Decentralized Crowdfunding Platform - Project Report

Team Name : Team KanyaRashi

Team members :

L. Yashwanth Chowhan	- 230001046
M . Mohan Prudhvi Sai	- 230001047
M. Rishik Preetham	- 230001048
M . Rajavardhan	- 230001053
Mohit Katariya	- 230001055
L. Praveen Kumar	- 230041018

Introduction

This project is a decentralized crowdfunding platform, inspired by Kickstarter, built on the Ethereum blockchain. The platform allows users to create and fund campaigns in a transparent and decentralized manner using smart contracts. It integrates blockchain-based backend logic with a React-based frontend, providing a complete DApp (Decentralized Application) experience.

Technologies Used

- **Solidity**: For writing the smart contract (`Crowdfunding.sol`).
- **Truffle**: For compiling, deploying, and testing smart contracts.
- **Ganache**: A personal Ethereum blockchain used for local development.
- **React.js**: Frontend user interface.
- **Web3.js**: To connect the frontend with the Ethereum blockchain.
- **MetaMask**: Browser extension for Ethereum wallet and signing transactions.

- **OpenZeppelin**: Library for securing smart contracts (`ReentrancyGuard`).

Backend (Smart Contract)

The backend logic is implemented using a smart contract written in Solidity. The `Crowdfunding.sol` contract allows users to:

- Create campaigns with a funding goal and deadline.
- Contribute ETH to campaigns.
- Request refund if campaign goal is not met by the deadline.
- Claim funds if the campaign is successful.

The contract also ensures security against reentrancy attacks by using `ReentrancyGuard` from OpenZeppelin.

Main functions:

- `createCampaign()`
- `contribute()`
- `releaseOrRefund()`
- `getCampaign()`
- `getContribution()`

All data (campaigns, contributions) is stored directly on the blockchain, making the system transparent and immutable.

Frontend (React.js)

The frontend is built using React.js. It connects to the smart contract using Web3.js and allows users to interact with the blockchain via MetaMask. Features include:

- Form to create new campaigns.
- List of all existing campaigns.
- Option to contribute ETH to a campaign.
- Button to trigger release or refund based on campaign status.

The React app dynamically reads from the smart contract using its ABI and deployed address to keep the interface synchronized with blockchain state.

Workflow Summary

1. **Start Ganache** — launches a local Ethereum blockchain.
2. **Compile Contracts** — using `truffle compile`.
3. **Deploy Contracts** — using `truffle migrate`.
4. **Connect Frontend** — update the React app with deployed contract address.
5. **Launch Frontend** — using `npm start` inside `/client` folder.
6. **Use DApp** — create, fund, and manage campaigns using MetaMask.

Security Measures

The project implements the following security features:

- **Reentrancy Protection**: Uses non`Reentrant` modifier to block reentrancy attacks.
- **Validation Checks**: Verifies input data and enforces deadlines and contribution rules.
- **Event Logging**: Emits events for every major action (creation, contribution, refund) to keep UI updated and auditable.

Conclusion

This decentralized crowdfunding platform demonstrates a fully functional DApp with an Ethereum smart contract backend and a Web3-connected React frontend. The project highlights key blockchain principles including immutability, transparency, and decentralized control. It also ensures security and correctness through smart contract protections and automated testing.

This DApp serves as a scalable base for real-world blockchain-based crowdfunding solutions, and can be extended with features like milestone funding, contributor rewards, and admin dashboards.