Peer Review

Feature Name: Lifelog Reminders

Developer Name: Devin Kothari

Reviewed by: Yoshiki Yarlagadda

Date Developer Submitted: April 28th

Date Review Completed: April 28th

Date Review Approved: April 28th

Major Positives and Negatives:

- Positives:
    - Detailed design that laid out all the steps needed for the feature.
    - There are clear designs for the front end user action, which would make front end design and development more straightforward.
    - There is a clear separation of concerns within a class; the main functionalities of a class are broken into smaller helper functions that are more maintainable and understandable.
- Negatives:
    - The design only works if the system is running 24/7 even when there are no potential users on the system, due to the way that you send reminder emails on a recurring basis. This leads to an increase in costs.
    - There is a lack of separation of concerns when it comes to the javascript functionality for sending reminders. You can make more helper functions that handle different functionalities making the code more maintainable and readable.
    - There are instances of functionalities that are missing when creating the data and writing to the database throughout the LLD, which may create confusing scenarios when developing the feature.

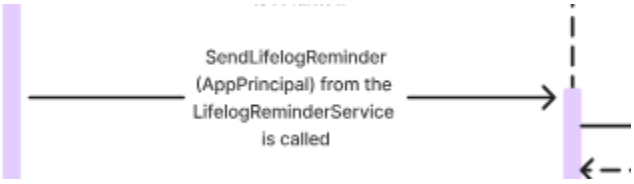
Unmet Requirements:

- There are no unmet requirements.

Design Recommendations:

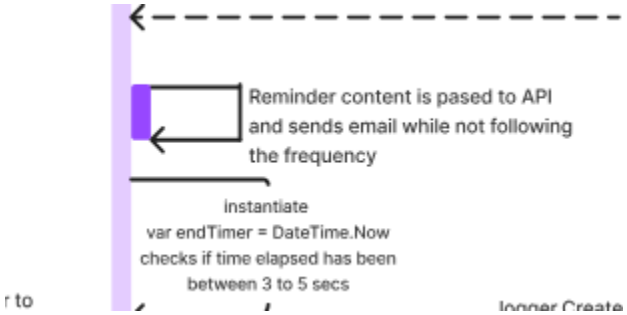
- You need to be more specific as to which API is being used here to send the emails:

- You should specify the data types of the function parameters in the back end. For example, here you should specify the data type of the appPrincipal

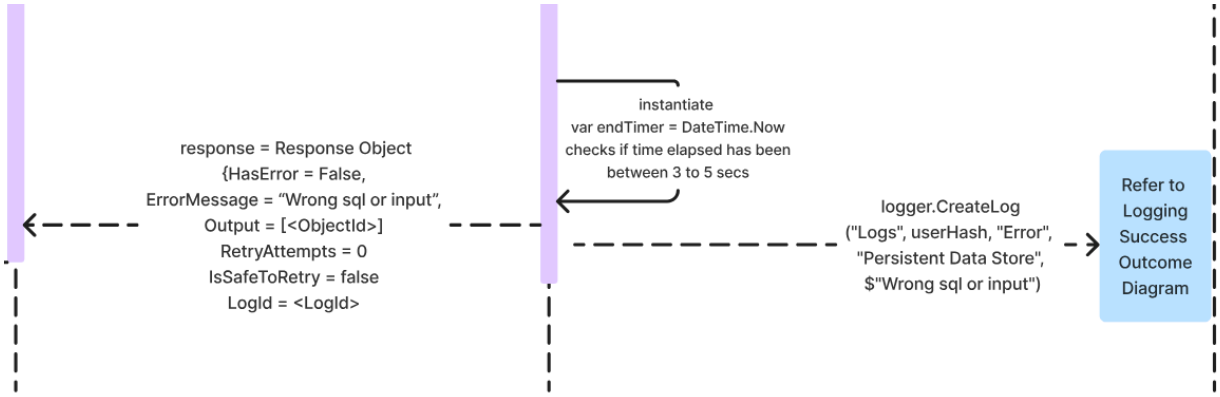


- The function ReadAllLLIInDB from the LRRE Repo layer, this is redundant code since we already have a function that Reads all the user's LLI data in the system.

- You need to show in detail what is happening when checking for a failed frequency of an email. There is no way to tell if the email is being sent at the right frequency to the user. You should implement a functionality that requests the API for information regarding the email to be sent, and check that the email was sent out at the right time with the right content

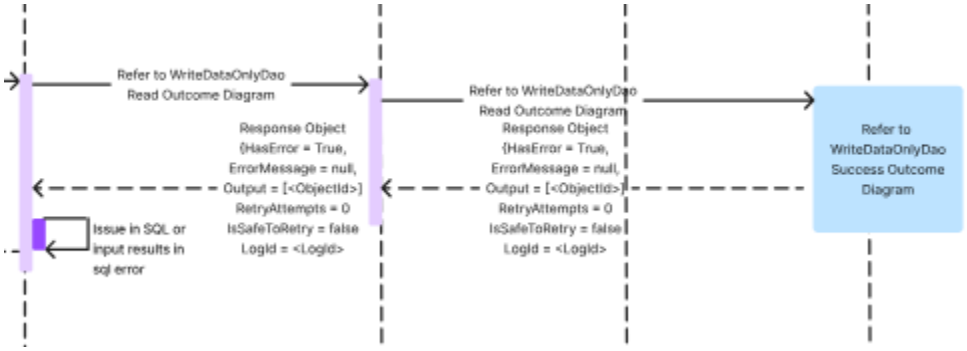


- The response object in failure outcome 1, is supposed to return a True for the HasError attribute. Instead, you say it returns a False.



- You have some confusing namings, do not abbreviate unless it is a common knowledge. What is "LRRE"?
- You need to be more specific with what you mean by "input", is it the SQL? Is it the content, frequency? Be more detailed including the data type the function takes in

- You need to be more detailed as to what is happening at the repo layer, there is no indication of any function from the SqlDAO being implemented in order to write data. You need to use the createData(sql) function to do so in our system.



Test Recommendations:

- You need to test your backend more thoroughly. Every backend function should be tested. Here are some tests that you should be implementing.
    - Service Class
        - What if the app principal is null. Try passing a null appPrincipal into your functions, they should be returning an error.
        - What if the app principal is unauthorized. Try passing an app principal with an invalid user role into your functions, they should be returning an error.