Errors and limitations of the app

I have listed some limitations in the application:

1. the user does not know what actions are available
2. there is no way to stop program without finishing the task
3. there is no way of adding new people if they are unknown to the program
4. if intent is low for something there is no confirmation if that is the action the user wants to do

Fixes

For the first problem my fix was to simply add a descriptive message from the program if the user asks for something that the program can not do.

For the second problem I wanted to add a single event that would always recognise the utterance *stop*. I tried to add a recognised-event to the main state but it did not work and I could not figure out how to do this. Instead I added a target and condition to all recognised events. In retrospect it may have been better practise to create a function that returns a template of a recognised event instead.

The fix for the third problem would require so much work that I did not try to implement it. You would need to add a state for adding new entities to grammar or to Rasa if grammar was replaced by the API. In this state you would handle adding a new person. This state, let's call it *new_person*, would be triggered in the following way:

P: "Who are you meeting with?" (in state *who*)
U: "Stacy" (in state *who*)
P: "I do not know them. Would you like to save Stacy as a new contact?" (in state *who*)
U: "Yes" (in state *who*)

transition to state *new_person*

P: "OK. Saving Stacy as a new contact. What is her full name?" (in state *new_person*)
U: "Stacy Star" (in state *new_person*)
P: "Saved Stacy Star" (in state *new_person*)

transition to state *day* and continue creating appointment

For the fourth issue I decided to add a threshold and check the intent confidence on that threshold. If the confidence is too low a confirmation message will be prompted. If the user confirms that the action is correct the program transitions to the state corresponding to the intent.

Other observations

The flow of conversation is not necessarily the most natural.

In some cases it is expected that the user will answer either yes or no. In these cases there is no error handling if the user answers something else.

Using the following:

```
      unknown: {
            entry: say(I did not understand that. I can create and
appointment, set a timer or add an item to your to-do list for you."),
            always: 'welcome'
      }
```

triggered the listen state in welcome. I guess this makes sense because the event ENDSPEECH is triggered and then you transition to the *ask* substate. I could not find another solution than to create the *unknown* state like this:

```
      unknown: {
            initial: "prompt",
            on: {
                  ENDSPEECH: { target: 'welcome' }
            },
            states: {
                  prompt: {
                  entry: say("I did not understand that. I can create
and appointment, set a timer or add an item to your to-do list for
you.")
                  }
            }
      }
```

Another general note worth mentioning is that the code is a bit messy, there is inconsistent use of for example semicolons and quotations, there is some repetitive code which could be refactored and there is a lack of proper documentation. Since I spent quite some time on other aspects of this assignment I did not fix these issues in the end but instead acknowledge their existence and realise that in a real-life project fixing these problems would require more attention.