COMP-4400 Final Project Documentation

**Project Title:** Covid-19 Chatbot Using Prolog

**Team Member:** Tony Alas

**Date Submitted:** December 8, 2020

This project consists of a Prolog program which was written with SWI-Prolog in mind. It runs best when consulting the program from within SWI-Prolog. You will then see the first prompt from the Covid Bot which will allow you to respond with your name.

**ALL** responses will be in the form of a procedure call (which the program will let you know), so for example:

The first question will be “what is your name?”

The user should respond with:

name(yourName). -> Sample Run: name(tony). (where the period is necessary).

**Therefore, any response to a question will need to be of the form:**

procName(yourResponse).

Here is what you should see on the initial program run after entering your name:

Text

Description automatically generated

Additionally, whenever the Covid Bot asks a question and when the user responds with a “yes” to any of the questions, the user will need to press the ‘.’ (period) key in order to be able to respond. This is due to the fact that the procedure will try to return any remaining searches, but the period will allow the user to skip the search. Two examples are shown below:

Text

Description automatically generated

For this part of the program, you can see that the program tells the user that they need to press the period key in order to continue to respond. You can also see that the first message from the Covid Bot in response to the “iam(yourFeeling)” call returns a specific response depending on what the user said. For example, if the user responded with something like “sad” or “bad”, then the bot will output a different message.

Text

Description automatically generated

For this portion of the program, **as well as the rest of the program when the user responds with “yes” to one of the questions**, the bot will output a series of messages and then ask the next question. The user will then be prompted to press the period key in order to be able to respond to the question.

Therefore, the program will output slightly different messages depending on whether or not the user responds with a “yes” or “no” to make it feel more personalized.

For each question that is asked, there is a respective procedure. The input will be the response from the previous question and the output will be a series of messages reacting to the response, as well as asking the next question. More specifically, the procedures will take in a previous response and then output a slightly different response based on if the user said yes or no, and then will ask them the next question. If the user answers yes, the counter variable will increase (by calling the add procedure). The procedures that follow this description include fever, drycough, tired, breath, chestpain, speech, sorethroat, aches, rash, taste. Each of these are formatted in the form: proc(X) where X is the response from the user.

The program will also keep a count of the amount of times that the user responds with “yes” to a question. There are a few procedures that handle this information, namely: add, addtwo, counterValue(), checkCounter(X), intermediateFunc(X). These functions make use of the native Prolog nb\_getval and nb\_setval procedures that originally set the counter to 0 at the beginning of the program and then increase the counter whenever the user responses with “yes”. For most of the questions, if the user responds with “yes”, the counter will increase by one, but the counter will increase by 2 if the user responds with “yes” to any of the 3 serious symptoms (which are breath, chestpain and speech). This is to make up for the fact that they are more important and if somehow the user responds with “no” to most of the other questions, the bot will still recommend that they consult a medical professional or keep an eye on their health for further complications. This is because the threshold for the bot to respond with letting the user know that they do not need to worry yet is a score of 3 or below. Therefore, by making the serious symptoms increase the counter by 2, it will help prevent the bot from thinking that the user is safe.

If the user's score is below 4 (exclusive), it will go to the procedure that tells the user that they should be safe for now, if it is above 4 (inclusive) then it will go to an intermediate procedure that will do another comparison. If the user's score is below 7 (exclusive), it will go to the procedure that tells the user that they are okay for now but should be wary, if it is above 7 (inclusive) then it will tell them to report to a doctor, etc.

Therefore, depending on the user’s final score, the bot will provide different responses and decide whether or not the user needs to keep an eye on their health, contact a medical professional immediately, or can relax for now. The bot will output a series of links, resources and more information if the user gets either of the two outcomes that do not say they can relax. However, if the user wants the information even if they do not need to worry, they can call the procedure “covidInfo.” to output the same information regardless.

The program will also give the user the option to take the test again by calling the procedure “takeAgain.” which will reset the total score and bring them back to the beginning.

Below is a sample run of the entire program where the user only answers “yes” **twice**:

Text

Description automatically generatedText

Description automatically generatedText

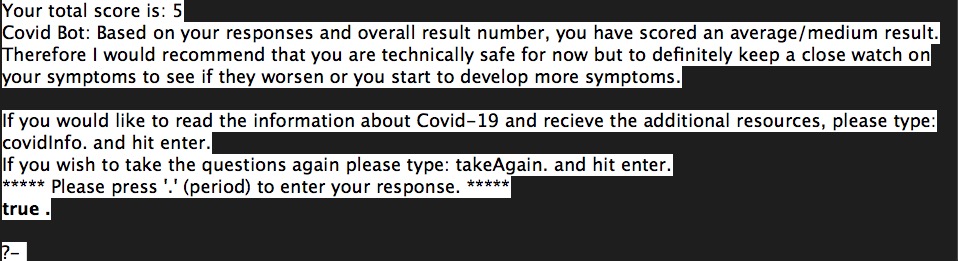
Description automatically generated

As can be seen, the Covid Bot asks a series of questions with the user responding in a specific manner and the bot will eventually output their final analysis and give its recommendation. The user then has the option to receive the information regardless of the bot’s positive recommendation and they can also choose to retake the questions (not shown in the above picture but will be shown below).

Below is a sample run of the entire program where the user answers “yes” enough times to land in the medium range (some parts are left out since it is similar to the run above):

Text

Description automatically generated



As can be seen, this result is slightly different than the first one where the user is safe. The bot recommends that the user is good for now but should keep a close eye on their health.

Below is a sample run of the entire program where the user answers “yes” enough times to land in the CRITICAL range (some parts are left out since it is similar to the first run above):

Text

Description automatically generated

Text

Description automatically generated

As can be seen, this is a result where the user has scored the maximum amount of points (they said yes to every question) and therefore immediately is brought to the information and links to what they should do.

Any additional documentation for each specific procedure and the program as a whole can be found inside of the chatbot\_Covid.pl file.

Thank you for reading this documentation.