Lab 3: Report

The goal of Lab 3 was to create a voice-driven appointment scheduler using XState and Speechstate, allowing users to select the person, day, and exact time for their meeting. While the system performs well and can successfully schedule appointments based on user responses, it still has several limitations and issues that prevent it from becoming a fully efficient chatbot.

The first and most frustrating limitation is that the chatbot only recognizes words or phrases explicitly included in its grammar. Users cannot use synonyms, variations, or filler words in their responses; they must provide the exact answer the system expects. For example, if the chatbot asks, *"Which day would you like to schedule the meeting?"* and the user replies *"On Tuesday"* or *"Maybe Tuesday,"* the system will not understand and will prompt the user to try again. The system also does not support more natural conversation patterns. Users must answer one question at a time in a specific order. If a user tries to provide all the information at once, such as saying the person, day, and time in a single sentence, the system will not be able to extract and process that information.

Moreover, unnecessary repetitions and the lack of guidance or hints about what the system expects can make the entire process frustrating and time-consuming. The NoInput states repeat endlessly if the user does not respond, which is both inefficient and irritating. When users provide input that is not recognized by the system's limited grammar, the chatbot almost never explains what it expects. The only exception is during the confirmation steps, where it clearly asks users to *"Answer yes or no."* This lack of guidance can cause users to get stuck in an endless loop of repetition. Additionally, during confirmation, if the user says *"No"* the system restarts the entire process by asking *"Who would you like to meet with?"* instead of identifying and correcting only the incorrect information. This results in unnecessary repetition and a poor user experience.

Another big issue is that the system does not remember anything or adapt to the user. If the page is refreshed or the app is restarted, all the information disappears, and the user has to go through the whole process again, which can be really frustrating. At the same time, the system doesn't learn from previous interactions or adjust to user preferences either. Not taking repeated mistakes into account or trying to make the experience smoother can make the interaction impersonal instead of natural and user-friendly.

To improve some of these issues, I made a few changes to make the system feel more reliable and easier to use. The chatbot now deals with invalid inputs more clearly by using dedicated NotInGrammar states, such as PersonNotInGrammar, DayNotInGrammar, TimeNotInGrammar, and ConfirmationNotInGrammar. Now nstead of ignoring what the user says when it doesn't recognize it, the system repeats what it understood and infroms the user that the input is not available. For example, in the AskForDay state, if the user says something other than a valid day or chooses a day that is not available, the system responds with a clear message such as: *"That day is not available"*. This makes the interaction clearer and avoids silent errors that could confuse the user.

I also improved how the system handles *yes* and *no* responses by adding helper functions like positiveAnswer() and negativeAnswer(). This allows the chatbot to understand variations such as *"yes" "sure" "fine"* or *"nope"* rather than only accepting a strict *"yes"* or *"no"*. Finally, in the whole day and confirmation states, the context variables agree and disagree are reset before the system listens again. This prevents previous answers from affecting new decisions and helps the conversation flow more smoothly and accurately.