Project Documentation

BlogChain

Members and Contributors

Orjan Monsen	**Nathan Robertson**	**Seth Piercey**	**Seyd Razavi Lopez**
B00697153	B00684827	B00725633	B00751312
mons@dal.ca	NathanRobertson@dal.ca	piercey@dal.ca	sy349773@dal.ca

For Faisal Abbas

CSCI 4145 - Cloud Computing

Faculty of Computer Science

Dalhousie University

August 6th, 2018

Table of Contents

1 User Documentation

1.1 What does BlogChain do?

BlogChain is a hub for exchangers of all different types of cryptocurrencies. It allows users to see updates in the market, stay up-to-date on cryptocurrency news, and make informed decisions regarding the buying and selling of different coins.

As its name implies, BlogChain also supports blog posts and user interactions. Users may post blogs discussing significant events in the market such as significant spikes or dips in the value of a coin. Blogs are organized chronologically so the most relevant ones are seen first. Users may also comment on these blogs to ask questions or fill in any information that may have been missed in the original post.

Prices for different coins across multiple exchanges are provided. Users can see the lowest and highest recent prices for each cryptocurrency as well as the most recent selling price. The coin list is provided in an organized table that can be sorted by any of the categories.

BlogChain also provides access to an arbitrage page which shows price differences for specific coins so users may capitalize on the opportunity to make money. This feature has a view similar to the prices as it is organized in a table and is able to be sorted by any of the various categories.

1.2 How to Use BlogChain?

In order to use all the services, a new user must register and login. The register and the login options are displayed in the navigation bar (navbar). Once the user has registered and is logged in, the user can view their profile, the arbitrage service, the current market prices on the dashboard, and all blog posts on BlogChain.

The dashboard displays the prices and the providers for each cryptocurrency. There is also an educational tool in the same page that explains the dashboard. The user can access it by clicking on the “Learn More” button.

Similarly, by clicking on the the arbitrage option in the navbar, the user can see the arbitrage page where all arbitrage opportunities across cryptocurrency market exchanges are listed. This page also has an educational tool that explains the arbitrage table in form of a “Learn More” button.

The user profile can be accessed through the personalized profile button in the navbar which displays the user’s username. Here the user can enter information about themselves, update the username, or delete the account and all data associated with their user account.

Blogs can be accessed through the Blogs button in the navbar, where the user can see all blog posts, or create a new blog. If the user desires to read a full blog, the user must click on the “READ MORE” button and will be redirected to the full blog. The comments for this blog are displayed at the bottom of this blog view. The user can write a new comment on the blog post as long as the user is logged in to their account on BlogChain. Further, the user has the option to update and delete their own comments by clicking on the “Update” or “Delete” buttons next to the comment. If the blog belongs to the user, “Delete Blog” and “Update Blog” buttons are displayed at the bottom of the blog’s text. By clicking on either one of these, the user will delete or will be redirected to a new page where the user can update its own blog.

To create a new blog, the user must click on the “Create New Blog” button on the blogs page. Here the user is redirected to the new blog page where there are 3 fields: Title, Content and Image. The user can update these fields and then click on submit to create a new blog. If no image is selected, a defaulted image is used.

1.3 Web-Application URL

BlogChain is hosted on Google Cloud Platform at the following URL:
https://dal-cloud-s2018-fiery-caldron.appspot.com/

2 Design Documentation

2.1 Database Design

Initially, a relational and simplistic database design was created to act as a guide for future development: by looking at the diagram, developers would have a basic idea of the kinds of functions which would needed in BlogChain. This design featured a parent “User” table which had “Session”, “Comments”, and “Blog” child tables (Figure 1a).

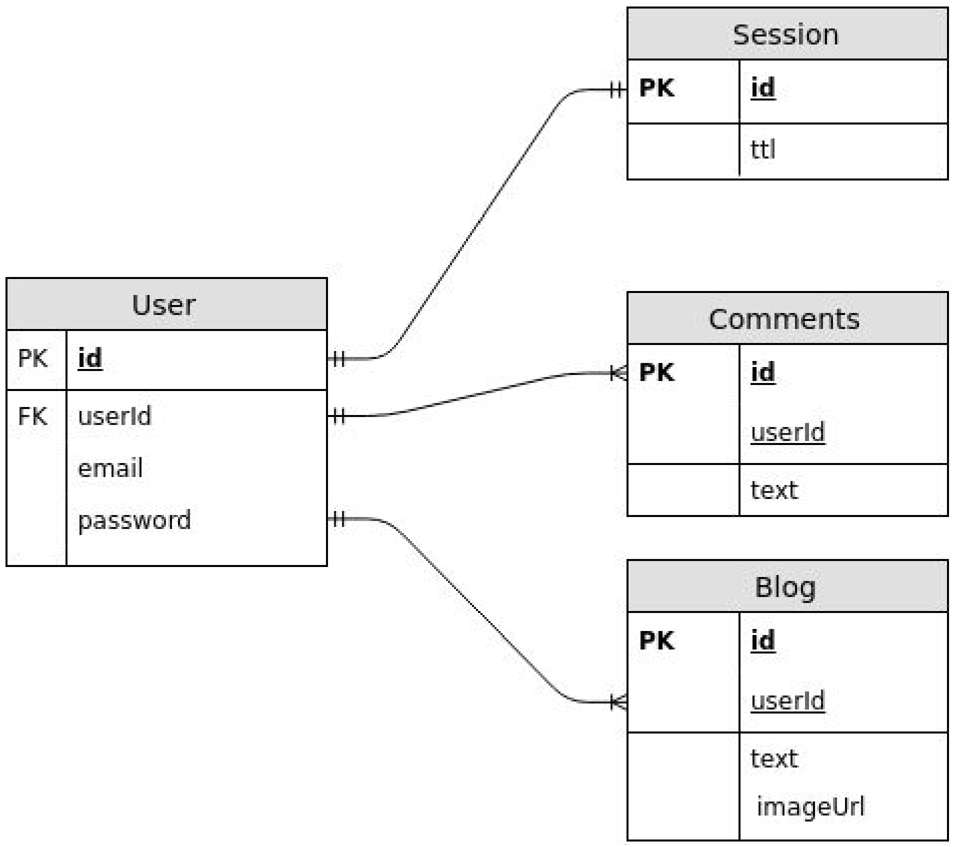


Figure 1a. Original entity-relationship diagram

This relational database model was eventually abandoned for a non relational, JSON based database, due to its synergy with our chosen server-side technology. This change afforded us increased data availability, system usability, and development speed. Primary tables in this model are the “User”, “Blog”, and “Arbitrage” tables, each of which contain BlogChain data corresponding to their names. Note that, due to being non relational, these tables are generally independent from one another, unlike in the original design. Figure 1b is a relational representation of the final database, but this is simply for ease of understanding; child tables in this instance only represent extended JSON values within each table.

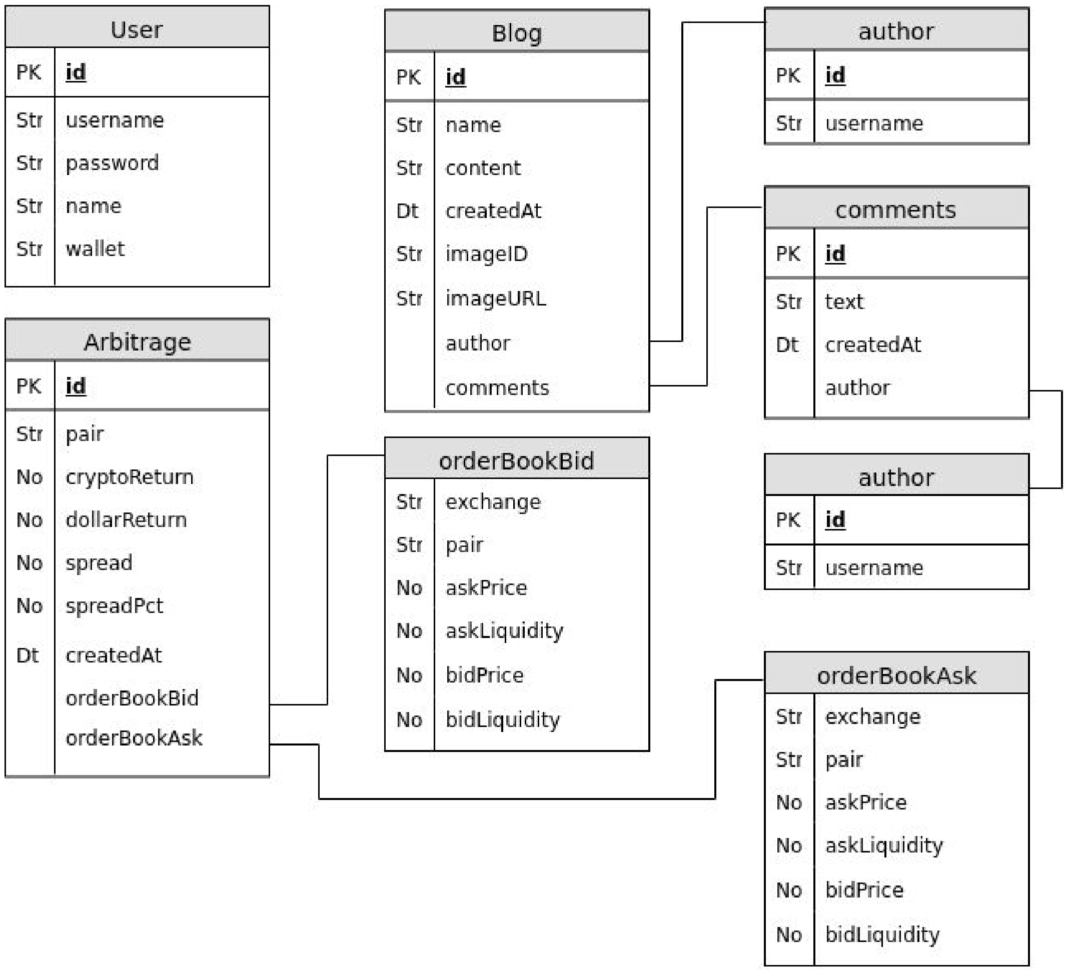


Figure 1b. Final table represented relationally. In reality, it is non relational and JSON based.

2.2 Architecture Design

BlogChain is hosted on Google Cloud Platform's App Engine solution. The container is setup to have 1 CPU, 6 GBs of memory, and 10GB disk space. It also utilizes the Dynamic Scaling cloud pattern where the minimum number of instances is 1, and the maximum number of instances is 10 at this time. It is configured to scale up once the CPU usage spikes above 70% of utilization. The server instances communicates directly with the CryptoWatch API to gather data about cryptocurrency markets. The server instances also communicates with a MongoDB hosted on Google Cloud Platform using Atlas. The MongoDB contains 3 clusters and uses a load balancer internally to balance requests amongst the clusters (see Figure 2).

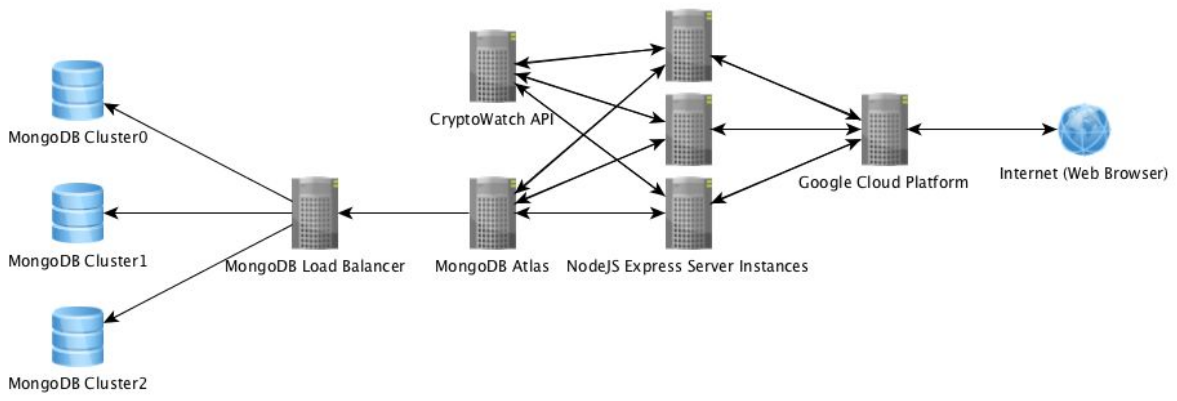


Figure 2 - BlogChain Architectural Overview

2.3 API Descriptions

BlogChain does not provide an API solution, therefore there are no API descriptions of an API developed for this project. However, BlogChain heavily relies on data pulled from the Cryptowatch API, which is described in section 2.4.8.

2.4 Description of Frameworks and Dependencies

2.4.1 NodeJS

NodeJS is a cross-platform run-time environment solely in JavaScript which executes JavaScript code outside of the browser, allowing us to run server-side scripts to produce dynamic web-page content.

URL: https://nodejs.org/en/

2.4.2 MongoDB

MongoDB is a cross-platform, NoSQL, document-oriented database which stores data in JSON format. MongoDB was a suitable database choice since we were already using JavaScript and NodeJS for developing BlogChain, which makes storing and retrieving JSON much more seamless and streamlined. MongoDB also provides high availability by using replica sets consisting of 2+ copies of the data.

URL: https://www.mongodb.com/

2.4.3 Mongoose

We used Mongoose on the NodeJS server to elegantly communicate with the MongoDB instances. Mongoose is an object modeling library for NodeJS and MongoDB which wraps validation, casting, and business logic into simple function calls, taking care of all the complicated things for us behind the scenes. It allowed us to write schema-based solutions to properly model the data we were fetching and storing in the database.

URL: http://mongoosejs.com/

2.4.4 Google Cloud Platform (GCP)

Our decision to use GCP is mostly based on the fact that we received a free project quota from the professor. This allowed us to host the web application for free on the cloud, and from there we also added dynamic scaling to auto-scale BlogChain up/down depending on the demand, and liveness checks to automatically restart unhealthy instances allowing for higher availability.

URL: http://cloud.google.com/

2.4.5 Cloudinary

Cloudinary is a toolkit that provides a cloud-based image and video management services. It allows to upload, store manage and manipulate images and videos. We used Cloudinary to allow users to store and manipulate images for their blogs. Cloudinary allows us to quickly retrieve the image URL, created by Cloudinary, so it can be displayed as a HTML element in our application.

URL: https://cloudinary.com

2.4.6 Passport

Passport is a NodeJS library which provides authentication middleware on the NodeJS platform. We used Passport to authenticate users and persist sessions throughout the application routes. We also used Passport to hash and salt passwords inputted by users upon user registration. This allowed us to quickly iterate on the development of BlogChain while not spending too much time implementing user authentication. Passport also allows authentication using social media such as Facebook, Google, Twitter and Linkedin which further allows the scalability of our application.

URL: http://www.passportjs.org/

2.4.7 Request

Request is an NPM package library which helps NodeJS developers send HTTP requests. We used the Request library to communicate with the CryptoWatch API as well as the Cloudinary API.

URL: https://www.npmjs.com/package/request

2.4.8 CryptoWatch

CryptoWatch is a REST API which provides basic information about all cryptocurrency markets on their platform. BlogChain uses the CryptoWatch API to collect information about specific markets and specific coins, as well as analyse markets and coins to find arbitrage opportunities.

The Cryptowatch API is rate limited by CPU allowance where each client is allowed to use 8 seconds of CPU time per hour. This drastically limited our abilities to generate real-time arbitrage analysis, and therefore we ended up searching for arbitrages between all the markets and coin pairs on Cryptowatch once every hour.

URL: https://cryptowat.ch/docs/api

2.4.9 MLab

We used MLab to host a MongoDB instance for the development environment. MLab gave us up to 512MB storage for free, but it has limited redundancy and resiliency features and is hence not suitable for production. MLab allowed us to rapidly start the design of our database during development without worrying about translating the application schemas during production deployment and avoiding technological incompatibility as our deployment database cluster (Atlas) can host MongoDB instances.

URL: www.mlab.com

2.4.10 Atlas

We used the Atlas DBaaS to host the production database cluster on GCP. Similarly to MLab, Atlas also gave us 512MB of free storage. However, Atlas provisions 3 clusters and contains DB sharding functionality (horizontal scaling), resulting in a database that's highly resilient and ready for production usage.

URL: www.mongodb.com/cloud/atlas

2.4.11 MaterializeCSS

We used the MaterializeCSS library to design our web application in a modern and simplistic style. MaterializeCSS provides responsive web-page styling, and their design is based off of Google's simplistic design. This library allowed us to easily design html code into cards, tables, navigation bars, and buttons in a consistent manner.

URL: https://materializecss.com/

2.4.12 Embedded JavaScript Templating (EJS)

EJS was used to display the dynamic content retrieved from the NodeJS server on the web pages. For example, querying the database and retrieving all blogs will not always have the same number of blogs in the response, and therefore we used EJS to repeat html code using JavaScript loops in the view. We also used EJS to divide our views into partials, allowing for cleaner code as well as code reuse.

URL: http://ejs.co/

2.4.13 JQuery

jQuery is a portable and small JavaScript library. It allowed us to manipulate HTML elements very easily so we could create a very dynamic user interface. jQuery is supported by many browsers which allows a much more dynamic product capable of working in any machine. Furthermore, other libraries that were used in this project such as: Materialize, DataTables, Cloudinary and Node.js support jQuery operations which allowed for a faster integration.

URL: http://jquery.com

2.4.14 DataTables

DataTables is a .net plug-in for the jQuery JavaScript library. It allowed us to display data in a dynamic table. The main features of DataTables are: Pagination, instant Search based on columns and column ordering. These features were used to display the data retrieved from the CryptoWatch API in the dashboard and the arbitrage page.

URL: https://datatables.net

2.5 How to Setup the Infrastructure

2.5.1 Project Dependencies

The project depends on the following tools:

- node
- npm

The developer must install these tools on their local machine, for example by using HomeBrew commands such as “brew install node”.

2.5.2 Local Environment Setup

The following steps indicate how to setup the local development environment and run the server locally:

1. Open a terminal and change directory into the directory where BlogChain is stored
2. Run the command “npm install” to install all dependencies
3. Run the command “npm start” to start the server
4. Open localhost:8080 (HTTP server) or localhost:8081 (HTTPS server) in your favourite browser to access the web application.

2.6 Web-Application Design

It was decided early in the conception phase that our intended audience would be both cryptocurrency pros and amateur enthusiasts. Hence, it was imperative that the application be designed simply enough to be welcoming to new users, but complex enough to provide sufficient information to seasoned veterans. The application was designed from the start with this in mind.

2.6.1 Welcome Page and User functions

BlogChain functions which explicitly required user input, like registration, login, and user profile (Figures 4-6) were designed to be simple and similar to what one would expect from other web applications with these functions. In the case of user profile, the design was much more complex than the end product, and represents the future of the user profile page. The only exception to this simplicity is the welcome page (Figure 3), which has a design some users may find unconventional. This however was ultimately left in the final product as we felt it contributed to a more welcoming platform.

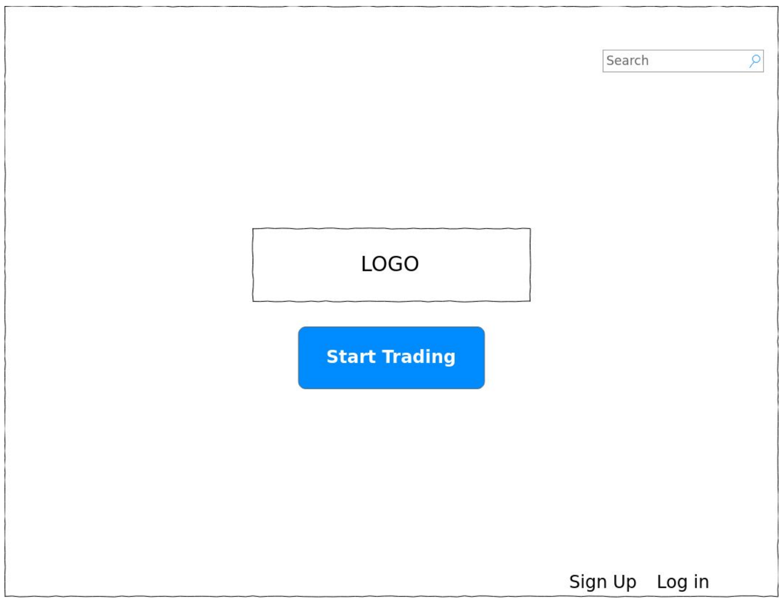


Figure 3 - A wireframe of the welcome page.

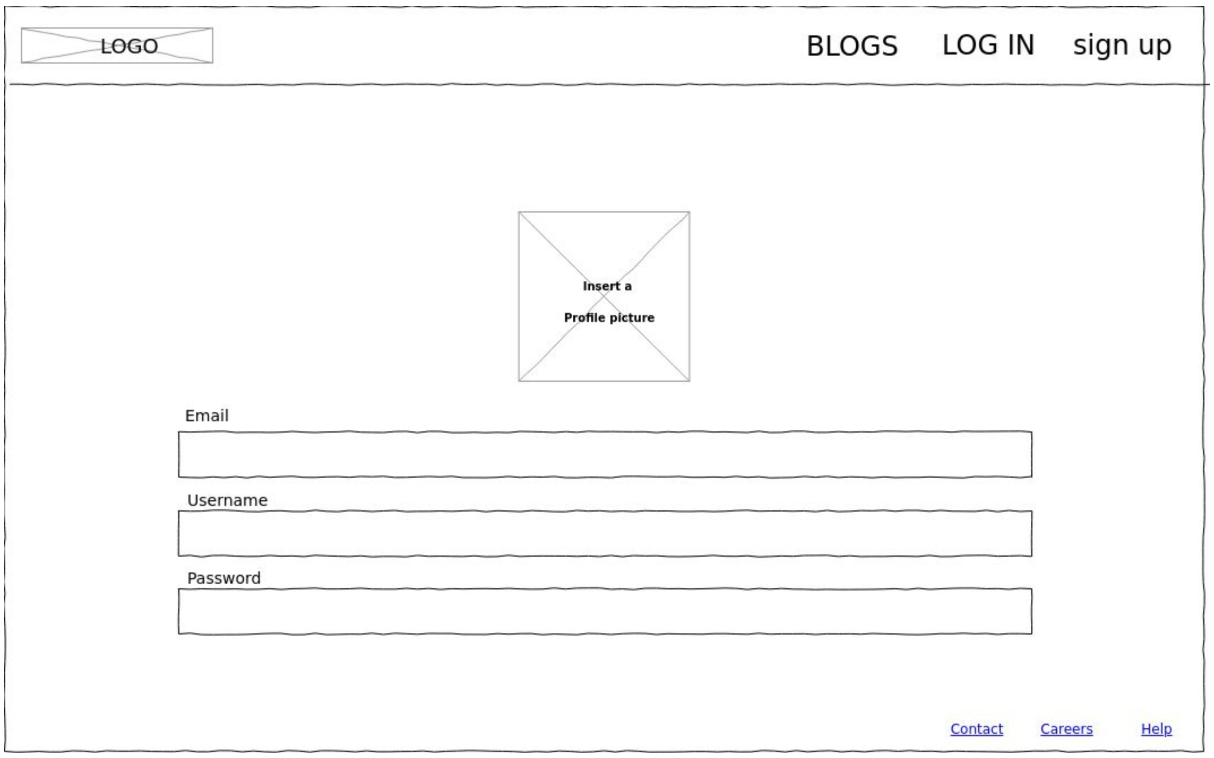


Figure 4 - A wireframe of the user registration page.

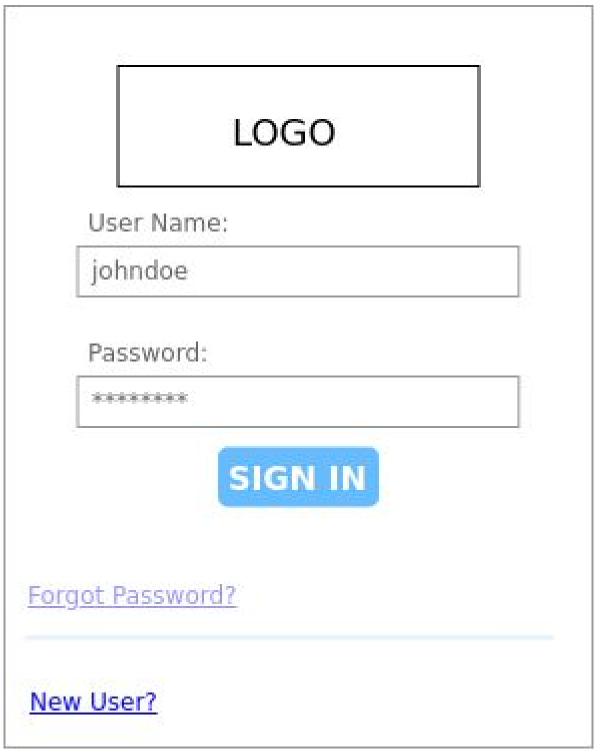
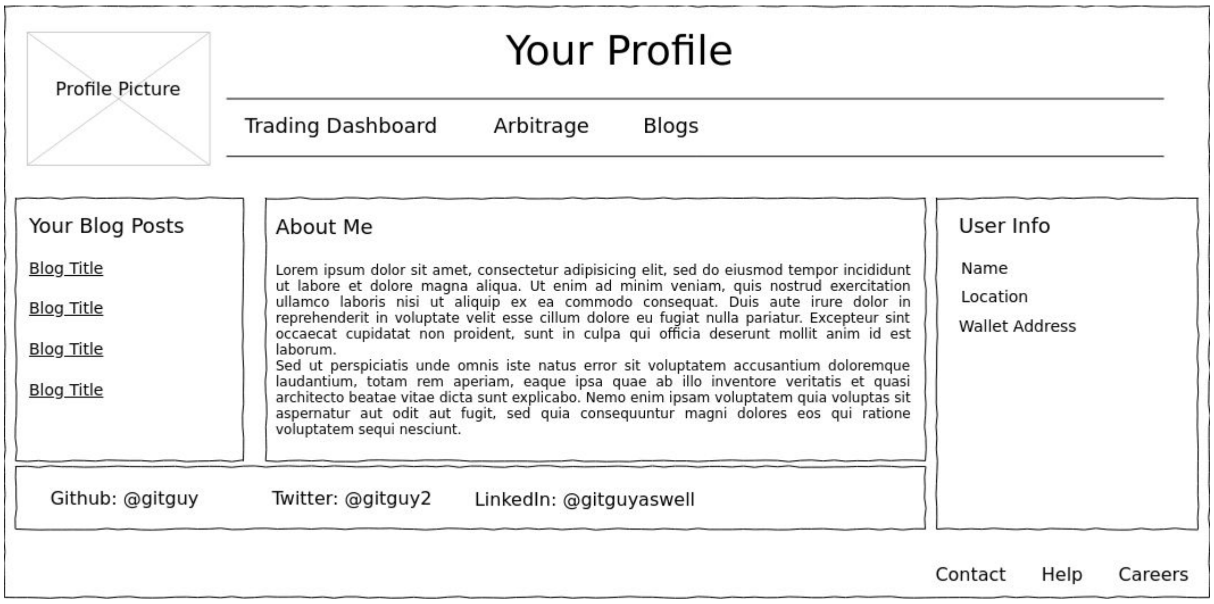


Figure 5 - A wireframe of the login page.

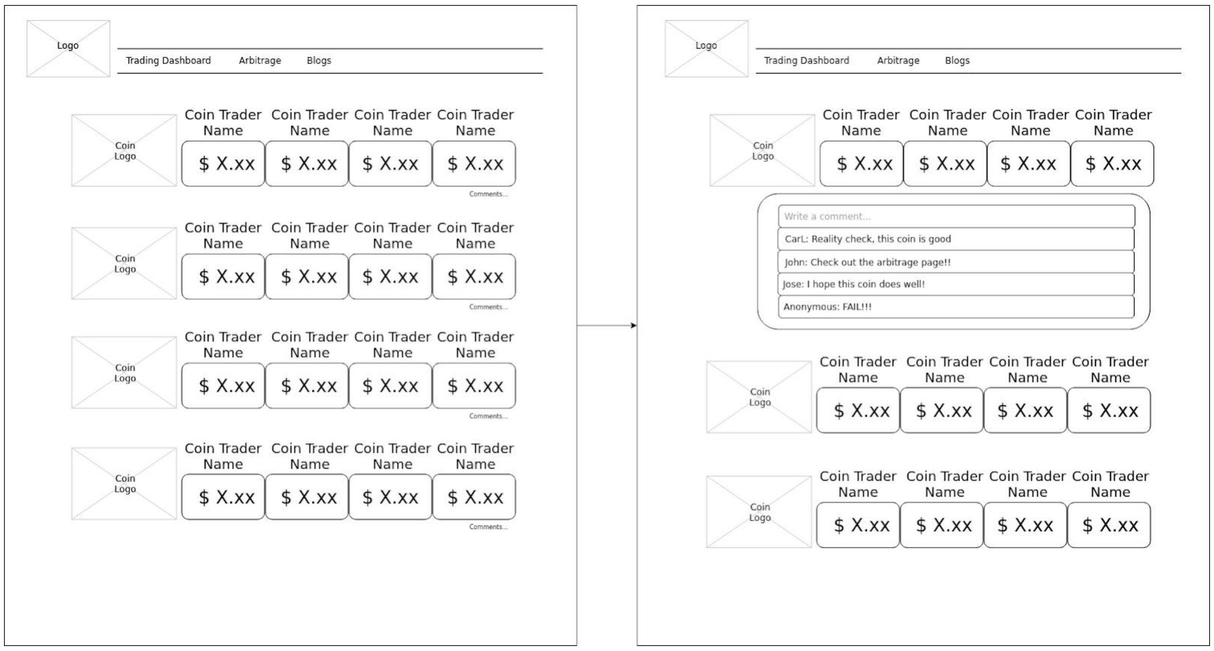


Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt.

Figure 6 - A wireframe of the user profile page.

2.6.2 Dashboard

Initially, the dashboard was designed to allow users to comment below real-time exchange prices, which would allow for extended user discussion and engagement. This design was dropped due to changing priorities, but could be implemented in the future.



CarL: Reality check, this coin is good
John: Check out the arbitrage page!!
Jose: I hope this coin does well!
Anonymous: FAIL!!!

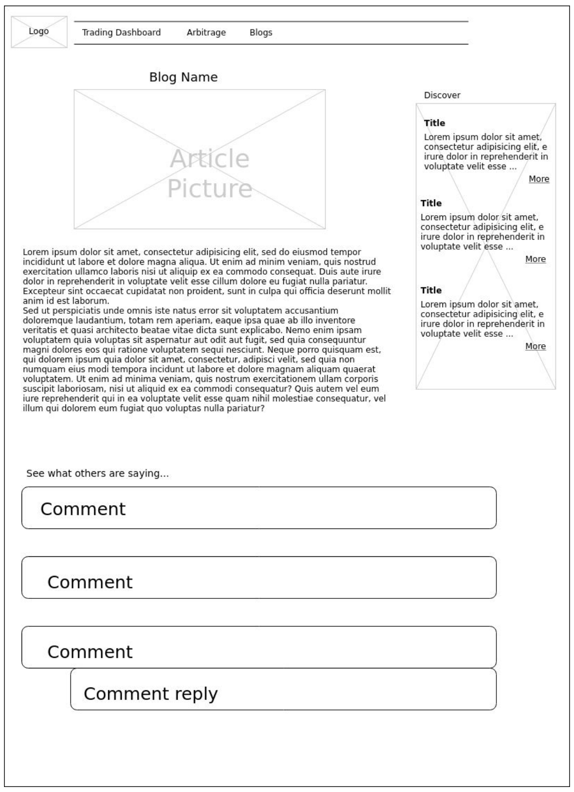
Figure 7 - A wireframe of the dashboard page.

2.6.3 Blogs

Like the user functions, blogging pages were designed to be similar to what one would find on other websites (Figures 8 and 9).

2.6.4 Arbitrage

Representing arbitrage information succinctly was the biggest challenge during the design phase; it was difficult to represent the information in such a way so as to avoid cramming too many numerical values in a small space. Thus, the arbitrage page was designed using percentages, representing user profit from the resulting transaction, and using a card-like interface. However, this was ultimately dropped for a more compact arbitrage page (Figure 10).


Logo
Trading Dashboard
Arbitrage
Blogs
Blog Name
Article Picture
Discover
Title
Lorem ipsum dolor sit amet, consectetur adipisicing elit, e irure dolor in reprehenderit in voluptate velit esse ...
More
Title
Lorem ipsum dolor sit amet, consectetur adipisicing elit, e irure dolor in reprehenderit in voluptate velit esse ...
More
Title
Lorem ipsum dolor sit amet, consectetur adipisicing elit, e irure dolor in reprehenderit in voluptate velit esse ...
More
See what others are saying...
Comment
Comment
Comment
Comment reply

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Figure 8 - A wireframe of the blog page.

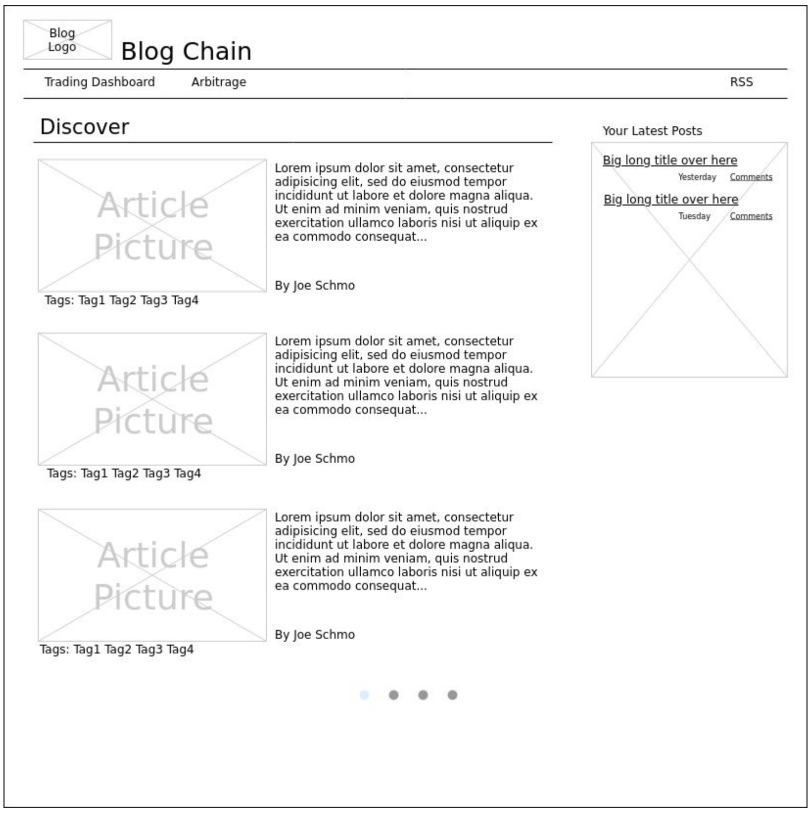

Blog Logo
Blog Chain
Trading Dashboard
Arbitrage
RSS
Discover
Article Picture
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat...
By Joe Schmo
Tags: Tag1 Tag2 Tag3 Tag4
Your Latest Posts
Big long title over here
Yesterday
Comments
Big long title over here
Tuesday
Comments

Figure 9 - A wireframe of the Blog Navigation page.

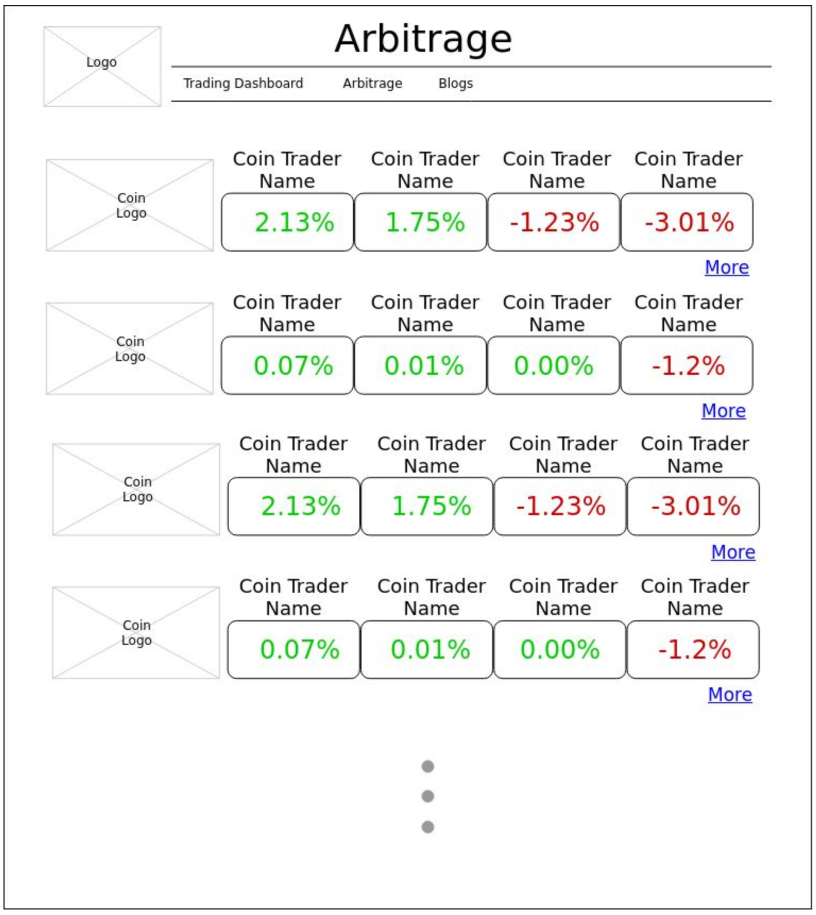


Figure 10 - A wireframe of the arbitrage page.

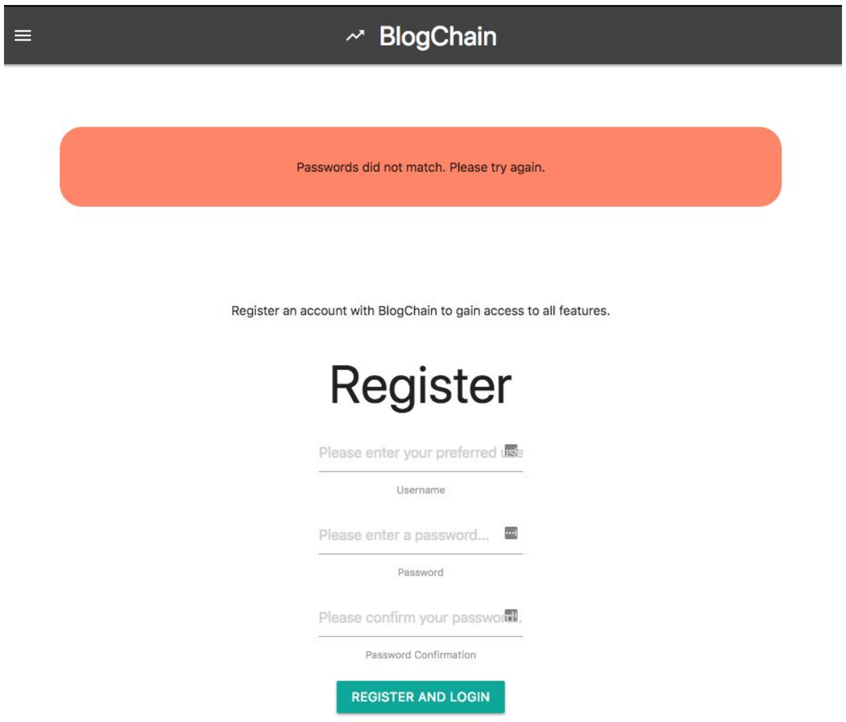
3 Software Testing

This section shows the software tests performed on the production environment on August 6th, 2018.

3.1 User Registration

3.1.1 Non-Matching Passwords

Expected output: A red flash message indicating that the passwords are not matching.
Actual output: as expected.



Passwords did not match. Please try again.

Register an account with BlogChain to gain access to all features.

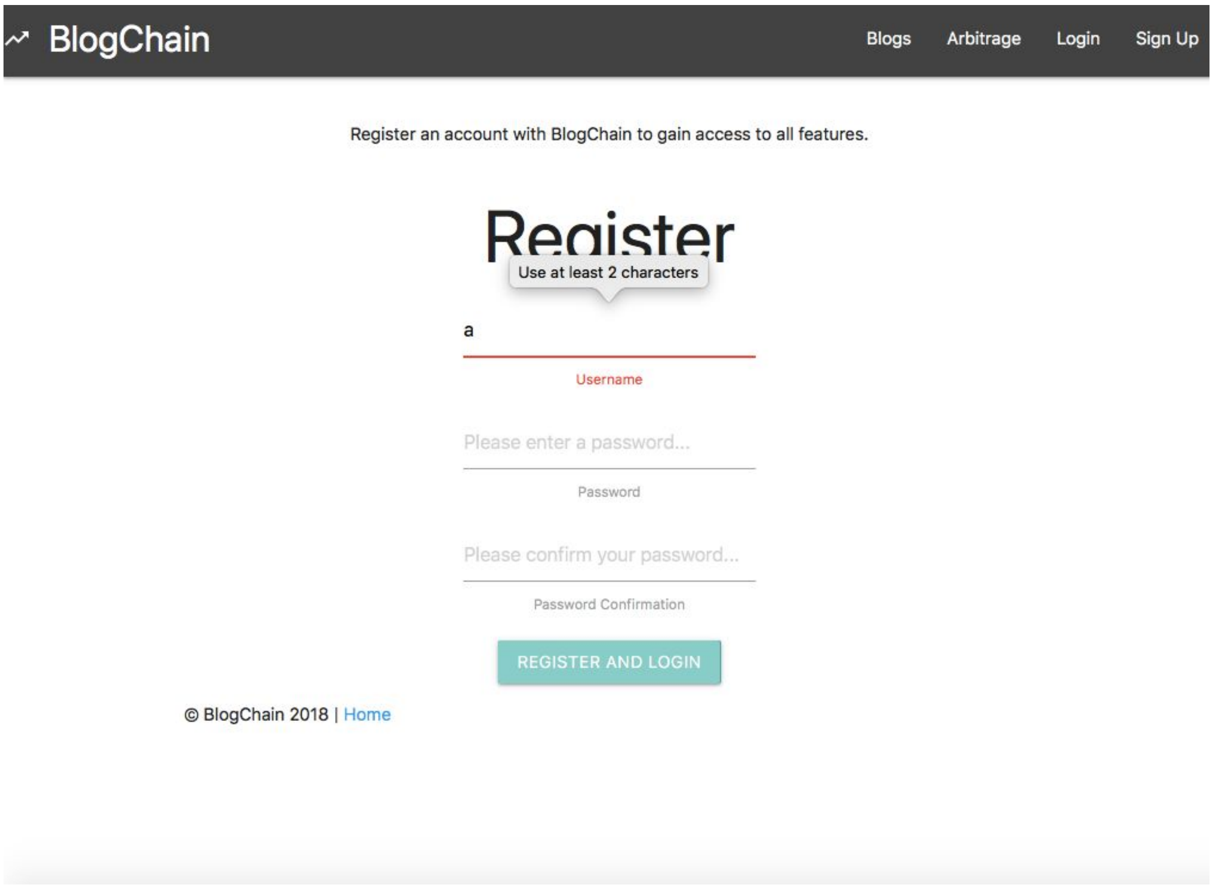
Register

Please enter your preferred use
Username
Please enter a password...
Password
Please confirm your password.
Password Confirmation
REGISTER AND LOGIN

Figure 11 - Test results of the non-matching passwords test.

3.1.2 Username or Password is Too Short

Expected output: A browser message indication the the minimum requirements for successful registration.
Actual output: as expected.



Register an account with BlogChain to gain access to all features.

Use at least 2 characters

a

Username

Please enter a password...

Password

Please confirm your password...

Password Confirmation

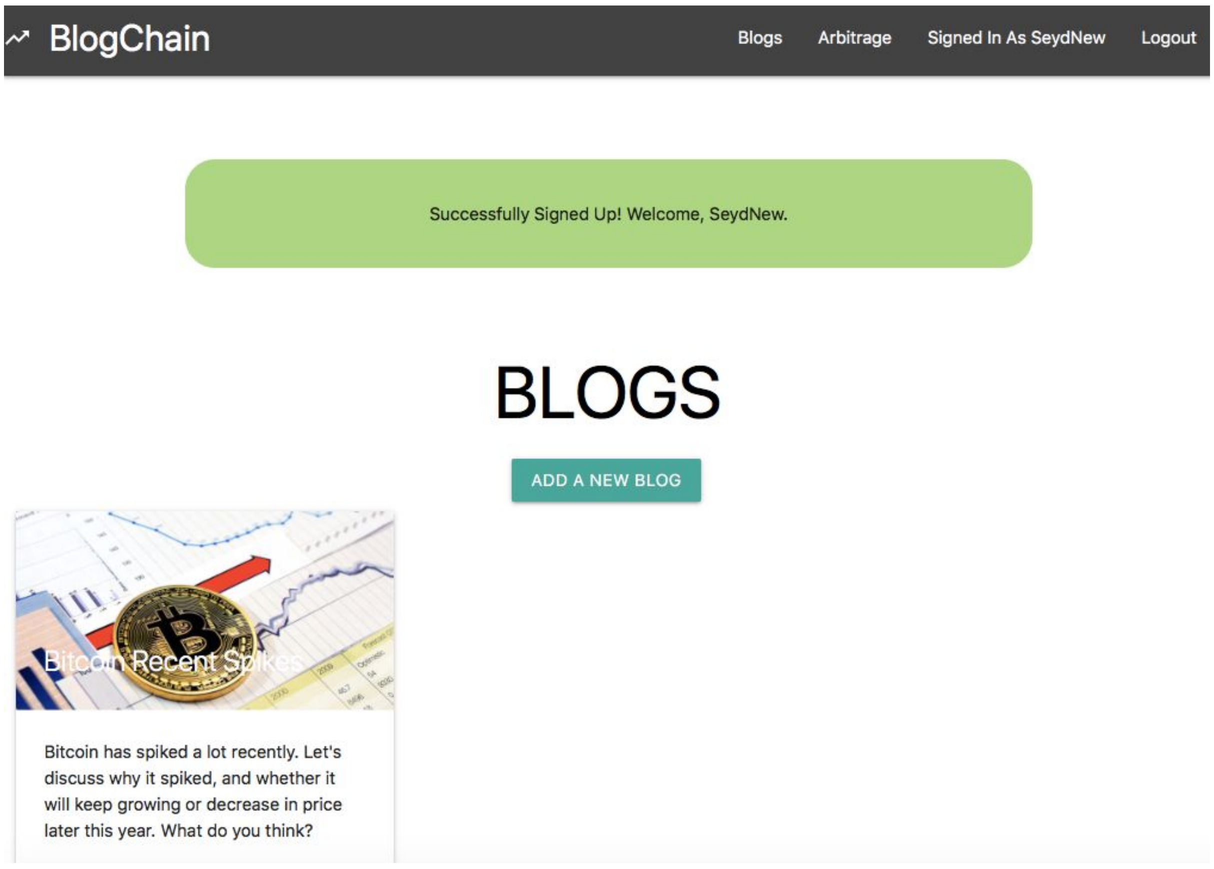
REGISTER AND LOGIN

© BlogChain 2018 | Home

Figure 12 - Test results of inputting a short username

3.1.3 Successful Registration

Expected output: A green flash message indicating that registration has been successful and a personalized message welcoming the user to BlockChain. The website redirects the user to the blogs page.
Actual output: as expected.



Successfully Signed Up! Welcome, SeydNew.

BLOGS

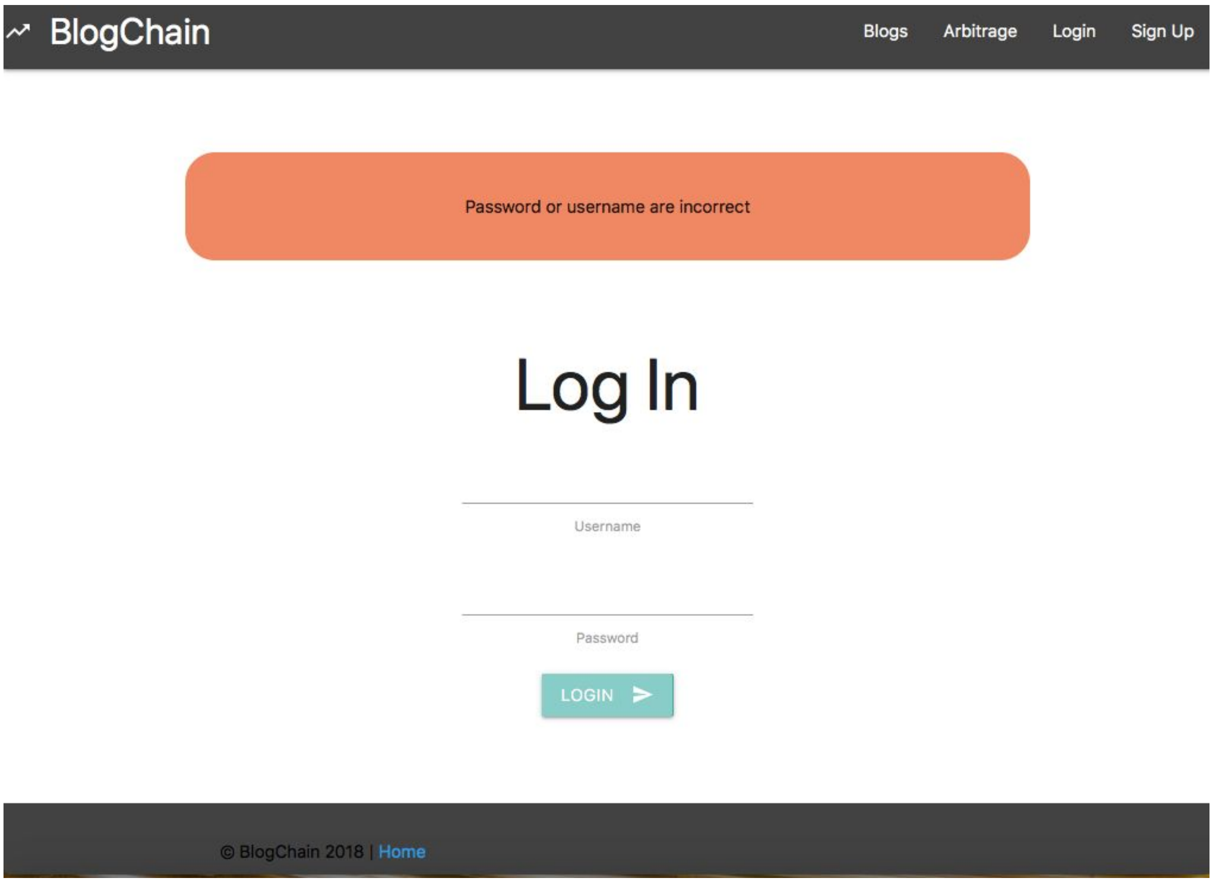
Bitcoin has spiked a lot recently. Let's discuss why it spiked, and whether it will keep growing or decrease in price later this year. What do you think?

Figure 13 - Test results from a successful user registration.

3.2 Login / Authentication

3.2.1 Login to a Non-Existing Account or with the Wrong Password

Expected output: A red flash message indicating that the password or username is incorrect.
Actual output: as expected.

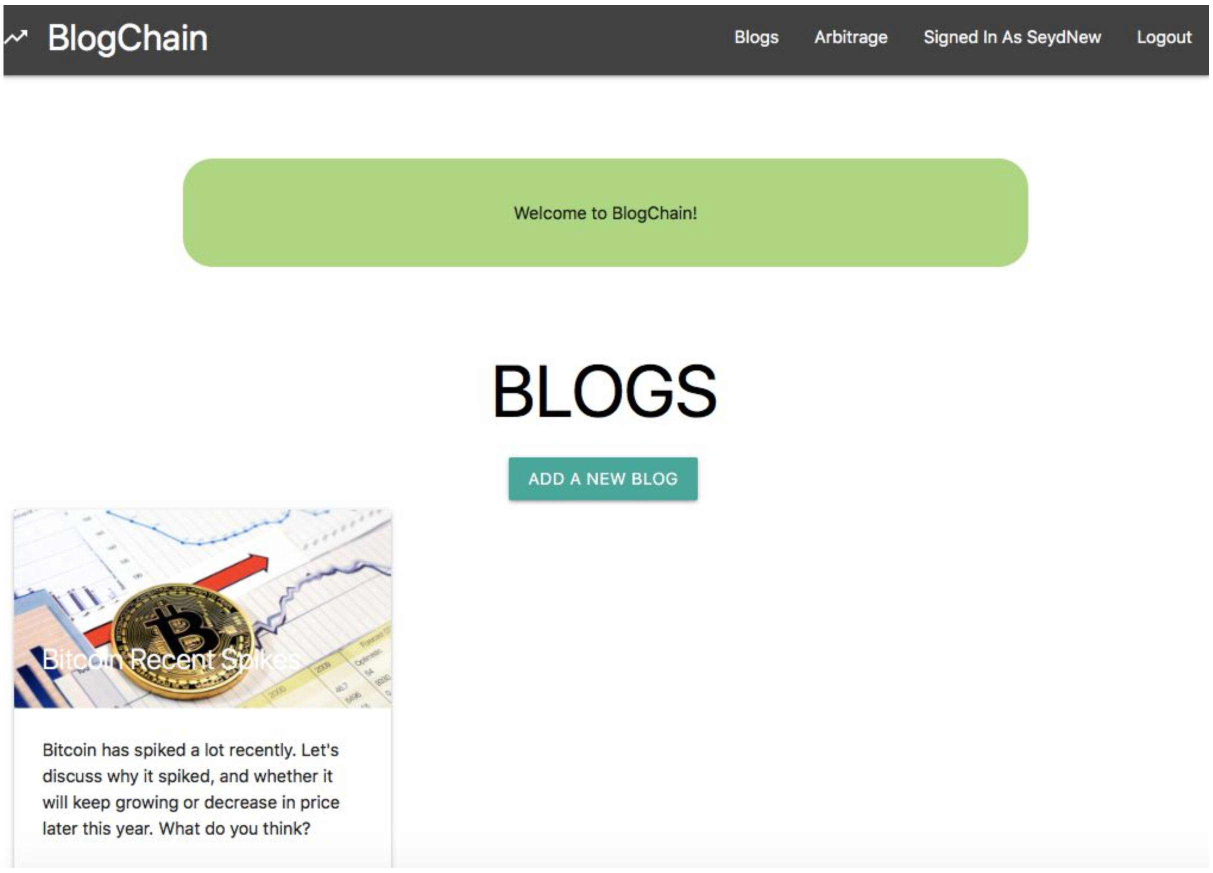


Password or username are incorrect

Figure 14 - Test results of logging in with the wrong credentials.

3.2.2 Successful Login

Expected output: A green flash message welcoming you to BlogChain. The user gets redirected to the blog page.
Actual output: as expected.



Welcome to BlogChain!

BLOGS

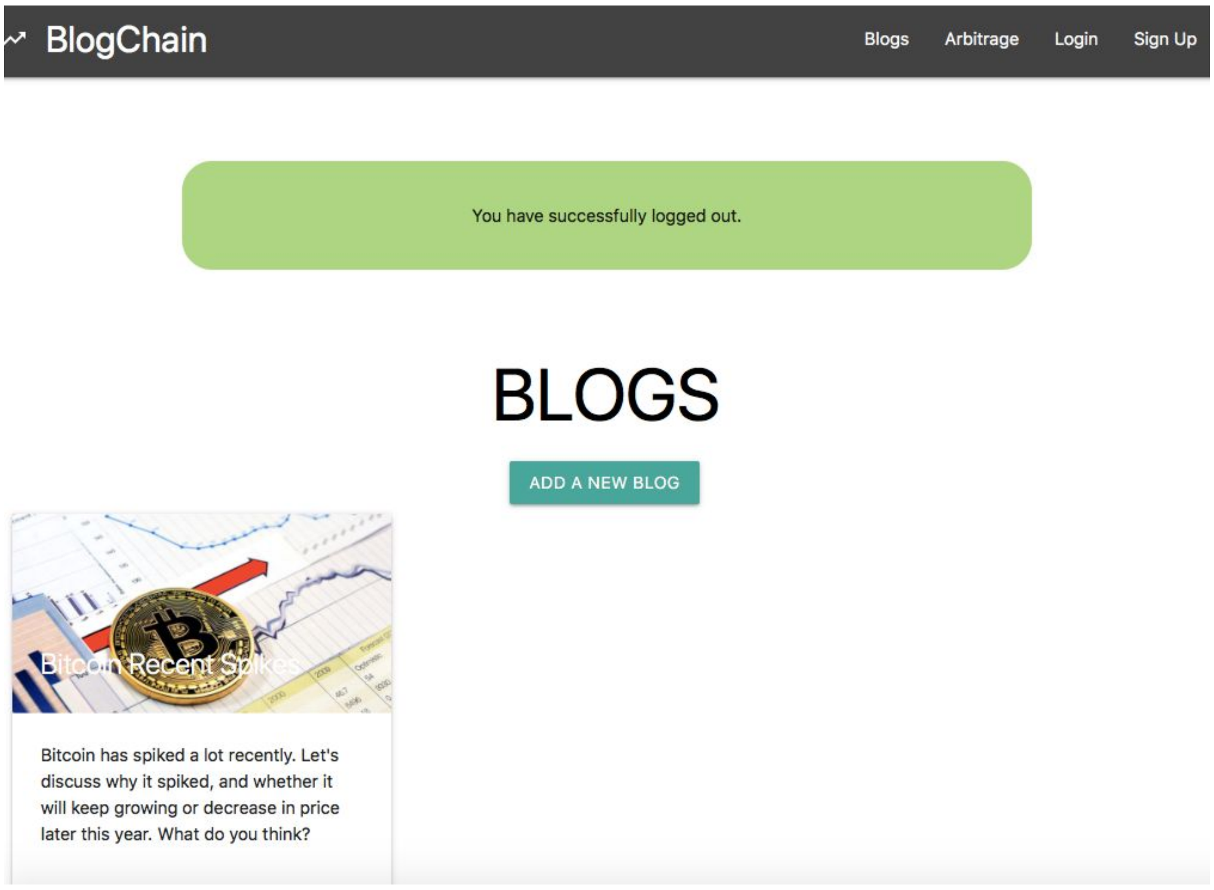
Bitcoin Recent Spikes

Bitcoin has spiked a lot recently. Let's discuss why it spiked, and whether it will keep growing or decrease in price later this year. What do you think?

Figure 15 - Test results of a successful login attempt.

3.2.3 Logout

Expected output: A green flash message indicating that the user has logged out. The user gets redirected to the blog page.
Actual output: as expected.



You have successfully logged out.

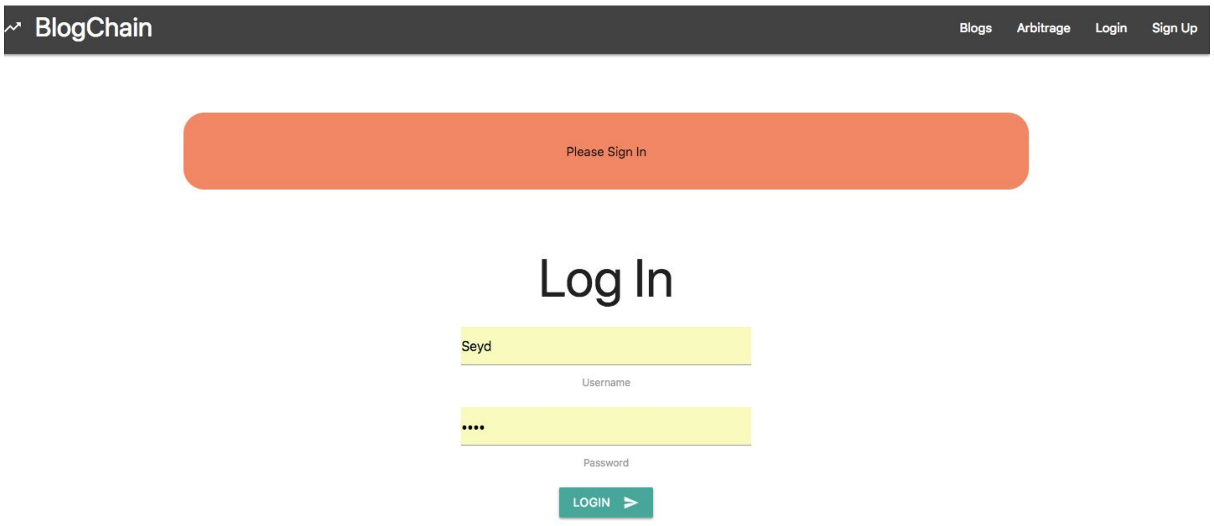
BLOGS

Bitcoin has spiked a lot recently. Let's discuss why it spiked, and whether it will keep growing or decrease in price later this year. What do you think?

Figure 16 - Test results of a successful logout attempt.

3.2.4 Accessing Member-Only Features Without Authentication

Expected output: A red flash message indicating that the user is required to sign in. The user gets redirected to the login page.
Actual output: as expected.



Please Sign In

Log In

Username: Seyd

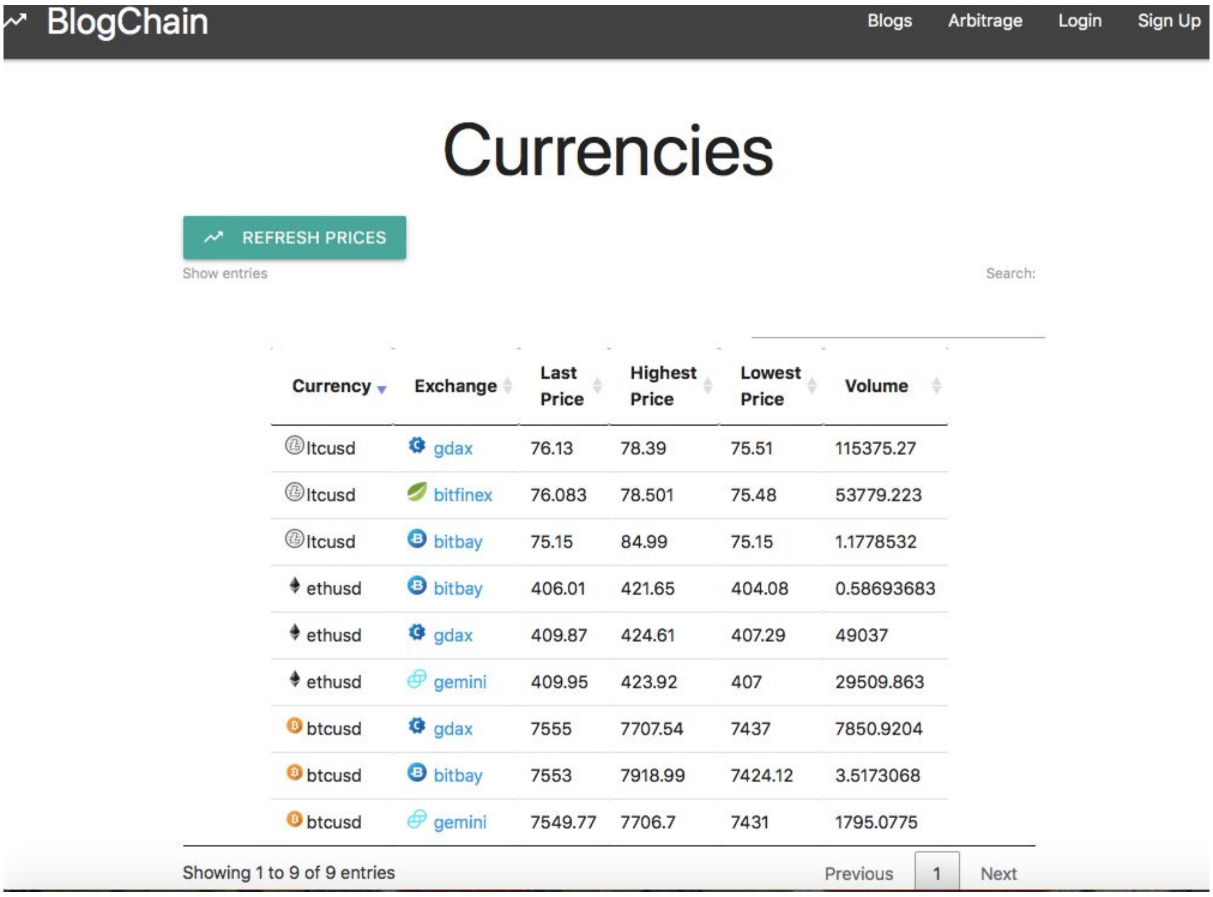
Password: ••••

Figure 17 - Test results of attempting to access member-only features in BlogChain (e.g. Arbitrages).

3.3 Dashboard

3.3.1 Dashboard Table

Expected output: The Dashboard page showing a table with the latest market data for each cryptocurrency and the exchange that they are traded with the currency data.
Actual output: as expected.



Currencies

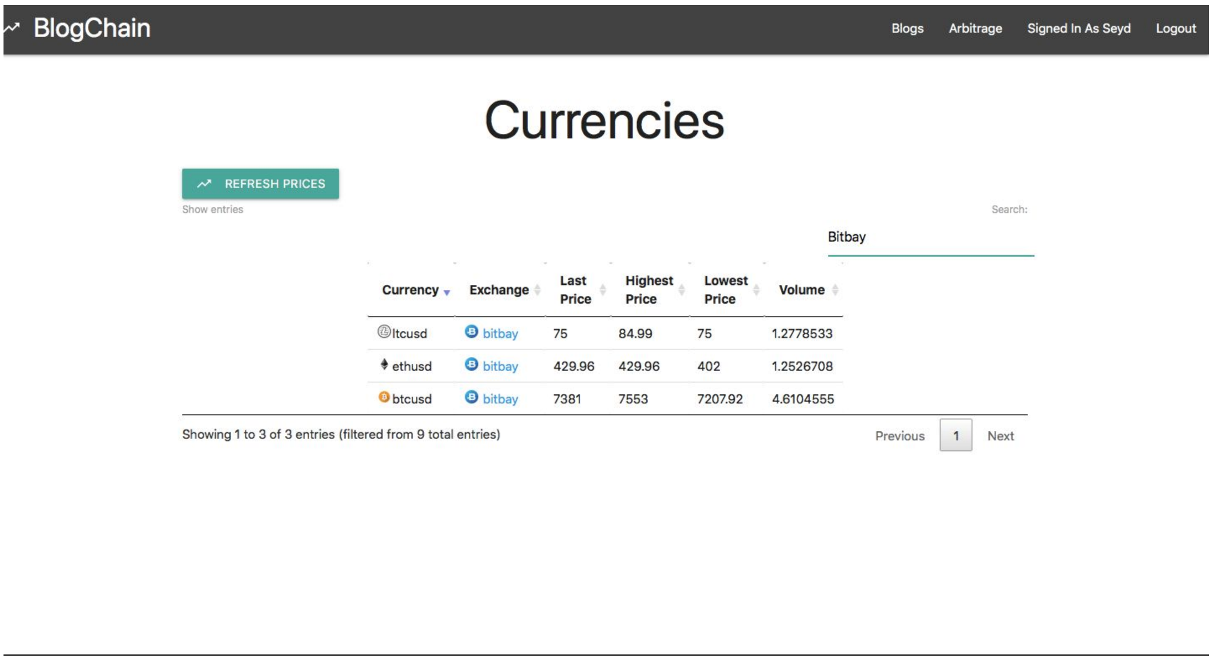
Currency	Exchange	Last Price	Highest Price	Lowest Price	Volume
ltcusd	gdax	76.13	78.39	75.51	115375.27
ltcusd	bitfinex	76.083	78.501	75.48	53779.223
ltcusd	bitbay	75.15	84.99	75.15	1.1778532
ethusd	bitbay	406.01	421.65	404.08	0.58693683
ethusd	gdax	409.87	424.61	407.29	49037
ethusd	gemini	409.95	423.92	407	29509.863
btcusd	gdax	7555	7707.54	7437	7850.9204
btcusd	bitbay	7553	7918.99	7424.12	3.5173068
btcusd	gemini	7549.77	7706.7	7431	1795.0775

Showing 1 to 9 of 9 entries

Figure 18 - Test results of loading the dashboard table.

3.3.2 Dashboard Table Sorting

Expected output: The matches for the selected column and the search parameter
Actual output: as expected.



Currencies

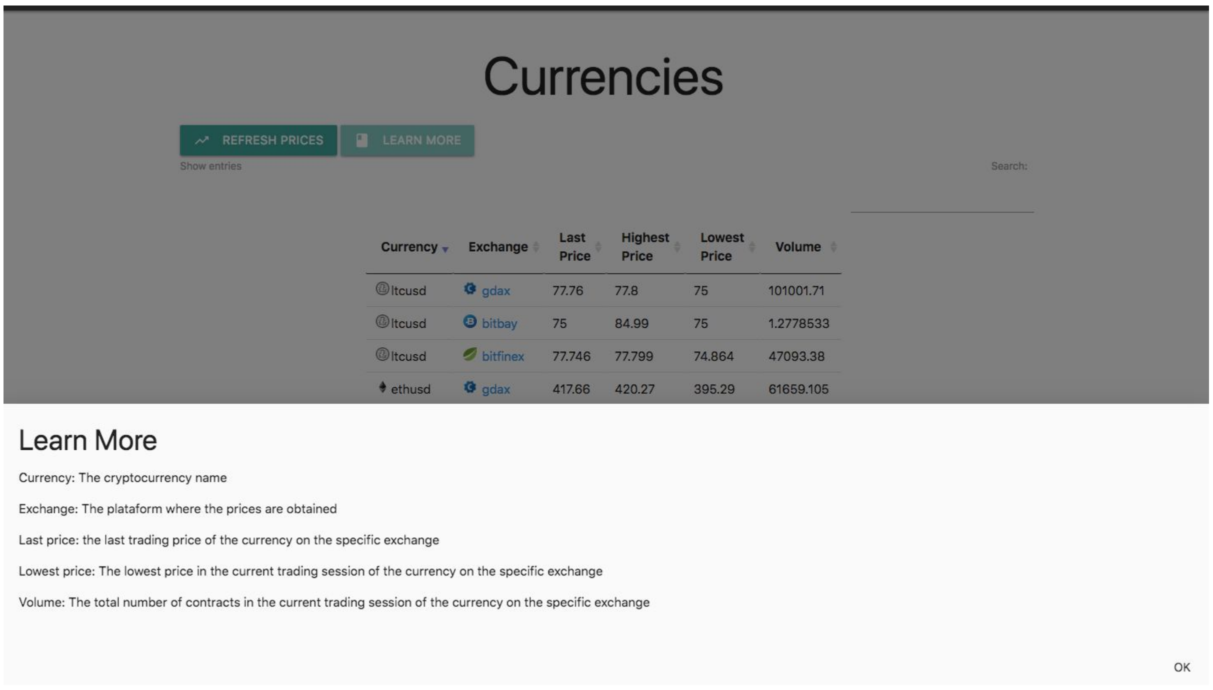
Currency	Exchange	Last Price	Highest Price	Lowest Price	Volume
ltcusd	bitbay	75	84.99	75	1.2778533
ethusd	bitbay	429.96	429.96	402	1.2526708
btcusd	bitbay	7381	7553	7207.92	4.6104555

Showing 1 to 3 of 3 entries (filtered from 9 total entries)

Figure 19 - Test results of sorting the dashboard table by entering a search parameter.

3.3.3 Dashboard Education

Expected output: A scrollable bottom fixed window explaining the table columns
Actual output: as expected.



Currencies

Currency	Exchange	Last Price	Highest Price	Lowest Price	Volume
ltcusd	gdax	77.76	77.8	75	101001.71
ltcusd	bitbay	75	84.99	75	1.2778533
ltcusd	bitfinex	77.746	77.799	74.864	47093.38
ethusd	gdax	417.66	420.27	395.29	61659.105

Learn More

Currency: The cryptocurrency name

Exchange: The plataform where the prices are obtained

Last price: the last trading price of the currency on the specific exchange

Lowest price: The lowest price in the current trading session of the currency on the specific exchange

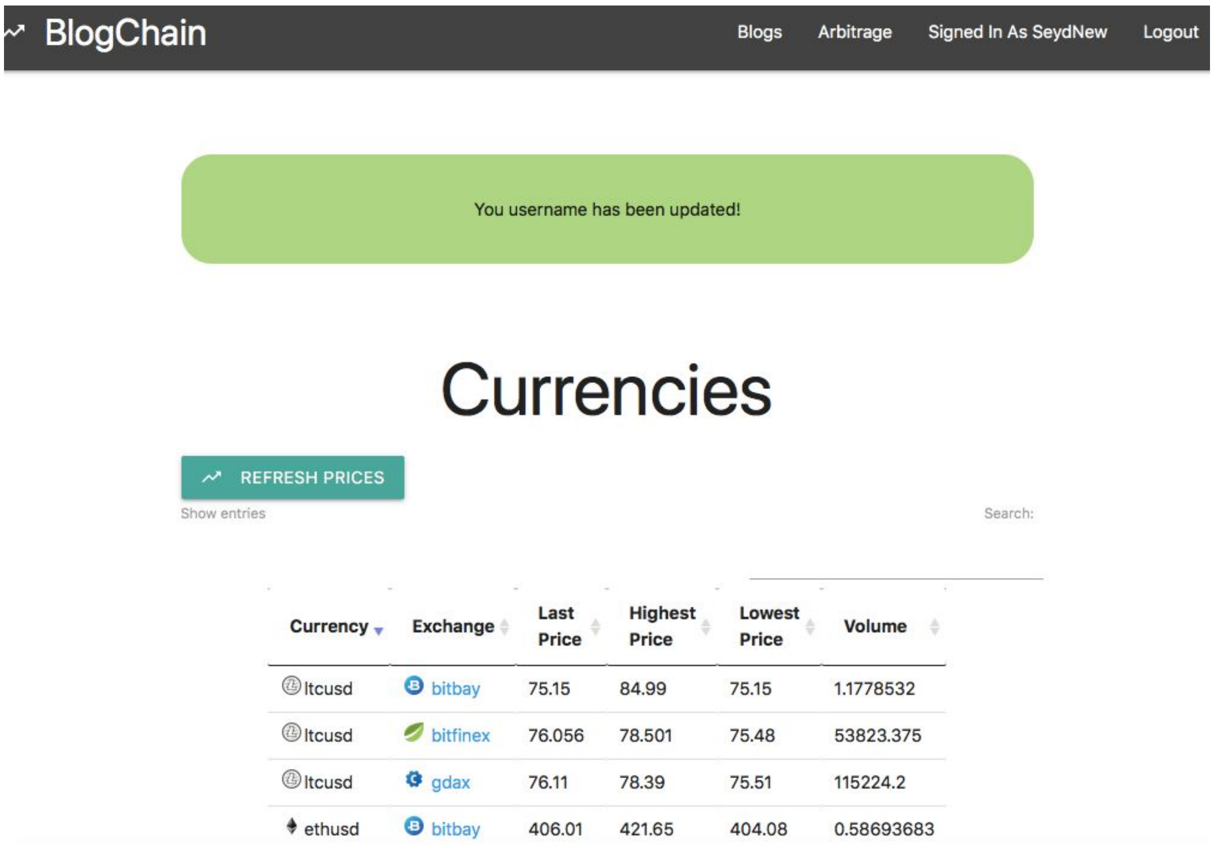
Volume: The total number of contracts in the current trading session of the currency on the specific exchange

Figure 20 - Test results of a bottom-fixed window providing an explanation with about the dashboard columns.

3.4 User Profile

3.4.1 Update User Information

Expected output: A green flash message indicating that the profile has been updated.
Actual output: as expected.



You username has been updated!

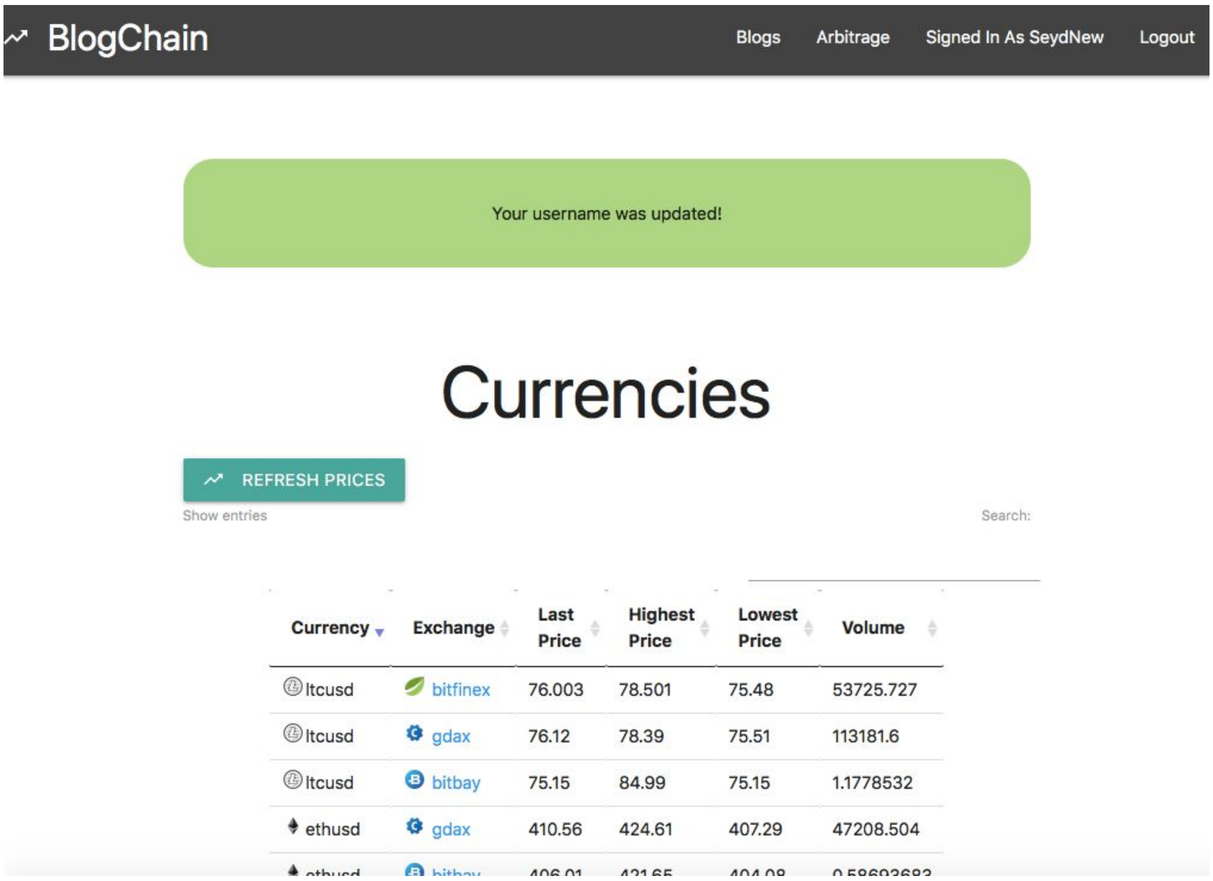
Currencies

Currency	Exchange	Last Price	Highest Price	Lowest Price	Volume
ltcusd	bitbay	75.15	84.99	75.15	1.1778532
ltcusd	bitfinex	76.056	78.501	75.48	53823.375
ltcusd	gdax	76.11	78.39	75.51	115224.2
ethusd	bitbay	406.01	421.65	404.08	0.58693683

Figure 21 - Test results of a successful attempt at updating user credentials.

3.4.2 Update Username

Expected output: A flash message indicating that the Username has been updated.
Actual output: as expected.


BlogChain Blogs Arbitrage Signed In As SeydNew Logout

Your username was updated!

Currencies

REFRESH PRICES

Show entries Search:

Currency	Exchange	Last Price	Highest Price	Lowest Price	Volume
ltcusd	bitfinex	76.003	78.501	75.48	53725.727
ltcusd	gdax	76.12	78.39	75.51	113181.6
ltcusd	bitbay	75.15	84.99	75.15	1.1778532
ethusd	gdax	410.56	424.61	407.29	47208.504
ethusd	bitbay	406.01	421.65	404.08	0.58693683

Figure 22 - Test results of successfully updating the username.

3.4.3 Delete User Account

Expected output: The user gets redirected to the application entry point.
Actual output: as expected.

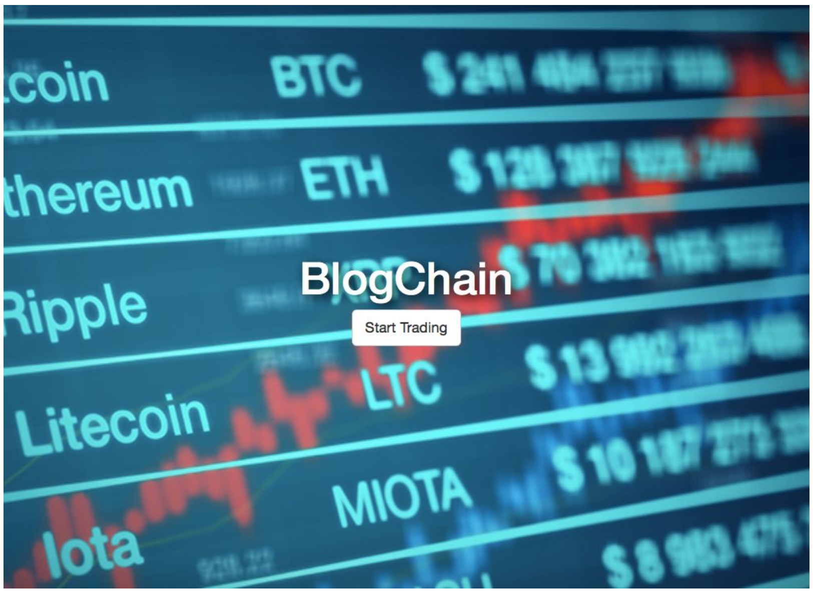


Figure 23 - Test results of successfully deleting a user account, which redirects to the intro page.

3.5 Blogs & Comments

3.5.1 Creating/Updating a Blog (Content Too Short)

Expected output: A browser message informing of the requirements for the input.
Actual output: as expected.

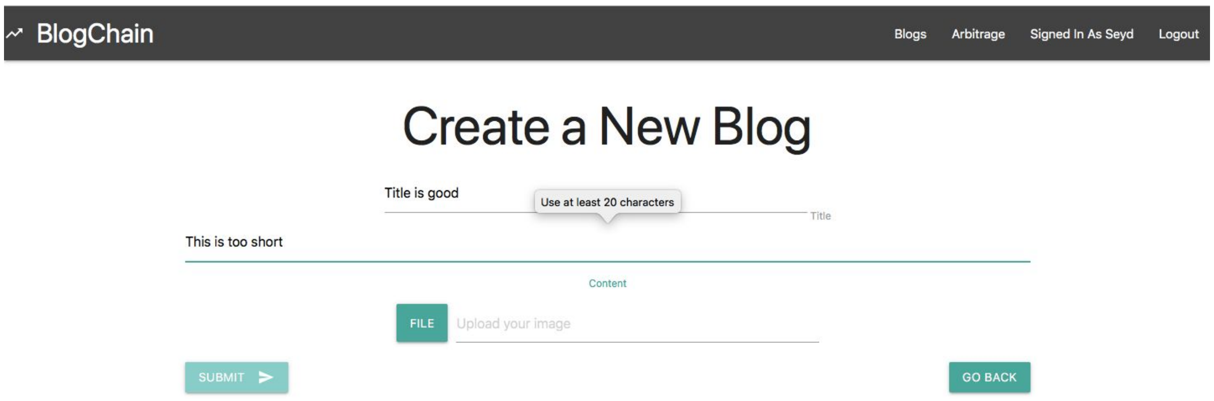
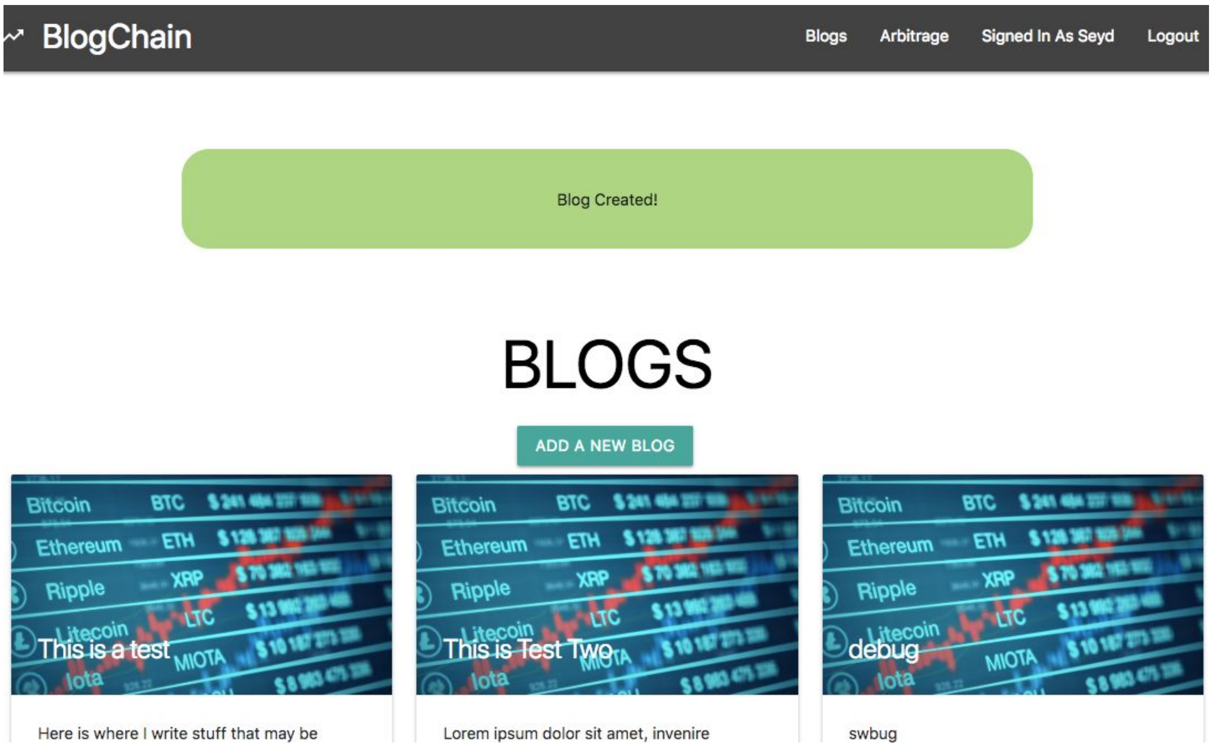


Figure 24 - Test results of entering a short text input in the content blog. The same message will display for any other text input.

3.5.2 Successfully Creating a New Blog

Expected output: A green flash message informing that the blog was created. The user gets redirected to the blogs page where the blog is displayed.
Actual output: as expected.


BlogChain Blogs Arbitrage Signed In As Seyd Logout

Blog Created!

BLOGS

ADD A NEW BLOG

This is a test

Here is where I write stuff that may be

This is Test Two

Lorem ipsum dolor sit amet, invenire

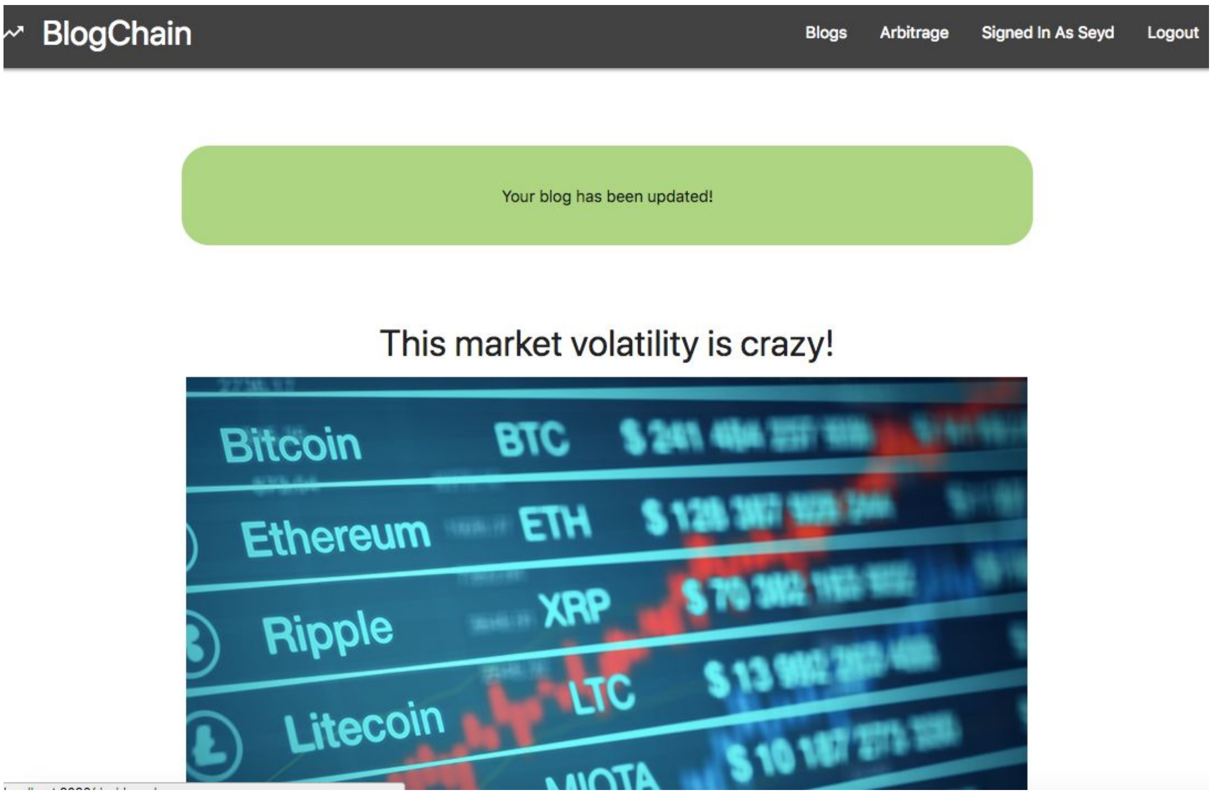
debug

swbug

Figure 25 - Test results of successfully creating a new blog.

3.5.3 Successfully Updating a Blog

Expected output: A green flash message informing that the user has updated the blog. The user gets redirected to the specific blog page with the changes.
Actual output: as expected.


BlogChain Blogs Arbitrage Signed In As Seyd Logout

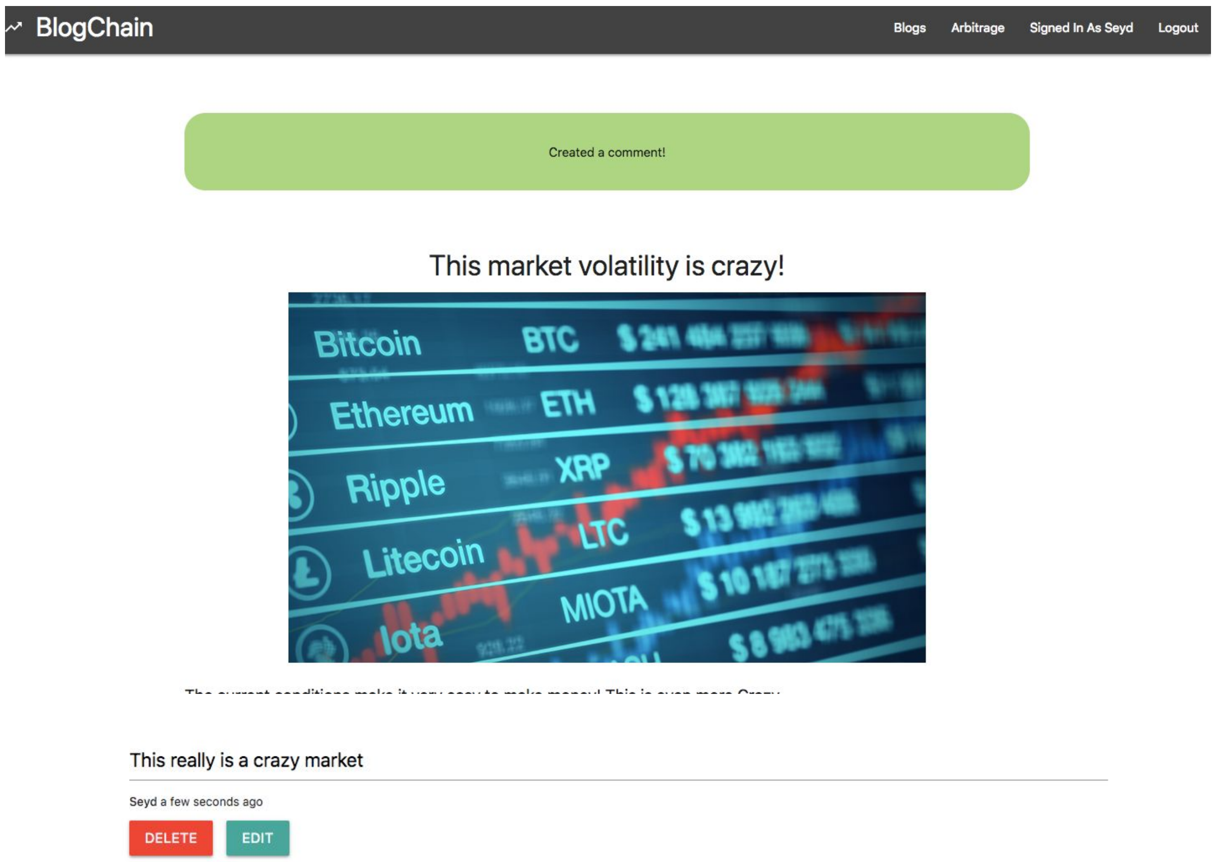
Your blog has been updated!

This market volatility is crazy!

Figure 26 - Test results of successfully updating a blog.

3.5.4 Commenting on a Blog

Expected output: A green flash message informing that the user has posted a comment. The user gets redirected to the blog. The comment is displayed at the bottom of the blog page while displaying the EDIT/CANCEL button.
Actual output: as expected.



Created a comment!

This market volatility is crazy!

The current conditions make it very easy to make money! This is even more Crazy

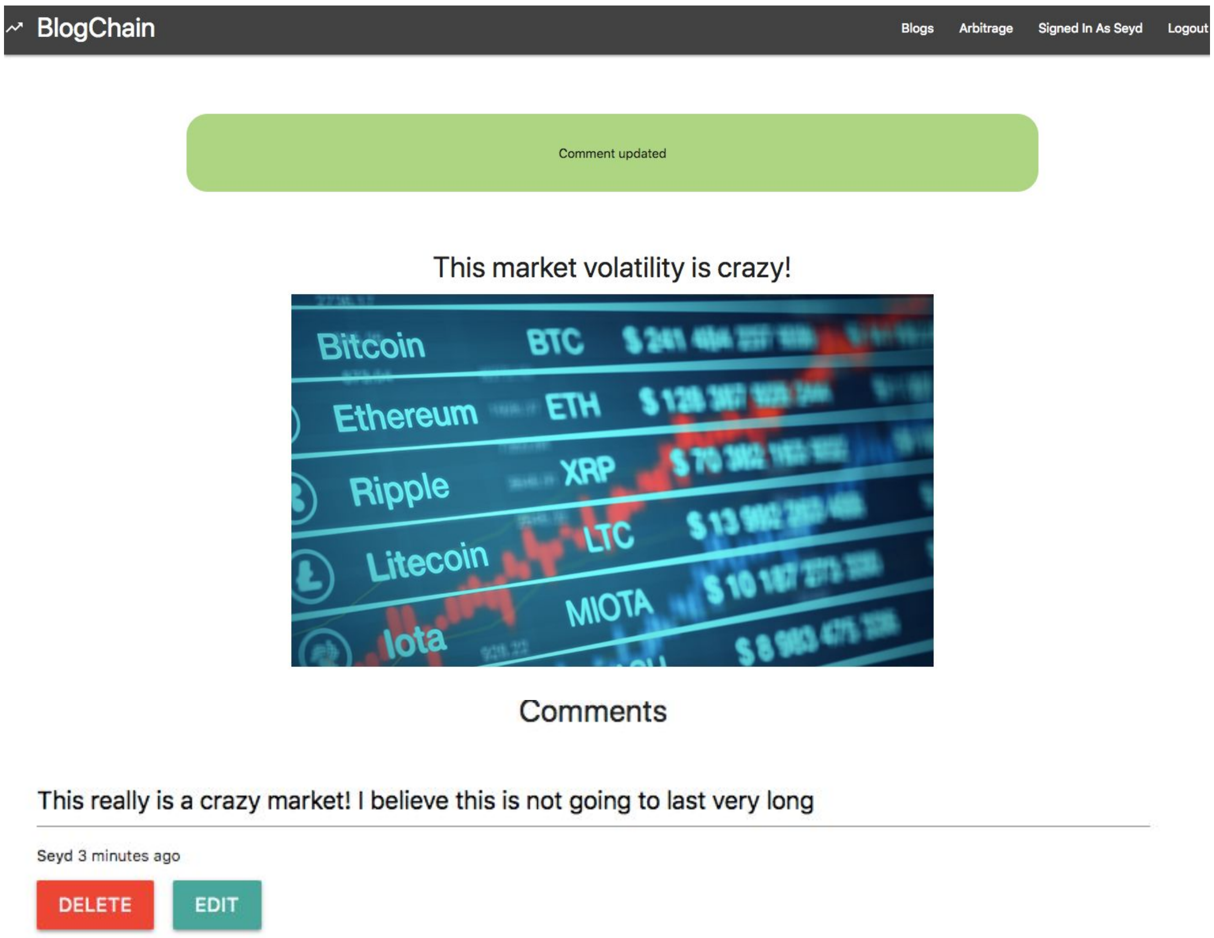
This really is a crazy market

Seyd a few seconds ago

Figure 27 - Test results of successfully writing a comment on a blog post.

3.5.5 Updating a Comment on a Blog

Expected output: A green flash message informing that the user has updated the comment. The user gets redirected to the blog page with the updated information.
Actual output: as expected.



Comment updated

This market volatility is crazy!

Comments

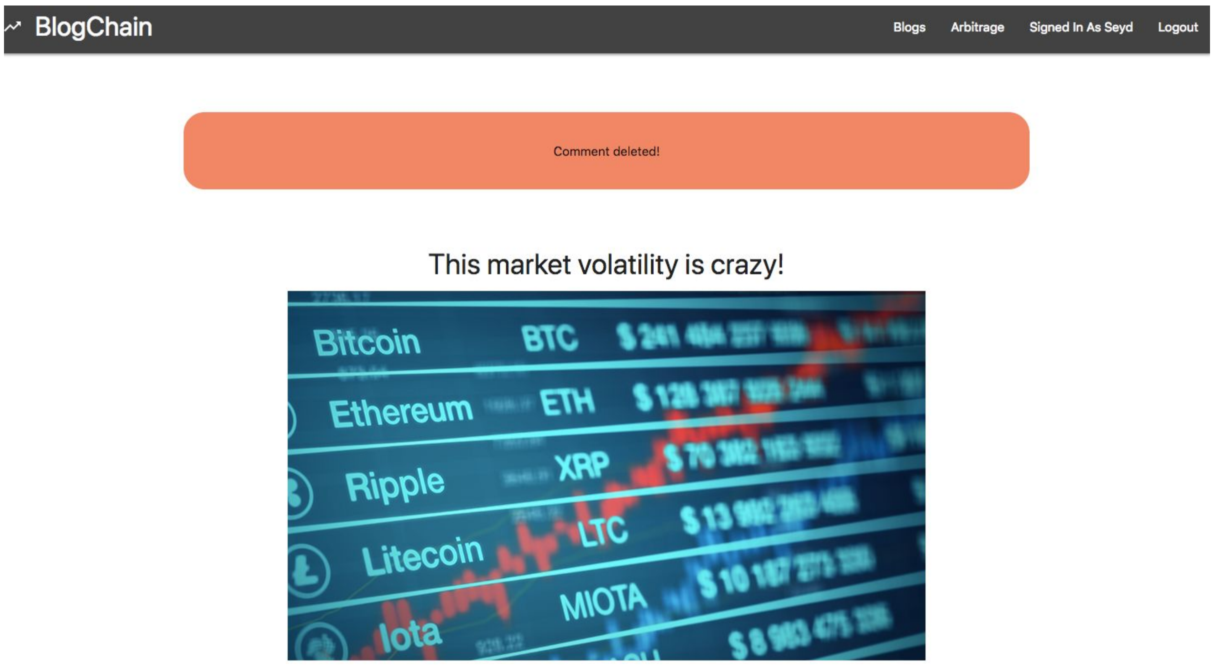
This really is a crazy market! I believe this is not going to last very long

Seyd 3 minutes ago

Figure 28 - Test results of updating a comment on a blog post.

3.5.6 Deleting a Comment on a Blog

Expected output: A red flash message indicating that the comment has been deleted. The user gets redirected to the blog page.
Actual output: as expected.


BlogChain

Blogs Arbitrage Signed In As Seyd Logout

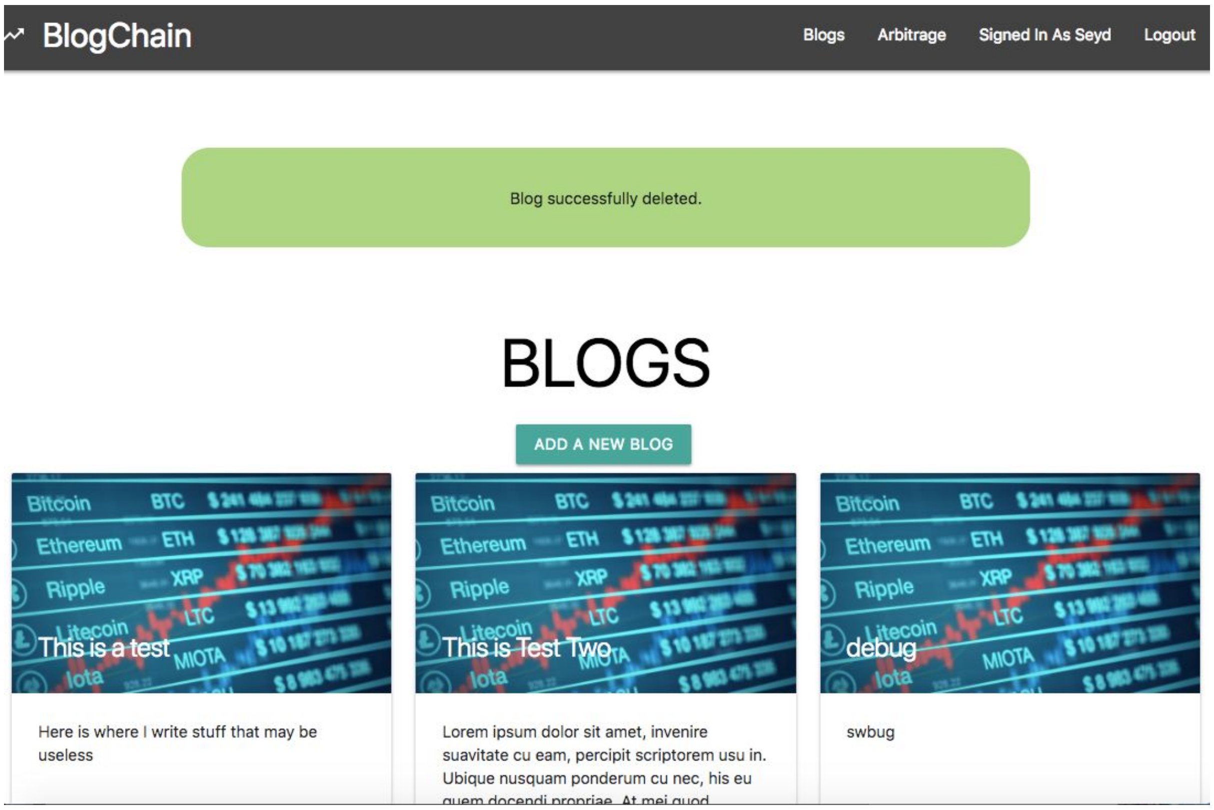
Comment deleted!

This market volatility is crazy!

Figure 29 - Test results of deleting your own comment on a blog post.

3.5.7 Deleting a Blog

Expected output: A green flash message informing that the blog was deleted. The user gets redirected to the blogs page
Actual output: as expected.


BlogChain

Blogs Arbitrage Signed In As Seyd Logout

Blog successfully deleted.

BLOGS

ADD A NEW BLOG

This is a test

Here is where I write stuff that may be useless

This is Test Two

Lorem ipsum dolor sit amet, invenire suavitate cu eam, percipit scriptorem usu in. Ubique nusquam ponderum cu nec, his eu quem docendi propriae. At mei quod

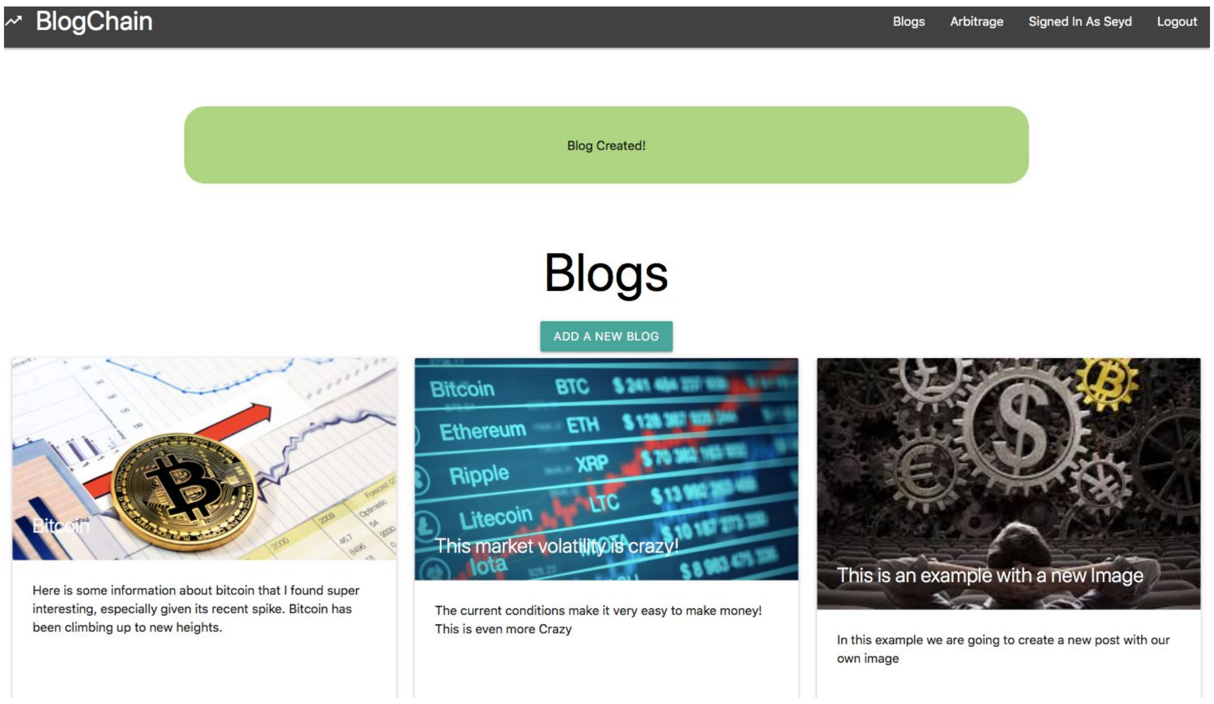
debug

swbug

Figure 30 - Test results of successfully deleting a blog post.

3.5.8 Upload Image

Expected output: A blog with the selected Image and a green flash message informing that a blog has been created.
Actual output: as expected.



Blog Created!

Blogs

Bitcoin

Here is some information about bitcoin that I found super interesting, especially given its recent spike. Bitcoin has been climbing up to new heights.

This market volatility is crazy!

The current conditions make it very easy to make money! This is even more Crazy

This is an example with a new Image

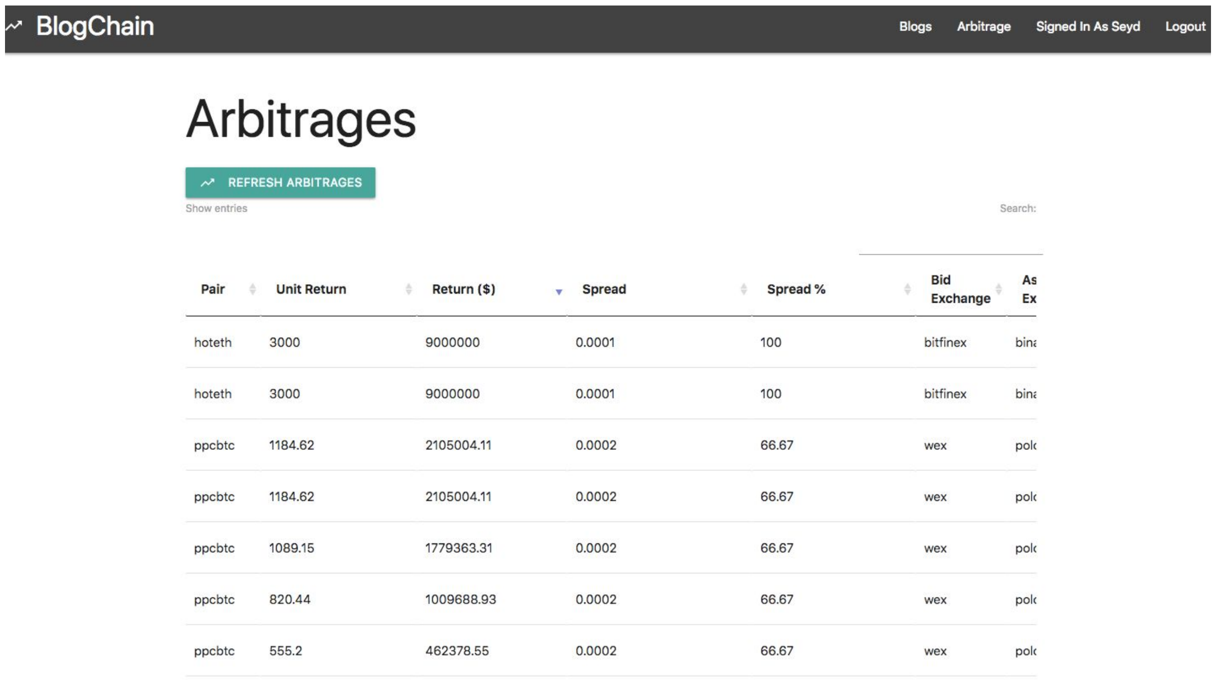
In this example we are going to create a new post with our own image

Figure 31 -A new blog a with a selected image.

3.6 Arbitrage

3.6.1 Arbitrage Table

Expected output: The user gets redirected to the Arbitrage page where the arbitrage information is displayed in a dynamic table.
Actual output: as expected.


BlogChain

Blogs Arbitrage Signed In As Seyd Logout

Arbitrages

REFRESH ARBITRAGES

Show entries

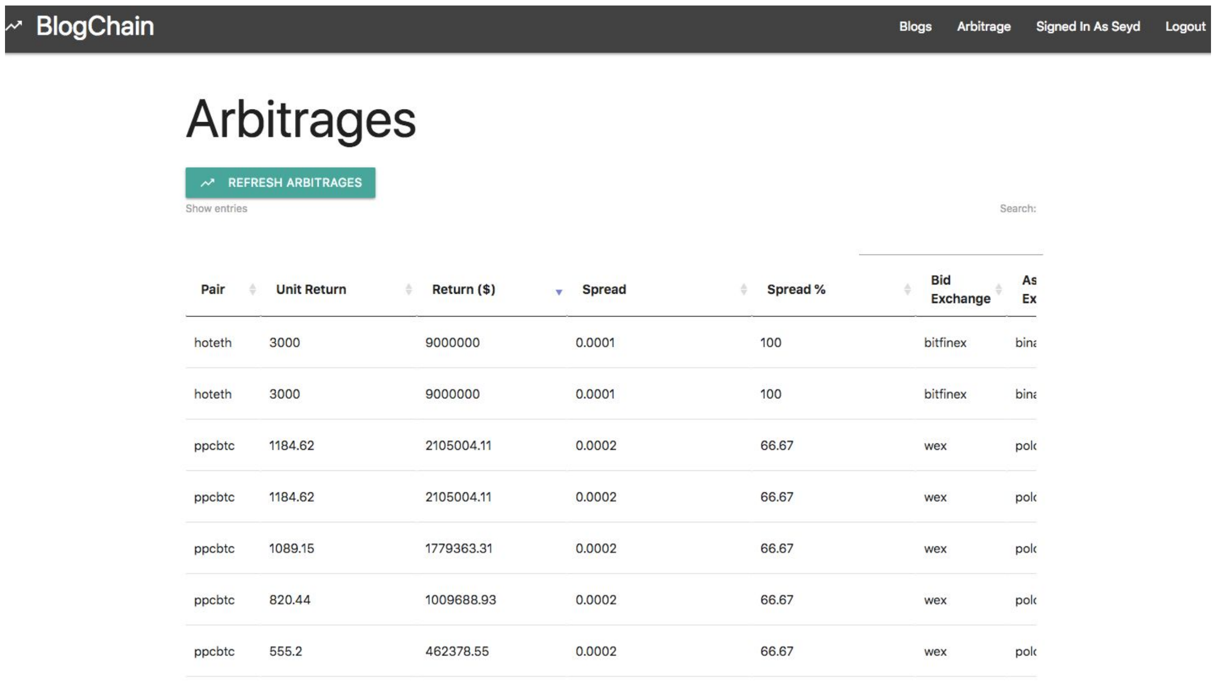
Search:

Pair	Unit Return	Return ($)	Spread	Spread %	Bid Exchange	As Ex
hoteth	3000	9000000	0.0001	100	bitfinex	bina
hoteth	3000	9000000	0.0001	100	bitfinex	bina
ppcbtc	1184.62	2105004.11	0.0002	66.67	wex	pol
ppcbtc	1184.62	2105004.11	0.0002	66.67	wex	pol
ppcbtc	1089.15	1779363.31	0.0002	66.67	wex	pol
ppcbtc	820.44	1009688.93	0.0002	66.67	wex	pol
ppcbtc	555.2	462378.55	0.0002	66.67	wex	pol

Figure 32 - Test results of loading the arbitrage table.

3.6.2 Arbitrage Table Sorting

Expected output: The matches for the selected column and the selected parameters
Actual output: as expected.


BlogChain

Blogs Arbitrage Signed In As Seyd Logout

Arbitrages

REFRESH ARBITRAGES

Show entries

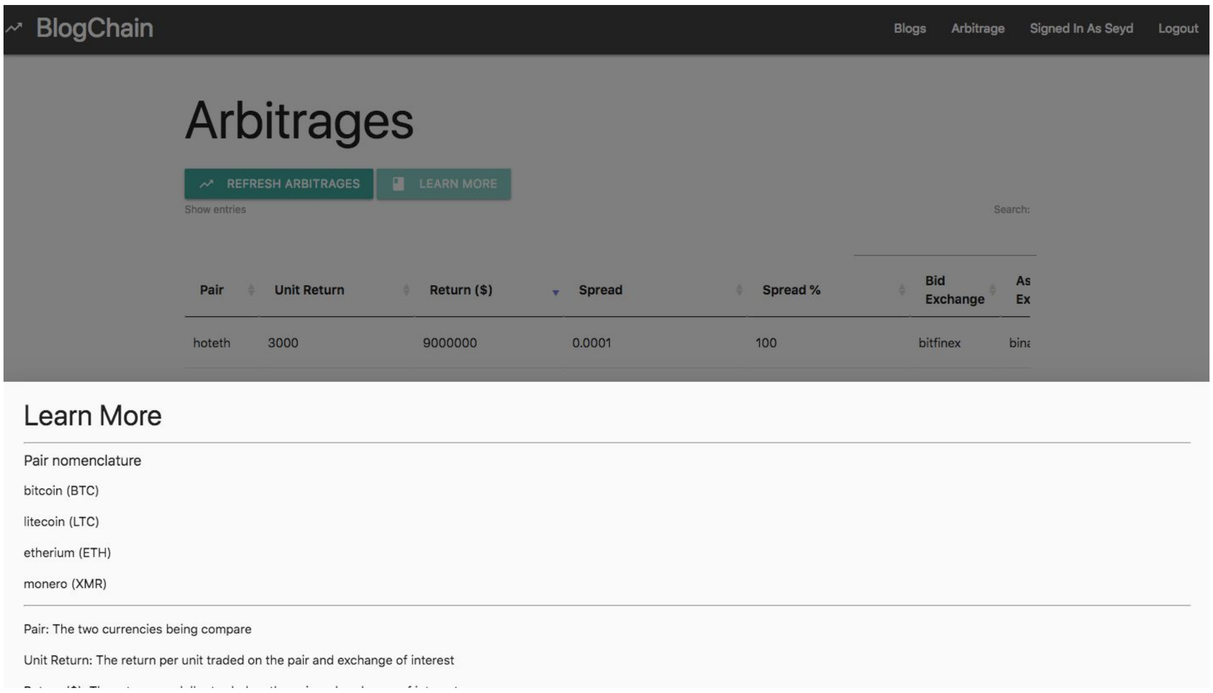
Search:

Pair	Unit Return	Return ($)	Spread	Spread %	Bid Exchange	As Ex
hoteth	3000	9000000	0.0001	100	bitfinex	bina
hoteth	3000	9000000	0.0001	100	bitfinex	bina
ppcbtc	1184.62	2105004.11	0.0002	66.67	wex	pol
ppcbtc	1184.62	2105004.11	0.0002	66.67	wex	pol
ppcbtc	1089.15	1779363.31	0.0002	66.67	wex	pol
ppcbtc	820.44	1009688.93	0.0002	66.67	wex	pol
ppcbtc	555.2	462378.55	0.0002	66.67	wex	pol

Figure 33 - Test results of sorting the arbitrage table in ascending order by the Spread column.

3.6.3 Arbitrage Education

Expected output: A scrollable bottom fixed window explaining the table columns
Actual output: as expected.


BlogChain

Blogs Arbitrage Signed In As Seyd Logout

Arbitrages

REFRESH ARBITRAGES LEARN MORE

Show entries Search:

Pair	Unit Return	Return ($)	Spread	Spread %	Bid Exchange	As Ex
hoteth	3000	9000000	0.0001	100	bitfinex	bin

Learn More

Pair nomenclature

bitcoin (BTC)

litecoin (LTC)

etherium (ETH)

monero (XMR)

Pair: The two currencies being compare

Unit Return: The return per unit traded on the pair and exchange of interest

Figure 34 - Test results of a bottom-fixed window providing an explanation with about the arbitrage columns.

4 Future Work

4.1 Algorithmic Trading

BlogChain allows the user to exploit market inefficiencies between different cryptocurrency providers. These inefficiencies are based on price discrepancies between multiple markets. While this creates an opportunity for our users, it creates difficulties when managing different trading accounts that may have different trading restrictions. One improvement to be made would be to have a single customer account linked to the different providers so all the trades could be done from a single platform.

Having a single platform where the user can manage their different accounts would allow to automate the process based on some parameters, so the user can select the conditions to place a trade. Subsequently, a notification would be sent to the user after the trade has been entered.

Another improvement would be to create a service that allows users to check their own assumptions by using Big Data technologies. At the moment, we are storing market data with the intention of creating this service in the future.

4.2 Social Ranking

A frequent problem with platforms that allow pseudonymous users to make posts is that false information can be passed off as legitimate with no repercussions for the culprit. One way to mitigate this would be a social ranking system.

Users would have a reputation that starts at 0 upon account creation. This reputation value is visible to all other users and would be found under a user's username when making comments or posting blogs. If a user posts false information anywhere on BlogChain and other users become aware of this, they can reduce the reputation of the offending user. On the other hand, if a user posts legitimate, helpful information then the affected users may increase the reputation of said user. This reputation value will help users discern whether information is valid or not by letting them see how reliable the source of the information is.

If a user's reputation falls to extremely low values then they will be punished accordingly. A user may be silenced or banned entirely based on how negative their reputation is.

Another use for a reputation score is rewarding the helpful users with exceptional reputation scores. Users with glowing reputations will be rewarded by having their transactions prioritized. Combining reputation with the aforementioned algorithmic trading allows us to give preference to reputable users when a group of users is capitalizing on the same arbitrage opportunity. This gives users even more of an incentive to develop a good reputation.

4.3 Cryptocurrency Comments

A Twitter-like news feed for each cryptocurrency would improve the service so users can post their ideas and knowledge on the product they trade. At this moment, users can create blogs where they can post anything they want and other users can comment on it. This, however, is based on a single blog. Having comments based on cryptocurrencies, users are able to interact in real time with each other and share their thoughts much faster in sorted fashion (one comment section for each cryptocurrency).

4.4 User Surveys

No product is a finished product. In order to create a customer friendly service we require input from our users. A survey service would allow us to get feedback from our customers and improve the BlogChain user experience, and ultimately find out what features customers want. User surveying is a solid method for finding out what problems customers have, and whether or not the features provided by BlogChain solves their problem. This could be done through each user's account or via email.

4.5 Additional Cloud Components

With more components and features to be added on to our project, we will need to implement more cloud elements. Implementing load balancing on our cloud server would be the first step to making our platform more reliable and sustainable in the long-term. Since our goal is to have as many users as possible enjoying BlogChain, load balancing will help keep our server stable and affordable.

Another cloud element to be added is a database for the prices. Currently the arbitrages are stored in a database but the prices are queried directly from the Cryptowatch API. As we support more exchanges and coins, we come closer to reaching the request limit for Cryptowatch. Storing the prices in a database after they are queried will help us avoid reaching this limit.

Even with storing prices, the rate limit still poses a problem for us as it makes it difficult to display recent information for a lot of exchanges at once. If we implement cloud servers with the sole purpose of querying for a certain amount of exchanges or coins, we could instead query those cloud servers for the information they retrieve. This allows us to avoid reaching the rate limit while having the most recent data possible.