SpectroCoin QA task

Tautvilis Savickas

2022-10-13

1 Write test cases to test view / function given below:


Test cases:

- Check if selected input/output currency shows the right amount.

- Check if history button shows old conversions.

- Check if clicking input/output currency displays other available currencies.

- Check if input/output currency field only allows numbers and regional decimal separator.

- Check if input/output currency symbol is correct.

- Check if input/output field correctly displays long numbers.

- Check if input value is correctly converted into output value.

- Check if **BTC USE ALL** button inserts the correct value.

- If input field is left empty, exchange button should do nothing.

- If input field has data entered, the exchange button should subtract input currency from the wallet and add output currency to the wallet.
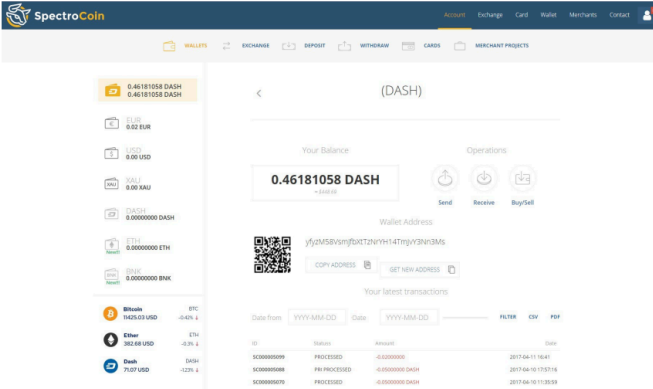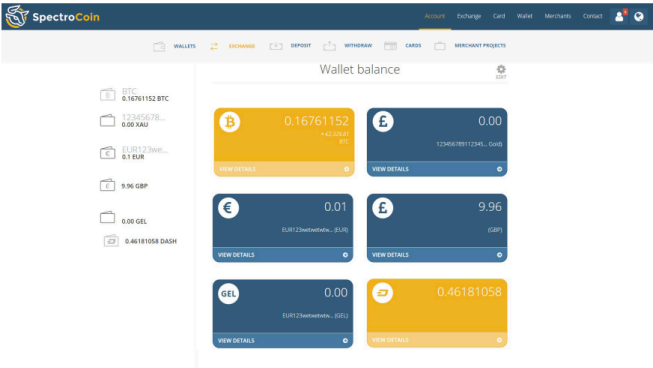
2   Find bugs in given UI examples
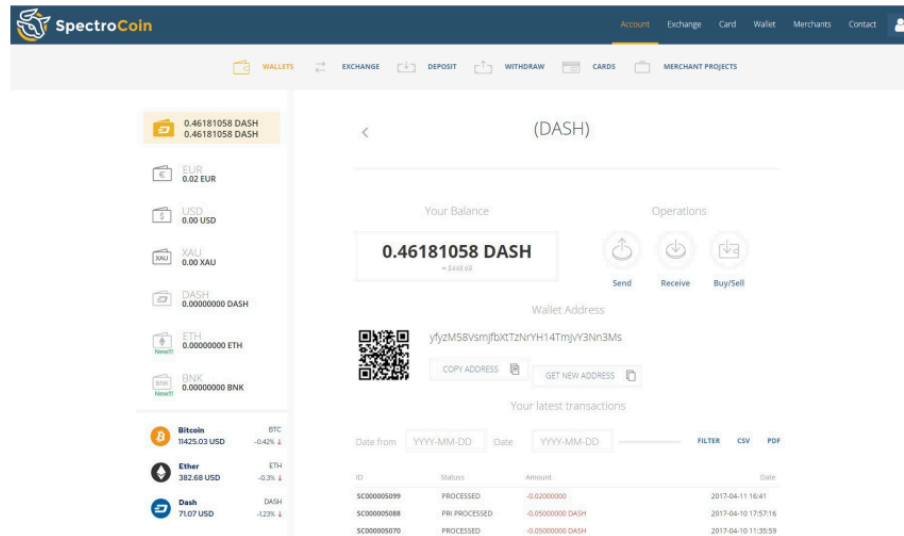
a)
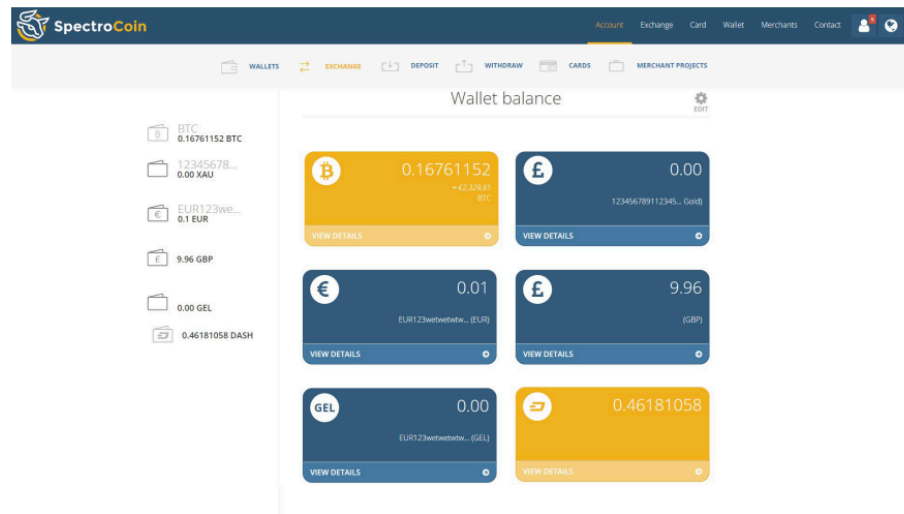
b)

2.1 Create bug reports for 2 bugs found in upper tasks

Name	Missing name on active currency in the sidebar
Description	Active currency name is missing in the sidebar, now it displays the currency twice, instead of name and currency.
Expected result	Currency name should be DASH
Steps to reproduce	Open **accounts** page and go to **wallet** tab
Image	

Name	Wrong info on Gold card in wallet balance
Description	Gold has GBP icon, wrong currency name.
Expected result	Card has Gold Icon and currency name is Gold
Steps to reproduce	Open **accounts** page and go to **exchange** tab
Image	

2.2 Create an API test for endpoint:

GET https://spectrocoin.com/scapi/ticker/usd/btc test using node.js. The code tests if the status code in 200 and currencies converted are USD and BTC.

```
const axios = require('axios')
var assert = require('assert')

axios.get('https://spectrocoin.com/scapi/ticker/usd/btc')
  .then(res => {
    assert(res.status == 200)
    assert(res.data.currencyFrom == 'USD')
    assert(res.data.currencyTo == 'BTC')
  })
  .catch(err => {
    console.log('Error: ', err.message)
  });
```

2.3 Create a test plan

Create a test plan for upcoming function described below: "As an user which has an active debit card, I am inactive on the web for 5 minutes, I want to be logged out from the system, so system should automatically log me out from the web"

1. **Objective** Test a new feature, which will log out an user from the system after 5 minutes if he is inactive and has an active debit card.

2. **Description** A new feature is being implemented which will, log out the user if he's inactive, its cookie should be terminated after 5 minutes and it should redirect the user to a public page. It's important that the cookie is terminated, because if not the system can be subject to cookie replay attack.

3. **Input**

   - Functions documentation
   - Testing environment

4. **Output**

   - All tests pass
   - All bugs are fixed
   - Test report

5. **Environments** All usable environments should be tested: mobile and web.

6. **Tests**

- Smoke testing
- Functional testing
- Regression testing
- Security testing

7. **Test examples**

- Create a test, which will wait for 5 minutes and checks if the elements are still visible. It should log out
- Create a test, which will create mouse events for 5 minutes. It shouldn't log out.
- Create a test, which will refresh the page every N time. It shouldn't log out.
- Create a test, which will wait for 5 minutes and after the user gets logged out, use the back button of the browser to check, to see the cached version and refresh the page. This will check if the cookie is terminated correctly, or refreshed.

2.4   End-To-End test

The end-to-end test is done in cypress and can be found in the git repository, under '/cypress/e2e/spectro.cy.js'