Lecturer: Mario Silic
Semester: Autumn Semester 2019
Group Project: Recipe bot, ID: 1171
Students: Tamir Metzner (14-608-202), Michael Greger (14-617-914)
Lecture: 7,789 | 8,789: Skills: Programming with Advanced Computer Languages

Date: December 20, 2019

_____

After almost 6 weeks of learning **Python** as a programming language, it was time for us to apply the acquired knowledge in a self-chosen group project within the master course "Programming with advanced computer languages" of the University of St. Gallen in November.

Contrary to the **thematic focus** of our university, we did not want to focus on any financial topic, but rather on a sustainable topic that would help the world and at the same time make life as a student easier. We quickly came across the UN's 17 "Sustainable Development Goals", which were launched in 2016. Achieving these goals in all 193 UN member states by 2030 should ensure sustainable development on an economic, social and ecological level. A closer analysis of the 17 targets revealed that we were particularly interested in target 12, "Responsible consumption and production", with a partial focus on "Food". It states that "*each year, an estimated 1/3 of all food produced - equivalent to 1.3 billion tons worth around $1 trillion - ends up rotting in the bins of consumers and retailers, or spoiling due to poor transportation and harvesting practices*".[1]

We then set ourselves the **goal** of counteracting these incredible figures with a software program. This program should help to reduce the food waste of private households, especially of students. Specifically, as a first step, the user of the software should be able to enter the ingredients that are currently in his fridge and need to be processed. The user should then be able to enter the number of recipes desired, select a specific recipe and finally display nutritional values and cooking steps for the selected recipe. Lastly, we decided that all communication should be conducted via a chatbot to make the program as user-friendly as possible.

In a **first step**, we searched for a database that we could use to access the necessary recipes. Here we came across the recipe collection of "spoonacular.com", which could be ideally accessed via its own API. [2]

The **second step** was now to directly program a program in line with our objectives. Unfortunately, there were no tutorials available, but there was enough documentation about the API available on the website, so that we were able to build a working prototype within two weeks thanks to our efforts. This prototype already had all five essential functions:

- getRecipeByIngredients(): input of 1-10 ingredients and desired number of recipes
- chooseRecipe(): selection of one of the recipes
- getRecipeInstructions(id): display of the cooking instructions of selected recipe
- getRecipeIngredients(id): display of the ingredients of selected recipe
- getRecipeNutrition(id): display of the nutritional values of selected recipe

[1] https://www.un.org/sustainabledevelopment/sustainable-consumption-production/
[2] https://spoonacular.com/food-api

In programming the prototype and these five functions we had to overcome some *challenges*. First, we both had no experience accessing an API. After some research, we came across the "requests" module, which makes it quite easy to send GET requests via the API. Then all you have to do is save the results and present them in a user-friendly way. The latter leads to the second challenge, the handling of the output format of the API: Json. On the one hand, This format structures the data very well, on the other hand you have to get used to the format in order to be able to present it in a user-friendly way. On top of that, the recipe data in the "spoonacular data base" was not always structured in the same way, e.g. recipe instructions are sometimes subdivided into steps, but sometimes they are all gathered under "step 1" – and sometimes it is a mix of both. This required a lot of time and effort. In the end we assumed the best data structure we could observe within the data base as coding basis. When we had programmed all functions separately, we had to connect the functions bug-free. This was our third challenge, which also required some patience and trial and error attempts. Throughout all these challenges, we were limited to only 100 free API calls per day. However, we were able to overcome this challenge relatively quickly by caching Json files locally, so that we did not have to make unnecessary API calls anymore.

In the **third step**, the time had come to simplify the interaction between the program and the user using a chatbot. After some research we came across an online tutorial on YouTube which explained how to program a Deep Learning chatbot[3]. We definitely wanted to build a machine learning chatbot, because it doesn't require you to define the exact questions in advance. Instead, the chatbot processes any input from the user and mathematically calculates to which predefined "intent" the respective question phrase can most likely be assigned. If you also define a confidence level, you can define the threshold value above which the chatbot should spit out the answer sentence belonging to the intent - or, in the opposite case, up to which it should ask the user for clarifications. We could take much of the code from the tutorial but had to adapt the input file with the topic-specific questions and answers. The biggest challenge, however, was to integrate the chatbot into our previous program, which required a lot of time and effort. After some time, however, we succeeded.

However, we could not solve one **problem** concerning the chatbot: the appearance of various hints or error messages. These have to do with the fact that the modules used for the chatbot do not support the newest version of TensorFlow yet. Therefore, we had to use an older version of TensorFlow, which then resulted in some error messages saying that certain commands are deprecated. Nevertheless, the error messages have no impact on the functionality of the chatbot.

In **summary**, we are very happy about the Python knowledge we were able to acquire with the help of Mr. Silic's learning platform on the one hand and how we were able to apply this knowledge directly in our project on the other hand. The program works and meets all of our objectives defined at the beginning of November.

In order to be used by private households and students without IT skills and thus contribute to the twelfth goal of the UN "Sustainable Development Goals", a suitable **Graphical User Interface** would be ideal. Therefore, we want to develop a cross-platform Graphical User Interface in a next step in order to maximize the number of users of the program.

[3] https://www.youtube.com/watch?v=wypVcNIH6D4