1. **Introduction.** Provide a one or two paragraph overview of your chatbot project. What parts of the world will it chat about? What motivated you to select this particular domain? Are there any interesting features you want to highlight?

I would like to make a chatbot which would talk about a game called Path of Exile (PoE), particularly the items of POE. My motivation on this project would stem from my browsing of forums about the game, where there would be issues about the complexity of PoE.

PoE is a game which is notoriously known for being difficult to understand. Comparing the game now and when it was released, many new features were added. Usually, veterans of the game who have played since it's conception would thrive when playing the game, but newer players would be intimidated by it. I would like to further encourage new players to play and learn more about the game by asking the chatbot about it. A good starting point for players to learn about the game would be to learn about items. They would be able to ask questions about items such as how effective the item is, how much it is worth to buy, and what the requirements are needed to use the item.

2. **Technical Overview.** What programming language(s) will you use? Will you use APIs? You should not use external libraries; if you have any uncertainty about whether something written by someone else is acceptable, contact Prof. Frost in advance.

I will be using python to implement this chatbot, and I would like to use an API, from a website called poe.ninja to acquire the statistics for the items that I need for the chatbot. If permitted, I will only use the json and request libraries to utilize the API. Information about the API can be found here: https://github.com/5k-mirrors/misc-poe-tools/blob/master/doc/poe-ninja-api.md

Part 3 update: I have contacted Professor Frost and have approved the usage of the API. The libraries stated have also been used.

Part 4 update: I have added the tkinter library so that my project would utilize gui elements instead of console inputs. I have also added the Pillow and urliblibraries.

The commands used to install all needed libraries in Windows was done in command prompt.

Commands:

py -m pip install requests

py -m pip install Pillow

py -m pip install urllib3

To run the code, type in:

py CS171Proj.py

For the library installation and running the project, python could be typed instead of py depending on the environment setup.

3. **Phases.** Divide your implementation (which will comprise parts 3 and 4 of the project) into between 2 and 4 programming phases. Each phase should be a complete and working subset of your chatbot, building on the code from previous phases (and from part 1 of the project, if you so choose). Indicate which phase(s) you will implement for Part 3 (tentatively due March 1) and which remaining phases you will implement for Part 4 (tentatively due March 13). Do not include "blue sky" phases that you do not plan to complete this quarter.

   For part 3 of the project, I would like to have 2 phases:

   1. I would like to make sure I am able to correctly import data from the API. The API provides many details of an item, but I would only need to use a select few.

   2. I would also like to make sure my parsing algorithm will still work with the added variables from the API.

   For part 4 of the project, I would like to have 2 phases:

   1. I want to make sure I am able to parse input from the user.

   2. I want to use the information from the user to generate an output.

Part 3 update: I have successfully imported data from the API which added words to the lexicon list. I added grammar rules which would work with sample inputs that I have tested. Correct outputs have also been tested with these inputs. The quota for part 3 has been met, but I am lacking with the number of unique sentences. I have only tested for around 3 kinds of sentences.

Part 4 Update: I have successfully implemented input and output for the project. I have added more sentences that are able to be parsed, around 6 more. In addition to

adding more sentences, I added another goal for part 4 which was to make the project have GUI elements, which has been satisfied.

4. **Examples.** Write down examples of interactions between a human and your chatbot. Can questions or answers in the conversation impact subsequent answers by the chatbot? If so, include examples. You are free to make substantial changes later in the project, but these samples should convey a clear sense of your plans for the chatbot.
   Questions can include phrases such as:

“How expensive is [item name]?” or “What is the price of [item name]?”. There can be different variations of the question.

“What level do I need to be to use [item name]?”,

“What kind of weapon is [item name]?”.

These questions may be subject to changes. Depending on what information I can get from the API, I believe I can handle more complicated questions. I might also be able to extend the questions to handle other types of items, with these items having different and more complex characteristics.

Part 3 update: The sentences above have been implemented, but more will be added such as “Is [item 1] more expensive than [item 2]?” Or is [item1] higher level than [item 2]”.

Part 4 update: All of the sentences above have been implemented.

More questions which ask for item descriptions or item modifiers have been implemented(asking for modifiers will just output a list of characteristics an item has, they will make sense to the average player).

Example sentences available for input will be available for testing, and I will have screenshots of some examples as well. I will also include a list of available items (names) for testing in another test file, but it is not necessary to test all of them.

5. **Input Handling.** Your project will no doubt include code to convert text inputs from the user into some kind of internal representation. Outline the design of this code. Creating a parse tree is optional. If you plan on using a grammar, provide a partial version here. If you plan on following an algorithm described in the textbook or from another source, specify your sources. If you are inventing a novel approach, state that and give an overview.

   I would like to use the CYK-Parse algorithm from part 1 to be able to parse input from the user. I will also be reusing the grammar, but I will also add lexicons as necessary to cover as many questions as I can. Once the input has been parsed, then the input will be legible and will be able to be used in logic to generate an output.

   Part 3 update: I have not implemented input from the user, but I have been testing inputs with lists which resulted in correct outputs. I will leave those inputs in as well as leave a couple more examples of input lists that are able to be put into the chat bot. To parse in input from the user, I plan to tokenize item names since some item names have multiple words.Tokenizing these words would change “Soul Taker” into “Soul_Taker” which would make it easy for the parsing algorithm.

Part 4 update: Input has been implemented. I have successfully solved the issue of multi-word names and they are able to be parsed in. In addition to this, the names are added to the lexicon list. I have also improved user input by making it GUI based instead of console-input based. The sentences are specific, all sentences start with capital letters and the names need capital letters as well, but unknown words are handled and the program will ask for another input.

To use the GUI, just input a sentence and press the enter button.

6. **Internal Representations and Data Sources.** Outline your plans to represent and process data internally. Options might include (some combination of) logic, data structures, and tables. Will all information used by the chatbot be hard-coded in your code? If your chatbot will access an internal or network-available database in real time, describe that. Describe any interesting capabilities of your chatbot.

I will be using logic in order to determine how the output will be. Values will be taken from the API, and some values such as the price of an item will change in real time. It can be said that PoE has an economy of it's on based on events in the game and how players value an item. The poe.ninja API is able to give this real time data. Data structures such as a dictionary will be used to hold this information.

Part 3 update: I am able to use the API to get item names, item prices, item level, and item types. In the future, I may be able to acquire more information which will be included in the future report.

Part 4 update: I am able to get even more item characteristics to answer questions. I am also able to get an image from a URL to display what an item looks like. This feature uses the Pillow library as well as the urllib library.

7. **Output handling.** Outline the design of your code that generates English language replies to the user. See "Input Handling" for things to write about.

Based on the logic from section 7 as well as the input, the chatbot will simply print an English phrase to the user.

Part 3 update: No changes here, there will still be an English reply to the user.

Part 4 update: There will still be an English reply for most sentences, but there will also be an image as well as an English sentence for some inputs as well.

Miscellaneous:

There may be some useless grammar, but this doesn't affect functionality, this more likely affects a not-as-important feature such as runtime.

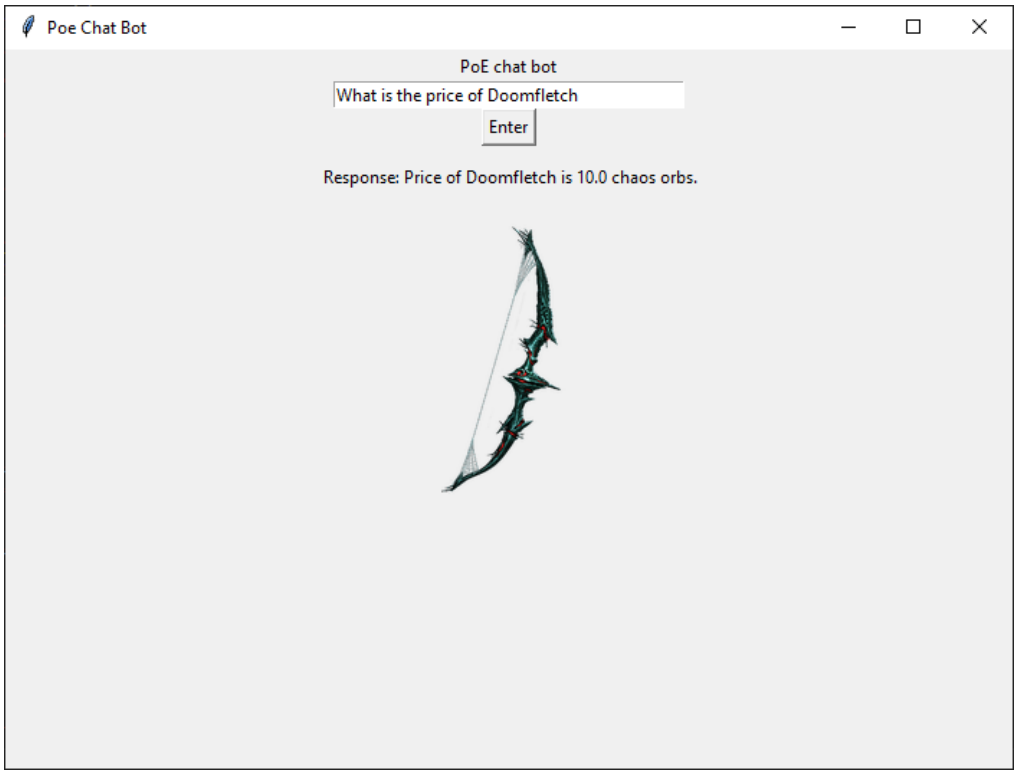
Items like **Mjölner** don't work due to **ö** not being recognized. The rest of the item names without accent marks work with no issue.

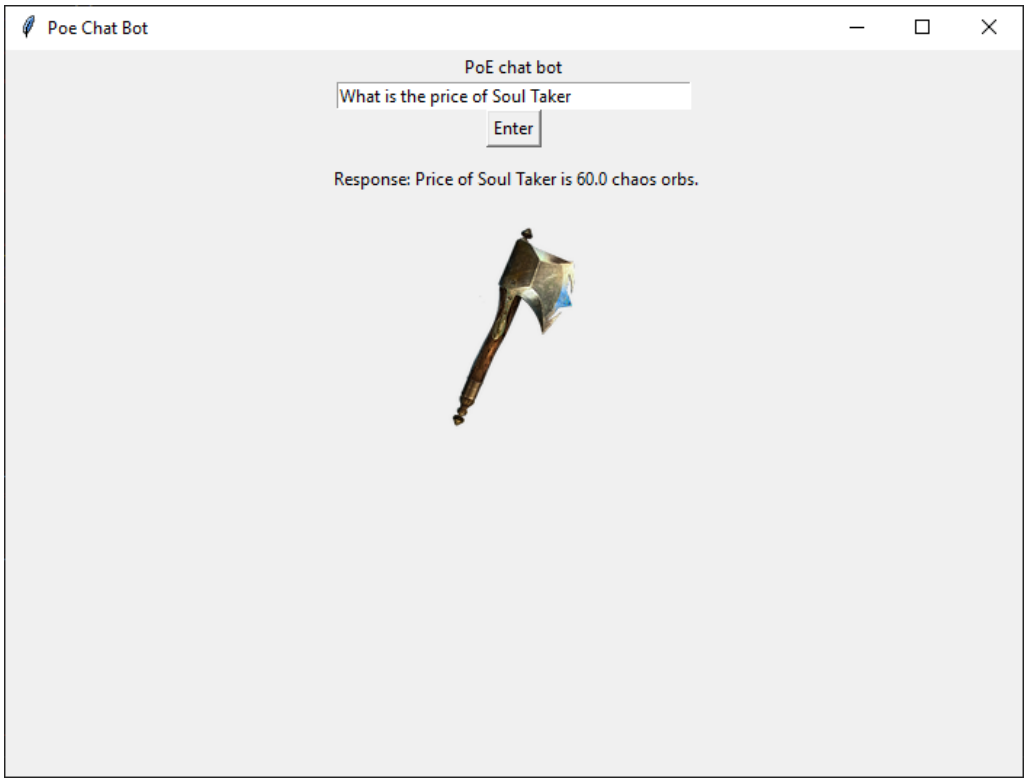
Since console inputs show more history about previous inputs and my GUI implementation doesn't, I will have more screenshots than normal.

Questions asking about single items will automatically display the image, questions about 2 items won't.

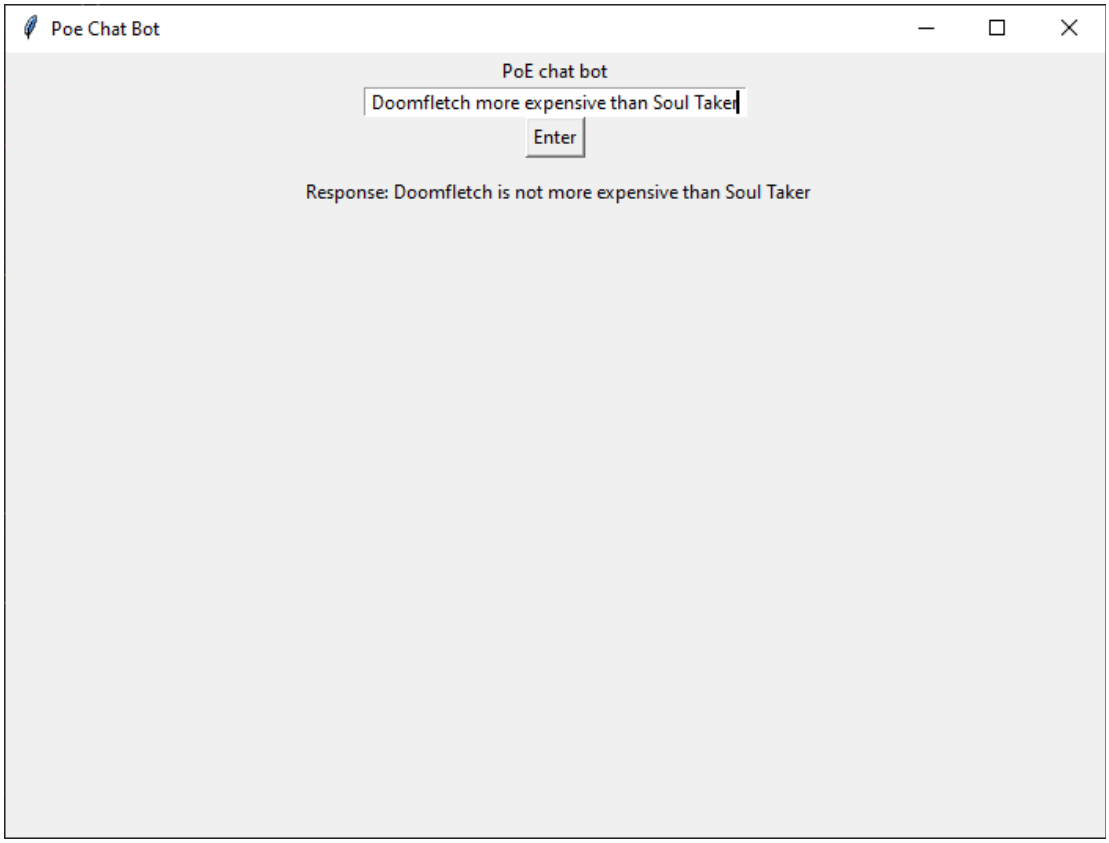
Screenshots:

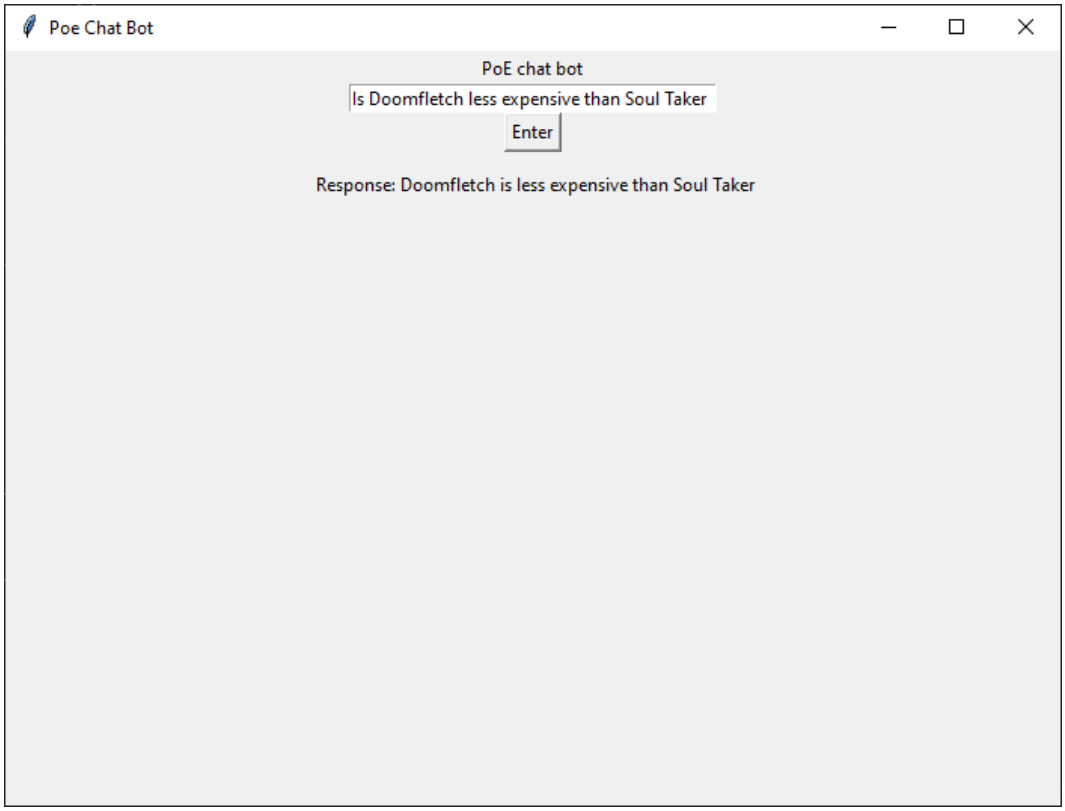
(If sentences are too large for the text box, they will be displayed on top of the screenshot)

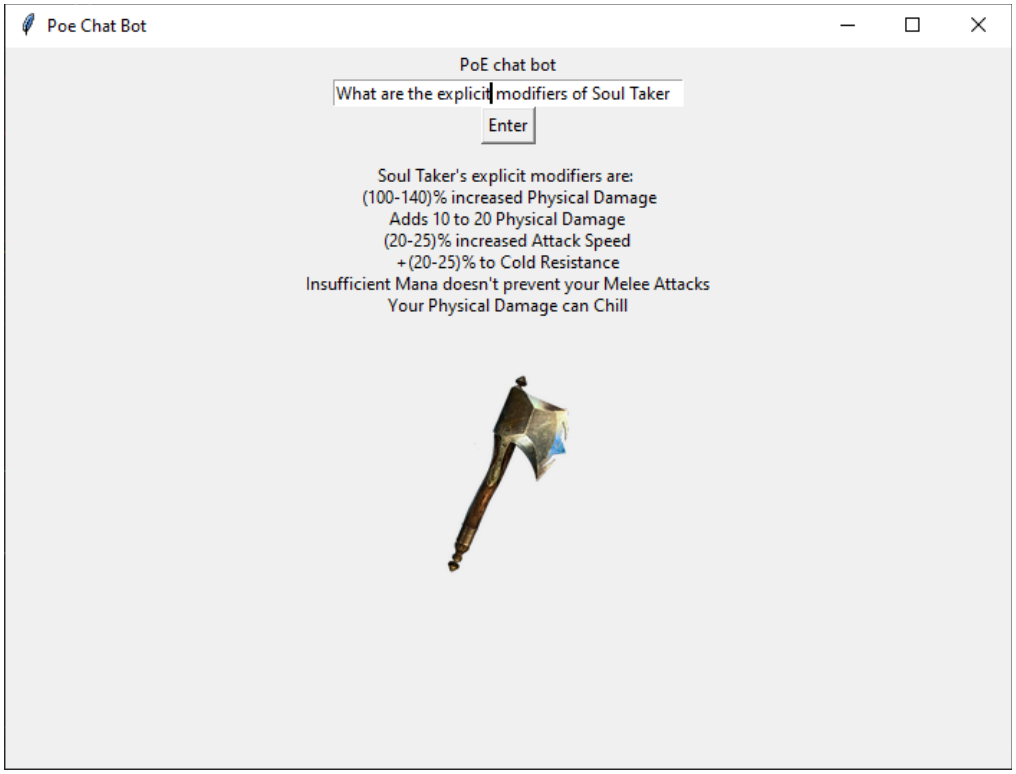




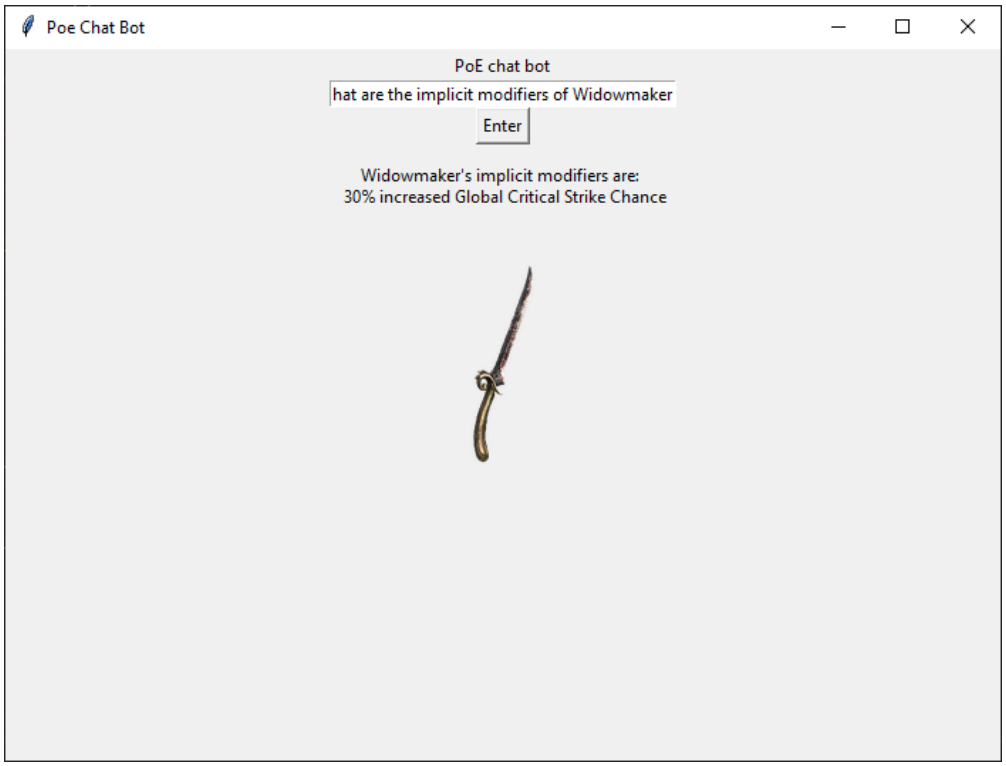
Is Doomfletch more expensive than Soul Taker:

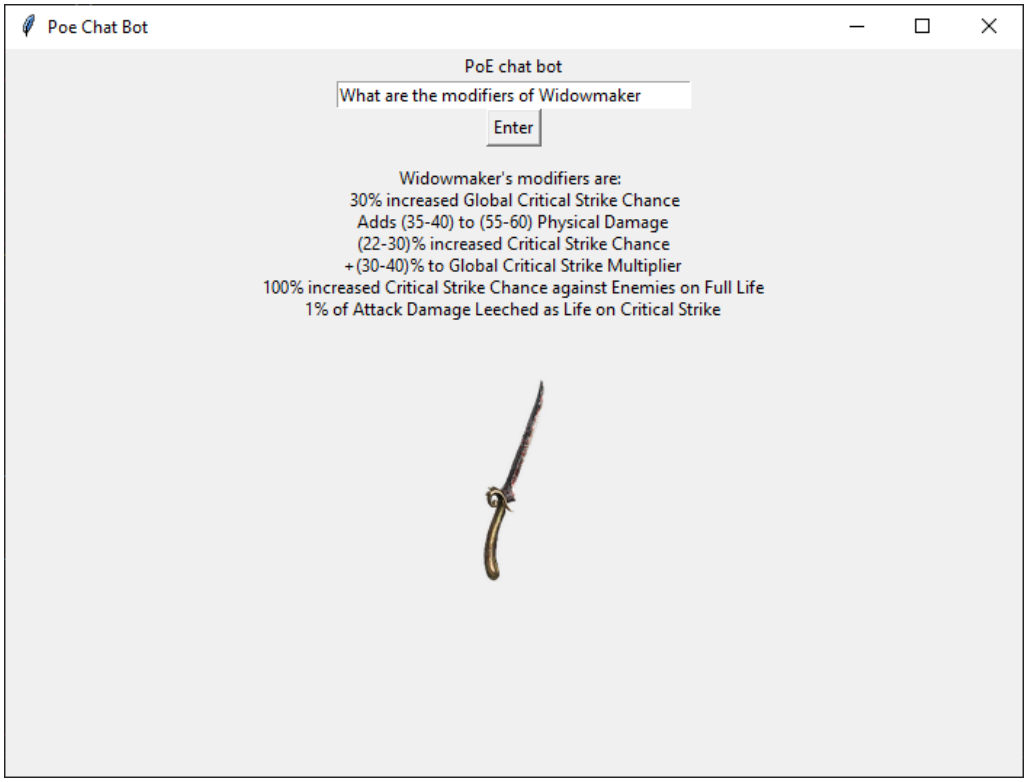




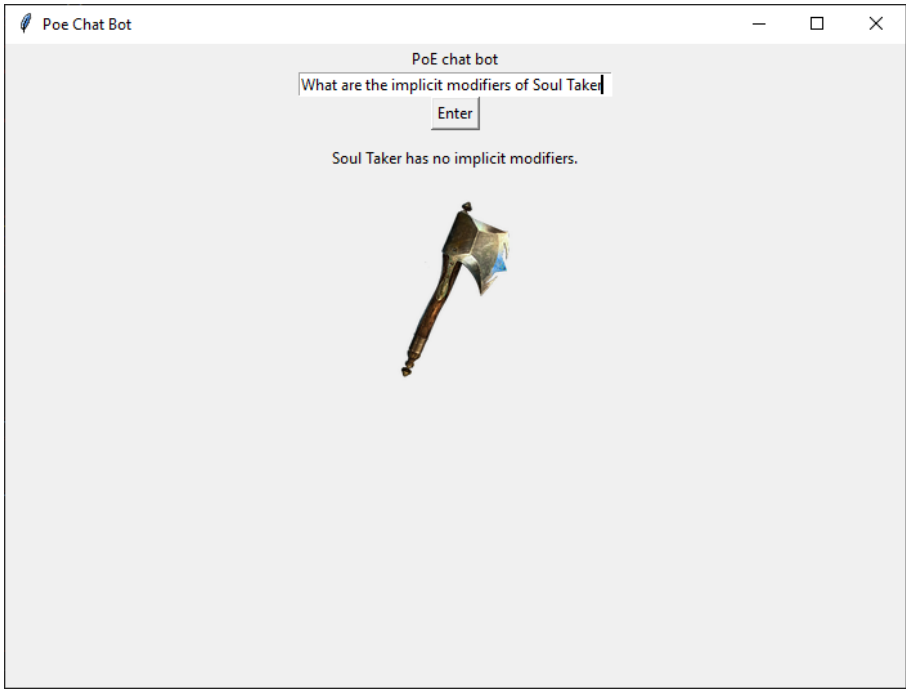


What are the implicit modifiers of Widowmaker:





(The difference between this screenshot and the previous screenshot is that this one displays both implicit and explicit modifiers. Some items do not have implicit modifiers, which is why I couldn't demonstrate this example properly with Soul Taker.)



Here are the list of questions that be tested. The item names can be interchangeable with item names from the text file included in the submission:
What is the appearance of [item name]
What is the price/value of [item name]
What is the exalted price/value of [item name]
What is the requirement of [item name]
What are the implicit modifiers of [item name]
What are the explicit modifiers of [item name]
What are the modifiers of [item name]
Is [item name] more expensive than [item name]
Is [item name] higher/lower level than [item name]