CurrencyHFT

Cha Ching

STAKEHOLDER REVIEW # 2

Jack Cusick
James Nakashian
Patrick Engelsman
Matthew Cordone
Ayushi Mishra

Working Beta Release

During lecture, the team was able to demonstrate to the course instructor the core progress and functionality of the CurrencyHFT application. This demonstration provided the team with useful feedback for the upcoming final demonstration. By detailing the features and implementation of the application, the team displayed their ability to show a key stakeholder how CurrencyHFT's functionality can benefit target users. The stakeholder 2 demonstration also provided beneficial practice in presenting the features of the application and explaining the technical aspects of CurrencyHFT's implementation. The team has completed a majority of the application's functionality. The most up-to-date release of the application can be found at currencyhft.com.

Source Code

The source code for CurrencyHFT is included in the zip file "CurrencyHFT.zip". The directory structure of the source code is as follows (displaying only the major folders for brevity):

- database/ - holds DB schema file and web scrapers
    - data/ - stores text files of scraped currency rates
- frontend/ - holds files used in generating frontend using Vue framework
    - src/ - holds frontend functionality
        - assets/ - holds images and icons
        - components/ - holds Vue components and
        - router/ - holds Vue router and navigation
    - theme/ - hold stylesheets
- node/ - holds files used for running the backend using node
    - build/ - holds files to build the server
    - src/ - holds C++ files to run backend functions
        - sql/ - holds API class to make queries to SQL DB

Code written by the development team is contained in the database, node, and frontend/src/ folders.

Testing Strategy

Statement of Strategy

The development team will utilize a series of test cases to test the features and functionality of the CurrencyHFT web application. The preliminary results of each test case are recorded in the tables below. For each test case, the team member executing the test will follow the actions laid out in the chart, first by assuring the preconditions have been met then by following the assigned steps. Once the process has been completed, the team member will then check to ensure the post conditions have been properly met. If the test worked properly, the team member will record a “Pass” in the appropriate space and write a note of anything found while executing the test if necessary. If the test did not work, the team member will record a “Fail” in the appropriate space and add a note about the improper response and what went wrong.

Testing Plans

The CurrencyHFT website will be tested across Chrome, Firefox, and mobile browsers, using the team’s desktops, laptops, and smartphones. Each team member will perform every test for each iteration. Additional testers will be recruited from other RPI students that the team knows, and these testers will perform the tests with their own devices. The application testing will occur iteratively. For each iteration, the team members will work through the test cases then compare and review their results. The first testing iteration will be based from the preliminary information provided below. Each preceding test iteration will involve the fixing of any bugs related to any of the failed test cases followed by another iteration through the test cases to see the progress and improvements made. This process will continue until all major bugs have been found and corrected. At the end of each test iteration, the found bugs will be reported to the whole team via the GitHub Issues page.

The first iteration of testing (after the preliminary testing displayed below) will begin on 5/22. Iterations will continue daily until all major bugs have been fixed, or the deadline for the final presentation is reached. Testing will occur throughout the day, with the exception of 12:00AM-1:00AM, as daily updating of data occurs during this time.

Quality Goals

The team’s expectation is that the number of errors should decrease with each testing iteration. This will continue until the team has reached a point where all major bugs have been eliminated. After its completion, this iterative testing process should provide a much more well formatted and user friendly application.

Test Cases

Test Case: Arbitrage Using Forex (different currencies)	
Identifier	TC1
Description	User wants to find optimal path for an exchange in the forex market and get new amount
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a different ending currency 3. User enters an amount 4. User clicks calculate ○ The conversion from the amount is displayed ○ The optimal path of currencies is displayed ○ The regular and path rate are displayed

Test Case: Arbitrage Using Forex (same currencies)	
Identifier	TC2
Description	User wants to find optimal path for an exchange in the forex market, but selects same start and end
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects the same ending currency 3. User enters an amount 4. User clicks calculate ○ The conversion from the amount is just the amount ○ The optimal path of currencies is just the currency ○ The regular and path rate are both 1

Test Case: Arbitrage Using Forex (no amount)	
Identifier	TC3
Description	User wants to find optimal path for an exchange in the forex market to find the rate

Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a different ending currency 3. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed ○ The regular and path rate are displayed

Test Case: Arbitrage Using Forex (exclude different currencies)	
Identifier	TC4
Description	User wants to find optimal path for an exchange in the forex market but exclude a currency
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a different ending currency 3. User selects a currency not the starting or ending (then try selecting multiple currencies) 4. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed ○ The regular and path rate are displayed ○ Excluded currencies are not used in path

Test Case: Arbitrage Using Forex (exclude a same currencies)	
Identifier	TC5
Description	User wants to find optimal path for an exchange in the forex market but exclude a currency
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a different ending currency 3. User selects a currency that is the same as the starting or ending 4. User clicks calculate ○ Returns an error that says “Cannot exclude a currency in the exchange”

Test Case: Arbitrage with Forex (setting max number of exchanges)	
Identifier	TC6
Description	User wants to find optimal path for an exchange in the forex market but wants to limit the number of exchanges
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a different ending currency 3. User selects a max number of exchanges they'd like (try many different values with same currency pair) 4. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed, this path is not longer than the max selected ○ The regular and path rate are displayed

Test Case: Arbitrage Using Banks (different currencies or different banks)	
Identifier	TC7
Description	User wants to find optimal path for an exchange with banks and get new amount
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects an ending currency a. Different with same bank b. Different with different bank c. Same with different bank 4. User selects an ending bank 5. User enters an amount 6. User clicks calculate ○ The conversion from the amount is displayed ○ The optimal path of currencies/banks is displayed ○ The regular and path rate are displayed

Test Case: Arbitrage Using Banks (same currencies and bank)	
Identifier	TC8

Description	User wants to find optimal path for an exchange with banks, but selects same start and end
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects the same ending currency 4. User selects an ending bank that is the same 5. User enters an amount 6. User clicks calculate ○ The conversion from the amount is just the amount ○ The optimal path of currencies is just the currency ○ The regular and path rate are both 1

Test Case: Arbitrage Using Banks (no amount)	
Identifier	TC9
Description	User wants to find optimal path for an exchange with banks to find the rate
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects a different ending currency 4. User selects an ending bank 5. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed ○ The regular and path rate are displayed

Test Case: Arbitrage Using Banks (exclude different currencies)	
Identifier	TC10
Description	User wants to find optimal path for an exchange with banks but exclude a currency
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects a different ending currency 4. User selects an ending bank 5. User selects a currency not the starting or ending (then try selecting

	multiple currencies) 6. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed ○ The regular and path rate are displayed ○ Excluded currencies are not used in path

Test Case: Arbitrage Using Banks (exclude a same currencies)	
Identifier	TC11
Description	User wants to find optimal path for an exchange with banks but exclude a currency
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects a different ending currency 4. User selects a ending bank 5. User selects a currency that is the same as the starting or ending 6. User clicks calculate ○ Returns an error that says “Cannot exclude a currency in the exchange”

Test Case: Arbitrage Using Banks (setting max number of exchanges)	
Identifier	TC12
Description	User wants to find optimal path for an exchange with banks but wants to limit the number of exchanges
Preconditions	User is on arbitrage page
Sequence of Actions	1. User selects a starting currency 2. User selects a starting bank 3. User selects a different ending currency 4. User selects an ending bank 5. User selects a max number of exchanges they’d like (try many different values with same currency pair) 6. User clicks calculate ○ The conversion from the amount is 0 ○ The optimal path of currencies is displayed, this path is not longer than the max selected ○ The regular and path rate are displayed

Test Case: Choose Ticker in Dashboard Graph (different)	
Identifier	TC13
Description	Change the currencies being used to display the graph
Preconditions	User is on dashboard page
Sequence of Actions	1. Select a starting currency in the first drop down ○ Graph will adjust to use the new first currency 2. Select a different ending currency in the second ○ Graph will adjust to use this as the ending 3. Select a data view (try multiple views) ○ Graph will change to match the data view

Test Case: Choose Ticker in Dashboard Graph (same currencies)	
Identifier	TC14
Description	Change the currencies being used to display the graph, but chooses the same start and end currencies
Preconditions	User is on dashboard page
Sequence of Actions	1. Select a starting currency in the first drop down ○ Graph will adjust to use the new first currency 2. Select the same ending currency in the second ○ Dashboard does not allow you to select the same

Test Case: Choose a Certain Data Set	
Identifier	TC15
Description	User can select certain data sets to be used in the graph
Preconditions	User is on dashboard page
Sequence of Actions	1. Select a starting currency in the first drop down ○ Graph will adjust to use the new first currency 2. Select a different ending currency in the second ○ Graph will adjust to use this as the ending 3. Select a data view ○ Graph will change to match the data view 4. Click “high” above the graph ○ The “high” data should be removed from the graph

	5. Click “close” above the graph ○ The “close” data should be removed from the graph 6. Click “low” above the graph ○ All the data is shown again on the graph

Test Case: Calculator Conversion (different)	
Identifier	TC16
Description	User can convert an amount of one currency to another
Preconditions	User is on calculator page
Sequence of Actions	1. Select a starting currency 2. Select a different ending currency 3. Enter a value to convert ○ The converted amount is displayed 4. Switch starting and ending currencies ○ The converted value should update

Test Case: Calculator Conversion (same)	
Identifier	TC17
Description	User tries to convert an amount of a currency to the same currency
Preconditions	User is on calculator page
Sequence of Actions	1. Select a starting currency 2. Select the same ending currency 3. Enter an amount to convert ○ It should return the same value as entered

Preliminary Testing Iteration:

Test Case	Results	Comments
TC1	Pass.	This test case works properly. The user is able to input a different starting and ending currency along with a limit for the number of currencies that can be used in the path. The site will return a display of the optimal path to the user along with all the related information.
TC2	Pass.	This test case works properly. When the user selects the same starting and ending currency the site will return the currency itself with all other information unchanged.
TC3	Pass.	The user is shown a zero field for the monetary profit and conversion fields, but is shown the actual values for the Optimal Rate and Direct Rate fields.
TC4	Fail.	When a single currency to exclude is given, the Calculate feature on the page does not display any information on the optimal path or the found exchange rates. However, if multiple are chosen the button works properly
TC5	Fail.	When a currency to exclude is given and matches a starting or ending currency, the Calculate feature on the page does not display any information on the optimal path or the found exchange rates. Attempting to exclude both the starting and ending currencies crashed the server.
TC6	Pass.	This test case works properly. The user is able to input a different starting and ending currency along with a limit for the number of currencies that can be used in the path. The site will return a display of the optimal path to the user along with all the related information.
TC7	Fail.	The ability to choose banks is not on the frontend yet. So this fails.
TC8	Fail.	The ability to choose banks is not on the frontend

		yet. So this fails.
TC9	Fail.	The ability to choose banks is not on the frontend yet. So this fails.
TC10	Fail.	The ability to choose banks is not on the frontend yet. So this fails.
TC11	Fail.	The ability to choose banks is not on the frontend yet. So this fails.
TC12	Fail.	The ability to choose banks is not on the frontend yet. So this fails.
TC13	Pass.	User is able to select two different currency tickers. User can freely change the currencies tracked by the tickers and the display graph will change accordingly.
TC14	Fail.	The same currency is able to be selected and the graph does not change at all.
TC15	Pass.	User can select and deselect all the data sets and they would display if selected or not display if they are deselected
TC16	Fail.	Users can select two different currencies and the conversion is correctly displayed. Conversion is incorrect if user switches currencies.
TC17	Pass.	The same value is returned whenever the same two currencies are selected.

Code Review

CurrencyHFT Reviews Community

The core functionality of the Community application is to allow users to both post and respond to community service jobs. Community is a web based application which allows users to create profiles and propose community service jobs as well as volunteer to attend those prospective community service jobs. The team seemed to have a solid idea of their project and where they wanted to be by the final iteration. The team showed a good understanding of their project scope and feasibility of implementation.

The team had implemented what is essentially a trustworthiness score for each user, allowing other users who attended the event to rate the user who hosted the event. This score was designed to work as a summation of each of the events hosted by the user, showing a more generally reliable user the higher their score is. Our team proposed an alteration to this feature, showing a user score that is an average of each of their projects rather than a summation, that way a less reliable user cannot make themselves seem better by posting many jobs. We also suggested adding a feature to allow the event host to rate the attending users on their reliability. We also suggested a verification by email upon each new user profile creation.

The Community code base seemed well organized, relatively simple, and fairly well commented. The code was structured such that we believe we would be able to relatively quickly familiarize ourselves with the code and gain a solid understanding of the implementation given more time to review. The team was using MySQL as their database system in order to store all the user profile and project information. The team was utilizing DJango as their main server technology. The team was using the Amazon Web Server EC2 service to host their server due to the fact that it provides a simple and proficient web server interface and also allows for one year of free hosting.

The Community team was utilizing GitHub for development due to its helpful set of features and proficient code and branch organizational structure. However, the team was not utilizing GitHub to the best of its abilities. The team was only using the coding and branching functionality of GitHub. Our team suggested that they start making use of the Issues and Wiki features of GitHub. The Wiki feature is very useful for creating useful instruction sets and for logging any information or practices that all team members should keep in mind while developing. The Issues feature is very useful for keeping track of the tasks that must be done, which team members are in charge of completing those tasks, and logging when each task has been completed. The Issues feature is also very useful for pointing out bugs in the code.

Null Pointers Reviews CurrencyHFT

The Null Pointers team was only able to give us limited feedback during their review of our CurrencyHFT website demonstration. They gave us a few suggestions for additions to our web page in order to increase the ease of use. The first suggestion was for the addition of a user manual area on the website to describe to the users how to navigate the web page and its features. The second suggestion was for the addition of a tooltips for the UI section which will display a brief explanation of each of the website's tools and how to use them. We will be taking those suggestions into consideration and adding these descriptions into the home page.

Due to the limited time constraint of the review process we were only able to show the Null Pointers team our primary sections of the code base for our core features and were not able to get very deep into the implementation details. Their main suggestion towards the code base implementation was in dealing with unexpected data received through the Node.js server. As it was implemented, anyone was able to send a get request through the server. The Null Pointers team suggested for us to implement a way to sanitize the inputs which will help protect and consolidate the received data and push the exchanges forward more efficiently.

The Null Pointers team also recommended us to more succinctly connect our code on the front-end and back-end sides through the Node.js. They also suggested for us to keep in mind that we need to be careful for SQL injections through our development.

Contribution Summary

All team members contributed to editing, reviewing, and approving the Stakeholder Review 2 deliverable for final submission. Breakdown of individual contributions is as follows:

Jack Cusick
Code commenting, repo restructuring, full document editing.

James Nakashian
Code commenting, directory structure outline, edited and added to Testing Documents, test cases 16-17

Patrick Engelsman
Code commenting, wrote test cases, completed/edited some of the preliminary testing results

Matthew Cordone
Full document editing, wrote code review section, wrote testing strategy section, wrote beta release section, wrote status report, tested the test cases 1-13.

Ayushi Mishra
Code commenting, directory structure for front end

Status Report

The Status Report Document identifies the team's progress throughout development.

The team is currently correcting minor bugs throughout the implementation. The team has finalized the main functionality of the code and are now working to fix specific issues within the code base and format the front-end website display to be viewed in the most user friendly format. The team has gone through and optimized the formatting of all the code to the accepted coding standards. The team has read through and added descriptive comments to the code base.

Moving forward, once the team has completed the Stakeholder Review #2, the team will prepare for the final presentation the following week and prepare for the final release in the transition phase. Moving towards the final transition the team will also work to fix any minor bugs that may still exist in the implementation, add support for currently missing features, and optimize the front-end display on the webpage, currencyhft.com.