CS 5100 Foundations of AI

Assignment 4:
Creating a Sandwich Ordering System

Value: 10%

Pinterest

In this assignment, you will design and develop a sandwich ordering system. The system you develop will take in a typed customer request for a sandwich, engage in a dialog with the customer to clarify details, and then print out the complete order. For simplicity we will ignore details about cost. The system does not really have to make the sandwich either! ☺

The goal in this assignment is to create a dialog system that does something useful. And in the process, to get an idea of the logic used, and the issues involved in understanding natural language.

Chatbots are becoming common in such situations. For some sample use-cases, see for instance https://www.linkedin.com/pulse/5-ways-grow-revenue-using-chatbots-sales-tool-alexandre-debecker/ or https://www.salesforce.com/eu/blog/2019/04/what-is-a-chatbot.html. However, our emphasis is not on the highly polished natural language interface you may see with a chatbot. Instead, **our goal is to have you concentrate on the logic behind such systems, finding out what information is available in the customer's order and what is not, and ensuring you get all the required information from the customer by asking specific questions.** Of course, in theory, you could create a fine chatbot using the information you put together in this assignment.

You can implement the coding part of this exercise using either Python or SWI-Prolog, as detailed in Step 3.

Follow the steps below in doing this assignment.

Step 1: Decide on the menu: the sandwiches and options you offer

In this step, you will decide what sandwiches and options you will offer in your sandwich ordering system. Customers can only order the sandwiches you offer, and they cannot create a completely new sandwich from scratch. However, they can modify any of the sandwiches by choosing a (different) type of bread or spread. They may also add options or exceptions (examples below) to their sandwich order.

a. Your menu must include at least 4 types of sandwiches, one of which has **your name** in it, such as "Anne's Veggie Special". You should decide what ingredients usually go into each sandwich, but you can leave the bread or spread unspecified in some types of sandwiches. Use this information to make up a description for each sandwich you offer. For example, you may describe "Anne's Veggie Special", as "Swiss Cheese and avocado, with tomato, lettuce, onion, carrot, and cucumber"

b. The menu must offer at least 3 options for breads and spreads, and at least 3 additional options you can add and at least 3 exceptions you can leave out.

c. For inspiration, see something like the Jimmy John's or Subway pages at https://www.jimmyjohns.com/downloadable-files/jj_menu_no_prices.pdf or https://www.subway.com/en-US/MenuNutrition/Menu/Sandwiches. These pages will give you some ideas. Get creative, but don't make things too complicated!

d. Decide on what you offer, including:

- What kinds of sandwiches you offer (e.g. Veggie Delight, BLT, etc.)
- What the usual ingredients of each sandwich are (e.g. bacon, lettuce and tomato for a BLT)
- What kinds of bread options are available (e.g. rye, wheat, French, etc.)
- What kind of spread options are available (e.g. butter, mayonnaise (mayo), etc.)
- What additional options are possible (e.g., toasted or not, "extra cheese", "salt and pepper", etc.) and
- What exceptions are possible ("no onions", "hold the mayo", etc.)

e. Create a file named **README.txt .** In that file, create a section titled "**Step 1**". In this section, list the sandwich choices available (including the one named after you) with the description of each, and the breads, spreads, options, and exceptions offered.

Step 2: Decide on the logic of the system

In this step, you interact with the customer to get complete details of the sandwich they want. From each initial customer request, you have to figure out what's specified. If some details are not specified by the customer (e.g., they did not specify the bread they want), you should tell the customer what detail you're looking for, what the options are, and what the default is. You should then ask the customer to specify the details you require.

To get these details right, imagine you are using the customer input to fill out a form (or data structure) with something like the following details:

Name of sandwich:
Usual ingredients:
Bread-type:
Spreads:
Options:
Exceptions:

Of these, the name , bread type, and spread are required to be specified by the customer. Customers may specify additional options or exceptions are optional information; often they may specify neither. The usual ingredients are known to the program from the name of the sandwich.

Here's what we expect your program to do:

a. Have your program greet the customer and ask them what they want to have.

b. If the customer asks for the menu with a request like "What are the choices?", "What's on the menu?" etc., display a menu with the sandwich choices and a description of each. Ask them what they would like to have.

c. The customer should come up with an (initial) request that looks something similar to one of these (these are only some examples; you can think of so many variations of these):
   - *Anne's Veggie please with no onions.*
   - *I want the BLT, Italian bread, with cheese.*
   - *I'd like the Meat Lover's on rye please, mustard, hold the mayo*

   Remember that customers can only order the sandwiches you offer, but they have a choice of breads, spreads, options and exceptions, and they can ask for any of these choices with any sandwich they order.

d. For example, if a customer just asks for "Anne's Veggie on whole wheat please with no onions", you can imagine the form or data structure you saw above filled-in with the following details:

   Name of sandwich: Anne's Veggie Special
   Usual ingredients: Swiss cheese and avocado, tomato, lettuce, onion, carrot, and cucumber
   Bread-type: Whole wheat
   Spreads: mustard
   Options:
   Exceptions: Onions

   In this case, the customer has completely specified what they want. No other details are required.

e. If some details are not specified in the customer's initial request, you must get them to clarify their choice for each detail that is required. For example, if the customer does not specify any particular bread for a sandwich, you may say "This sandwich usually comes with Italian bread. Press return to accept this choice of bread, or choose one of the other alternatives: 1. wheat bread, 2. rye bread." Take the customer's input and fill in the details provided. Decide also how you will respond if the customer specifies irrelevant options, for example, asking for "No carrots please" even when the sandwich they order does not have any carrots.
   Note:
   1. **You are NOT expected to create a fancy UI.** I'm expecting you'll use simple print and read commands (or the equivalent in Prolog) to interact with the user. Keep it simple!
   2. You must have a list of terms that you treat as equivalent. Customers may say veg, veggie or vegetarian (or mayo or mayonnaise) and you must treat each element in a set of these terms the same as the other elements in the set.
   3. Customers (the polite ones at least!) may use words/phrases such as "Please", "I'd like", "Thanks", etc., which you must learn to ignore.
   4. Phrases like "Hold the onions" or "Hold the mayo" mean "I do not want onions" and "I do not want mayonnaise" respectively, and you must include any such item(s) in the *exceptions list* in any sandwich order.

f. After all the steps above, you will have complete details for the sandwich the customer is requesting. Show the details that you have obtained, and get the customer to confirm that all choices are correctly understood.

g. In file **README.txt**, create another section, and title it "**Step 2**". In that section, provide:
   - The list of terms you treat as equivalent
   - The list of terms like "Please" etc. that you will ignore.
   - The different ways you will allow customers to specify exceptions like "Hold the mayo".

Step 3: Coding and testing the system
Once you have decided what your menu options will be, and you have decided on the logic of the system, you are ready to implement the system in this step.

a. Your system should do the following, for each run of the system:
   i. If requested, show a menu with the names and types of sandwiches available.
   ii. Ask the customer what they want.
   iii. Take input from the customer.
   iv. Check to see if any details are missing, and if so, *for each detail that's missing*, get the required details from the customer as detailed in Step 2e above.
   v. Once you have all the details required, and confirmed with the customer, print out the details in the format shown in Step 2d above.

b. You can implement this system in Python or SWI-Prolog. **Do NOT use any chatbot package or a bot framework.** You may use any regex package to identify relevant words in the customer's input**.** Make sure you implement what we have defined above, but keep it simple. Name the main program **sandwich** (i.e. sandwich.py or sandwich.pl, as appropriate).

c. **Do NOT use any natural language library such as nltk or spaCy** – these packages are way more complex for what is required for this assignment. These packages could make the program simpler in some ways, but the idea is to develop the program without using the abstraction that these tools provide.
On this note, remember: if you are thinking of using any other package that you think will be useful, check with us on Piazza ***before*** using these packages.

d. In file **README.txt**, create another section, and title it "**Step 3**". In that section, tell us the programming language you used to implement the system, and list the commands required to run the system. Also list the files that you wrote to implement this assignment and write a short one-line description of the contents of each file.

Step 4: Demonstrating the system

In this step, you demonstrate what your program can do, and how it does what it does.

a. In file **README.txt**, create another section, and title it "**Step 4**".
b. Identify 3 examples where the sandwich ordering system takes customer input and creates something for the customer based on incomplete specifications for the sandwich. At least 2 of these examples should be incomplete specifications requiring the system to check with the customer to get additional details. Run these examples on your system and cut and paste the whole interaction for these examples in section Step 4 of **README.txt**, under subsections titled **Example 1**, **Example 2**, and **Example 3**.
c. Identify a 4th example where the sandwich ordering system takes customer input, and fails to create an order for the customer, because the logic of the system does not support something the customer is requesting. Run this example on your system and cut and paste the whole interaction for this example in section Step 4 of README.txt, under a subsection titled **"Example 4"**. Explain in 1 or 2 sentences why the system fails in this example.

Step 5: Submitting the assignment

In file **README.txt**, create another section, and title it “**Step 5**”. In this section:

a. Write 1-2 sentences about what the easiest part of the assignment was.

b. Add another 1-2 sentences about the hardest part of the assignment.

c. Finally, add 1-2 sentences about what you learned from doing this assignment.

Put the file **README.txt** and the program files you wrote to implement the sandwich ordering system into a folder on your computer. Minimally you will have **README.txt** and either **sandwich.py** or **sandwich.pl** in that folder. You will have all other related (helper/auxiliary) files in the folder. When you have all your files ready, zip up these files into one .zip file named **yourFirstName_yourLastName_Assign4.zip** and submit the zip file via Canvas.

Hope this assignment is a fun and learning experience for you! We will try out a few assignment submissions in class – remember, your program could be the one selected to be tried out!

https://online.jimmyjohns.com/menu/jj3137/products/28122354