Task 3 - Improvements

Since this is a very basic dialogue application, there are several limitations that make the user experience less than ideal. The biggest limitation is that this assignment makes use of a "scripted"/ rule-based dialogue system. These types of systems prevent the user from being able to ask questions or respond in ways that were not anticipated by the programmer. However, since this is an introductory course in dialogue systems, I will not examine the applications and limitations of rule-based dialogue systems any further. Other limitations of this app that are due to my own development choices are the following: rigid grammar, ambiguity in interpreting the time, lack of functionality to handle repairs, and not allowing backtracks.

Looking closer at the first limitation; the app currently has a very limited grammar. It only has six people stored (who can only be referenced using their first name/nickname), seven days of the week, and ten ways to confirm or deny a statement. This means that the system only understands these limited words, and the user must only respond with single-word answers. Even if a word that is in the grammar is sandwiched in a larger context, the machine will not be able to interpret it. This is restriction is very unnatural, and requires that the user has some knowledge of what they are allowed to say. For example, when the dialogue system asks: "who are you meeting with?", the user can respond with "Anna", but they cannot respond with "I'm meeting with Anna".

Furthermore, something that occurs in natural speech but is not handled in this dialogue machine is the ability to understand and do repairs. This system is not designed to perform self-repairs, respond to repair requests from the user, or recognize when a user makes a repair mid-utterance. For example, if the DM asks "will it take the whole day?", and the user responds with "Yes. No", or "Yes I mean no", a human listener would interpret this as the user correcting themselves and ultimately answering "no". However, this system treats the entire utterance as a single input, and upon finding it doesn't match anything in the grammar, routes to the generic fail state of "I didn't understand that" and then repeats the question.

In a similar vain, the DM does not have the ability to backtrack in the conversation. If the user realizes that they said the wrong person when the DM first asked who they are meeting with, they are forced to continue through with the flow of the conversation until the DM asks them if they have all the correct information, and at that point the user can say no. This is very time consuming and inflexible because the DM will start back at the beginning with the questions.

Lastly, I will discuss a limitation that I have provided some improvements to in the code: interpreting various time formats when the DM asks at what time the meeting is. In the getTime() function I have tried to anticipate as many variations in time utterances as possible to make it more flexible. It now knows how to handle if the user just says two numbers (for example, "fourteen thirty-five"), if they use the word "o'clock" (for example "five o'clock"), if they say "half past" or "quarter to", and if they use AM/PM (for example "five thirty-five PM").

There is still a limitation in the getTime() function that I don't think I can solve without more advanced tools, or a more specific setting. The problem occurs when the DM does not know if a person is speaking in 24h format or if they colloquially want to refer to a time in the afternoon without saying AM/PM. For example, if someone says "three fifteen" they most likely have a meeting at 15:15 and not 03:15 in the morning. If the app is used for a specific business then one could check that any meetings that are requested by the user fall within the business' hours of operation, but without that knowledge then the machine is required to just take the user's words at face value. Another improvement that could be made in the future is that if the user does not specify AM/PM and they say an hour that is lower than 13, then there could be a state that confirms whether they want the time in the morning or the afternoon.

Separate note: I was not able to make task 2 as dynamic as I had hoped. I was trying to save the context of the previous question so that I could have a generic re-raise state that would raise the previous question again if the response was invalid/missing. I could successfully save the previous state in the context, but I was not able to get the target to read the reference to the state that I saved.