Content

Introduction

Pagacoin is the solution for the management and transfer of users' virtual currencies, called PagaCoins, used by administrators.

Functionalities

Creation of users and virtual wallets

Due to the fact that the creation of users and virtual wallets by users was outside the scope of the project, this possibility has been added for administrators, in order to test the performance of the rest of the functionalities.

To access, click on the “Users and wallets creation” button in the header (Figure 1).

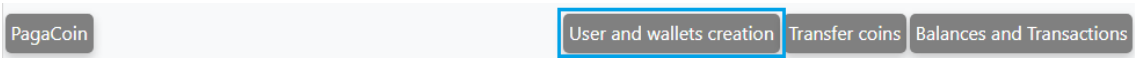


Figure 1

The first step is to set a user name and add virtual wallets (minimum one).

As shown in Figure 2 , the portfolios that are created appear in the list on the right.

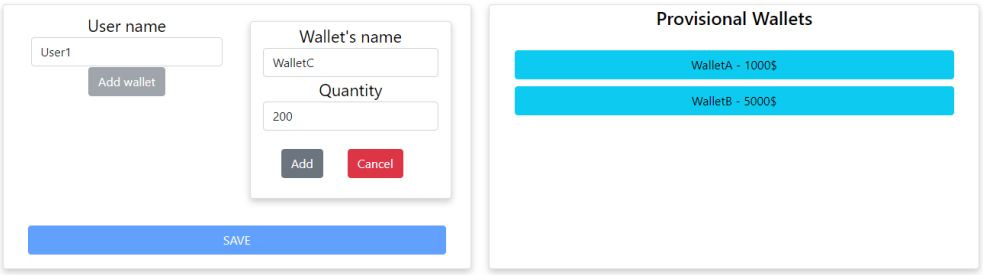


Figure 2

Once you have finished adding virtual wallets, press the “Save” button to add the user to the database.

Data persistence

Until the "Save" button is pressed, the user's filled data will be saved as long as the user remains in the application.

Transfer of Pagacoins between users and wallets

The administrator has the possibility to transfer PageCoins between users. To do so, click on the “Transfer coins” button in the header (Figure 3).

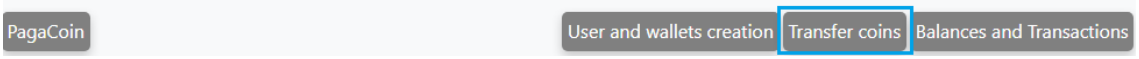


Figure 3

To make a transfer, we must go step by step (Figure 4):

- Step 1. Selection of the user who will send the PagaCoins.
- Step 2. Selection of the virtual wallet of the user chosen to send the PagaCoins
- Step 3. Selection of the user who will receive the PagaCoins
- Step 4. Selection of the virtual wallet of the chosen user to receive PagaCoins
- Step 5. Initiation of the transaction or cancellation of the transaction.

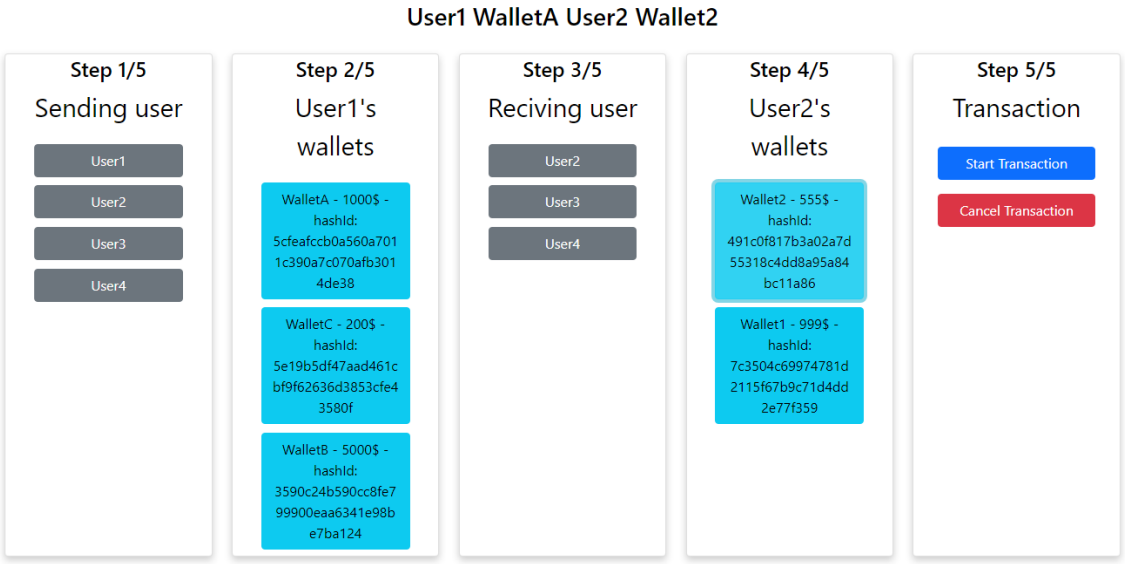


Figure 4

If the “Start transaction” button has been pressed, the following modal will appear to enter the amount to be sent. (Figure 5)

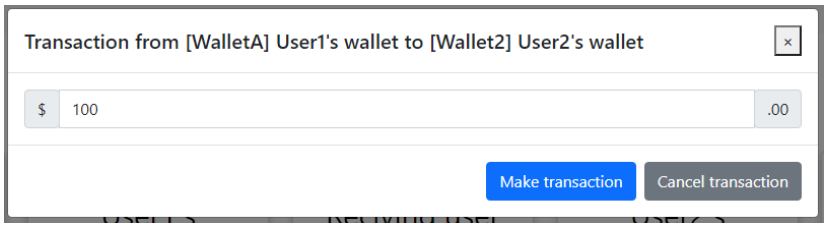


Figure 5

Data persistence

Until the “Make transaction” button in Figure 4 or the “Make transaction” button in Figure 5 is pressed, the selected users and virtual wallets shall be saved as long as the user remains in the application.

Visualisation of balances and transfers of PagaCoins between users

To view the transfers and balances, click on the "Balances and Transactions" button. (Figure 6)

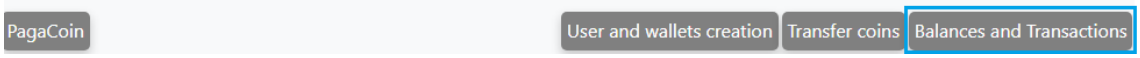


Figure 6

A list of users will appear with two buttons each; "Transactions by wallets" and "All Transactions", display of transfers filtered by user and virtual wallet or display of all transfers by users. (Figure 7)

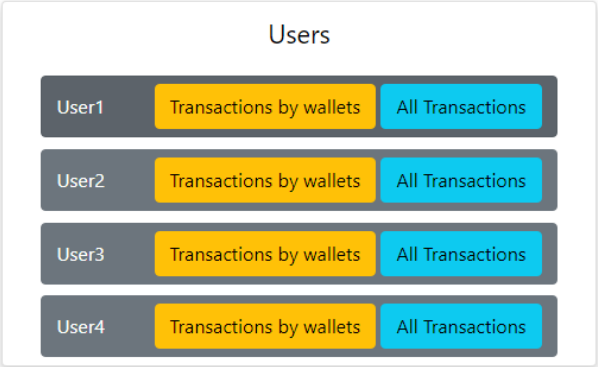


Figure 7

Transactions by user

The second list shows the user's sent (red) and received (green) transactions. (Figure 8)

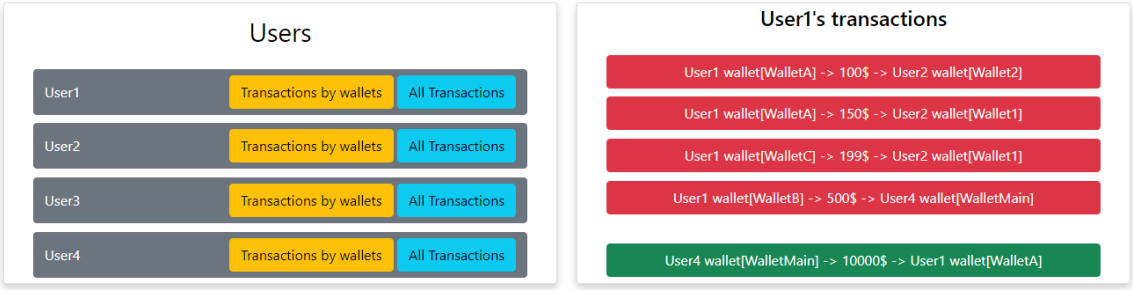


Figure 8

Transactions by user and virtual wallet

The second list shows the user's virtual wallets.

Once the virtual wallet is selected, the sent (red) and received (green) transactions of the user and wallet will be displayed. (Figure 9)

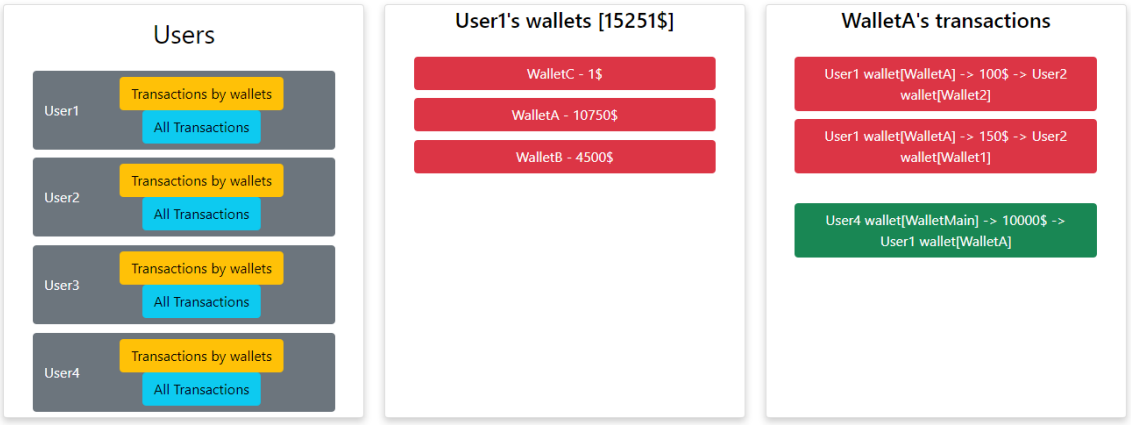
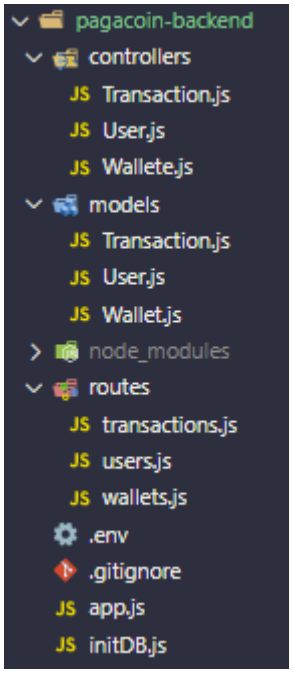


Figure 9

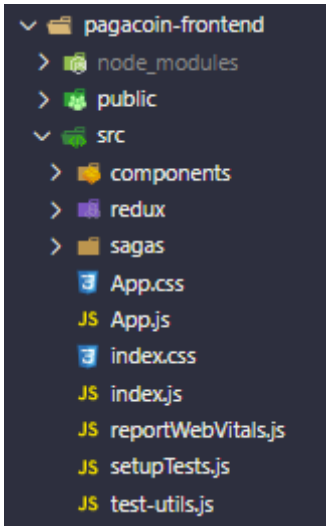
Arquitectura del proyecto

Backend (pagacoin-backend)



- Programming language: JavaScript.
- Database: MongoDB
- Server: express as Node.js web application
- Structure:
  - controllers: route implementations
  - models: scheams
  - routes: route interfaces

FrontEnd (pagacoin-frontend)



- Programming language: JavaScript
- Views library: React
- UI libraries: Bootstrap – reactstrap
- Style sheet language: Css
- States: Redux
- Test: Jest - @testing-library/react – jest-dom
- SideEffects management: Redux-Saga
- HTTP client: Axios

Ejecución

The easiest way to try this application is deploying both parts in local.

Frontend part:

Execute the following command on the command line, in the root path of the project

- npm install
- npm start

Once finished, we have running it in http://localhost:3000/

Backend part:

Execute the following command on the command line, in the root path of the project

- npm install
- npm start

Once finished, we have running it in http://localhost:5000/

Testing

Frontend part:

Execute the following command on the command line, in the root path of the project

- npm run test

Approach and solution

First of all, I had to separate the frontend and backend tasks and decide on the languages and technologies, being React and JavaScript the chosen ones.

The reason for not choosing Java as a backend programming language is due to limited time to refresh those skills (I currently work with NodeJs); so being more familiar with JavaScrip, I made that decision.

For the persistence in a database of wallets, transactions and user; MongoDb was chosen for its fast implementation and simplicity in simple CRUDs.

For frontend state management; I started with state in components, but added Redux and Redux-sagas for persistence and side effect management.

Bootstrap and Reactstrap are the UI libraries I decided to use because of their fast implementation and because, to a large extent, I work with them on a daily basis.

"I have enjoyed developing this solution with full decision making."