Fortune

A Cryptocurrency Trading Game

Sprint 2 Retrospective

Team 11: Ryan Huff, Sam Kravitz, Akash Lankala, Raziq Raif Ramli, Tyler Stanish, Blake Steel

What went well?

General:

At the beginning of Sprint 2 we decided to split up stories based on person rather than by task to avoid having blockers between different parts of a story and being dependent on another person to finish their task before being able to continue on. This worked very well and made development much more efficient from an individual standpoint. Basic functionality of the game was completed and we are in a good position to fine tune gameplay and add additional features to the game for Sprint 3.

User Story Specific:

User Story # 13: View Prices of Coins On Landing Page

As a player, I would like to see a landing page that includes the current prices of several cryptocurrencies.

#	Description	Estimated Time	Owner
1	Create a component where pricing information about	3 hrs	Ryan

	different cryptocurrencies can reside and can dynamically update based on an arbitrary choice of coins		
2	Fetch data from the database to serve the current prices of ten arbitrary coins	2 hrs	Ryan
3	Connect that data onto the created module to display onto the landing page	2 hrs	Ryan
4	Sort the listed coins based on price over a base currency (such as USD)	1 hr	Ryan

This story went fairly well. The landing page displays a coin table that displays data coming from the backend and can be dynamically displayed for any amount of coins necessary.

User Story # 14: Landing Page Graphs

As a player, I would like to see a landing page that shows graphs of historical data of cryptocurrencies.

#	Description	Estimated Time	Owner
1	Create a module to graph historical data based on data from our database	4 hrs	Ryan
2	Create a backend module to request the relevant data from the database to use in our graphing module	2 hrs	Ryan
3	Ensure proper error handling when given bad or incomplete data from the database	2 hrs	Ryan
4	Add the created module onto the landing page dynamically based on an arbitrary number of coins	1 hr	Ryan

Graphs for the historical data of coins from our database was successfully implemented, including creating a backend API endpoint for this data as well. Erroneous data is handled appropriately and the graphs are displayed in the correct row of the coin table.

User Story # 16: Joining the Global Indefinite Game

As a player, I would like to be added to the global indefinite game, so that I can play and compete with strangers.

#	Description	Estimated Time	Owner
1	For development purposes, create an entry for the global indefinite game in the database each time the development server is run, via migration scripts	1 hr	Ryan
2	In the registration process, when a user registers for an account, create a GameProfile database entry for the global indefinite game of that user's game information (money, etc)	1 hr	Ryan

An entry for a global indefinite game is seeded into the database upon creation of the database and an accompanying GameProfile entry for the global indefinite game is created for every newly registered user.

User Story # 17: Play Page

As a player, I would like to have a play page that displays the different types of games that I could play.

#	Description	Estimated Time	Owner
1.	Create a new dedicated page.	45 mins	Raziq
2.	Research about button components and create dedicated buttons to navigate to the global game, global timed game, and create game page.	1 hour	Raziq
3.	Write tests to ensure that all buttons are rendered.	30 mins	Raziq
4.	Write tests to ensure that each button redirects the player to the correct page.	1 hour	Raziq

A page for users to navigate to the join, create, and game page was implemented.

User Story # 18: Navigating Current Active Games

As a player, I would like to be able to navigate to any of my currently active games.

#	Description	Estimated Time	Owner
1.	Learn how data is structured in the backend and pull player's active games.	2 hours	Raziq
2.	Create dedicated buttons to navigate to all of the player's active games.	1 hours	Raziq
3.	Research suitable components to group a list of components into pages.	2.5 hours	Raziq
4.	Split the buttons into several pages if the player has many active games.	2 hours	Raziq
5.	Add a search bar to filter active games by name.	3 hours	Raziq
6.	Add a dropdown widget to sort the active games by name (other attributes can also be considered in the future, if needed).	2 hours	Raziq
8.	Write tests to ensure that clicking a button will redirect users to the correct page.	30 mins	Raziq

In the play page, the user can see "cards" that are each associated to a game that the user is in. A search bar, a dropdown menu for sorting, and a pagination bar were also successfully implemented.

The major difficulty faced when completing this story is when doing the API for getting active games. I underestimated the time that I require to learn using our ORM library and integrate the API to the frontend. However, the API was successfully implemented, and future similar work should take me less time.

User Story # 19: Joining Private Games

As a player, I would like to be able to join a private game that has been created.

#	Description	Estimated Time	Owner

1.	Create a button to join a game.	30 mins	Raziq
2.	Research suitable react components to be used and implement a popup page.	3 hours	Raziq
3.	Create a textbox inside the popup page for users to enter the game code.	30 mins	Raziq
4.	Request the game from the backend.	1 hour	Raziq
5.	Add the player into the game at the backend.	1.5 hours	Raziq
7.	Research a suitable react component to be used and display an error message if the user enters a code to a non-existing game.	1 hour	Raziq

A join button was created in the play page, and clicking it will open up a modal window. Users can use the modal window to enter a code and join the game that is associated with the code. The API to check if the game exists and add players into the game was also implemented.

User Story # 20 Game screen

As a player, I would like to have a dedicated game screen for every game that I join.

#	Description	Estimated Time	Owner
1	See title, current cash, current net worth, time remaining.	6.5 hrs	Sam
2	See liquidate button, time span button array	3 hrs	Sam
3	See “share” button (icon) which has a dropdown of shareable code and shareable link	1 hrs	Sam
4	Create deeply-rendered black box visual tests	3 hrs	Sam

The game screen was created in this sprint. It displays current information about the user’s status in the game as well as general information such as whether a game is public or private. This game screen was further split into stories #32-35.

User Story # 22

As a player, I would like to receive accurate game information.

#	Description	Estimated Time	Owner
1	Backend: Create “Get game by id” endpoint, which returns Title, current cash, end date, and coins by user	5 hrs	Blake
2	Frontend: Parse data	3 hrs	Sam
3	Create black box integration tests	1.5 hrs	Blake

This story went well. I worked with Sam on the frontend to make sure he was getting the data in the correct form, and there were limited merge conflicts.

User Story # 30

As a developer, I would like to calculate a player’s net worth, so that a player can see an accurate representation of their standing compared to others.

#	Description	Estimated Time	Owner
0	Dependency on Story #22	0 hr	N/A
1	Backend: Add net worth to the get game by id api call	3 hrs	Blake

As a later separation from story 22, this story went fine. It was just a matter of adding another variable to the transaction with the frontend.

User Story # 24: Buy/Sell endpoint

As a player, I would like to be able to buy and sell mock cryptocurrency and have it attached to my account.

#	Description	Estimated Time	Owner

1	Backend: create “Buy/sell currency” endpoint, which takes a coin id and amount and updates the user’s profile with that coin	3 hrs	Blake
2	Create black box integration tests	2 hr	Blake

This story went well. With limited data objects to be sent to the frontend, it did not need to have that much communication and I was able to resolve merge conflicts and fix.

User Story # 27

As a developer, I would like to create a service/daemon to fetch live cryptocurrency data.

#	Description	Estimated Time	Owner
1	Write service in background for fetching live data and algorithms for parsing data and then storing data into the database	7 hrs	Tyler
2	Search for websites serving real-time data and read API documentation	3 hrs	Tyler
3	Write stubbed implementation for development with randomly generated data	3 hrs	Tyler
4	Unit test ability to parse data	3 hrs	Tyler
5	Integration test service’s ability to parse data together with ability to interact with database	3 hrs	Tyler

The backend web service was built with both production and development/testing versions. A significant testing suite surrounded this service and it was able to be used by the main web server. It runs independently from the main server in a background thread that will die when the server ends.

User Story #29: Coin table

As a player, I would like to be able to view data about a currency (Price, history, % change, amount, min/max price on various exchanges)

#	Description	Estimated Time	Owner
1	See table of information, graph not functional	3 hrs	Sam

2	Can type in amount to amount buyer, buy/sell sends that amount of that coin to backend.	2 hrs	Sam
3	Create modal to confirm a buy/sell transaction	1 hr	Sam

Everything about the coin table besides the graphs was properly implemented this sprint. Every coin in a game will have a corresponding row in the graph that displays the current price, amount the player owns, and buttons to buy/sell the coin. A confirmation of transaction modal was also successfully implemented this sprint.

User Story # 31

I would like to parse the raw data from (potentially) several APIs.

#	Description	Estimated Time	Owner
1	Dependency on Story #27	0 hrs	N/A

The web service implemented the interaction with the nomics api and was able to successfully parse that data without any problems. We wrote tests to verify this behavior and all of the tests are passing.

User Story #32: Game screen title

As a player, I would like to see the title of the game.

#	Description	Estimated Time	Owner
1	See title of game at top of game screen.	30 min	Sam
2	See "access modifier" applied to game title (global, private)	30 min	Sam
3	The URL should reflect the game ID of the current game.	30 min	Sam

This was completed adequately during this sprint. The game title is shown as well as whether the game is public or private. The game title is truncated if it is too long. Tests were written to ensure proper functionality.

User Story #33: Share game button

As a player, I would like to see a button that reveals the 4 digit code to join the game.

#	Description	Estimated Time	Owner
1	See share button at the top of the game screen	30 mins	Sam
2	Clicking the share button will trigger a dropdown That contains the 4 digit "shareable code"	30 mins	Sam
3	Clicking the share button will trigger a dropdown That contains the long "shareable link"	1 min	Sam

The share button was fully implemented this sprint. Upon clicking the button, a modal appeals displaying the game's shareable link and shareable code. Tests were written to ensure that clicking the button actually opens the modal and the full code was displayed even if the code is not one of our existing games' code (for future proofing).

User Story #34: Current cash in game

As a player, I would like to see the current cash I have in a game.

#	Description	Estimated Time	Owner
1	Show current cash in game	30 mins	Sam
2	Format cash so it always has 2 decimal places	1 hr	Sam
3	Create methods that will send calls to the backend that alter cash.	1 hr	Sam

I modified the redux store this sprint so the cash that was returned from the backend can be utilized in multiple components. Methods were created to buy and sell coins. Tests were written to ensure that cash was always formatted properly (having 2 decimal places).

User Story #35: Current net worth in game

As a player, I would like to see my current net worth.

#	Description	Estimated Time	Owner
1	Show current net worth in game	30 mins	Sam
2	Format net worth so it always has 2 decimal places	0 min	Sam
3	Create methods that will send calls to the backend that alter net worth.	1 hr	Sam

I modified the redux store this sprint so the net worth that was returned from the backend can be utilized in multiple components. Tests were written to ensure that net worth was always formatted properly (having 2 decimal places).

User Story #36: Game time remaining

As a player, I would like to see the current time remaining in a game.

#	Description	Estimated Time	Owner
1	Calculate time remaining in a game	1 hr	Sam
2	Create a countdown that displays the time remaining	1 hr	Sam
3	Navigate to the leaderboard page once the countdown reaches 0.	1 hr	Sam

For any game with a time (any game besides the global indefinite game), a countdown was added to the game page informing the user of how much time remained in the game. Once the countdown reaches 0 the screen will redirect to the leaderboard. If a game screen is accessed after the countdown has expired redirection automatically occurs. Tests were written to ensure a private game had a countdown and the global game did not. Tests were also written to make sure proper redirection occurs.

User Story #4

As a developer, I would like to integrate historical cryptocurrency prices into the platform.

#	Description	Estimated Time	Owner
1	Obtain historical data files	0 hr *	Tyler
2	Parse historical data files	0 hr *	Tyler
3	Write script that inserts data into database	0 hr *	Tyler

The historical data script was written and can even be used while the server is running. We gathered lots of data and stored it using the git lfs plugin.

User Story #6

As a developer, I would like to only allow authenticated users to access some routes in Fortune.

#	Description	Estimated Time	Owner
1	Decorate route functions that we want to only be accessible to logged in users.	1 hr	Tyler
2	Persist user's token in localStorage on browser	2 hrs	Sam

We fixed many bugs surrounding the storage of a user's authentication token, and we added the ability to only allow certain routes to require an auth token. Unauthenticated users that try to access pages that require authentication are redirected to a login page.

User Story # 26: Leaderboard

As a player, I would like to be able to navigate to the leaderboard for the current game.

#	Description	Estimated Time	Owner
1	Create “Leaderboard” game page that can be accessed by players	2 hrs	Akash

What went well: The leaderboard page displayed correctly and accessible by the players. The navigating to the leaderboard from the game screen was easier than anticipated, and there was good cross-collaboration from the leaderboard story owner (myself) to the team member that created the game screen.

User Story # 38: Leaderboard

As a player, I would like to see players’ usernames and current net values on the leaderboards.

2	Create SQL queries to fetch users net worth along with their usernames, sort by decreasing net worth to be placed in hiscores page	7 hrs	Akash
5	Ensure smooth UI on leaderboard page	1 hr	Akash

Displaying the current leader of high scores was not too tasking. It involved using a sorting function which worked well. Because a module was used, a smooth UI did not take too much manual coding to achieve. In the future I will be looking to use more modules to develop features as they are already constructed well.

User Story #11

As a player, I would like a choice of initial cash amount each player has.

#	Description	Estimated Time	Owner
3	Check for invalid input and alert the user when invalid input is entered	1 hr	Akash

This story carried over from Sprint 1. There was overlap, and Blake carried out fixing this story early in this sprint. There did not appear to be any blockers as it was a fairly easy fix.

What did not go well?

General:

While we had great separation between users stories and tasks, once it came down to the end of the sprint we had many merge conflicts that caused a lot of headaches and unnecessary work at the very end trying to merge everything together for the sprint review. We have a code review and merging system in place and throughout the sprint many features had pull requests to get merged into the development branch. However, once a developer created a pull request for a completed story, that developer moved on to a new story and their previous pull request never made it through the code review process.

User Story Specific:

User Story # 13: View Prices of Coins On Landing Page

As a player, I would like to see a landing page that includes the current prices of several cryptocurrencies.

5	Create tests to ensure that an arbitrary number of information components are displayed on the landing page	1 hr	Ryan
6	Create automated tests to ensure displayed data matches current data.	2 hrs	Ryan

User Story # 14: Landing Page Graphs

As a player, I would like to see a landing page that shows graphs of historical data of cryptocurrencies.

5	Create tests to check for error handling and displaying a blank graph if errors persist.	2 hrs	Ryan

For both User Story #13 and User Story # 14, I did not make creating tests a priority and as a result they did not get completed.

User Story # 15: Joining the Global Timed Game

As a player, I would like to be able to join the global timed game.

#	Description	Estimated Time	Owner
1	For development purposes, create an entry for the global timed game in the database each time the development server is run, via migration scripts.	1 hr	Ryan
2	When a user selects the global timed game option on the select a game page, create a GameProfile database entry for that user, if not yet already created	1 hr	Ryan
3	Once the global timed game is completed, automatically create a new global timed game and add it to the database.	2 hr	Ryan
4	Test that the creation of global timed game is automated and creates a game	1 hr	Ryan

This story did not get implemented because I spent more time on getting the landing page working and looking clean since it has more of an "outward facing" interface. The global timed game and automation scripts aren't as visible as the landing page and I didn't make it a priority to get done.

User Story # 18: Navigating Current Active Games

As a player, I would like to be able to navigate to any of my currently active games.

7.	Write tests to ensure that correct results are returned when user filters games	1 hour	Raziq
9.	Write tests to ensure that active game buttons are split into several pages.	1 hour	Raziq
10.	Write tests to ensure that active game buttons are sorted properly when the option is used.	1 hour	Raziq

I did not manage to implement these tests. *(See User Story #19 below)*

Also, during the implementation of this story, the sort, filter, and pagination logic was delegated to the backend. Therefore, these tests should be replaced with API tests.

User Story # 19: Joining Private Games

As a player, I would like to be able to join a private game that has been created.

6.	Learn to write backend tests and test that the user was added into the game properly.	3.5 hours	Raziq

I underestimated the time I required to learn and implement the APIs for both stories (but mainly for story 18). As a result, I did not have enough time to implement these tests.

User Story # 21 Coin table graphs

As a player, I would like to see a time series graph displaying the historical price of a cryptocurrency from the exchange(s) of my choice.

#	Description	Estimated Time	Owner
2	Create graphs for each coin	3 hrs	Sam
3	Combine historical data with live data to continuously update the graph for whatever the selected time span is.	3 hrs	Sam
4	Create deeply-rendered black box visual tests	2 hrs	Sam

This story was almost completed by the sprint review but there were some bugs when switching between hour, day, week, month, and year views. Instead of refreshing the graphs when the view was switched, the new data was appended on the end of the graph. I was not able to fix these bugs by the sprint review, so this story was moved to the backlog so the team did not lose points for unfinished acceptance criteria.

User Story # 23: Get Coins by game ID (filtered)

As a player, I would like to modify the time span of crypto data to display (min/hr/day/month/yr), so that I may modify the data view to fit my buying and selling needs.

#	Description	Estimated Time	Owner
1	Backend: Take game ID, time span, and sort by categories (coin name, price, % change, your amount, num per page, page num)	5 hrs	Blake
2	Backend: Return appropriate coins	5 hrs	Blake
3	Frontend: Send time span and above sort by categories	2 hr	Blake
4	Frontend: Send appropriate coins to table	1 hr	Blake
5	Create black box integration tests	1 hrs	Blake

User story 23 was not adequately communicated with the rest of the team, which caused merging problems I wasn't able to solve by the sprint review.

User Story # 25: Liquefy endpoint

As a player, I would like to be able to liquefy all current assets immediately.

#	Description	Estimated Time	Owner
1	"Liquefy" endpoint, which sells all of a users' coins	3 hrs	Blake
2	Create black box integration tests	1 hr	Blake

Liquefy just didn't get done in time, the functionality on the backend exists, but due to testing issues it was not tested, so not pulled in in time.

User Story # 28

As a developer, I would like to serve live cryptocurrency prices (and potentially other cryptocurrency statistics) to the user with websockets.

#	Description	Estimated Time	Owner
1	Create websocket endpoints to serve cryptocurrency prices	5 hrs	Tyler
2	Create stubbed implementation for development	3 hrs	Tyler

3	Manually test that the websocket serves the correct prices and that it is generally working	2 hrs	Tyler
4	Smoke test that the websocket doesn’t crash or stop the server from serving other requests	2 hrs	Tyler

In development the websockets seemed to work just fine. We received updates from the server roughly every 10 seconds (as it should have) but for some reason on our production server it took upwards of 30 seconds for the client to receive this data (even though we use Docker and have identical environments in development and production). However, we suspect that the problem is due to a client-side issue but this still requires further investigation during the next sprint.

User Story # 26: Leaderboard

As a player, I would like to be able to navigate to the leaderboard for the current game.

#	Description	Estimated Time	Owner
1	Create “Leaderboard” game page that can be accessed by players	2 hrs	Akash

What did not go well: I had difficulty in understanding the requirements originally as I thought we only had a global leaderboard. Understanding leaderboard-by-game took some time to understand as well as implement. I changed the way I implemented the leaderboard twice during the sprint to better accommodate future improvements, features, and updates.

User Story # 38: Leaderboard

As a player, I would like to see players’ usernames and current net values on the leaderboards.

2	Create SQL queries to fetch users net worth along with their usernames, sort by decreasing net worth to be placed in hiscores page	7 hrs	Akash
4	Create tests to ensure correct values for hiscores leaderboard are displayed based on games	2 hrs	Akash

The team and I brainstormed our goal for how the leaderboard will be fetching information for a good deal of the beginning of the sprint. After spending a good amount of time understanding how SQL queries work in Postgres, we changed our implementation plan and decided to go another route since we don't have live data yet. This was a roadblock that used up some time that could have been avoided. Because the implementation of the leaderboard changed a few times throughout the sprint, having a functional leaderboard became priority by the end of the sprint, and consequently I did not have enough time to write formal tests.

User Story # 39: Leaderboard

As a player, I would like to receive in-app notifications if there are any significant movements in the leaderboard.

#	Description	Estimated Time	Owner
1	Write scripts to notify user if there are any large movements on the leaderboard	3 hrs	Akash

This story was not implemented as we did not have live data, and a means to provide notifications seemed complex without the leaderboard otherwise functioning as it should in the live game.

***Please note, User Story #37 does not exist due to an error in numbering the stories in the Sprint 2 Planning Document.**

How should we improve?

For Sprint 3, we plan on enforcing a rule that a fully completed story's pull request must be submitted to our code reviewing Slack channel before a developer is able to move onto a different story. This will help circumvent having halfway done stories by the end of the sprint and hopefully keep progress moving forward story-wise. We do not want to be in the position again where we're trying to merge in stories that aren't necessarily fully completed just so that the work that is actually completed appears in the sprint review. Staying on top of stories can also potentially decrease the amount of merge conflicts, as everyone should be staying up to date with the development branch for only one story.

During the sprint review we also got the comment that we really need to sit down and rehearse our sprint review beforehand and work on getting through the acceptance criteria as quickly and efficiently as possible. That is a good thing to keep in mind for the final few days leading up to the sprint review, but it can also be applied to the development process as well. We were more conscious of following the acceptance criteria during Sprint 2 as opposed to Sprint 1, but in some cases we weren't able to necessarily show that an acceptance criteria was met. A very good example for this would be with developing the backend. It is sometimes difficult to show that an API endpoint is operational if the frontend does not outwardly show the result of the backend call. Therefore, during development we need to keep in mind how we can present the acceptance criteria and possibly code in extra measures just for demonstration purposes (such as writing to console, showing backend responses on screen, etc.)