Work

Sushant:

1. Create testnet accounts for receiver (Bitcoin wallet), sender (Eth wallet), and server (both Bitcoin and Eth wallet).
2. Get access tokens for coinbase using OAuth
3. Create a Rest endpoint for sending money -> You might need another API for getting a conversion rate
4. Find a way to save transaction history (extra)

Saarim:

1. Download React bootstrap (or any other CSS framework for react. Semantic-UI React is also good)
2. Add React Router to our program. (We will have 3 pages. Login, Orgs, and Result/Transaction history)
3. Implement React pages -> Login will be a little tricky. You should have a way to know if you are logged in or not. So do login part later. (We can skip login feature if we have no time)

Sang:

1. Setup Heroku
2. Implement login with session
3. Finishing setting up mongodb and create CRUD functions & Schema

API:

Coinbase API: Support OAuth 2.0. Bitcoin & Ethereum & Litecoin supports
Stellar API: Support Stellar API

What to do next?
Figure out if we can use testNet for cointbase & stellar API

Architecture
Client (React) - Server(NodeJS) - MongoDB - firebase query is not strong (it can be fine since our program is very simple)
Coinbase API, Stellar API

Flow:

Client

1. User decides to send $100 to the organization A.
2. User needs his/her configuration. (Configuration will be saved in firebase or we can use default configuration) -> saved in database
3. User can set what blockchain it wants to use. -> saved in database

Server & Client

4. Server requests OAuth of coinbase to grant access from the user, /auth

Client

5. Once user finishes OAuth process, it can send money to stellar
6. Choose the organization that user wants to send money
7. Set how much it wants to send (ex, $100)
8. Click a send button. Since Oauth is already done, server can use user's account from coinbase

Server

9. Convert US dollar to user's cryptocurrency based on its settings (let's say bitcoin) -> need API to see conversion rate
10. Send money to server_wallet[BITCOIN]
11. Collect transaction logs using Coinbase API
12. Once money is transferred, convert the same amount of money to other cryptocurrency (In our case Stellar) by referencing the conversion API
13. Send money to the organization A's address saved in org_address[A]
14. Collect transaction logs using Stellar API
15. Once transfer has been confirmed, change user's state. Also update the blockchain transaction log for user's transaction.
16. Now client side can verify the transaction, /transaction?user=[userName]

Database Schema:

1. User
Name -> String
Config -> Config
TransactionRecords -> list(TransactionRecord)

2. Config
DefaultBlockchain -> String.

3. TransactionRecord
BlockchainLog -> list(Log)
DestinationAddress -> String
MoneySent -> float

4. Log
Logs -> list(String)

Rest Endpoint requirement

OAuth

1. /auth: start OAuth
2. /callback: OAuth webhook callback
   (if there are any other endpoints required for Oauth add here)
   (Please update http methods here)

Client

1. POST /config?user=[userName]: Set user config (write body here)
2. GET /config?user=[userName]: get config information. For debugging purpose.
3. POST /send?user=[userName]&amount=[amount]&destination=[destinationString]
4. GET /transaction?user=[userName]&howMany=[number latest transaction]

External API

Coinbase API for OAuth, sending bitcoin&ethereum&litecoin to server address
Coinbase API for transaction logs
Stellar API to send lumen to destination address
Stellar API for transaction logs

Frontend:

- User Information:
  - initial coin, final coin, amount, their public address, receivers public address,
- Website Information:
  - estimate of how much final coin will be sent to the receiver, our public address that the user will send their coins to
- Backend Information:
  - Our public address that will receive the coins

Initial Coin

Final Coin

Your public address in Inital Coin

Receiver's public address in Final Coin

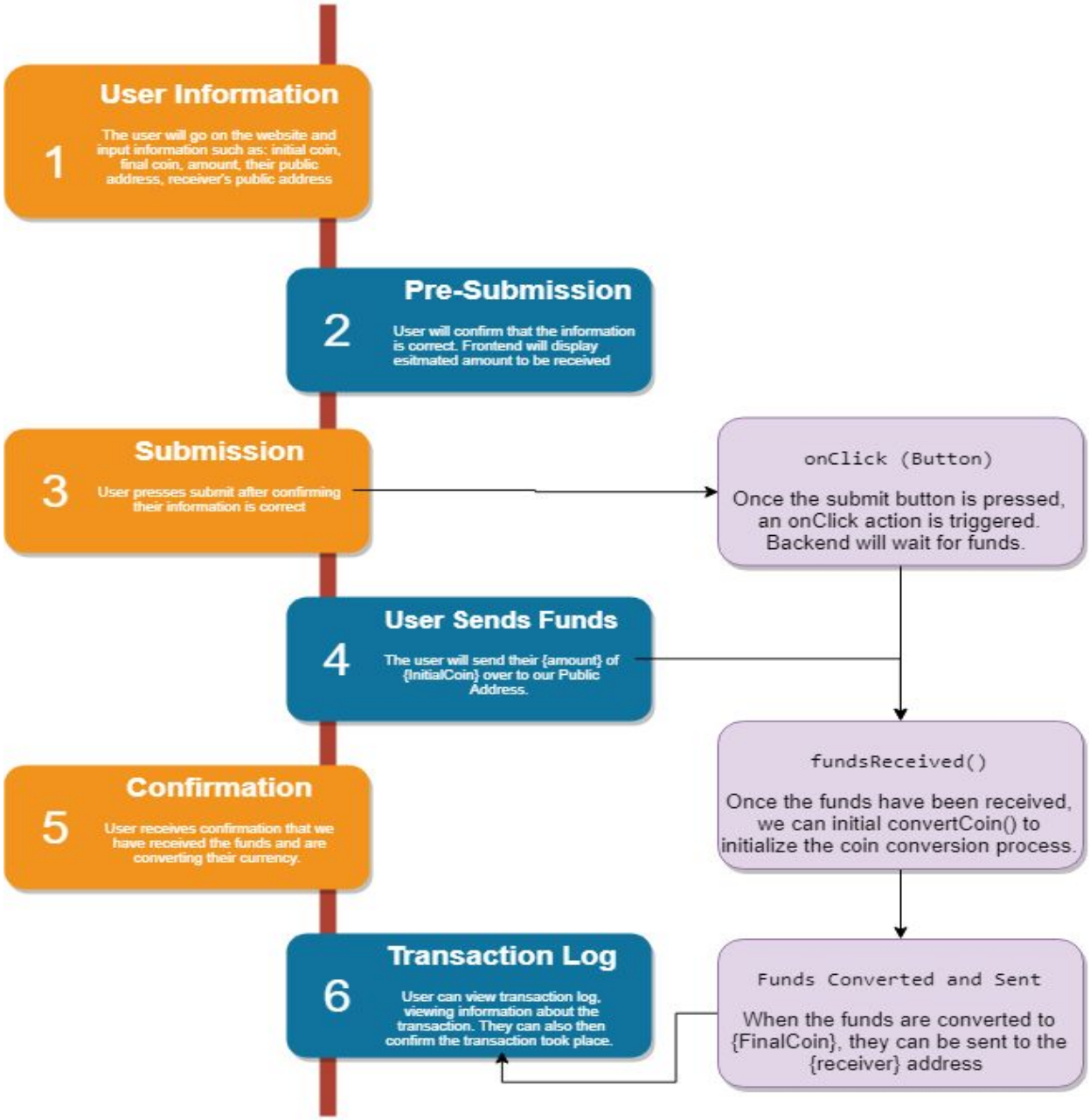
Submit

Please send your Inital Coin to 1F1tAaz5x1HUXrCNLbtMDqcw6o5GNn4xqX.

Your 10 Inital Coin will be transformed to approximately 5 Final Coin

Note that in this preliminary outline of the website,

- **Initial coin represents the coin that the user wants to send**
- **Final coin represents the coin that the receiver wants**



Build Instructions:
The repository can be cloned from https://github.com/rkooo567/ip-finance.

Dev Instructions:
If you wish to add on to the project, feel free to fork and develop from https://github.com/rkooo567/ip-finance.

Problem and Solution:
Currently, in the Blockchain space there exist many exchange websites. Ranging from Coinbase, Binance, Bittrex, StellarX, and the list goes on. However, often these websites have a tedious sign up process. In addition, for less experienced users, an exchange page can look daunting. From candlestick charts to constantly changing trade volumes to changing currency values to learning about fees, the whole process can be very daunting. However, if we were to abstract away that concept for the user, the whole experience will be much more seamless.

What our website aims to do is solve this issue of learning how to use an exchange and circumventing the process of creating an account. When a user visits our website, they are presented the ability to send money from on cryptocurrency to different cryptocurrency, without worrying about exchanging it. We will be providing that service for them.

However, with exchanges, there are some inherent issues. Once of the issues is security. In order to prevent our access to a user's private key, we will require the user to send their {InitialCoin} funds to our respective public address. Once those funds are received on our end, we will convert then to {FinalCoin} and send them over to the {receiver}. The other option was to have the user enter their private key on the website. However, we decided against this to help protect the user's security in this exchange.

Another problem we have to address is the constant varying cost of most cryptocurrencies. As we all know, cryptocurrency prices are the furthest from stable. To address this, we will provide a disclaimer to the user informing them that their final funds that are sent will be a product of the current price conversion of the coins minus the fees that are required.

Above all, our main selling point is convenience. We understand that there is inherent trust required in our platform to convert from one coin to another, but we will solve this by providing a complete transaction history showing the logs. In addition, it is important to understand that the funds may differ slightly based on market conditions.

Additional comments: