The Benefit Bot

Everyone needs a friend

University of British Columbia Okanagan
COSC 310, Software Engineering
PLANNING DOCUMENT

Team Information

Jean-Philippe Abadir
Rohan Chauhan
Keith Nicholas
Jan Reisler

Mission Statement

Have you ever wondered what it would be like to have a friend? No not an acquaintance you happen to encounter at the supermarket only to have them stroll by you pretending like they don't see you, no not that friend that you think you have but haven't had contact with in 3 months - we're talking about a *real* friend. Despite a whopping 7.7 billion people in the world, some people to this day have either neglected a meaningful relationship or just cannot seem to get one - that is where we come in. Enter The Benefit Bot, an intelligent chatbot whose only purpose in life is to help you become a better you, like a true friend would. The Benefit Bot allows you to practice your dusty social skills and hone them into something you can truly be proud of, and apply in real life situations. The way Benefit Bot works is such that you engage in a conversation with it, and your main goal is to have a happy outcome; however, this being said, the chatbot has consequences, for example you can make the bot sad, angry, etc. and these actions will lead to their respective outcomes. From scratch, the team behind Benefit Bot will be creating this software utilizing scrum methodology and java. We hope to see the world shine a little brighter, and more friendships formed thanks to The Benefit Bot, because as the team at Benefit Bot says - everyone needs a friend.

Repository URL: https://github.com/Rohanc98/COSC-310-Assignment-2

SDLC Selection & Rationale

We chose the scrum model for our SDLC. The Scrum model allows us to have flexible requirements which can be changed easily and allow for various stages of the project to be modified or removed. It also conveniently prioritizes important tasks to finish first allowing quick prototyping and a functional program. Small features and bug fixes are implemented/polished out later. With frequent group meetings everyone is also aware of the progress and issues of the project.

Phases of Scrum

Product Backlog:

- The program should be able to carry conversion for at least 30 turns
- The program should be able to handle bad user input & spelling

- Need to determine an algorithm that will be used for pattern matching and responses.
- The program should be able to give us an indication of what the final outcome will be.
- A GUI for the chatbot.

1st Sprint Backlog by priority (highest to lowest):

- Develop a skeleton program that acts as a template for the whole project (eg: crude command line app that accept user input and return sample output.)
    - Create a basic main loop for the program.
    - Determine flow of the program from input to output.

- Need to finalize high level implementation of the algorithm used for pattern matching & context recognition.
    - Determine appropriate data structure to store the answers
    - Decide between implementing our own code or using API
    - Determine how the program should understand context
    - Start coding the implementation

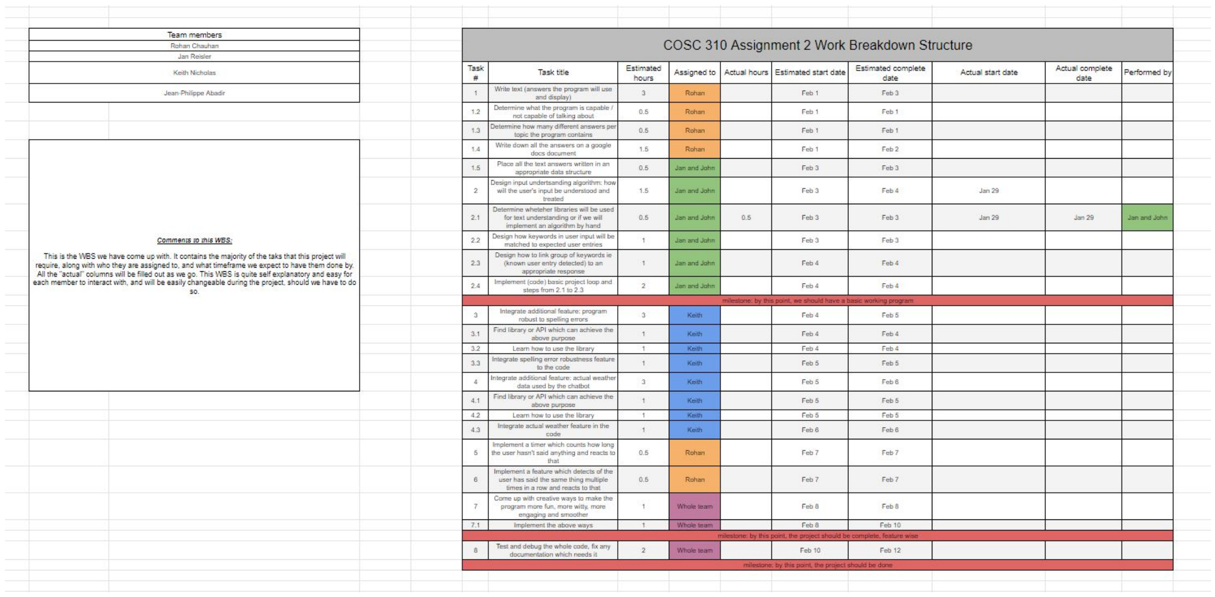
- Create a category of possible topic and responses for each topics
    - Determine the number of topics & answers the program could handle
    - Write the possible responses in google docs

- Create a class that handles spell checking using an API
    - Decide what API to use
    - Determine how spell checking would fit in the program (before or after pattern matching)

- Create a class that determines current weather of a specific location
    - Find a suitable API
    - Find appropriate library in Java to consume REST services and parse JSON

Work Breakdown Structure

Please see detailed version attached in the submitted zip file. Below is an image of the WBS extracted from the excel file.



Team members
Rohan Chauhan
Jan Reisler
Keith Nicholas
Jean-Philippe Abadir

Comments to this WBS:

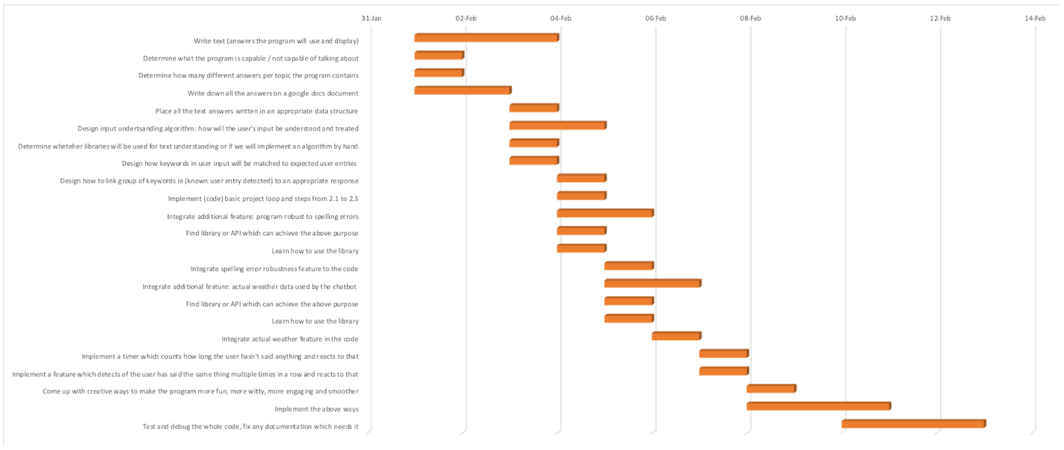
This is the WBS we have come up with. It contains the majority of the taks that this project will require, along with who they are assigned to, and what timeframe we expect to have them done by. All the "actual" columns will be filled out as we go. This WBS is quite self explanatory and easy for each member to interact with, and will be easily changeable during the project, should we have to do so.

COSC 310 Assignment 2 Work Breakdown Structure

Task #	Task title	Estimated hours	Assigned to	Actual hours	Estimated start date	Estimated complete date	Actual start date	Actual complete date	Performed by
1	Write text (answers the program will use and display)	3	Rohan		Feb 1	Feb 3			
1.2	Determine what the program is capable / not capable of talking about	0.5	Rohan		Feb 1	Feb 1			
1.3	Determine how many different answers per topic the program contains	0.5	Rohan		Feb 1	Feb 1			
1.4	Write down all the answers on a google docs document	1.5	Rohan		Feb 1	Feb 2			
1.5	Place all the text answers written in an appropriate data structure	0.5	Jan and John		Feb 3	Feb 3			
2	Design input undertsanding algorithm: how will the user's input be understood and treated	1.5	Jan and John		Feb 3	Feb 4	Jan 29		
2.1	Determine wheteher libraries will be used for text understanding or if we will implement an algorithm by hand	0.5	Jan and John	0.5	Feb 3	Feb 3	Jan 29	Jan 29	Jan and John
2.2	Design how keywords in user input will be matched to expected user entries	1	Jan and John		Feb 3	Feb 3			
2.3	Design how to link group of keywords ie (known user entry detected) to an appropriate response	1	Jan and John		Feb 4	Feb 4			
2.4	Implement (code) basic project loop and steps from 2.1 to 2.3	2	Jan and John		Feb 4	Feb 4			
milestone: by this point, we should have a basic working program									
3	Integrate additional feature: program robust to spelling errors	3	Keith		Feb 4	Feb 5			
3.1	Find library or API which can achieve the above purpose	1	Keith		Feb 4	Feb 4			
3.2	Learn how to use the library	1	Keith		Feb 4	Feb 4			
3.3	Integrate spelling error robustness feature to the code	1	Keith		Feb 5	Feb 5			
4	Integrate additional feature: actual weather data used by the chatbot	3	Keith		Feb 5	Feb 6			
4.1	Find library or API which can achieve the above purpose	1	Keith		Feb 5	Feb 5			
4.2	Learn how to use the library	1	Keith		Feb 5	Feb 5			
4.3	Integrate actual weather feature in the code	1	Keith		Feb 6	Feb 6			
5	Implement a timer which counts how long the user hasn't said anything and reacts to that	0.5	Rohan		Feb 7	Feb 7			
6	Implement a feature which detects of the user has said the same thing multiple times in a row and reacts to that	0.5	Rohan		Feb 7	Feb 7			
7	Come up with creative ways to make the program more fun, more witty, more engaging and smoother	1	Whole team		Feb 8	Feb 8			
7.1	Implement the above ways	1	Whole team		Feb 8	Feb 10			
milestone: by this point, the project should be complete, feature wise									
8	Test and debug the whole code, fix any documentation which needs it	2	Whole team		Feb 10	Feb 12			
milestone: by this point, the project should be done									

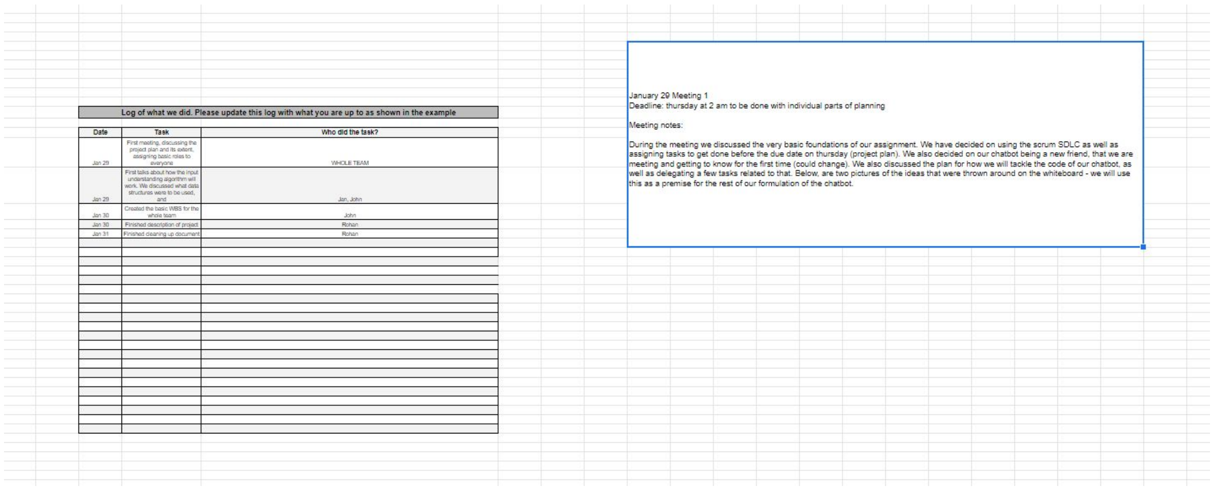
Gantt Chart

The following Gantt chart shows, at a very quick glance, which tasks should be performed, and within what timeframe. The tasks are displayed in chronological order (which is also in order of priority, since we are using Scrum) from top to bottom. We can clearly see the dependencies across the tasks from how they overlap vertically. If a rectangle is before another one, it is a prerequisite to another task. If a rectangle is above or below another one, it is a corequisite to another task. Please find the Gantt chart attached in the submitted zip file.



Work/Meeting Notes Log

The following image shows our meeting notes as well as a work log that we update as soon as significant work is done. Please find the log attached in the submitted zip file.



Log of what we did. Please update this log with what you are up to as shown in the example

Date	Task	Who did the task?
Jan 29	First meeting, discussing the project plan and its extent, assigning basic roles to everyone	WHOLE TEAM
Jan 29	First talks about how the input understanding algorithm will work. We discussed what data structures were to be used, and	Jan, John
Jan 30	Created the basic WBS for the whole team	John
Jan 30	Finished description of project	Rohan
Jan 31	Finished cleaning up document	Rohan

January 29 Meeting 1
Deadline: thursday at 2 am to be done with individual parts of planning

Meeting notes:

During the meeting we discussed the very basic foundations of our assignment. We have decided on using the scrum SDLC as well as assigning tasks to get done before the due date on thursday (project plan). We also decided on our chatbot being a new friend, that we are meeting and getting to know for the first time (could change). We also discussed the plan for how we will tackle the code of our chatbot, as well as delegating a few tasks related to that. Below, are two pictures of the ideas that were thrown around on the whiteboard - we will use this as a premise for the rest of our formulation of the chatbot.

Please note that this document is a work in progress and items such as the log, meeting notes and WBS will be updated as the assignment is carried out.