list.
3. getBalance() Function
    - This function returns the total Ether stored in the contract.
    - Only the manager is allowed to call this function.
4. random() Function
    - This function generates a pseudo-random number using blockchain properties such as timestamp and participant count.

    - It is used only internally to select a winner.

5. selectWinner() Function

    - This function can be called only by the manager.
    - It checks whether there are at least three participants.
    - A random participant is selected as the winner.
    - The entire contract balance is transferred to the winner.
    - After the transfer, the participants list is reset for the next round.

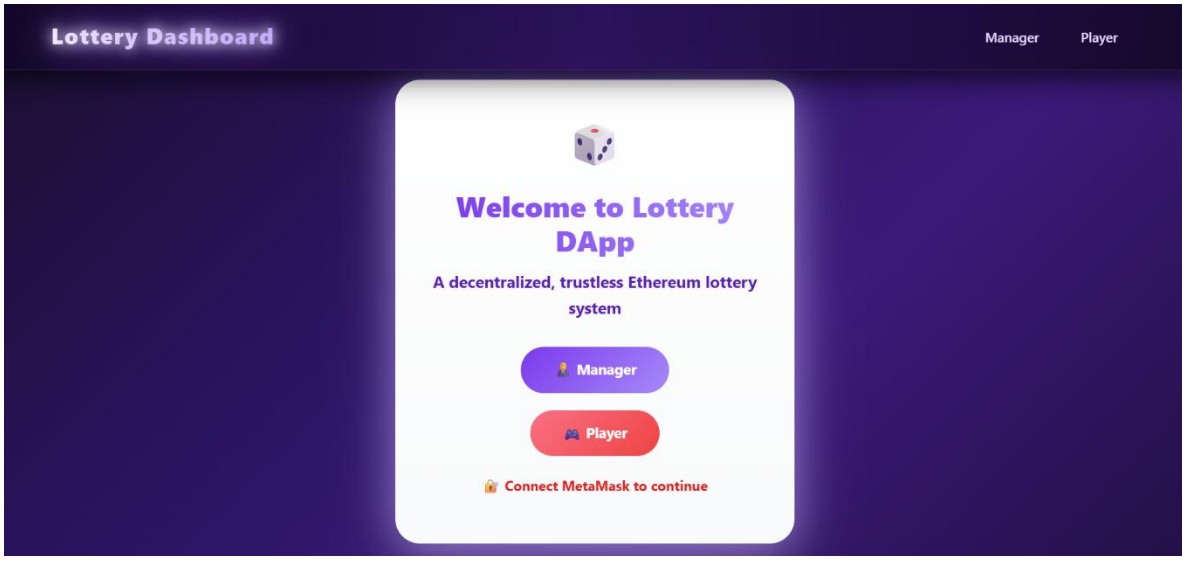
Security and Fairness

- Only the manager can select the winner.
- Each participant must send exactly 1 Ether, ensuring equal chances.
- The blockchain ensures transparency and prevents manipulation.

Experimentation and Results

1. Homepage / Lottery Status

- Open your DApp (https://riyaram05.github.io/Lottery-Dapp)

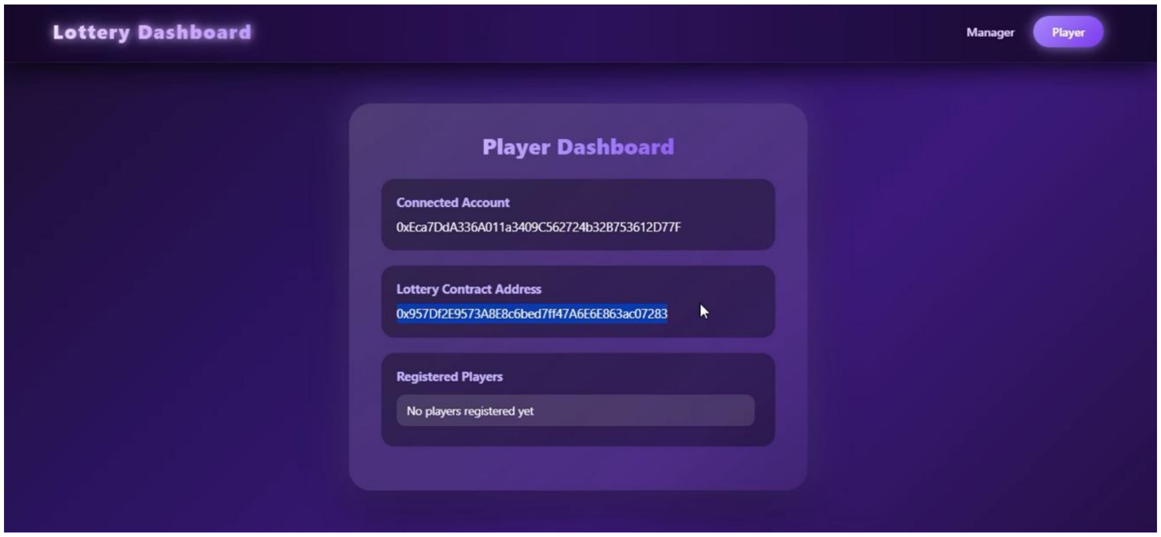


Welcome to Lottery DApp

A decentralized, trustless Ethereum lottery system

Connect MetaMask to continue

This figure shows the workflow of players in the Lottery DApp, where players connect their MetaMask wallet, send 1 ETH to the smart contract, and get registered automatically in the system.



Player Dashboard

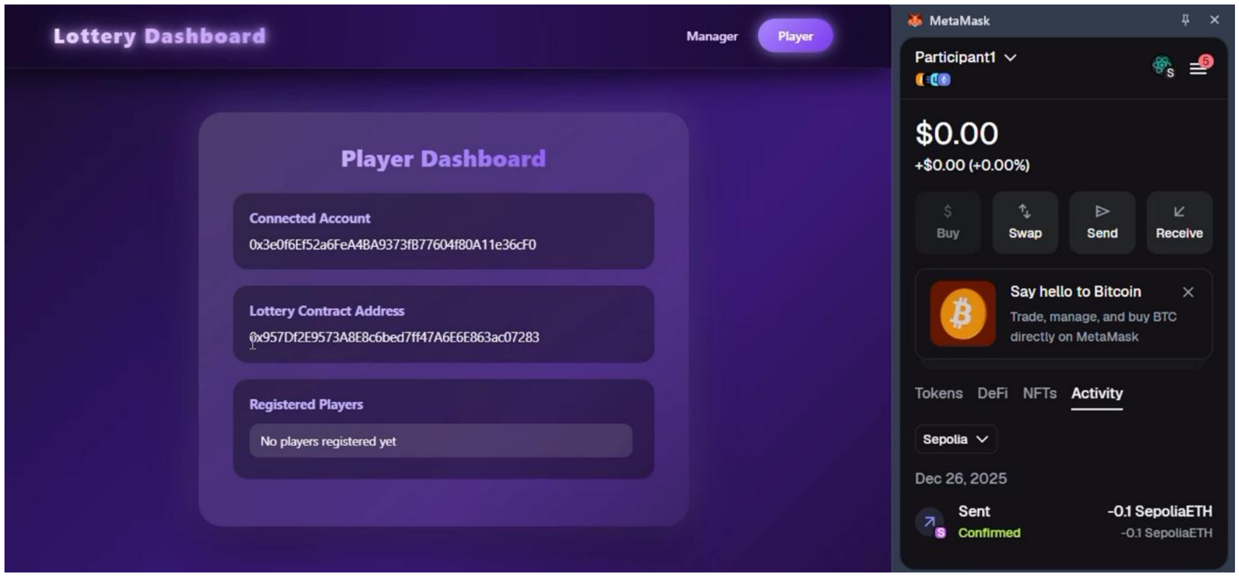
Connected Account
0xEca7DdA336A011a3409C562724b32B753612D77F

Lottery Contract Address
0x957Df2E9573A8E8c6bed7ff47A6E6E863ac07283

Registered Players
No players registered yet

2. Wallet Connection

- Connect the user wallet to the DApp



Connected Account
0x3e0f6Ef52a6FeA4BA9373fB77604f80A11e36cF0

Lottery Contract Address
0x957Df2E9573A8E8c6bed7ff47A6E6E863ac07283

Registered Players
No players registered yet

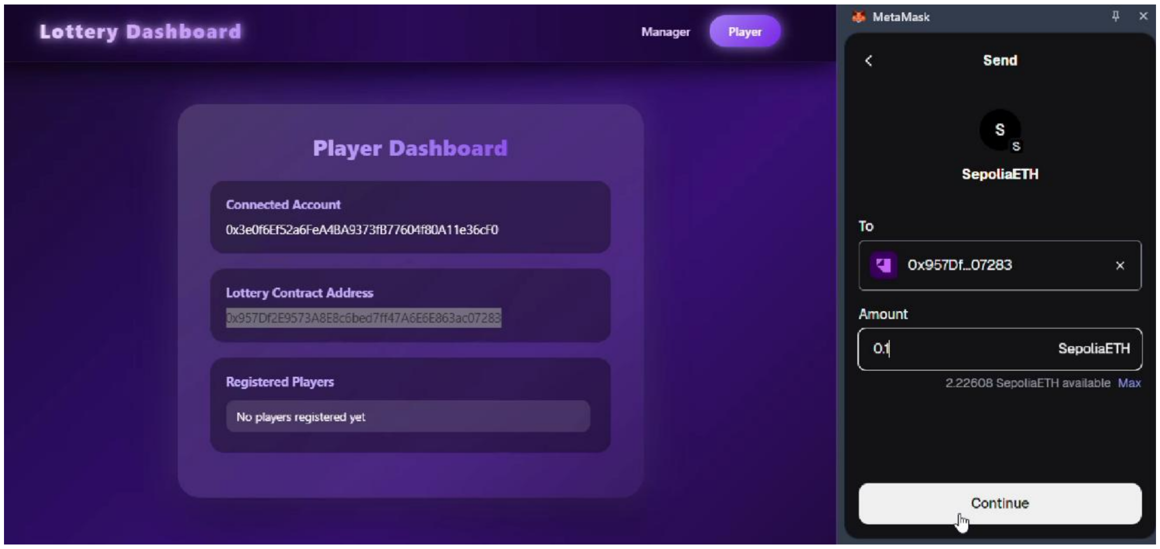
$0.00
+$0.00 (+0.00%)

Say hello to Bitcoin
Trade, manage, and buy BTC directly on MetaMask

Dec 26, 2025
Sent -0.1 SepoliaETH
Confirmed -0.1 SepoliaETH

3. Ticket Purchase

- Enter the required amount and purchase a lottery ticket



Connected Account
0x3e0f6Ef52a6FeA4BA9373fB77604f80A11e36cF0

Lottery Contract Address
0x957Df2E9573A8E8c6bed7ff47A6E6E863ac07283

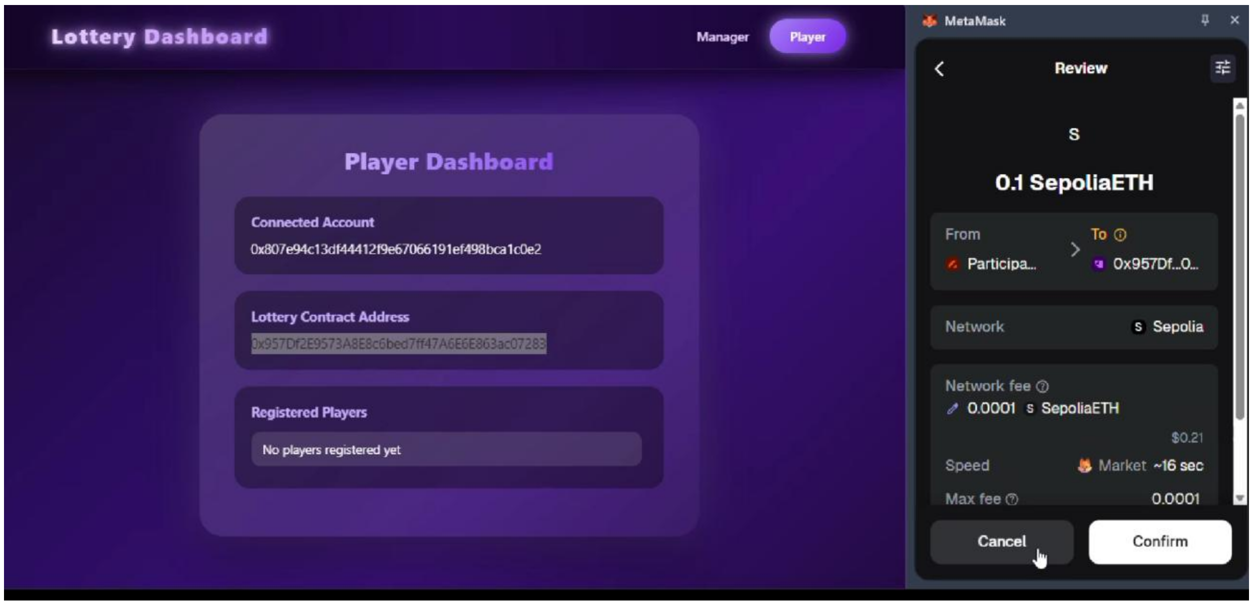
Registered Players
No players registered yet

To
0x957Df...07283

Amount
0.1 SepoliaETH
2.22608 SepoliaETH available Max

4. Transaction Confirmation

- After confirming the transaction in the wallet



Lottery Dashboard

Manager | Player

Player Dashboard

Connected Account
0x807e94c13df44412f9e67066191ef498bca1c0e2

Lottery Contract Address
0x957Df2E9573A8E8c6bed7ff47A6E6E863ac07283

Registered Players
No players registered yet

MetaMask

Review

0.1 SepoliaETH

From: Participa...
To: 0x957Df...0...

Network: Sepolia

Network fee: 0.0001 SepoliaETH ($0.21)

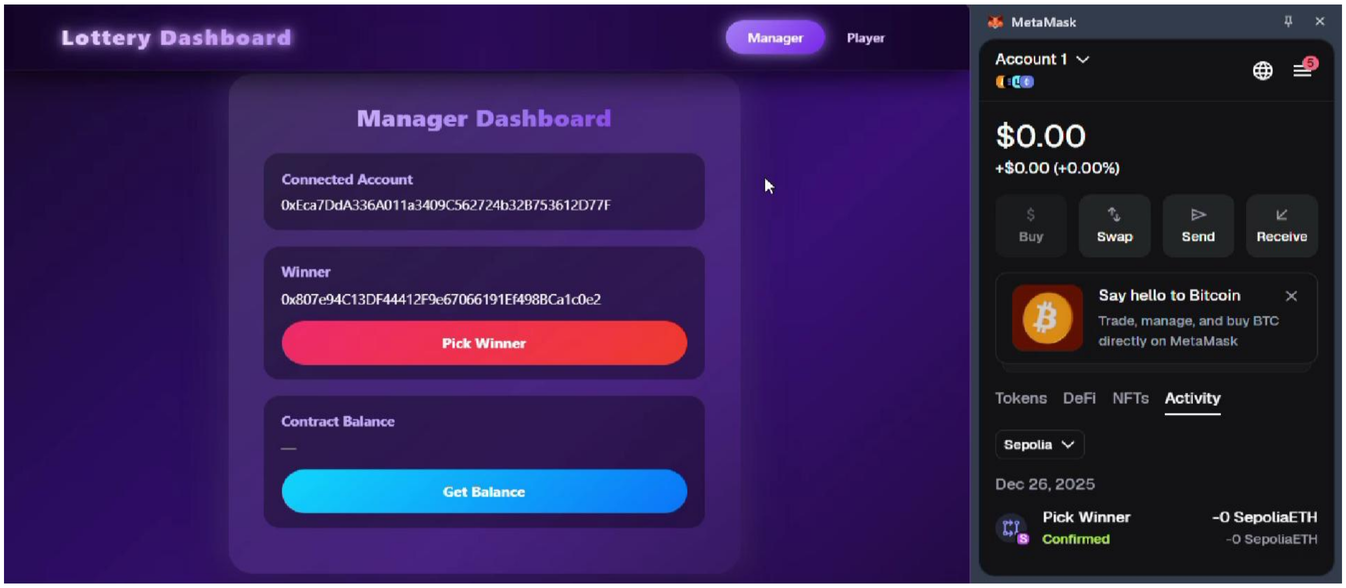
Speed: Market ~16 sec

Max fee: 0.0001

Cancel | Confirm

5. Winner Selection / Result Display

- Execute the draw function



Lottery Dashboard

Manager | Player

Manager Dashboard

Connected Account
0xEca7DdA336A011a3409C562724b32B753612D77F

Winner
0x807e94C13DF44412F9e67066191Ef498BCa1c0e2

Pick Winner

Contract Balance
—

Get Balance

MetaMask

Account 1

$0.00
+$0.00 (+0.00%)

Buy | Swap | Send | Receive

Say hello to Bitcoin
Trade, manage, and buy BTC directly on MetaMask

Tokens | DeFi | NFTs | Activity

Sepolia

Dec 26, 2025

Pick Winner — Confirmed — -0 SepoliaETH — -0 SepoliaETH

Conclusion

The Lottery Decentralized Application (DApp) successfully demonstrates how blockchain technology can be used to create a transparent, secure, and trustless lottery system. By leveraging smart contracts, the application eliminates the need for intermediaries, ensuring that all lottery rules are executed automatically and fairly.

The use of Ethereum blockchain guarantees immutability and transparency, allowing participants to verify transactions and lottery results independently. Smart contracts handle ticket purchases, random winner selection, and prize distribution without human intervention, reducing the risk of fraud or manipulation.

Additionally, integrating a web-based user interface with blockchain wallets (such as MetaMask) makes the system accessible and user-friendly. The DApp ensures secure fund handling, real-time updates, and decentralization, which are major advantages over traditional lottery systems.

Overall, the Lottery DApp proves that decentralized applications can provide fairness, security, and efficiency, making blockchain a promising solution for online gaming and financial applications.

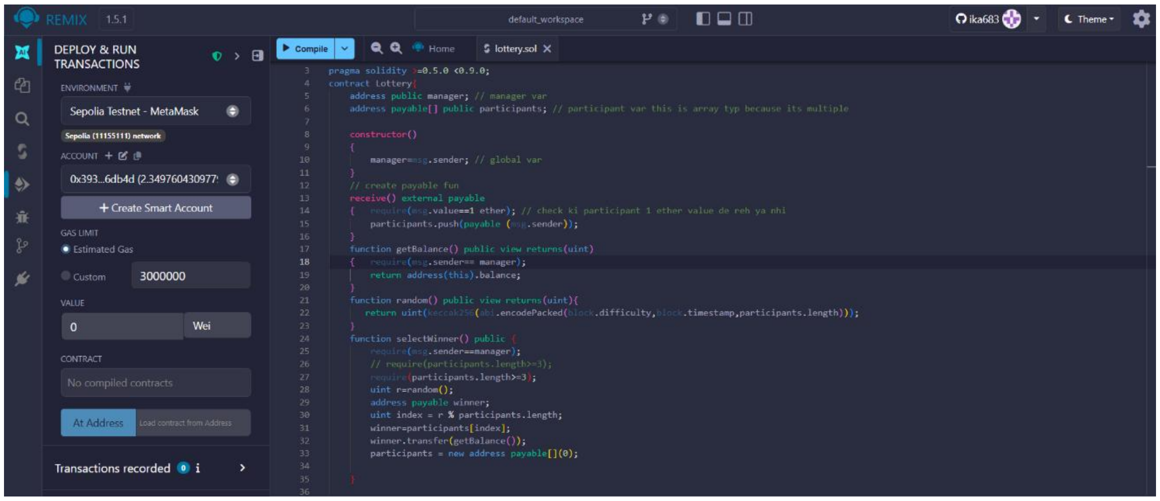
Appendices

Appendix A: **Smart Contract Code (Solidity)**

This appendix contains the Solidity smart contract used to implement the Lottery system on the Ethereum blockchain. The contract handles participant entry, manager control, prize distribution, and winner selection.

Key functionalities:

- Accepts exactly **1 Ether** from each participant
- Stores participant addresses
- Allows only the manager to select the winner
- Transfers the total prize pool to the winner
- Resets the lottery for the next round



ENVIRONMENT: Sepolia Testnet - MetaMask
Sepolia (11155111) network
ACCOUNT: 0x393...6db4d (2.349760430977
GAS LIMIT: Estimated Gas / Custom 3000000
VALUE: 0 Wei
CONTRACT: No compiled contracts

```solidity
pragma solidity >=0.5.0 <0.9.0;
contract Lottery{
    address public manager; // manager var
    address payable[] public participants; // participant var this is array typ because its multiple

    constructor()
    {
        manager=msg.sender; // global var
    }
    // create payable fun
    receive() external payable
    {   require(msg.value==1 ether); // check ki participant 1 ether value de reh ya nhi
        participants.push(payable (msg.sender));
    }
    function getBalance() public view returns(uint)
    {   require(msg.sender== manager);
        return address(this).balance;
    }
    function random() public view returns(uint){
      return uint(keccak256(abi.encodePacked(block.difficulty,block.timestamp,participants.length)));
    }
    function selectWinner() public {
        require(msg.sender==manager);
        // require(participants.length>=3);
        require(participants.length>=3);
        uint r=random();
        address payable winner;
        uint index = r % participants.length;
        winner=participants[index];
        winner.transfer(getBalance());
        participants = new address payable[](0);

    }
```

Appendix B: Frontend Code (React – App.js)

This appendix includes the React frontend code that interacts with the smart contract using MetaMask and ethers.js.

Features implemented:

- MetaMask wallet login
- Display connected wallet address
- Show participants count
- Show prize pool
- Participate in lottery
- Manager-only “Select Winner” option
- Side popup panel UI

import React, { useState, useEffect } from "react"; import getWeb3 from "./getWeb3"; import Lottery from "./contracts/Lottery.json";

import Manager from "./components/Manager"; import Players from "./components/Players"; import Intro from "./components/Intro";

import { HashRouter as Router, Route, NavLink } from "react-router-dom";

import "./App.css";

const App = () => { const [state, setState] = useState({ web3: null, contract: null, }); const [address, setAddress] = useState(null);

useEffect(() => { const init = async () => { try { const web3 = await getWeb3(); const networkId = await web3.eth.net.getId();

```
const deployedNetwork = Lottery.networks[networkId];
const instance = new web3.eth.Contract(
  Lottery.abi,
  deployedNetwork.address
);

setAddress(deployedNetwork.address);
setState({ web3, contract: instance });
```

```
  } catch (error) {
    alert("Failed to load web3 or contract");
    console.error(error);
  }
};
init();

}, []);

return ( {/* ===== NAVBAR ===== /}

{/ BRAND */} Lottery Dashboard
    {/* LINKS */}
    <div>
      <NavLink
        to="/manager"
        className="navtext"
        activeClassName="active"
      >
        Manager
      </NavLink>

      <NavLink
        to="/players"
        className="navtext"
        activeClassName="active"
      >
        Player
      </NavLink>
    </div>
  </nav>

  {/* ===== ROUTES ===== */}
  <Route exact path="/">
    <Intro />
  </Route>

  <Route path="/manager">
    <Manager state={state} />
  </Route>

  <Route path="/players">
    <Players state={state} address={address} />
  </Route>
</Router>

); };

export default App;
```

Appendix C: Tools and Technologies Used

- Solidity – Smart contract development
- Remix IDE – Contract compilation and deployment
- Ethereum Sepolia Testnet – Blockchain network
- MetaMask – Wallet and transaction signing
- React.js – Frontend development
- Web3.js – Blockchain interaction library
- Node.js & npm – Development environment

Appendix D: System Flow Diagram

User

↓

Connect Wallet (MetaMask)

↓

Send 1 Ether

↓

Smart Contract

↓

Prize Pool Updated

↓

Manager Selects the ‘Select Winner Button’

↓

Winner Receives Prize

References

[1] Ethereum Foundation, “Ethereum Documentation,” 2024. [Online]. Available: https://ethereum.org/developers

[2] Ethereum Foundation, “Solidity Documentation,” 2024. [Online]. Available: https://docs.soliditylang.org

[3] Remix Project, “Remix IDE Documentation,” 2024. [Online]. Available: https://remix.ethereum.org

[4] MetaMask, “MetaMask Developer Documentation,” 2024. [Online]. Available: https://docs.metamask.io

[5] Facebook Open Source, “React – A JavaScript Library for Building User Interfaces,” 2024. [Online]. Available: https://reactjs.org

[6] OpenZeppelin, “Smart Contract Security Best Practices,” 2024. [Online]. Available: https://developer.mozilla.org

[7] Ethereum Foundation, “Ethereum Testnets (Sepolia),” 2024. [Online]. Available: https://developer.mozilla.org

[8] MDN Web Docs, “JavaScript Documentation,” 2024. [Online]. Available: https://developer.mozilla.org

Reflection by team members on the Project

Working together on the *Decentralized Lottery Application* was a valuable and enriching experience for our entire team. Through this project, we collectively gained practical exposure to blockchain technology, smart contracts, and decentralized application development. Each team member contributed in different areas such as smart contract design, frontend development, MetaMask integration, testing, and documentation.

As a team, we learned the importance of collaboration, clear communication, and task distribution while working on a real-world technical problem. We faced challenges related to contract deployment, transaction handling, and frontend-blockchain interaction, which helped us improve our debugging and problem-solving skills. By supporting each other and sharing knowledge, we were able to successfully overcome these challenges.

Overall, this project strengthened our technical understanding as well as our teamwork and project management abilities. It provided us with hands-on experience that bridges the gap between theoretical knowledge and practical implementation, preparing us for future academic and professional projects.