Decentralized Cryptocurrency Exchange

Term Project Report

Group 6

Reetesh Zope
Samihan Jawalkar
Tina Andre
Mohit Gupta

Table of Contents

Abstract

A blockchain is a shared distributed database or ledger between computer network nodes. A blockchain serves as an electronic database for storing data in digital form. The most well-known use of blockchain technology is for preserving a secure and decentralized record of transactions in cryptocurrency systems like Bitcoin. The innovation of a blockchain is that it fosters confidence without the necessity for a reliable third party by ensuring the fidelity and security of a record of data.

Users can trade cryptocurrencies in a non-custodial setting on a decentralized exchange (DEX), which is a peer-to-peer marketplace that eliminates the need for a middleman to handle the transfer and custody of funds. With blockchain-based smart contracts that facilitate the exchange of assets, DEXs replace intermediaries—traditionally, banks, brokers, payment processors, or other entities.

In our Project, We aim at building such a Decentralized exchange for Ethereum.
Our app's goal is to cut out intermediaries so that average people may do business directly with one another. Users do not receive custody of their crypto assets when using a DEX. Instead, consumers always keep all of their possessions in their wallets.
On top of blockchain networks like Ethereum, DEX application uses self-executing smart contracts that generate different outputs in response to specific inputs.
In this project, we want to create our own DEX for exchanging cryptocurrencies.

Introduction

Problem Description

There are many Centralized exchanges in the World. Example Coinbase, Gemini, Binance etc. These Exchanges charge high transfer fees for crypto transfers. Our project aims at reducing these costs and overheads. The solution to this is DEX or Decentralized Exchanges. Decentralized exchange.

Users can trade cryptocurrencies in a non-custodial setting on a decentralized exchange (DEX), which is a peer-to-peer marketplace that eliminates the need for a middleman to handle the transfer and custody of funds. With blockchain-based smart contracts that facilitate the exchange of assets, DEXs replace intermediaries—traditionally, banks, brokers, payment processors, or other entities.

DEXs provide complete transparency into the movement of funds and the processes supporting exchange, in contrast to typical financial transactions, which are opaque and carried out through middlemen that provide very little insight into their actions. DEXs also lessen counterparty risk and can lessen systemic centralization problems in the bitcoin ecosystem because user money don't transit via a third party's cryptocurrency wallet during trading.

Project Team Details

- Introduction - Our team consists of four software engineers: Reetesh Zope, Samihan Jawalkar, Tina Andre and Mohit Gupta. Our strength is our eagerness to learn and work on niche software development projects. We share a common interest in Blockchain.
- Team Profile -
  Reetesh Zope (Programming & Tech Writer)
  Samihan Jawalkar (Microservices Developer & Frontend Developer)
  Tina Andre (Scrum Master, Project Management)
  Mohit Gupta (Design & Solution Architect)

- First meeting: We held our first Face-To-Face meeting on 1st June. 2022 on Zoom.
- Team Agreement:
  1. **Methods of communication** - Zoom, Google Meet, Email, Google Chat and instant messaging using WhatsApp. In case of an urgent message, phone calls are welcome.
  2. **Expected response time** - The response time of message was within hours of delivery. Everyone followed the given timeline and participated in all Zoom meetings.
  3. **Meeting attendance -** Every team member attended all planned meetings. In case of non-avoidable emergencies, meetings should be postponed to a date when everyone is available.
  4. **Running meetings -**All formal meetings were conducted on zoom during.
  5. **Meeting preparation -** Each member elaborated on action items that he/she has completed in the last sprint. The last 15 minutes of every meeting will be reserved for discussing road blockers and planning the next sprint.

6. **Version control -** We will be using GitHub for version control. No pull requests were merged to the main branch without code review by the owner of the code repository. Commit messages were short and sensible.
7. **Submission of work -** Reetesh Zope or Samihan Jawalkar will be submitting the work once it has been reviewed.
8. **Contingency planning-** In case a team member drops out, the team reserves the right to continue and demonstrate the entire work. In case a team member does not contribute or is being dishonest then the rest of the team can reach out to the Professor to exclude that member from the team.
9. **GitHub Accounts -**

Name	GitHub Account
Reetesh Zope	https://github.com/reeteshz
Samihan Jawalkar	https://github.com/samihan123
Tina Andre	https://github.com/marxeen
Mohit Gupta	https://github.com/mohitguptagp500

Project Proposal

As per the instructions received, the project proposal is submitted on canvas under assignment 'TERM PROJECT PROPOSAL SUBMISSION LINK' on 2nd June 2022.

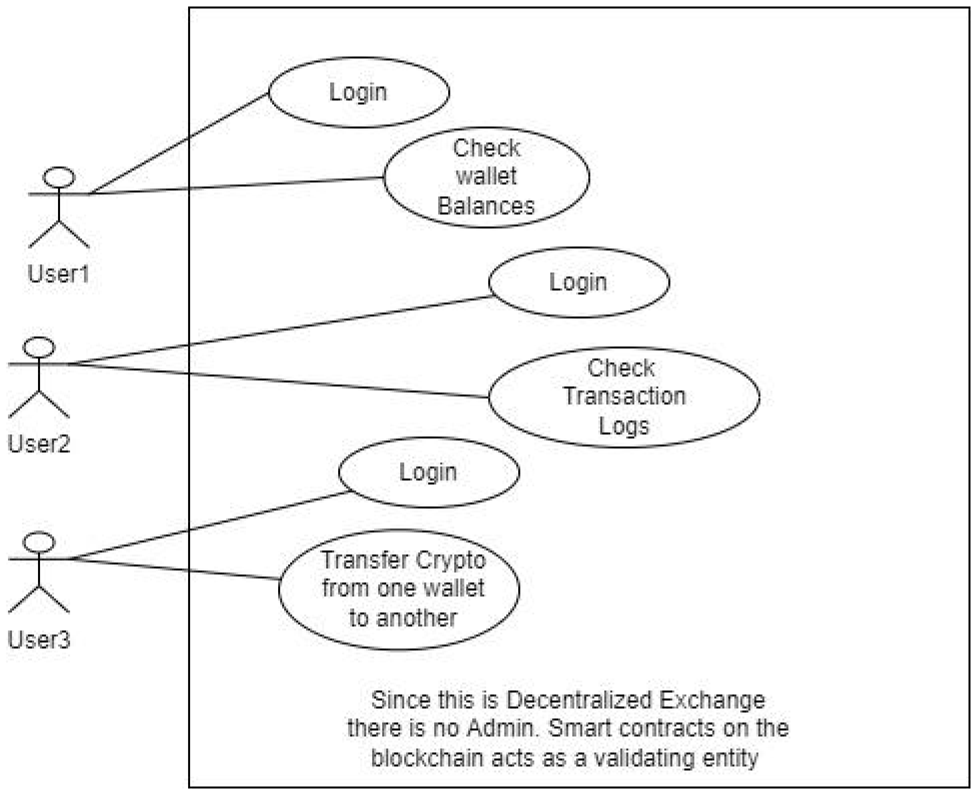
System Requirements

The overall system should demonstrate Login, generation of log messages, User data storage in a blockchain, retrieval of logs for system administration on the web app dashboard and run transactions from one wallet to another.

Use cases

1. End users will have to authenticate themselves before they can access their wallets and user profile.
2. End users will be able to see their previous transactions.
3. End user will be able to see logs of recent transactions on the dashboard.
4. End user will be able to initiate and authorize transactions from one wallet to another.

Use case diagram:


Login
Check wallet Balances
User1
Login
Check Transaction Logs
User2
Login
Transfer Crypto from one wallet to another
User3
Since this is Decentralized Exchange there is no Admin. Smart contracts on the blockchain acts as a validating entity

User Stories

User Story One:
We want an Fornt end for developing decentralised exchanges. Ui should be simple for now. Basic requirement should be fullfilled

User Story Two:
We need to integrate metamask with our app. Metamask is A software cryptocurrency wallet is used to communicate with the Ethereum network.

User Story Three:
We need to create contracts, So we can exchange our cryptocurrency.
In order to automatically execute, control, or document legally significant events and activities in accordance with the provisions of a contract or an agreement, a smart contract is a computer program or transaction protocol.

User Story Four:
We need to create a transaction history and display it. This will help user to keep the track on his transaction.

User Story Five:
we need to do testing on whole platform. Software testing is the process of assessing and confirming that a software application or product performs as intended. Testing has advantages such as bug prevention, lower development costs, and better performance.

User Story Six:
Enhance User interface and user experience.

User Story Seven:
Alpha and beta testing on the application.

Product Backlog

User Story	**Task**	**Subtask**	**Estimate**
1	Creating Next app with tailwind.	Develop webUi	1
		create a header	1
		Add form	1
2	Integrate Metamask	Install and integrate Metamask	2
		Add metamask authentication using ether and API	2
3	Create Contracts	Create simple contracts	2
		currency transfer testing.	2
		Create process for Crypto transfer with ether.js	2
4	create a transaction history.	Create transaction history according to the current user.	2

		Display transaction history on the web page.	
5	Testing the application	Create various scenarios and test if currencies are transferred without being altered.	1
			1
6	Enhance User interface.	Do some detail study on User experience and User interface.	1
		Enhance and upgrade the UI/UX.	2
7	Alpha/Beta testing	Do Alpha testing between team.	1
		Do beta testing by sending the application to other user.	2

Sprint Reports

Sprint Report 1

Implemented functionality	Planned user stories for this sprint	Completed user stories for this sprint	Planned user stories for next sprint
1. Creating Next app with tailwind CSS. 2. Integrate Metamask 3. Create Contracts	1,2,3,4	1,2,3,4	3,4,5,6,7,13

4. create a transaction history.			

Sprint Report 2

Implemented functionality	Planned user stories for this sprint	Completed user stories for this sprint	Planned user stories for next sprint
5. Testing the application 6. Enhance User interface. 7. Alpha/Beta testing	5,6,7	5,6,7	

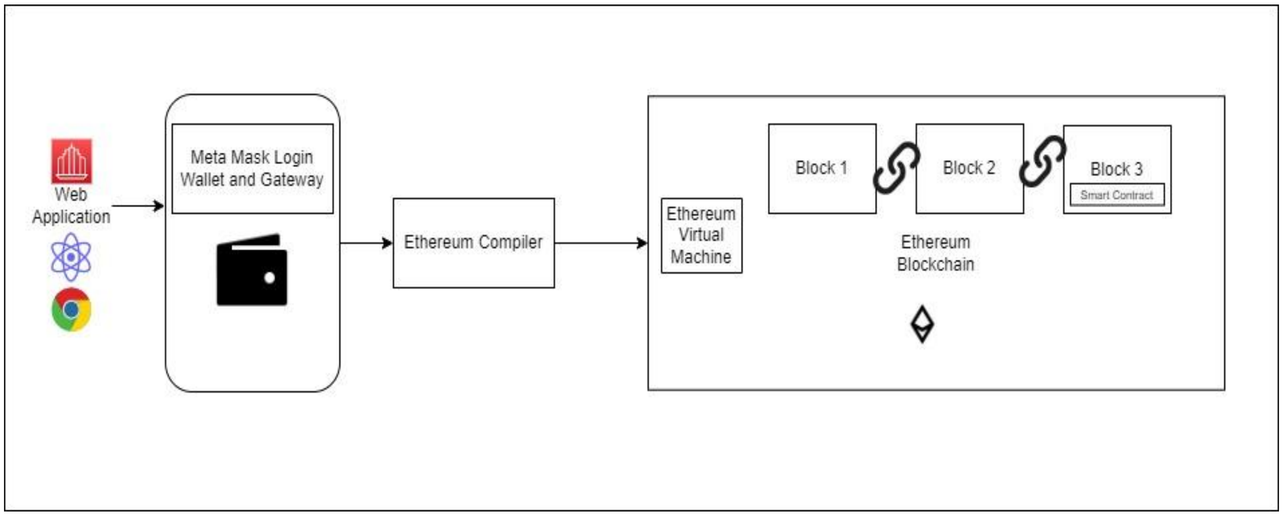
System Decomposition

1. Web Application: This application will be responsible to initiate login from the user. Once the user logs in the user will get a front-end. The web application will help the user initiate and authorize a transaction. Logs will be updated on the dashboard after every transaction. Web App also provides the user with wallet balances and individual wallet transaction logs.
2. Meta mask connection: This will help us have a user profile and provide a wallet connection for transactions. Available as a browser extension and as a mobile app, MetaMask equips you with a key vault, secure login, token wallet, and token exchange—everything you need to manage your digital assets.
3. Blockchain: Functions similar to a distributed virtual computer in operation. On this virtual machine, smart contracts that were created using high-level programming languages like Solidity act as the backend APIs. The frontend

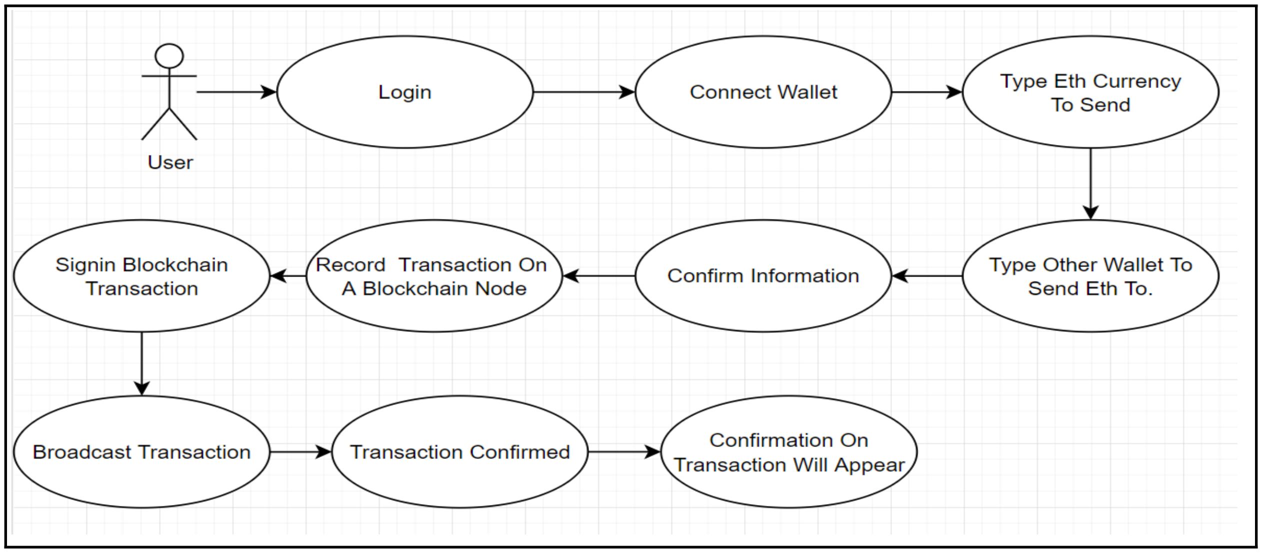
broadcasts user requests in the form of transactions to these smart contracts using a node on the blockchain network (usually accessed through a node-as-a-service provider), which then executes the invoked logic on the virtual machine. Upon completion, the transaction information and state changes are cryptographically verifiably stored in the blockchain ledger.

UML Diagrams

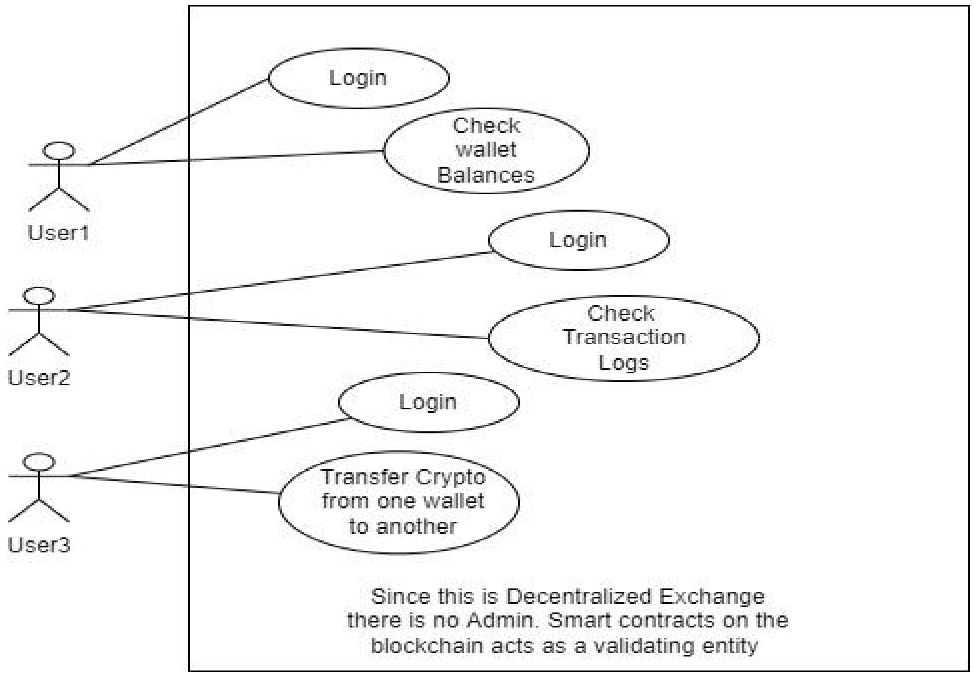
a. Software architecture diagram:



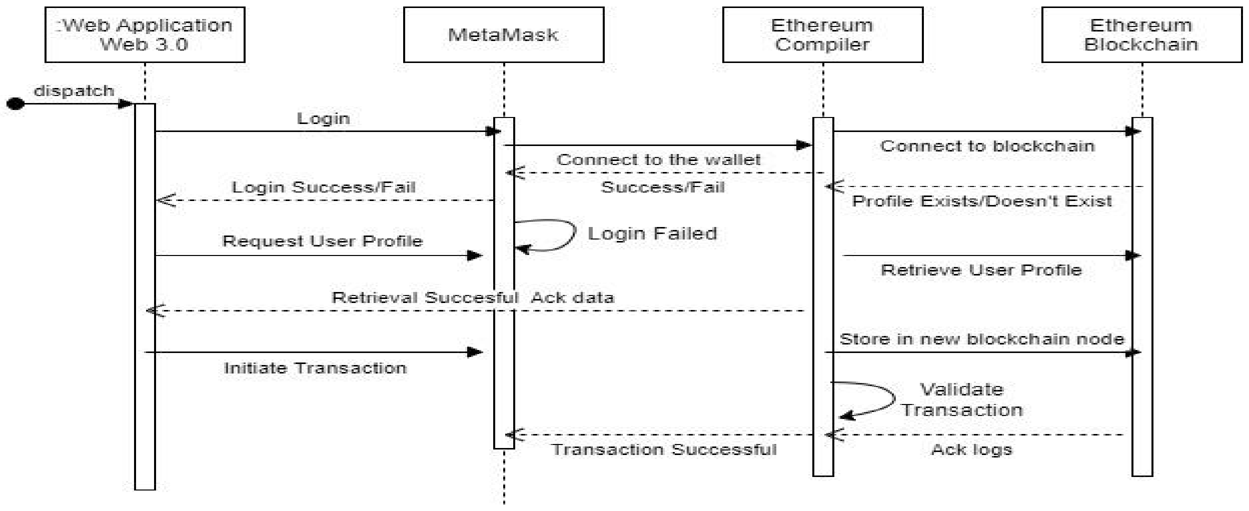
b. Activity diagrams:


User
Login
Connect Wallet
Type Eth Currency To Send
Type Other Wallet To Send Eth To.
Confirm Information
Record Transaction On A Blockchain Node
Signin Blockchain Transaction
Broadcast Transaction
Transaction Confirmed
Confirmation On Transaction Will Appear

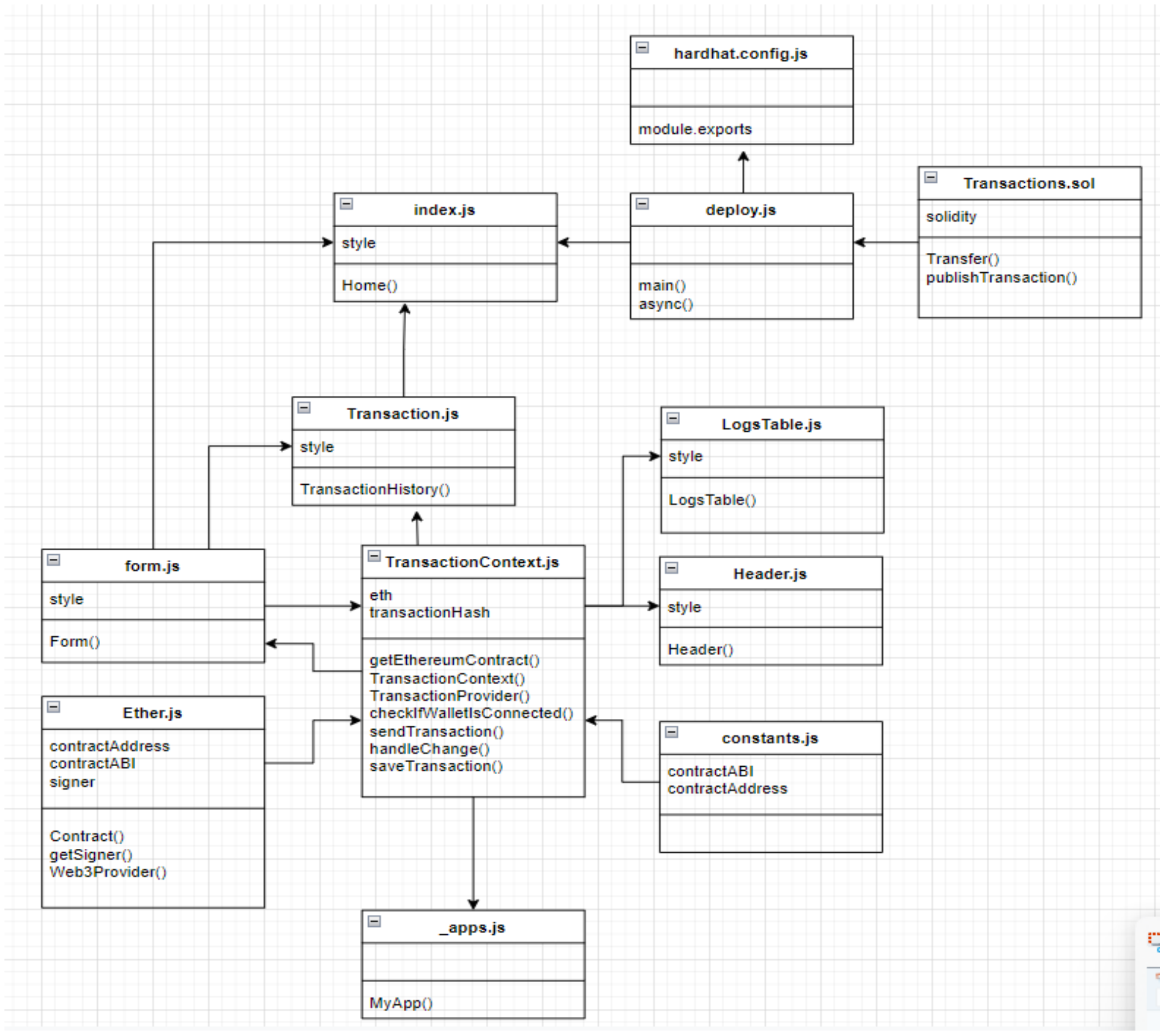
c. Use case Diagram:


Login
Check wallet Balances
User1
Login
Check Transaction Logs
User2
Login
Transfer Crypto from one wallet to another
User3
Since this is Decentralized Exchange there is no Admin. Smart contracts on the blockchain acts as a validating entity

d. Sequence diagram:



e. Class diagram:



Programming Languages

The frontend application has been developed with **Next JS**. This is a **react** framework by **Vercel**. Smart contracts are made using **Solidity**. **Web3.0** Authentication is done by using **MetaMask**. The send and receive functionality for tokens on blockchain was made using **Solidity**.

Reused Components and Programs

The npm logger library is the reusable component of the project. We import this library into our frontend applications and configure the logs based on this logger library.

Tools and Environments

- **JIRA** was used for sprint planning and backlog management.
- **Github** was used for version control of the project.
- **Visual Studio Code** was used as an IDE

Database System and Documents Stored

Since this is a Decentralized exchange project all the data is encrypted and stored in one block of the blockchain.

Text Files

There are no text files produced or consumed in this project. The data produced by the application gets stored in and is retrieved from the blockchain using the key to encrypt or decrypt.

User Manual

Dependencies

- Install Git - instructions at https://github.com/git-guides/install-git
- Install Node - instructions at https://nodejs.org/en/download/
- Install Next Js - instructions at https://nextjs.org/docs

- Install npm package manager - Instructions at https://docs.npmjs.com/downloading-and-installing-node-js-and-npm
- Install solidity - Instructions at https://docs.soliditylang.org/en/v0.8.13/installing-solidity.html
- Install MetaMask from - https://metamask.io/
- Check if all dependencies are installed successfully on your system

Crypto Exchange platform

GitHub Link:

Steps to run on the local environment:

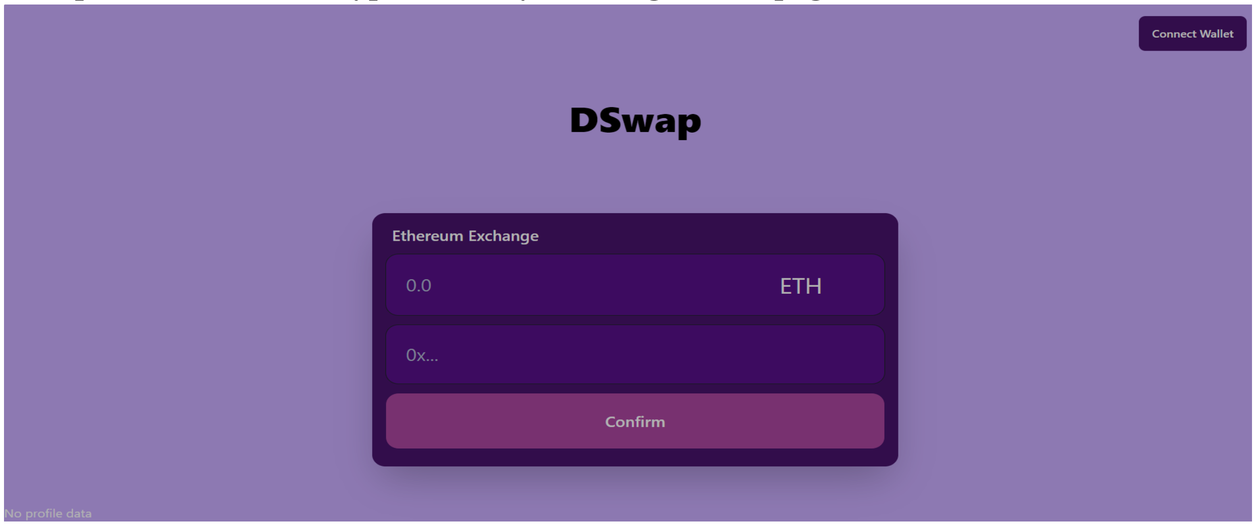
1. Clone the GitHub repository
2. Open a command prompt and navigate to the cloned repository
3. Run command npm install in client folder and studio folder - This will automatically install all dependencies.
4. Go to client folder and then run `yarn dev` command and open http://localhost:3000 on your browser
5. To start solidity server, go to studio folder and run command `sanity start`

Json part generated by the code:

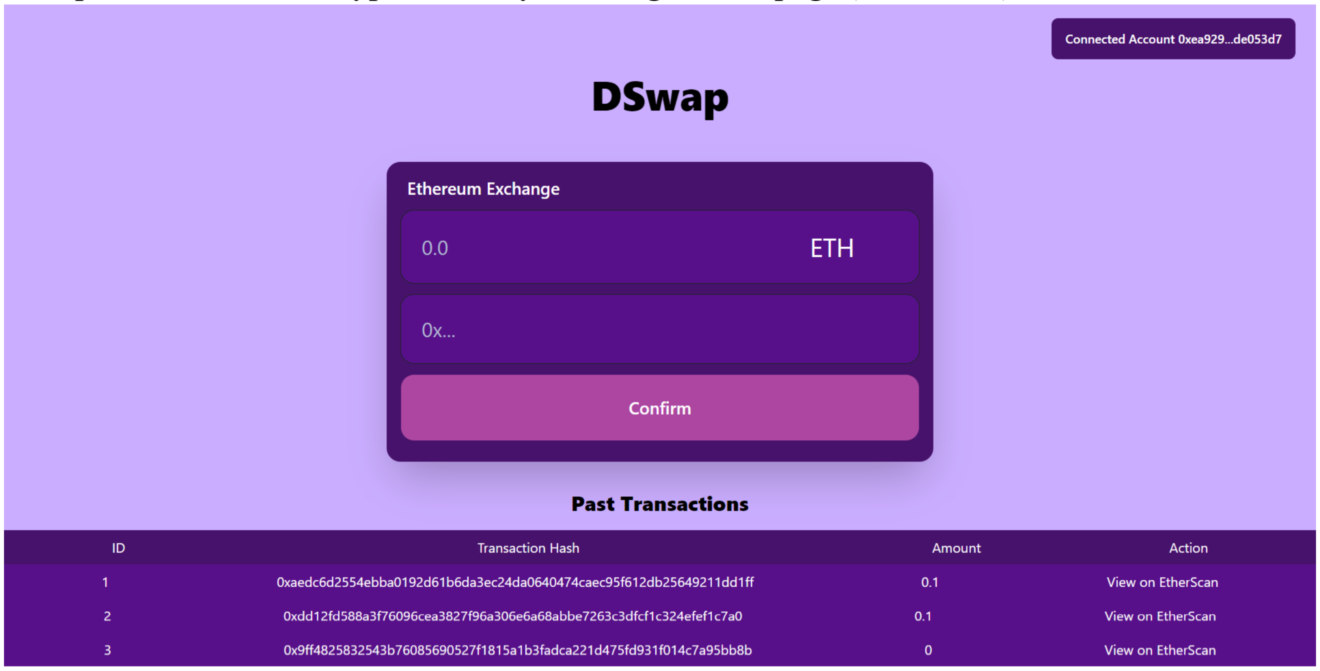
```
{
  "blockHash":"0x616528a16d7aa732e8b981d493929d04148201f89b7d10068376bd09dd9a31ce",
  "blockNumber":"10915875",
  "confirmations":"28",
  "contractAddress":"",
  "cumulativeGasUsed":"9891351",
  "from":"0xda500c9444b7c9a376758086c1573755d69f53ee",
  "gas":"520000",
  "gasPrice":"1940000000",
  "gasUsed":"21000",
  "hash":"0x1e34dc114d69d235a739f48b5aafce9601ce022ca4f46be4a63f050d854fd045",
  "input":"0x",
  "isError":"0",
  "nonce":"20",
  "timeStamp":"1656193740",
  "to":"0x6704f716eb322f6cc53194d4495455da6f474729",
  "transactionIndex":"8",
  "txreceipt_status":"1",
  "value":"33000000000000000"
}
```

Sample.

Dswap - Decentralized Cryptocurrency Exchange Home page:

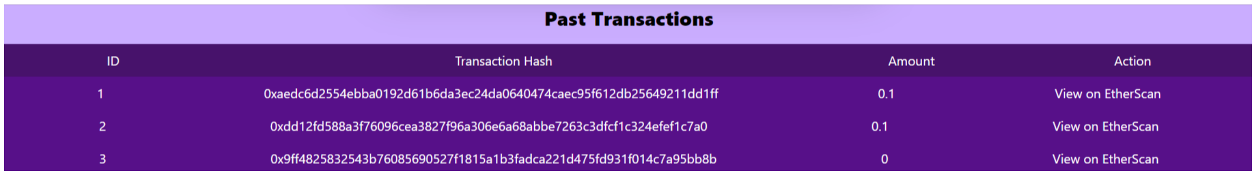


Dswap - Decentralized Cryptocurrency Exchange Home page (Connected):



ID	Transaction Hash	Amount	Action
1	0xaedc6d2554ebba0192d61b6da3ec24da0640474caec95f612db25649211dd1ff	0.1	View on EtherScan
2	0xdd12fd588a3f76096cea3827f96a306e6a68abbe7263c3dfcf1c324efef1c7a0	0.1	View on EtherScan
3	0x9ff4825832543b76085690527f1815a1b3fadca221d475fd931f014c7a95bb8b	0	View on EtherScan

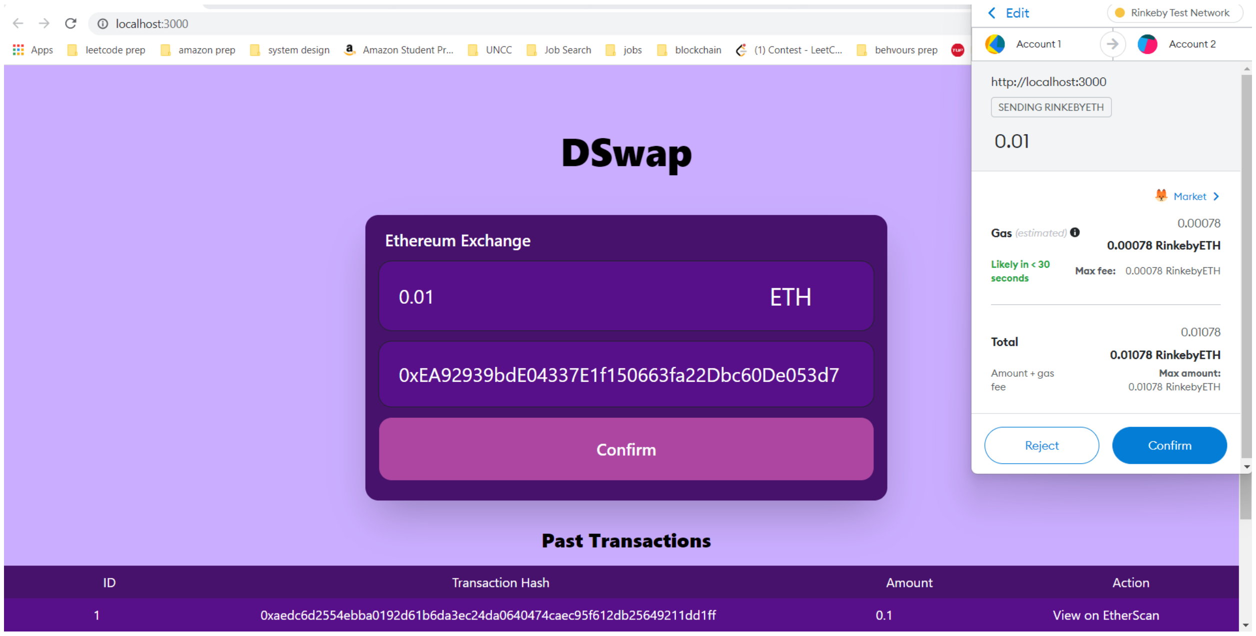
Past Traction



Past Transactions

ID	Transaction Hash	Amount	Action
1	0xaedc6d2554ebba0192d61b6da3ec24da0640474caec95f612db25649211dd1ff	0.1	View on EtherScan
2	0xdd12fd588a3f76096cea3827f96a306e6a68abbe7263c3dfcf1c324efef1c7a0	0.1	View on EtherScan
3	0x9ff4825832543b76085690527f1815a1b3fadca221d475fd931f014c7a95bb8b	0	View on EtherScan

Confirming the transaction



DSwap

Ethereum Exchange

0.01 ETH

0xEA92939bdE04337E1f150663fa22Dbc60De053d7

Confirm

Edit

Rinkeby Test Network

Account 1 → Account 2

http://localhost:3000

SENDING RINKEBYETH

0.01

Market

Gas (estimated) 0.00078
0.00078 RinkebyETH

Likely in < 30 seconds

Max fee: 0.00078 RinkebyETH

Total 0.01078
0.01078 RinkebyETH

Amount + gas fee

Max amount: 0.01078 RinkebyETH

Reject Confirm

Past Transactions

ID	Transaction Hash	Amount	Action
1	0xaedc6d2554ebba0192d61b6da3ec24da0640474caec95f612db25649211dd1ff	0.1	View on EtherScan

Conclusion

The objective of the project was to build and demonstrate the Decentralized exchange system for Ethereum transactions in the period of our coursework. We had an opportunity to learn and work on a new Front end framework like Next Js. The project also helped us understand the working of blockchain and its advantages. The team also practiced industry level agile model and worked in sprints. The project can be extended to other cryptocurrencies in future. The project can also be extended to include visual analytics showing transactions over a period of time and also fetch real-time data of cryptocurrency rates.