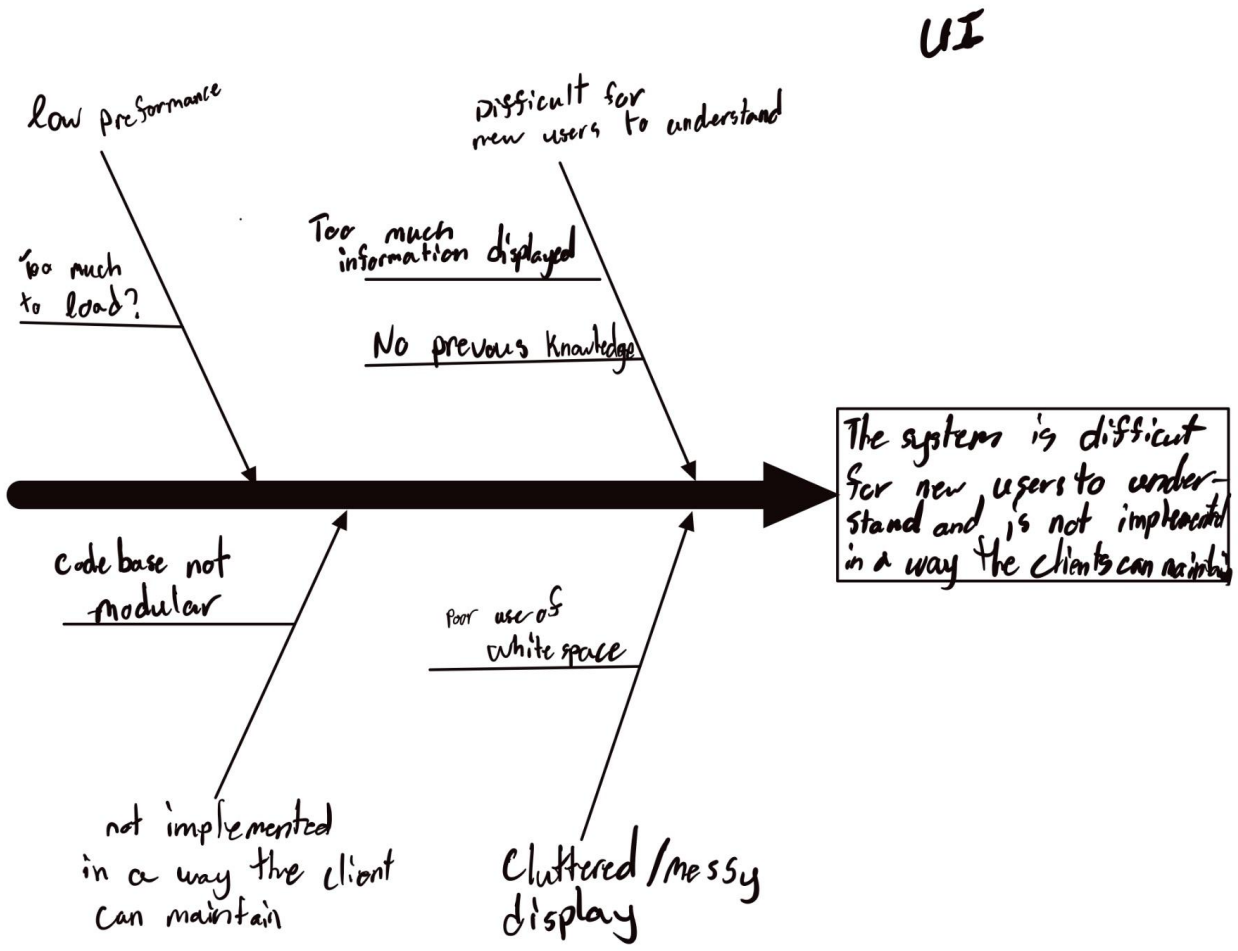
# Pelican Innovations Root Cause Analysis

The following Root Cause Analysis and Fish Bone Diagrams were constructed based on feedback received from the client and during team discussions internally.

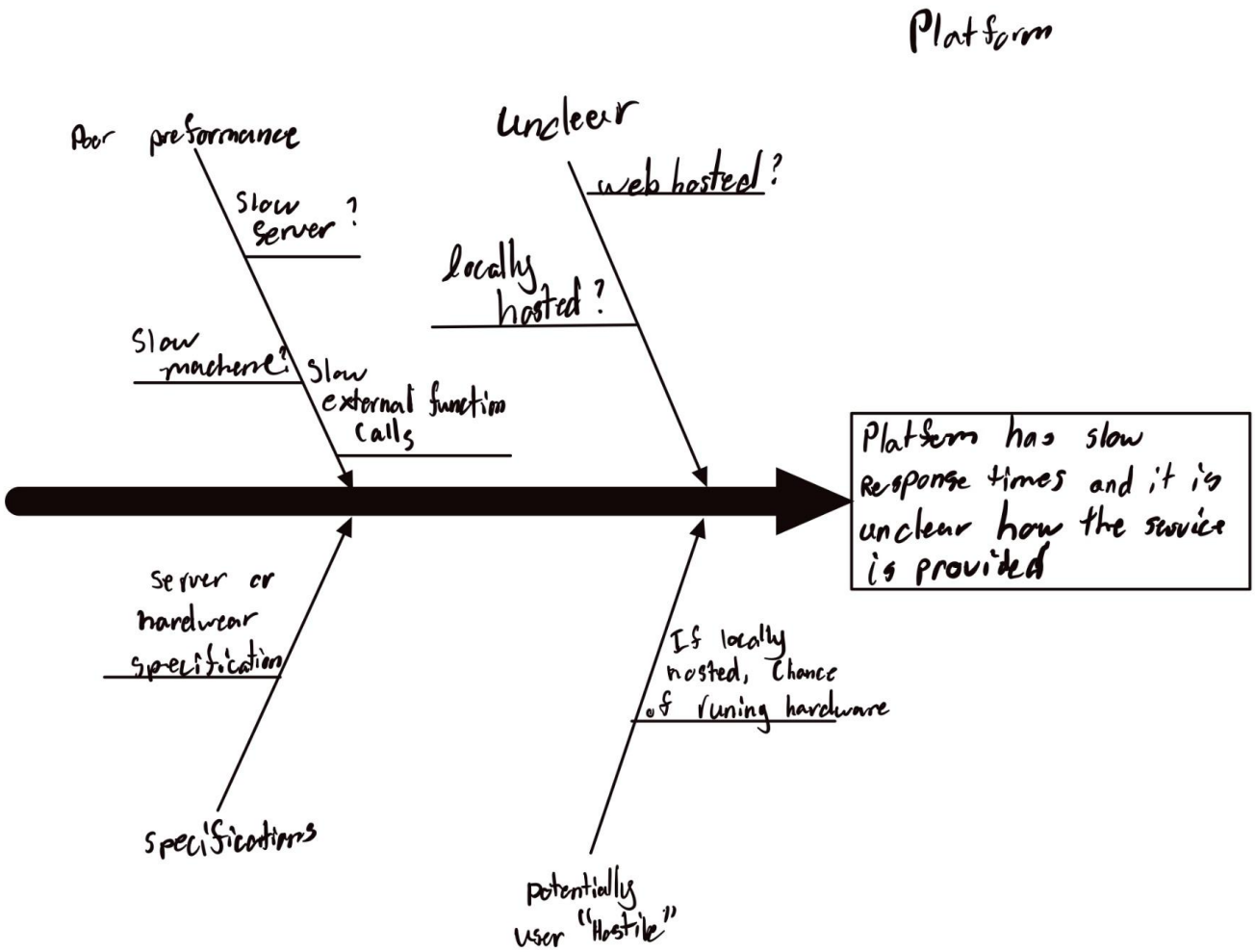
## UI

The UI for a web application is incredibly important for the way users perceive and use it. When performing a Root Cause Analysis on our web application, we noticed that users may get confused about how our stock application looks in comparison to other stock trading applications. As a result, we changed our buy/sell interface to mimic that of other popular trading options so users will be familiar with our interface from the start.



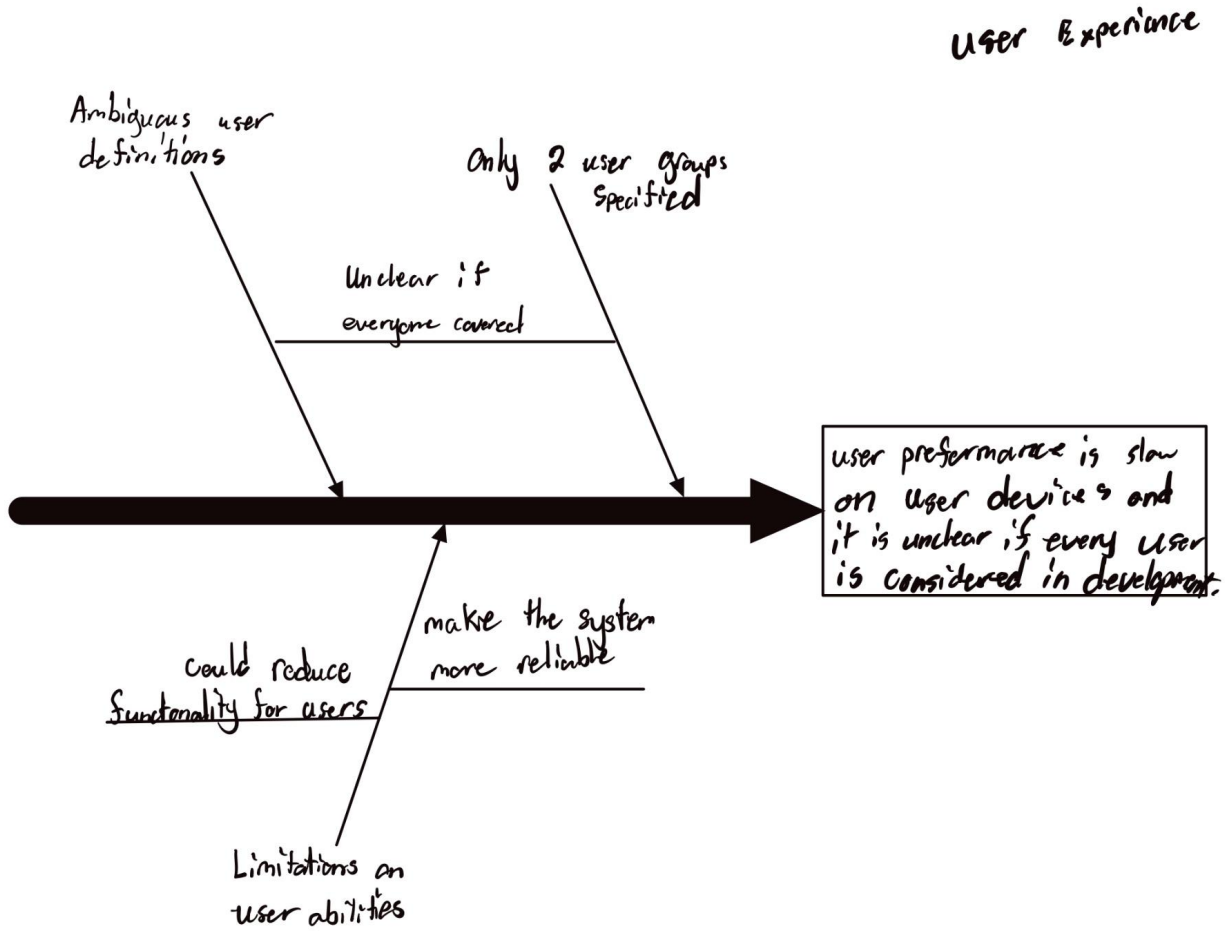
# Platform

It was never clear from our requirements document what type of hosting Pelican Innovations was planning for our stock trading application. By us not specifying this metric, this could result in a multitude of challenges, such as a poor performing web application, an unsecure “hostile” web application, and not specifying to users what kind of hardware they will need to run our web application. As a result, we have decided that we will host this web application from one of our team’s computers to be able to control these metrics, and the application will be very lightweight for any operating system.



# User Experience

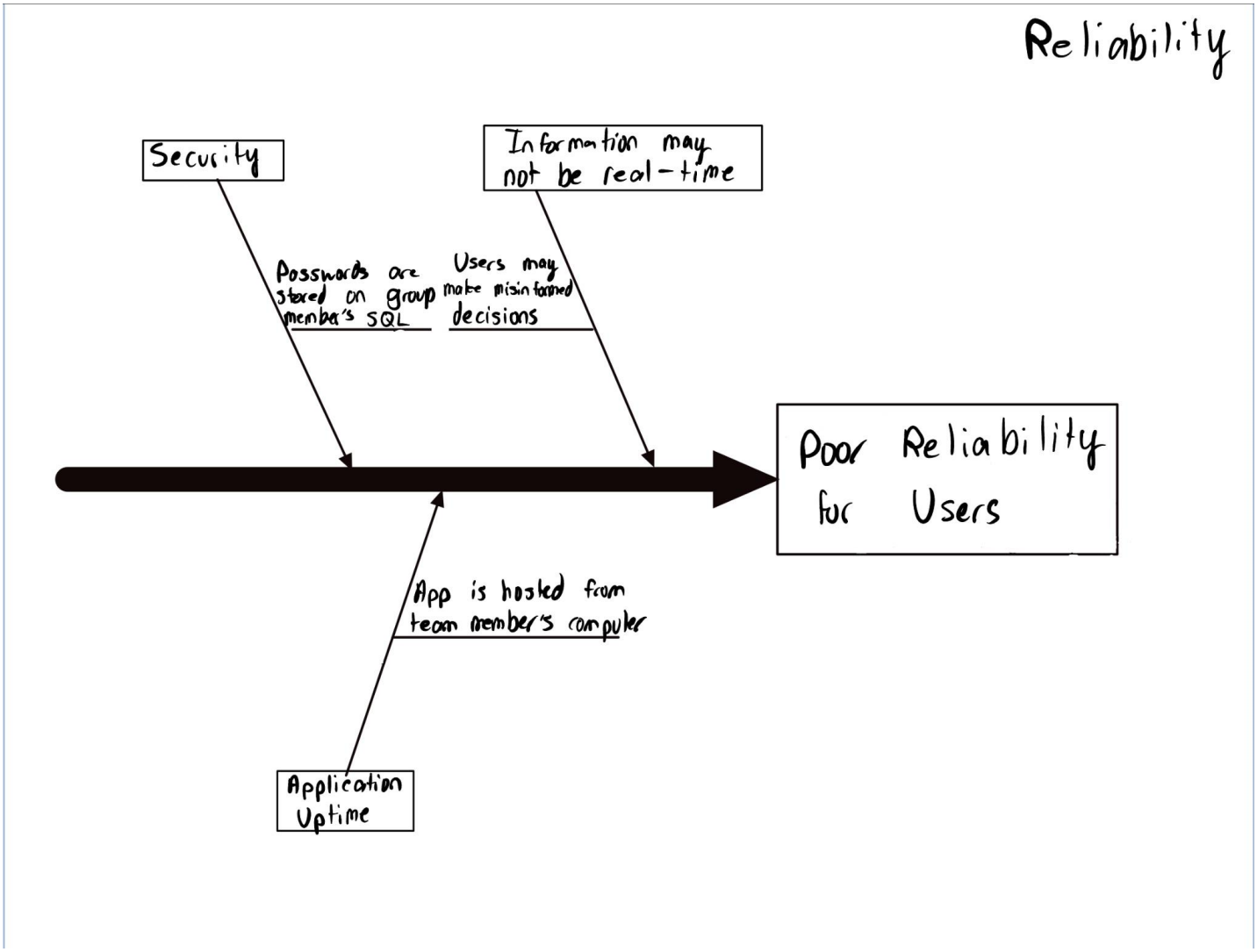
In our requirements document, we mentioned that we would have multiple classes of users that use this application, being “advanced” vs “beginner/youth”. But we have received some feedback on this requirement and are adjusting it for the following reasons: This could split our user base and put limitations on certain user’s abilities, which we do not want, and it is unclear if every user is encompassed by those two user groups. As a result, we are unifying the user groups into one generic user group for all users, as our application is mainly intended for one user group - being “beginners”.




.

# Reliability

Upon discussing further with the team members as the project continued development, it became clear that there could be some reliability concerns within our application regarding our method of implementation. Mainly being that most stock APIs do not support real-time stock data unless paid for, which could become detrimental for new users trying to invest and follow the current market. Additionally, there was the issue of parts of the web application being hosted on a team member's computer, mainly the SQL server storing user information. As a result of these vulnerabilities, Pelican Innovations made sure to use a real-time stock API called Finnhub: https://finnhub.io/ . And to protect the SQL database from potential SQL injection attacks using software protections built into our application.



# Dependencies

As the project progressed during development, there were some issues with the reliability of some of the hosts within our web application. Three external APIs are depended on for news data, real-time stock data, and for the plotting of stock graphs. As well as the use of an SQL server for containing user information. If any of these hosts go down, the functionality of key features within our application would not work. As a result, certain changes were made to the use of APIs within our application: News articles are not retrieved live anymore, the news API was used to generate a file with hundreds of articles, and the web application pulls articles from that list. Next, error handling was done for if the other APIs went down, such as if the graphing plot API is down, the graph simply does not display, rather than the web application crashing for instance.