FINEX

Sprint 1 Retrospective

Team 8: Hugh Bromund, Peyton Williams, Aditya Naik, Daniel Joseph, Nathan Ashta, Niyati Sriram

What Went Well?

Overall:

We completed a majority of the initial setup for our project and gave ourselves a strong foundation to build on in the future. We fully integrated the account authentication which gives us a strong base to start building out the budgeting part of our software with sprint 2. We also built out a majority of the stock tracking features during this sprint and created a system for finding stocks by name or ticker which will help us build in the future.

User Story #1

As a user, I would like to be able to register for an account

Description	Estimated Time	Owner
Create a UI page/form to allow users to register for an account	3hr	Hugh
Handle routing from the main page to the register page	1hr	Aditya
Integrate the register page with the rest of the application	5hr	Hugh
Insert user information into the database	3hr	Niyati
Handle registration logic and validate the request	3hr	Peyton
Include unit tests for registration	5hr (Each)	Peyton, Niyati, Hugh, Aditya

Completed: The user is able to register for an account, and that account is connected to our backend. If the user attempts to create an account with a username that already exists they are informed and prompted to try logging in to that account.

User Story #2

As a user, I would like to be able to login to my account

Description	Estimated Time	Owner

Create a UI page/form to allow users to login for an account	3hr	Hugh
Handle routing from the main page to the login page	1hr	Aditya
Integrate the login page with the rest of the application	3hr	Hugh
Handle login logic an validate the request	5hr	Peyton, Niyati
Create unit tests to ensure functionality	5hr (each)	Hugh, Aditya, Niyati, Peyton

Completed: Users are able to log in to their account and stay signed in during the course of their use session. If they attempt to log in with an incorrect username or password they are prompted to try logging in again. After logging in they are redirected to the user account screen where their account information is shown to them.

User Story #3

As a user, I would like to be able to change my associated email address

Description	Estimated Time	Owner
Create UI form which will allow for the input of a new email address	5hr	Nate
Integrate form with the rest of the application	3hr	Nate
Connect UI form with the backend	5hr	Nate
Handle backend logic to connect with the database to change the email address	5hr	Peyton, Niyati
Edit the database to reflect the change in user email address	3hr	Niyati
Create unit tests to ensure functionality	5hr (each)	Nate, Peyton, Niyati

Completed: On the user account screen, there is a button to change the email address associated with the account. Upon changing the email, the user's email is updated in the database and in the application. The form is connected with the backend and the backend makes that request to the cloud.

User Story #4

As a user, I would like to be able to search for a stock

Description	Estimated Time	Owner
Create input for user to search for stock by name or symbol	1hr	Aditya
Obtain stock names matching user input	5hr	Peyton
Create routing that returns autofilled list in backend	1hr	Peyton
Display list of potential search results dynamically from user input	5hr	Aditya
Create unit tests to ensure functionality	5hr (each)	Peyton, Aditya

Completed: The user is able to search for a stock and the stock ticker symbol is autocompleted as intended. An error page is also displayed if the user redirects to a stock that does not exist.

User Story #5

As a user, I would like to be able to view information about a stock

Description	Estimated Time	Owner
Obtain stock data from API	3hr	Daniel
Parse JSON for required information	2hr	Daniel
Set up routing and project structure	6hr	Daniel
Decide which API to use	2hr	Daniel
Display information on stock page from parsed JSON	2hr	Aditya
Create unit tests to ensure functionality	5hr (each)	Daniel, Aditya

Completed: The user is able to retrieve information based on the given stock such as high, low, volume etc. and it is displayed on the page. The routing backend has proper routing in order to service these requests as well, and the frontend parses the JSON correctly.

User Story #6

As a user, I would like to be able to view a stock's volume

Description	Estimated Time	Owner
Create routing for viewing stock volume	1hr	Daniel
Make API call, parse JSON, and return required stock's volume	2hr	Daniel
Display information on stock page from parsed JSON	2hr	Aditya
Create unit tests to ensure functionality	5hr (each)	Daniel, Aditya

Completed: The stock's volume is displayed in the daily summary section as a part of the stock information page which is routed to the associated pages. This is updated dynamically based on the stock, who's data is retrieved through an API call to the backend. Additionally, the volume may be viewed as a tooltip as part of the graph and updates based on the date hovered over.

User Story #7

As a user, I would like to be able to view a stock's trend/range

Description	Estimated Time	Owner
Create routing for viewing trend/range	1hr	Daniel
Make API call, parse JSON, and return required stock's trend/range	2hr	Daniel
Display information on stock page from parsed JSON	2hr	Aditya
Create unit tests to ensure functionality	5hr (each)	Daniel, Aditya

Completed: The user is able to see the open and close data that represents the range for each data point on the graph. They are also able to see the global range over the course of the entire time period at the bottom of the page.

User Story #8

As a user, I would like a graphical representation of stock prices and how they are changing over time

Description	Estimated Time	Owner
Create routing for viewing stock graph	1hr	Daniel
Make API call(s) format graph data, and return	2hr	Daniel
Format graph to display to the user	10hr	Nate
Create unit tests to ensure functionality	5hr (each)	Daniel, Nate

Completed: The backend has an endpoint to get the required information for a stock graph. The frontend calls that endpoint and then parses the information and remaps it in order to create a graph that is aesthetic to the user.

User Story #9

As a user, I would like the application to have an easily navigable UI

Description	Estimated Time	Owner
Verify that you can route easily from any component to another	2hr	Aditya
Create a Top Bar that is present on every page	4hr	Hugh
Handle any routing errors	3hr	Aditya
Create unit tests to ensure functionality	2hr (each)	Hugh, Aditya

Completed: The user is able to navigate using a very clean-looking top bar. This navigation bar can be used to route to different components, and if the user tries to navigate to a component that is not available, the user will be directed to a custom 404 page. The routing is very seamless as a result.

User Story #10

As a developer, I would like the user password and personal information to be encrypted to protect the privacy of the user

Description	Estimated Time	Owner
Implement password-encryption within the database	5hr	Niyati

Test and verify encryption methods to ensure the most secure option is used	5hr	Niyati
Severely restrict database access permissions to increase security	5hr	Niyati
Create unit tests to ensure functionality	5hr	Niyati

Completed: The users' password is hashed with 10 salt rounds and then stored in the database using the bcrypt algorithm. When a user logs in, their entered password is hashed and compared. Database access is restricted to those with proper login credentials.

User Story #11

As a developer, I would like the server to be able to store user data in a database

Description	Estimated Time	Owner
Create MongoDB database and push to cloud via Atlas	5hr	Niyati
Decide on and implement database organization and design	3hr	Niyati
Integrate database to backend code	10hr	Niyati, Peyton
Create methods to edit database based on user activity	10hr	Niyati, Peyton
Create unit tests to ensure functionality	5hr (each)	Niyati, Peyton

Completed: The database contains all user information, including a unique username and email. It is deployed on MongoDB Atlas for global connectivity, connected to our backend and methods have been written to edit and update all relevant database information.

What did not go well?

Overall:

One of our issues was keeping our standups concise and on-topic. In the future, we need to focus on being more efficient by letting everyone share their updates before we open up to a team discussion on specific implementations. We also had some issues with time management and allowing ourselves enough time in case certain things take longer than expected. Doing so will ensure we do not hold up other team members who are waiting on tasks to be completed before they can begin their own. In addition, our code standard suffered as we each had different preferences be naming, semicolons, and many other small things.

User Story 1: Given that the frontend UI is implemented correctly, when the user types in a weak password, then the frontend will alert the user and suggest ways to make the password stronger.

Not Completed: We didn't realize that we were being graded on the acceptance criteria and as a result overlooked that this was an acceptance criteria for the user story

User Story 2: Given that the backend logic is implemented correctly, if the user repeatedly fails login attempts to the same account, the user will be locked out.

Not Completed: Again, we didn't realize that we were being graded on the acceptance criteria and as a result overlooked that this was an acceptance criteria for the user story.

User Story 7: Given that a stock is trending upward or downward, when the user looks up that stock, then the UI will change color to match the current trend

Not Completed: Again, we didn't realize that we were being graded on the acceptance criteria and as a result overlooked that this was an acceptance criteria for the given sprint.

How can we improve?

Over the course of the first sprint, one of our biggest challenges was with roadblocks. This was our biggest issue because so many parts of both the frontend and backend required other parts to be completed. During the course of the sprint aligning these timelines was a large struggle for our team. Members of our team were blocked by each other for a large part of our sprint causing the amount of work we could complete to be reduced. For Sprint 2 and beyond we are planning on spending more time planning out the timelines to try and make sure that tasks are completed in a more logical order.

An issue we faced was that we did not have a unified coding standard for the frontend or the backend. This made some of the code difficult to read and difficult for other people to pick up other people's code since the readability was not very good. As a result, we decided that going forward, we will use camelcase for all variable names and unified our other styles to increase our readability.

Another challenge for us during this sprint was our weekly standup meetings. During these meetings, it was often easy for us to get off track and lose sight of the goal of the meeting. Our team was friends with each other before coming into this project and so it is very easy for us to get distracted by other things going on in our classes. For Sprint 2 and beyond we are going to work on staying more focused on the project during our weekly team meetings.

One area of improvement for us would be to manage our time better. Throughout sprint 1 we felt a sense of urgency to make progress towards the end of weeks. This feeling may have been prevented by working on features every day, rather than simply rushing production on the weekend. Granted, some members had other priorities on various weeks such as midterms and competitions. Knowing this, the team should have planned assignments and responsibilities slightly differently.

In addition, we did not fulfill our acceptance criteria fully. We all forgot about the criteria and failed to validate we had completed all of them. In the following sprints we can improve this by frequently reviewing our acceptance criteria and ensuring that our product fulfills it.

Overall Sprint 1 had several places that we could improve upon in the future. Having a better plan and working to integrate different people's work will serve us well in the future. Additionally, keeping our standup meetings short and on task will help keep the whole team organized. Finally, making sure that during the course of the sprint all acceptance criteria are completed on time will help us in our presentation at the end of the sprint.