Feature Name: Motivational Quote

Developer: Devin Kothari

Developer Submitted: 03/18/2024

Peer Reviewer: Jack Pickle

Review Completed: 03/18/2024

Review Updated: 03/23/2024

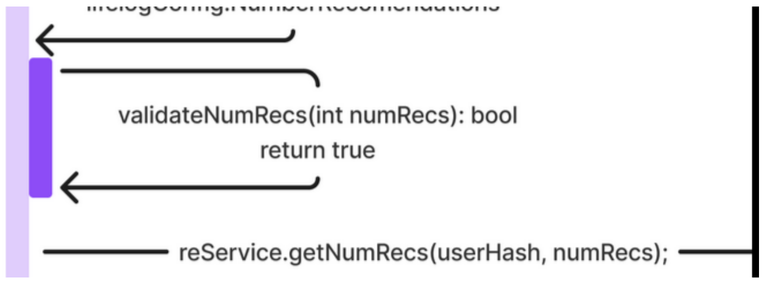
Major Positives:

1. Diagrams are succinct, clear and easy to understand.
2. Excellent normalization of data model
    - Trade off - lowers amount of duplicate data, could lower performance, developers choice
3. Response object contents are clear, easy to follow what should be happening.
4. Good idea to break validation down into helper methods
    - Show method signatures
5. Current design validates quote and author separately, accounting for unique error cases where author can be correct and quote isn't
6. Diagrams highlight that response objects are not being passed through, and are instead being evaluated before sending a new response back.

Major Negatives:

1. Missing Lifelines
    - Include Response, Logger, User, View, motivational quote.js, middleware, controller, motivational quote repository, mysql.
    - Including these lifelines will highlight where things come from, and improve the clarity of what is happening
2. System Lifeline
    - Unclear, we have more system understanding, system should be replaced with aforementioned lifelines since we have quantified what our system is
3. Missing Helper function activation blocks.

- Ex:


validateNumRecs(int numRecs): bool
return true
reService.getNumRecs(userHash, numRecs);

- Including these blocks highlights that the lifeline is still active, and is calling a helper function that is only active for part of the lifeline's lifetime

4. Include logging details
    - Detailed information about log: userhash, level, severity, and message.
    - This information is outlined in Lifelogs BRD, but your design documents are not as verbose.
5. User type
    - Anonymous, unauthenticated users can utilize this feature
6. Refer to other sequence diagrams
    - When utilizing data access, refer to data access success diagrams.
    - Same goes for logging, etc
7. More specific with method signatures
    - For example, what does validQuote take in? I understand that it "validates the currentQuote and currentAuthor, but what is it taking in? Are those part of an object together, or separate?

Unmet Requirements

1. Success Case - Quote Update time
    - Detail a success case that demonstrates that the quote should update at midnight.
2. Detail quote recycling process
    - Indicate somewhere in design how the process will.
3. Logging from front end success
    - You describe that you will log successes when the quote has displayed to the main page view for the user, but this is not specified in either success diagram.
    - May need a logging web service to accomplish this goal, ultimately should focus on the MVP
4. Placeholder message
    - Should be a failure outcome, as intended functionality of motivational quote is not accomplished, even if it is handled properly by the system.

Design Recommendations

1. **Identify what your personal design goals are**
    - Are you prioritizing extensibility in your system? Security? Reusability? Regardless, your personal goal is unclear. Team peace decided in our design phase to prioritize extensibility, so I would recommend making this your priority.

    Pros:

    - Easier design review, can evaluate design against general ideals.
    - Improves overall developer comprehension of your design, allowing possible integration of motivational quotes into other services of the Lifelog application.

    - Saves time in the long term, as having a more comprehensive design phase reduces aggregate refactoring time while developing.

2. Prioritize extensibility
    - Make motivational quote service return some type of collection, enabling the return multiple quotes, allowing the motivational quote feature to adapt to changing system requirements.

   Pros:
    - System will be more extensible, allowing for potential gold plating like an animated, rotating quote on the homepage to further entice the user to pursue their goals.

   Cons:
    - Implementation of collection could slow the development process, and hinder development of a MVP, which is the team's main priority.

3. Enforce business rules in a MotivationalQuoteService file
    - Separate your business logic and your data access (SQL statements) into separate classes inside of Motivational Quote.

   Pros
    - Clear enforcement of single responsibility principle (SOLID).
    - Better Open-closed principle adherence, by allowing easier additions of more business rules without being forced to modify already working sql logic.
    - Matches Lifelog's layered architecture detailed in a high level design document.

   Cons
    - Passing information through more different parts inside of the backend.

Test Recommendations

1. Business rules of quote could be tested:
    - Author names shouldnt include special characters like “*”, expand on this.
    - Quotes could be limited to alphanumeric characters.
2. Test amount of quotes, to ensure recyclability is possible.
    - Need 188 quotes, check that your table includes this many quotes (SQL query to count).