TABLE OF CONTENTS

1.0 Introduction ……1
- 1.1 Overview ……..1
- 1.2 Scope ……...1

2.0 Design Constraints, Assumptions, and Dependencies ……..1

3.0 Design Descriptions
- 3.1 Master Class Diagram ……...2
- 3.2 Module 1 ……..3
- 3.3 Module 2 ……..6
- 3.4 Module 3 ……..8

4.0 ER Diagrams ……13

5.0 User Interface Diagrams ……14

6.0 Report Layouts ……15

7.0 External Interfaces ……..15

8.0 Packaging and Deployment Diagrams ……..16

9.0 Help ……16

10.0 Reusability Considerations ……..16

11.0 Traceability Matrix ……..16

1.0 Introduction

This document contains an overview of the project, the system architecture, the hardware and software requirements, the performance characteristics of the product being developed. It also lists out the general characteristics of the eventual users of the product and their specific requirements.

1.1 Scope

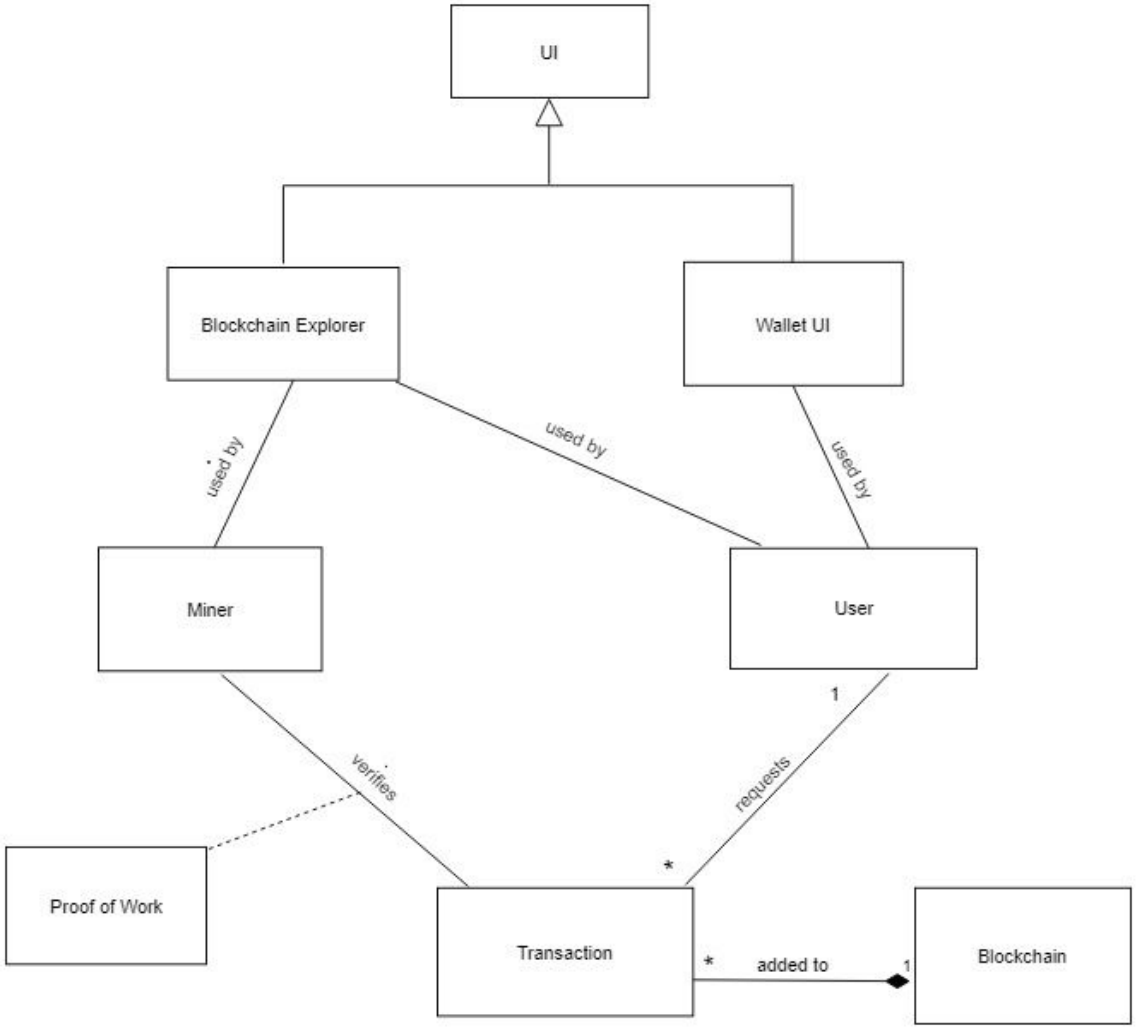
The aim of this project is to implement blockchain technology for the creation and use of a cryptocurrency which we have termed "Brownie Points". Blockchain for cryptocurrency involves trustworthy transactions in a trustless world. It enables anonymous transactions of currency. It transfers control from central authorities to anyone and everyone. Some of its features include decentralized authority, distributed system and secure transactions. The novelty in this project involves the use of quantum resistant hashes for securing transactions and the use of secure wallets.

2.0 Design Constraints, Assumptions, and Dependencies

- The software needs to store the private keys securely(cannot be easily stolen or copied) and still be easily accessible by authorized users.
- The current high level network status(like speed and its corresponding price) is difficult to be shared in real time to users, this might lead them paying more and still not be able to get the required speed due to network congestion.
- The user needs to double check before creating a transaction, as erroneous transactions are irreversible.
- The user needs to have a web browser that supports HTML5 to use the software, as it offers only a web based UI.
- Every transaction approximately takes 4-5 minutes to be committed to the ledger
- Since the transaction pool is publicly available, the primary constraint is to make individual transactions inherently anonymous. Wallet addresses are not directly indicative of identity, however, given a wallet, it is easy to identify all the transactions committed to/from that wallet and hence, by that user.

3.0 Design Descriptions

3.1 Master Class Diagram



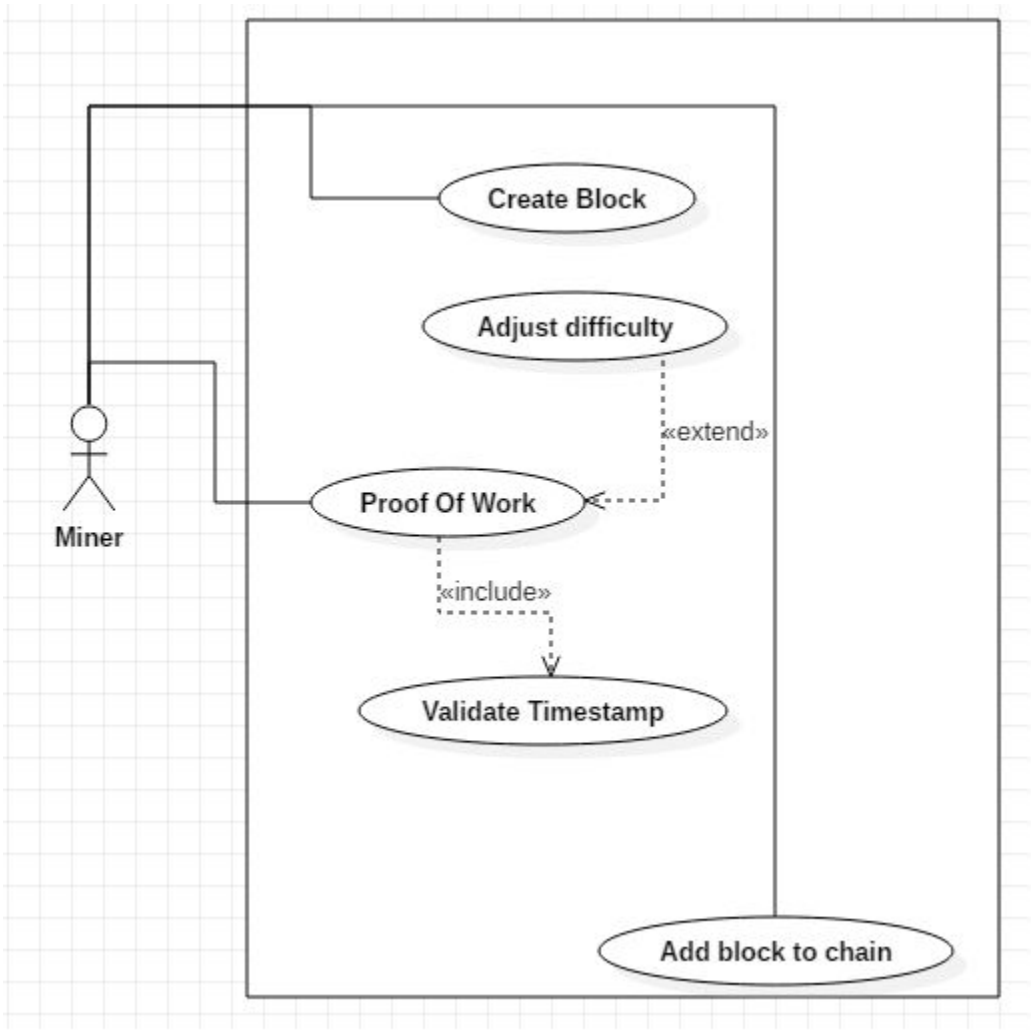
3.2 Module 1 - Working Blockchain

3.2.1 Description

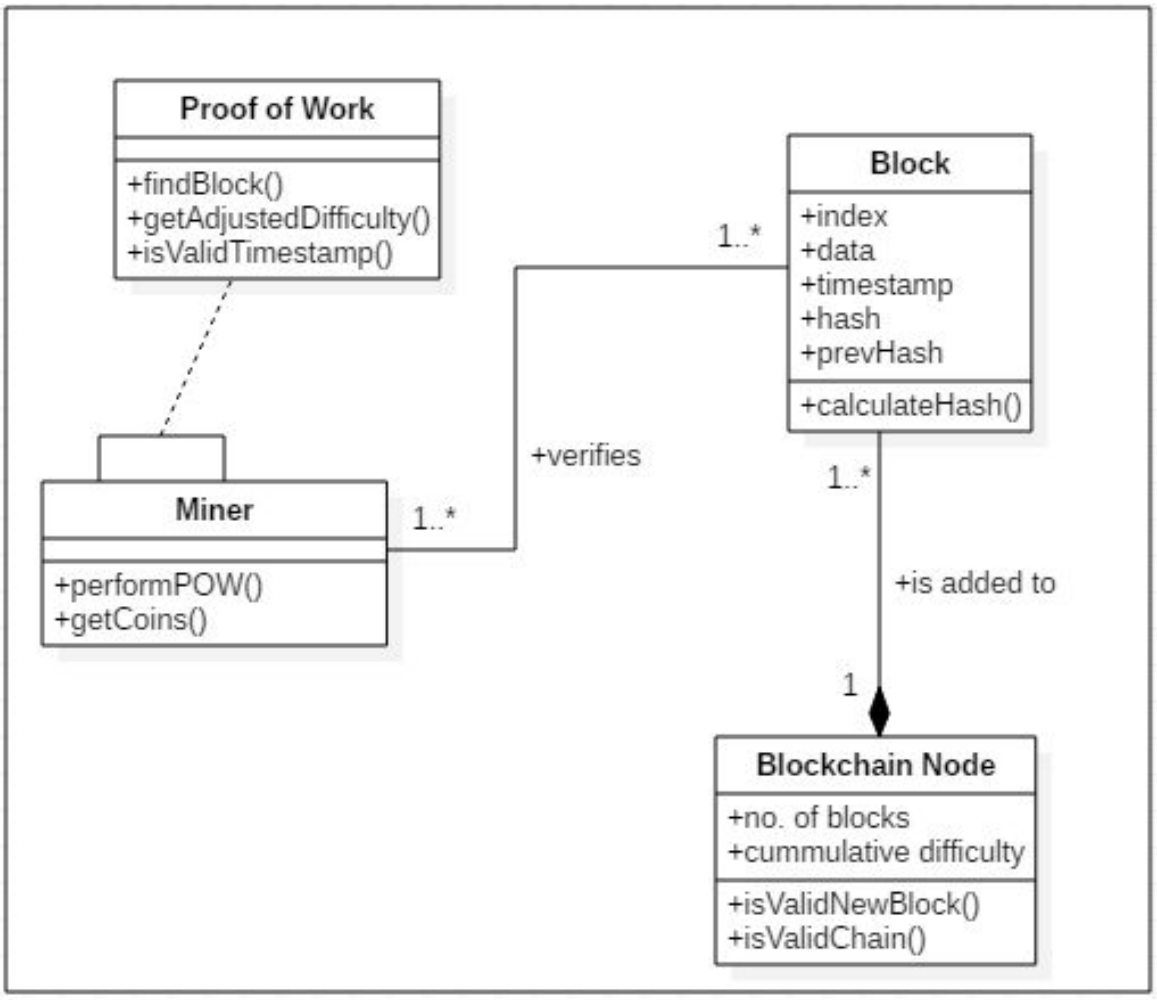
The **Working Blockchain** module consists of two submodules, **Blockchain** and **Proof of Work**. The Blockchain module is used for creating new blocks, broadcasting the newly created blocks in a **P2P network**, to validate the structure of the blocks and the entire blockchain itself.

The major task accomplished by the Proof of Work module is to solve a puzzle to add the block into a blockchain. This is dependent on the difficulty set for the puzzle which is in turn dependent on the BLOCK_GENERATION_INTERVAL, this ensures that blocks are not added very frequently. In addition to controlling the mining rate the Proof of Work module also validates the timestamps in the blocks. The Miner is the only actor who engages directly with the Working Blockchain module and earns coins for the work performed.

3.2.2 Use Case Diagram



3.2.3 Class Diagram



3.2.3.1 Class Descriptions

- **Block** is a class that represents an individual basic units of the blockchain. It has the attributes:
    - **index** : The height of the block in the blockchain
    - **data**: Any data that is included in the block.
    - **timestamp**: The time when the block was added.
    - **hash**: A sha256 hash taken from the content of the block
    - **previousHash**: A reference to the hash of the previous block. This value explicitly defines the previous block.

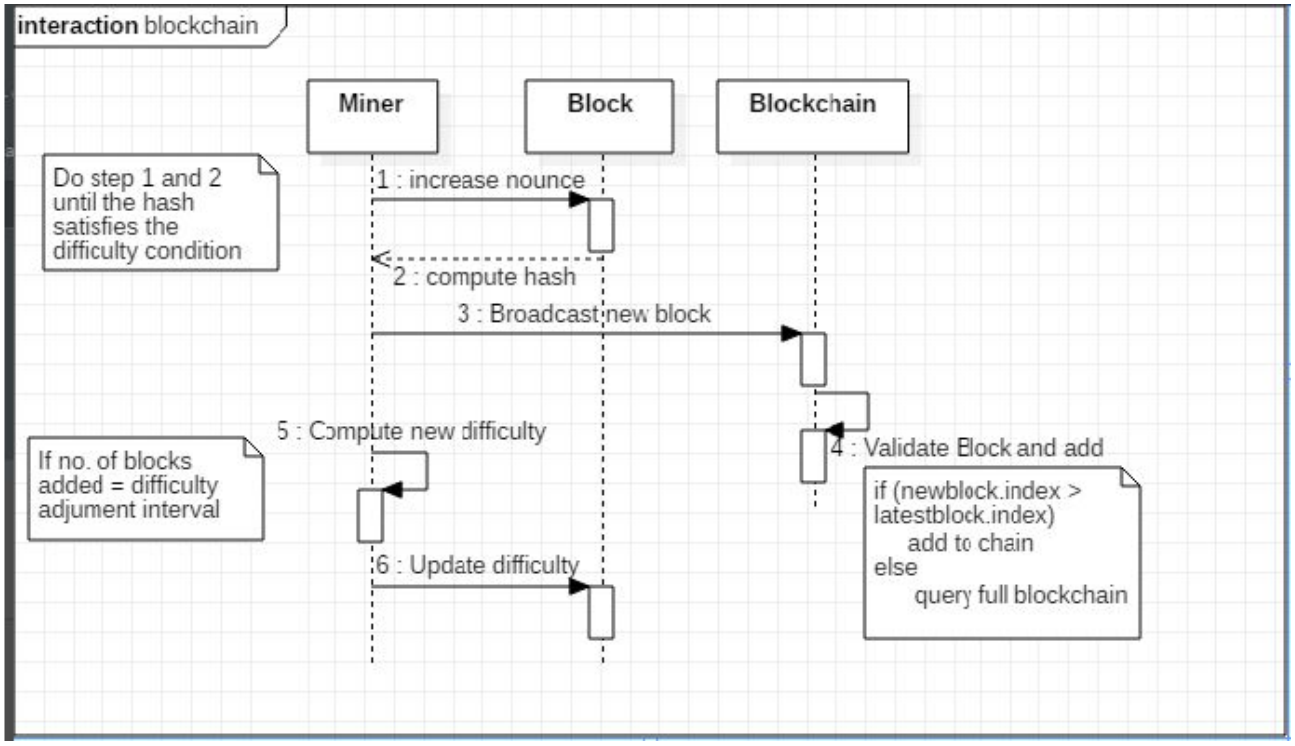
The operations include creating a new instance of the block by computing the hash using SHA256 with all attributes as parameters.

- **Blockchain Node** validates the new blocks that come in from other peers and also the entire blockchain.The cumulative difficulty is a attribute used to decide which chain to keep when there are duplicate blocks in two different chains.

- **Miner** class works coordinating all other classes. It creates a new block, performs proof of work and broadcasts it to all other peers in the network. Every miner is rewarded some coins for the work performed.

- **Proof of work** is an association class that is used by the miner to perform operations like finding a block that satisfies the difficulty condition for the hash. It also performs periodic adjustment of difficulty level for the puzzle.

3.2.3.2 Sequence Diagram



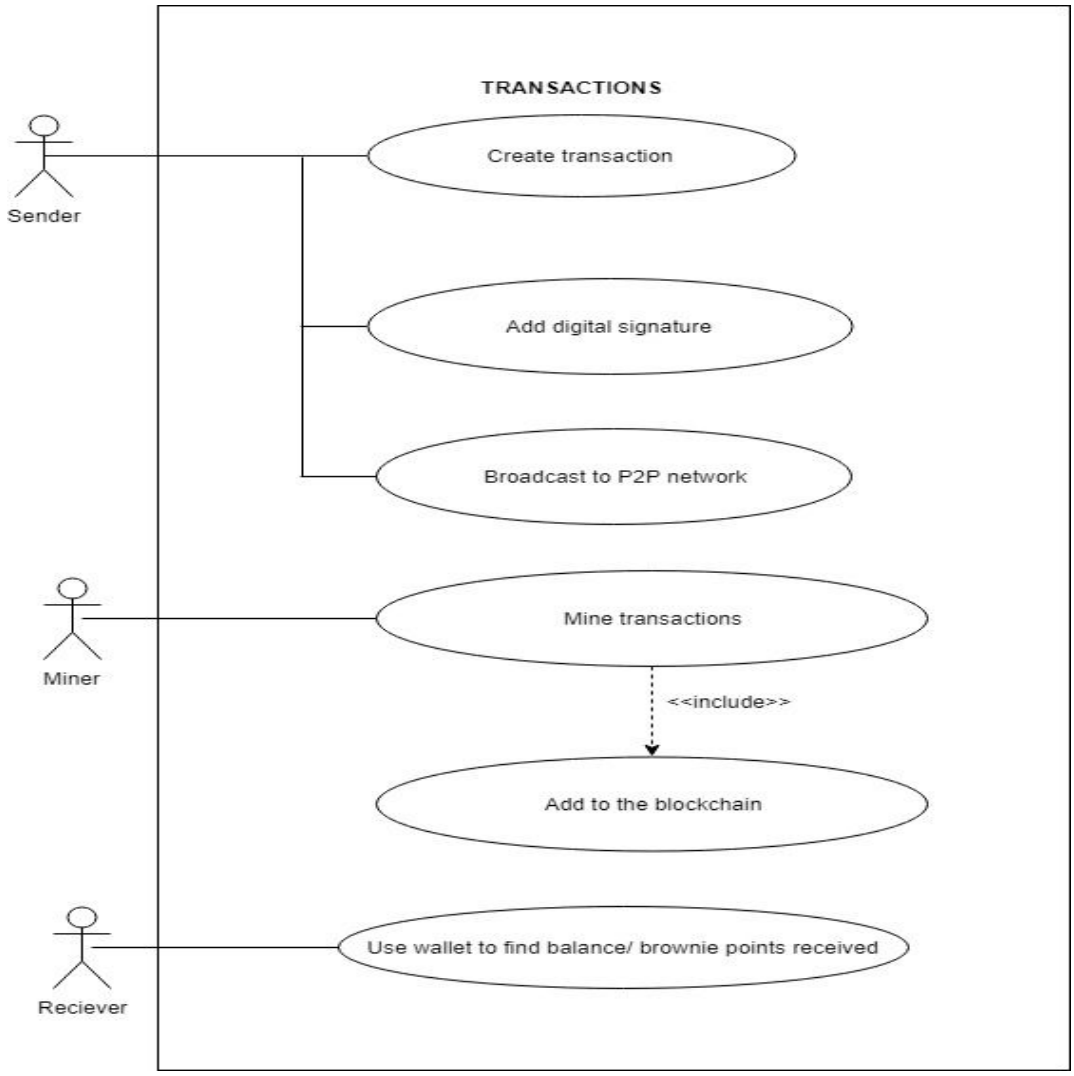
3.3 Module 2 - Transactions

3.3.1 Description:

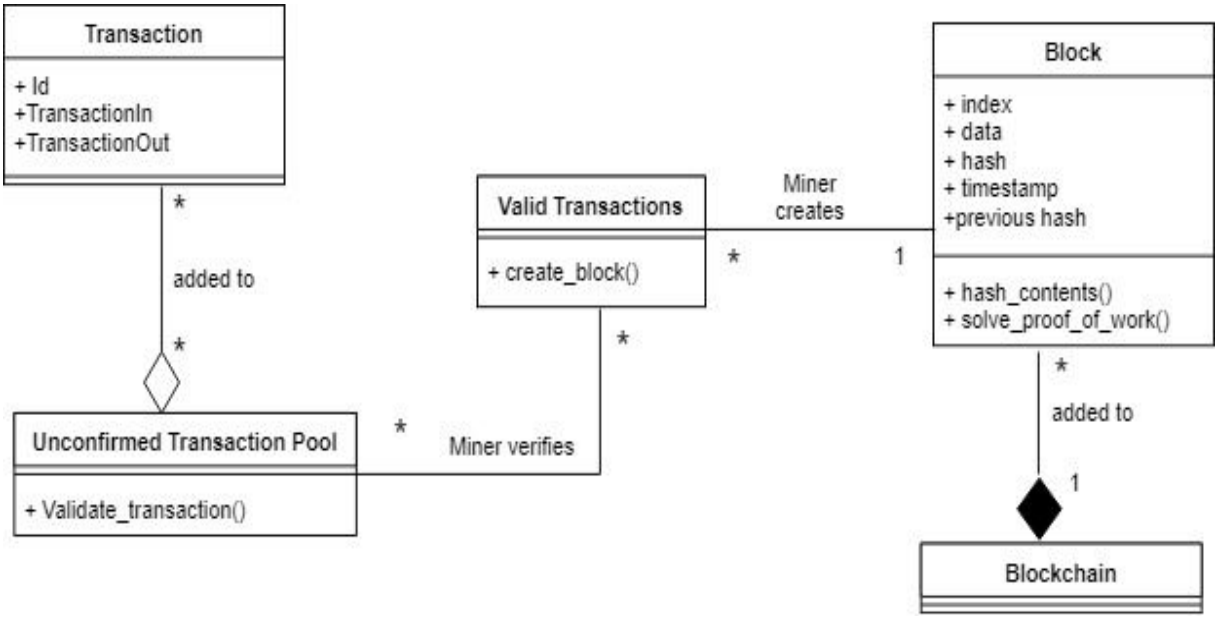
Transactions are made between a sender and a receiver. The sender signs the transactions and these transactions are public and broadcast to the entire **P2P network**, but only the person with the appropriate **private key** can obtain/access them.

The P2P network of nodes will now validate the transaction using a set of known algorithms with the criteria that the format of these transactions should match and the transaction inputs, outputs and signatures should match. On validation of the transaction, the miner creates a block , solves a proof of work algorithm and adds it to the existing blockchain.

3.3.2 Use Case Diagram



3.3.3 Class Diagram



3.3.3.1 Class Description 1

Transactions class consists of a

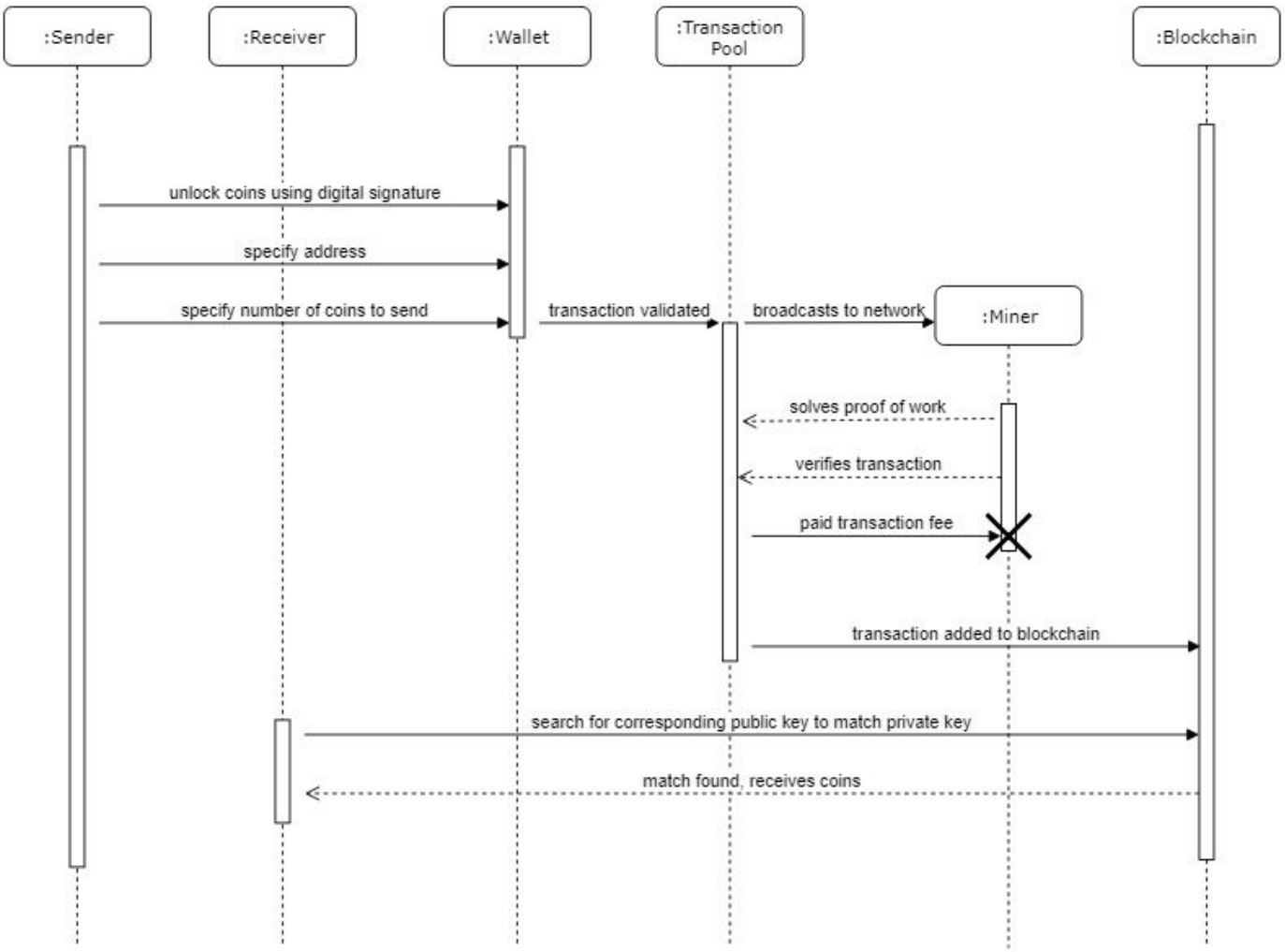
- **Transaction Id** is represented by taking a hash of the contents of the transaction.
- **Transaction input** contains the address from where the transaction is coming. It contains the digital signature of the sender which is signed using the public key.
- **Transaction output** contains the address (public key) of the sender and the amount of coins sent. This can be unlocked by the receiver using private key.

Unconfirmed Transaction Pool is a class which contains all unconfirmed transactions. These transactions are now broadcast to the entire network.

Confirmed and valid Transactions is a class which contains valid transactions and have been verified that it is an unspent brownie point and verified of its ownership by the P2P network.

Block is a class which has valid transactions bundled together. It is created a miner and a proof of work algorithm has to be solved to add this to the **Blockchain.**

3.3.3.2 Sequence Diagram



3.4 Module 3 - User Interface

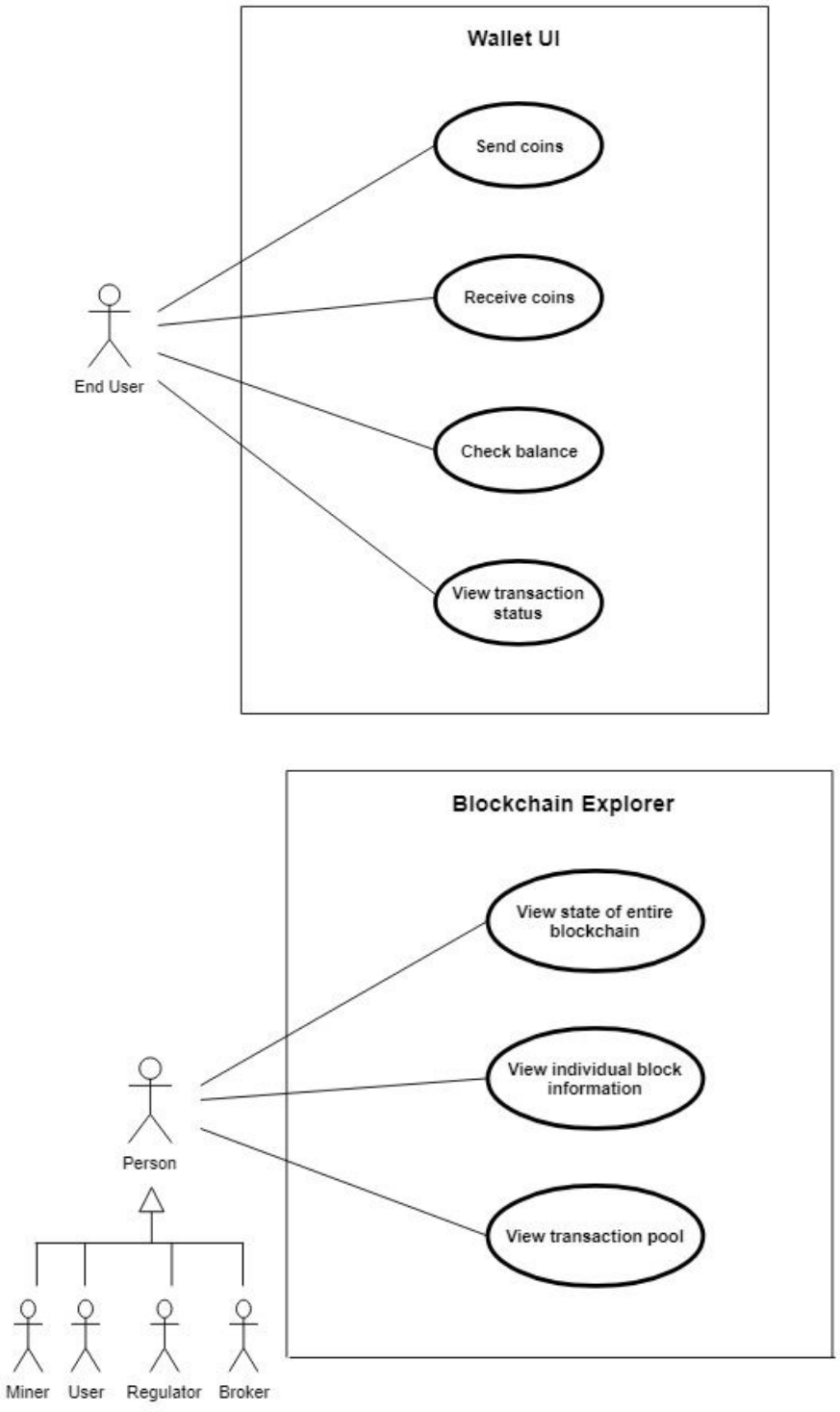
3.4.1 Description

The user interface comprises mainly of two parts:

- **Wallet UI :** It is a digital wallet which enables the customer to send coins, buy assets and check balance from their transactions. They contain the public and private key of the customer. They do not explicitly contain the currency.

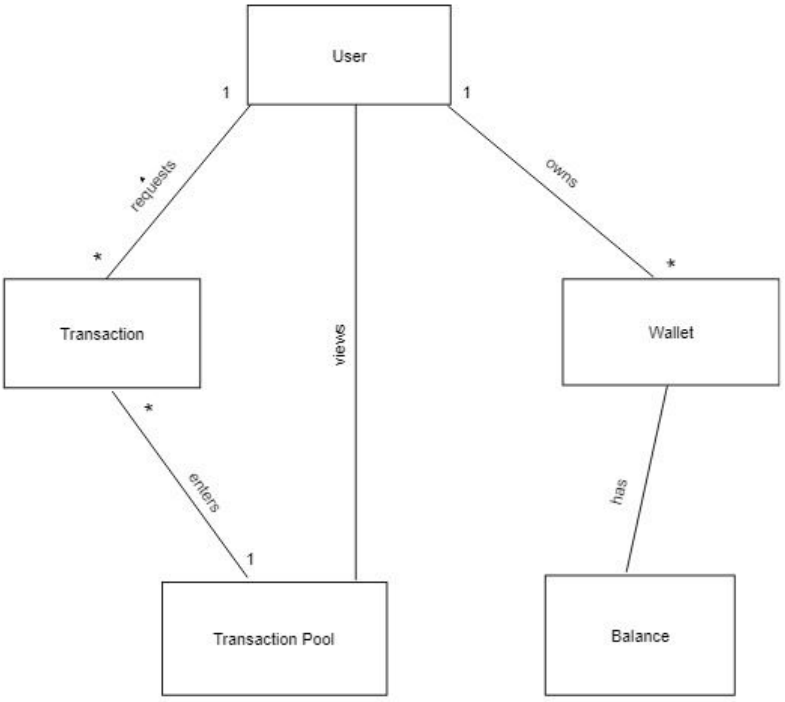
- **Blockchain Explorer:** It is a web interface which allows the customer to visualize the state of transactions.It also helps in visualising individual block information, the balance for a given public key.

3.4.2 Use Case Diagram

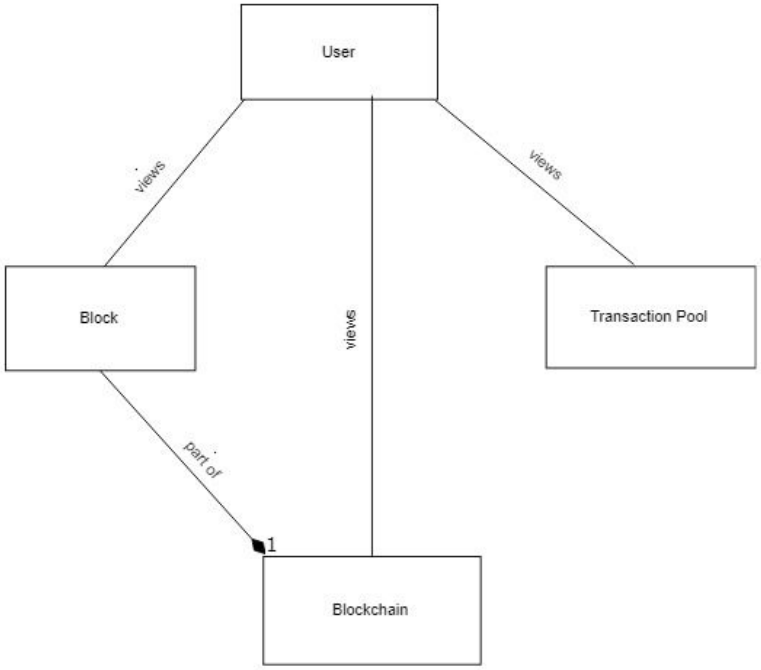


3.4.3 Class Diagram

Wallet UI



Blockchain Explorer



3.4.3.1 Class Description 1

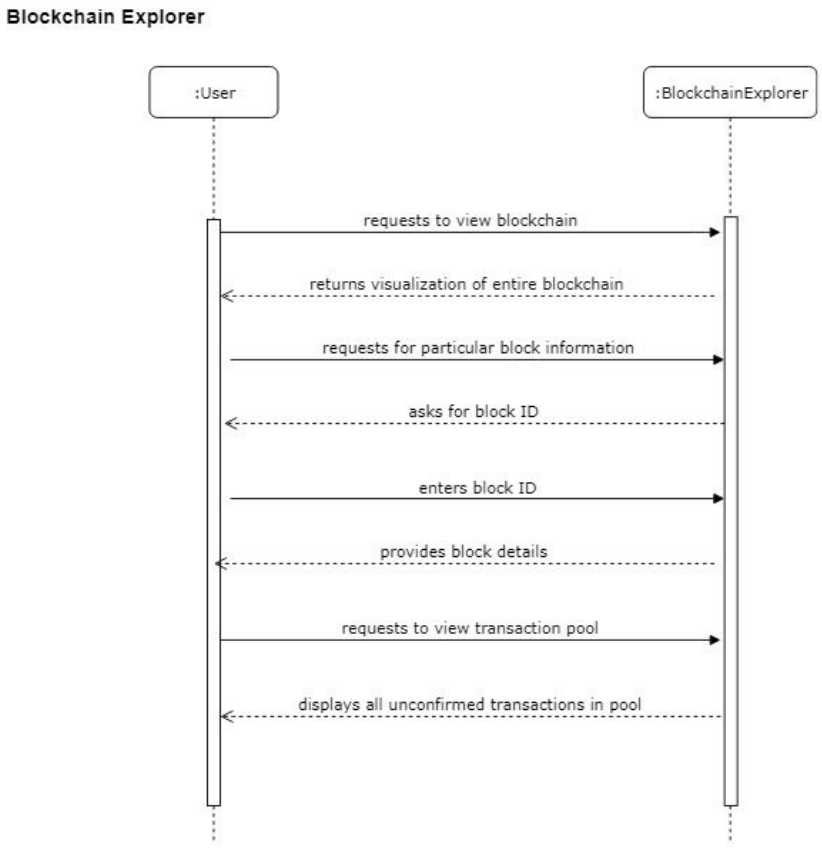
Wallet UI:

- **User** : He can own many wallets and make many transactions.
- **Wallet** : Contains public, private key and provides interface for user to make transactions. The wallet can be used in viewing the balance and the set of transactions made by customer.
- **Transaction** : It contains the address of the receiver and the amount of brownie points to be sent. It is digitally signed by the customer.Every transaction made by customer is added to a pool to be verified and added to the blockchain by the miner.

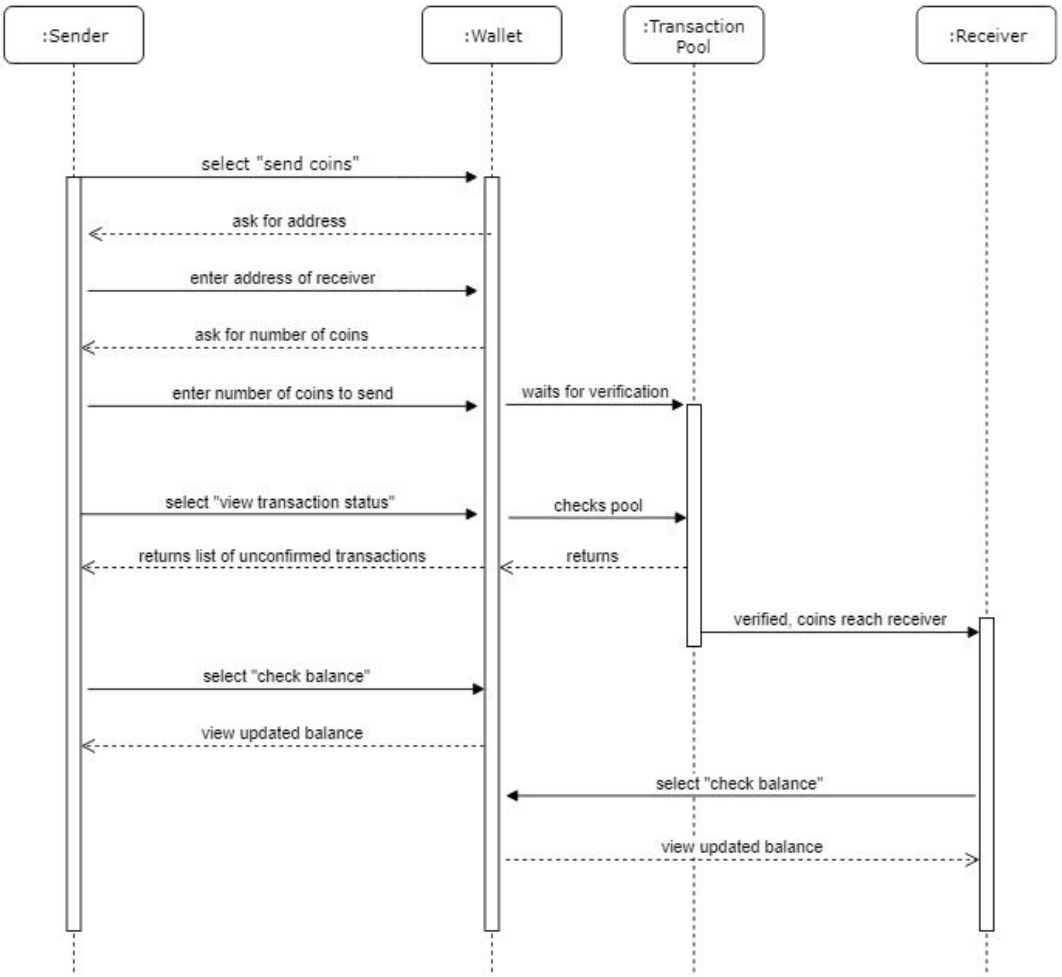
Blockchain Explorer:

- A user can:
  - State of blockchain
  - View information of a block
  - View previous transactions

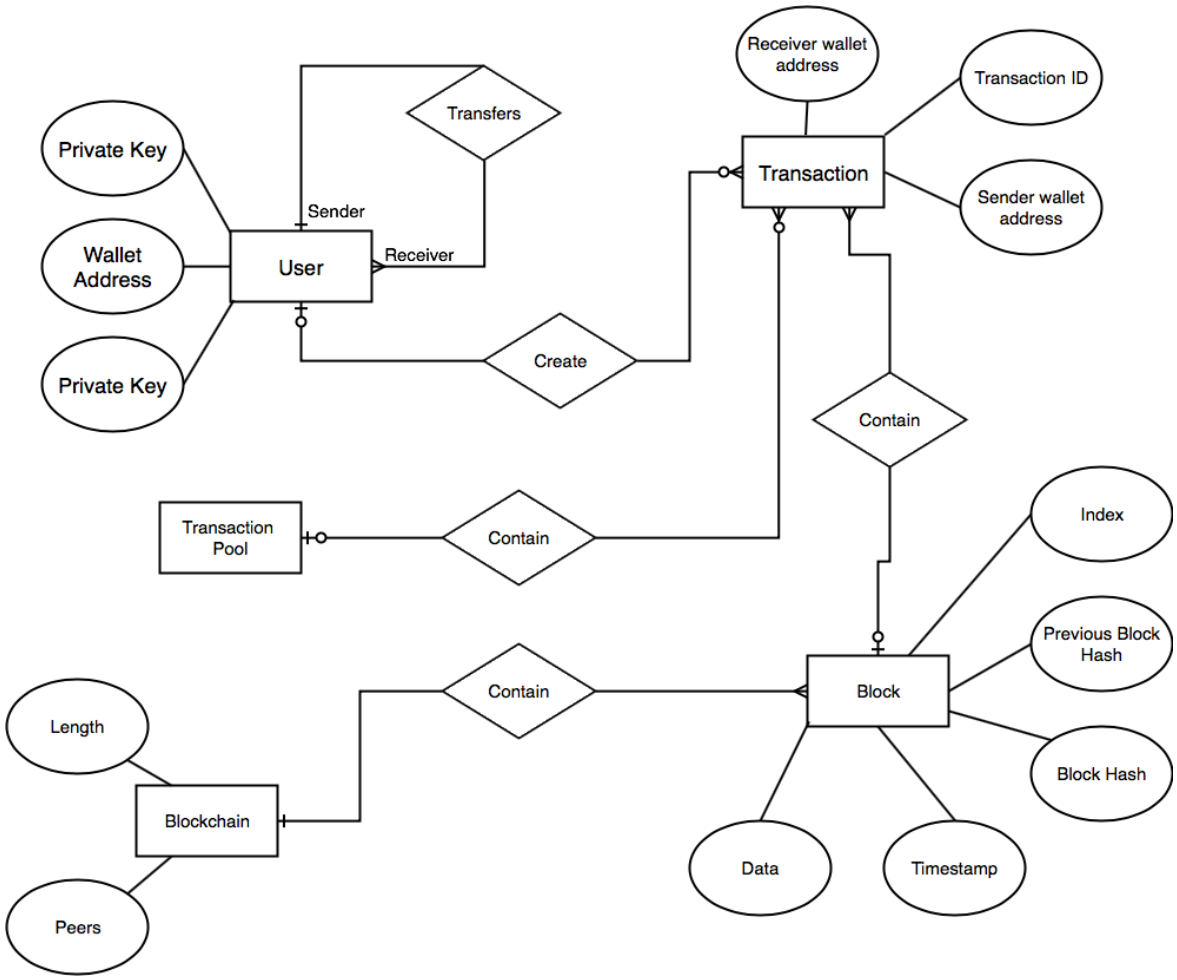
3.4.3.2 Sequence Diagram



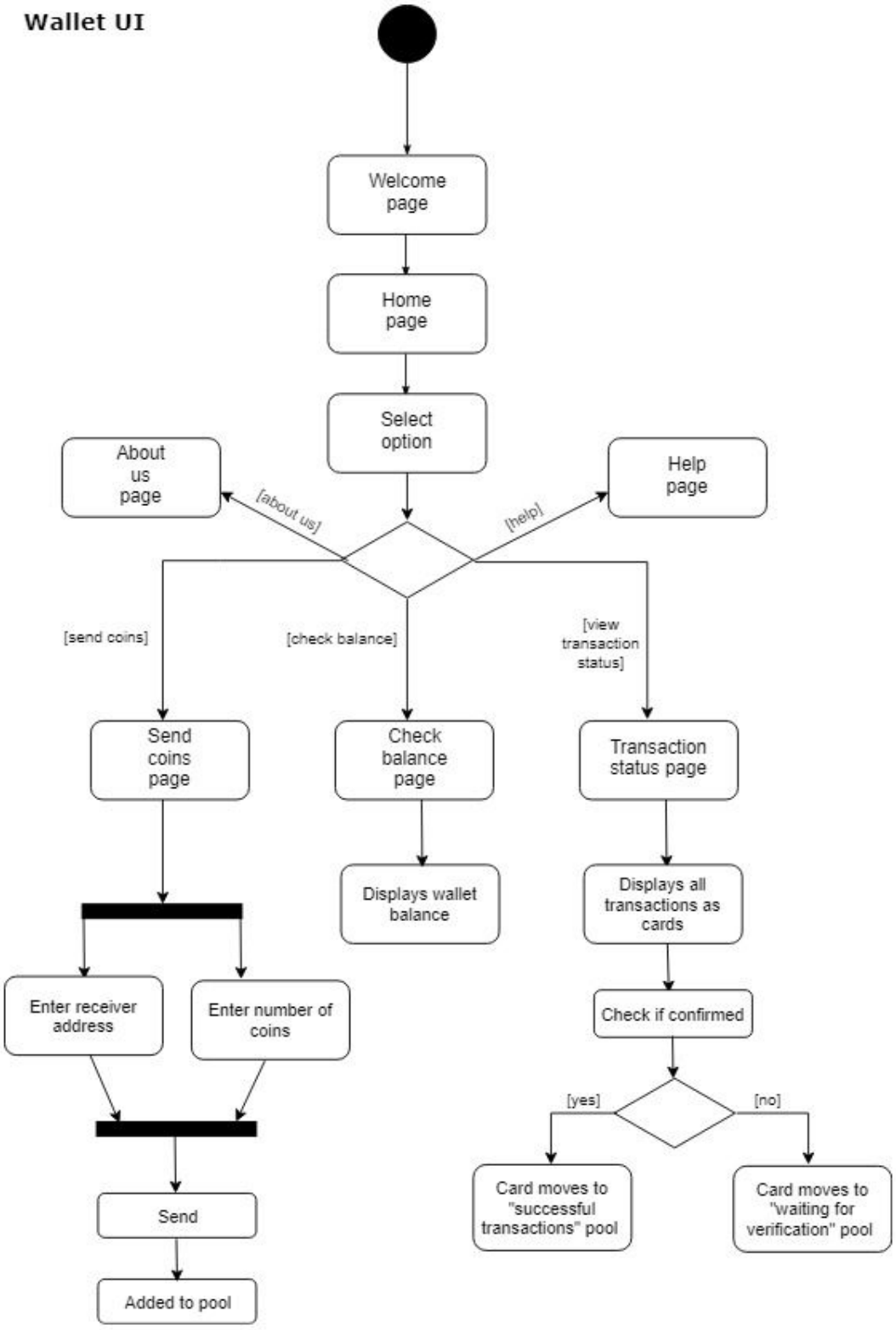
Wallet UI

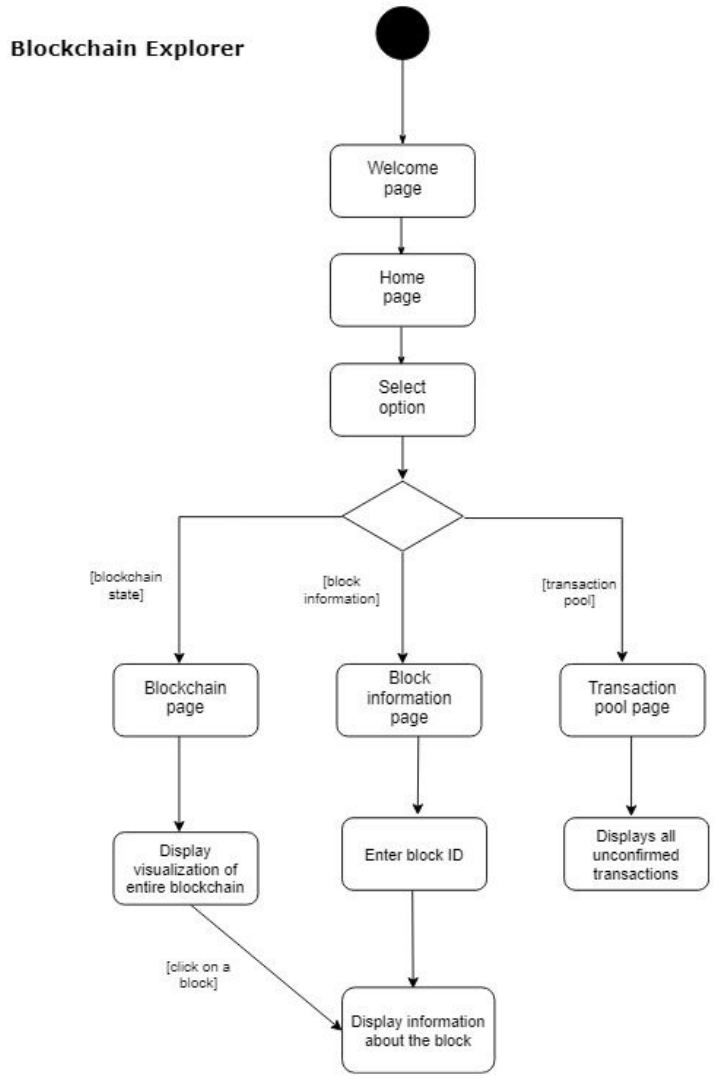

:Sender
:Wallet
:Transaction Pool
:Receiver
select "send coins"
ask for address
enter address of receiver
ask for number of coins
enter number of coins to send
waits for verification
select "view transaction status"
checks pool
returns list of unconfirmed transactions
returns
verified, coins reach receiver
select "check balance"
view updated balance
select "check balance"
view updated balance

4.0 Entity Relationship Diagram (ERD)



5.0 User Interface Diagrams





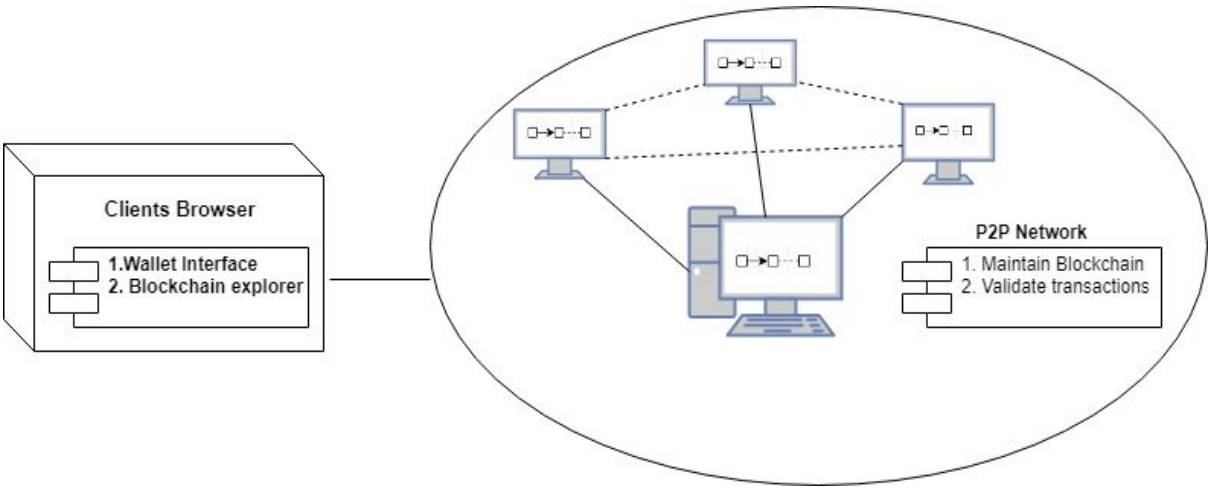
6.0 Report Layout

The reports will display the user's latest / most recent transactions. He / She can also view the details of these transactions such as the status of the transaction, the timestamp etc.

7.0 External Interfaces

There are mainly 2 external interfaces, namely, the blockchain explorer and the wallet UI. The wallet ui provides the means to send and receive coins, view balance and check the transaction status. The blockchain explorer is used to visualise the state of the blockchain. Information about the transactions added to the blockchain is available here, as well as a transaction pool with all the unconfirmed transactions. It is public and can be viewed by everyone, providing high transparency. REST APIs are another interface that is provided which can be used by any other program that is trying to communicate with the server.

8.0 Packaging and Deployment Diagrams



9.0 Help

Help will be provided in the form of documentation recorded prior to, during and after coding the solution. Documentation written prior to coding will help understand the intended use of the system. Detailed documentation on how to use each module / interface will be recorded for internal use.

10.0 Reusability Considerations

The modules have been written in a way such that code reuse is given the utmost priority. The design carefully incorporates the nuances of the system and detailed documentation for internal usage will allow for easy use of the same.

11.0 Traceability Matrix

CRS Reference Section No. and Name	Design/HLD Reference Section No. and Name

(Note - to be filled as and when a requirement has been satisfied.)