Project Plan Document

Matt Borle
Guy Kaminsky
Danielle Macdonald
Shawn Paul Mountenay
Wee Jia Quan
Harman Sahota

COSC 310: Software Engineering
March 5, 2021

Table of Contents

1.0 Project Description

Not Your Average Life Coach is a chatbot capable of holding sustained conversations and responding differently depending on your input. The goal of the bot is to give you life advice. The advice is poor by design. The user plays the role of someone in need of advice, help, or just someone wanting to have a general conversation. The bot will reply in the most disingenuous way it can, often giving the opposite of what's needed. The bot can be passive-aggressive or simply rude, and will never give real advice beyond the occasional unhelpful adage.

GitHub Repository:
https://github.com/Take-Your-Money-Corp/not-avg-lifecoach

2.0 SDLC

2.1 Rationale

Agile / Integration and Configuration:

With agile development being commonplace in the real world, agile development was of interest to our group. After analyzing our specific app's requirements, that is, creating a chat app, our interests were validated with logic. The ability to slice our app into achievable increments will allow us to pivot and shape our app as it grows. And doing each step in the Software Development life cycle as a natural need occurs will give us much more flexibility than a plan-based approach. That being said, building a chatbot from scratch without using any libraries would likely be futile. That is where the integration and configuration software engineering methodologies come into play. Thankfully, modern frameworks such as Express and VueJS will give us the power to build our client app and API respectfully. Furthermore, chatbot packages on NPM (Node Package Manager) will aid us in making the most of this project.

2.2 Task Breakdown

Specification

1. Define requirements
    a. Determine user requirements
    b. Determine software requirements
2. Check the validity of requirements
3. Define functionality of the software and constraints on its operation

Development

4. Write unit tests
5. Produce the software to meet the specification (define components)
    a. Implement pair programming
    b. Frequent informal reviews of code
6. Create this structure into an executable program (the whole system)

Validation

7. The software must be validated to ensure it meets the requirements
8. Formal code review when large components are completed
9. Identification of unstable components/functionalities

Evolution

10. The software must evolve to meet changing user and market requirement
11. The software must be modified to fix the errors discovered after delivery
12. Executing test cases that are defined from the specification

3.0 Requirements

User requirements:

- CORE: The ability to send a message, and have the bot reply in a meaningful way.
- CORE: The length of input and its syntax should be ambiguous. (any input is valid)
- CORE: The ability to hold a conversation without the bot hanging up and failing to reply.
- CORE: A 30 response long conversation must be possible.
- EXTRA: The bot should respond with useful advice consistently.

Software requirements:

- CORE: Take a text input (possibly voice?) and output a response.
- CORE: The response should depend on the input and be meaningful.
- CORE: The bot should not hang for several minutes while responding or time out.
- CORE: UI should allow basic functionality to any user.
- EXTRA: UI should be pleasing.

4.0 Work Breakdown Structure

The following work breakdown structure (WBS) is a tree-like structure that allows the chatbot project to break down into smaller and more manageable portions. Overall, the WBS comprises six parent nodes that need to be divided and conquered. In this case, each parent node divides into three levels. Each level provides more detailed work to be done to complete the parent task. Furthermore, at the bottom of each section, you will find the estimated number of hours needed to complete each task. Below that, you will see the actual hours it took. On the right-hand side, there is a legend that maps a letter to everyone's names. The letter is found on each bottom node to show who is responsible for each task.

Refer to 'Project Documentation' section within README file

5.0 Gantt Chart

The following Gantt chart is a bar chart that is used for planning and scheduling each project task. The horizontal axis of the Gantt chart represents the time it takes to complete the tasks. On the other hand, the vertical axis represents the tasks that need to be complete. Furthermore, the Gantt chart organizes each responsibility into groups as per the software development life cycle. Each bar in the Gantt chart contains several properties. These properties are a start and end date, plus the number of hours to complete a task. Lastly, each bar is color-coded for ease in readability and matches the project milestones on the left-hand side.

Refer to 'Project Documentation' section within README file

6.0 Project Limitations

In general, the chatbot can produce a sample output of good feasible conversation lasting a minimum of 30 turns. That being said, this chatbot has some limitations. Firstly, the chatbot will not produce the answer you expect. This chatbot uses natural language processing, which is a machine-based algorithm that builds a model based on the data given. Since our dataset is currently not large enough, the chatbot may repeat answers if the conversation is long enough, which means that the chatbot produces textbook-style conversation.

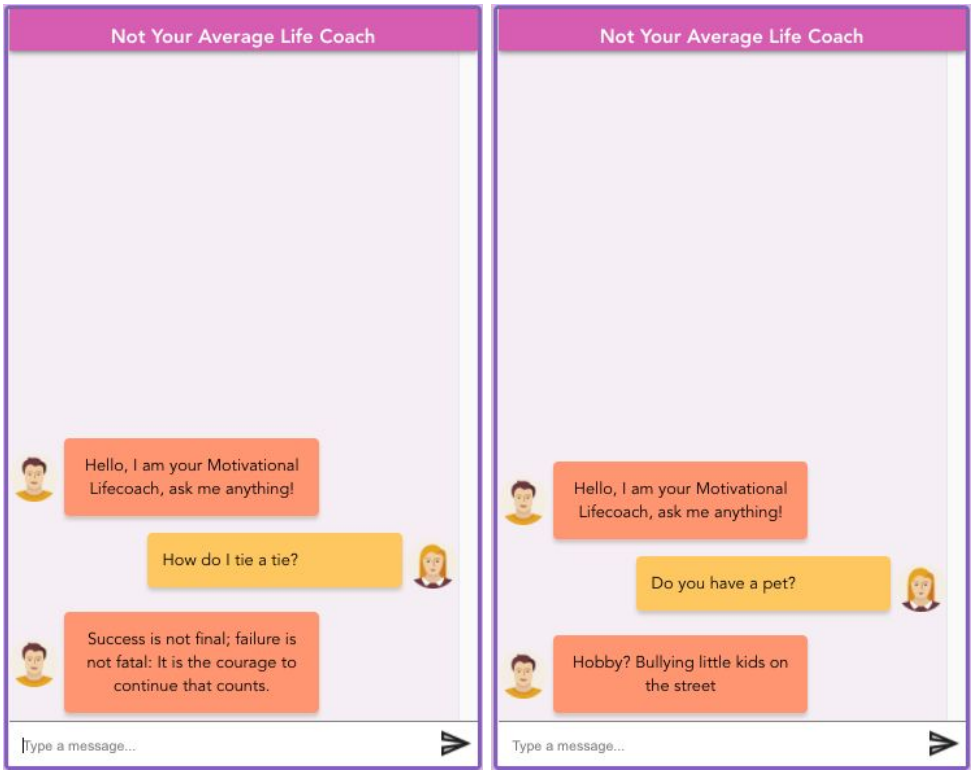
Given that information, the chatbot cannot handle incorrect spelling, as there is no natural spell checker attached. Also, the chatbot does not handle synonyms properly and treats them as

unique words. Lastly, the program does not remember the conversations it holds after refreshing the page.

7.0 Sample Output

Refer to the section within README file

7.1 Improper Conversation Handling



In both of these instances, the bot responds to user input with an invalid response that does not match the expected output.

8.0 Log of Meeting / Documentation Hours

1. G & D - Feb 10th - 0.5 hours
2. Everyone - Feb 10th - 3 hours
3. Everyone - Feb 15th - 3 hours
4. Everyone - Feb 17th - 3 hours
5. Everyone - Feb 20th - 2 hours
6. Everyone - Feb 24 - 3 hours
7. Everyone - March 1 - 2 hours
8. Everyone - March 2 - 2 hours
9. G & D - Mar 4th - 1 hour
10. G & D - Mar 9th - 1 hour