ProTech : Nutritive Coach

Robin Camarasa[1]

[1]Ecole des mines de Saint-Etienne, Saint-Etienne, France

1 Introduction

This project is an practical application of an 80 hours Artificial Intelligence (AI) lesson. That lesson is divided in three part, Problem Solving, Knowledge and Reasoning and Learning. The aim is to create a fully operational Android application that use 3 AI algorithms that gives nutritive advises to the user.

2 Project structure

In this section, we propose to explain the structure of our Android application solution.

2.1 Project overview

To create our solution, we divided the project into a back-end and a front-end. The back-end was divided on a MySQL[4] database, an heroku[2] server and Github repositories. The front-end was divided on an administrator web front-end hosted on a github.io page and an APK containing the Android application.

The front are connected to the heroku server with an REST API. The github repositories contain all the codes and automatically launch an heroku[2] deployment at each push and the MySQL database is an heroku add-on. You find the structure in Figure 1 at page 1.

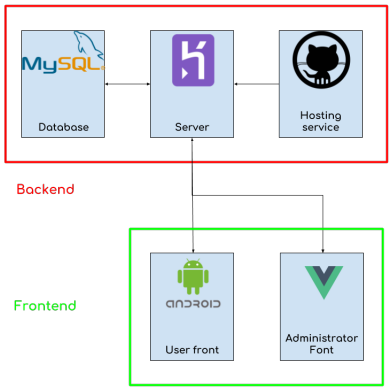

Figure 1: Project structure

2.2 Database

We used a governmental database from `https://www.data.gouv.fr` that contained 1878 entries in a csv file. Because of the encoding we created a shell script that transformed UTF-8 characters into ASCII one. An other shell script transformed the names of the columns and reduced their number. At the end our database contained 8 tables and you will find a full description in Figure 2 at page 2.

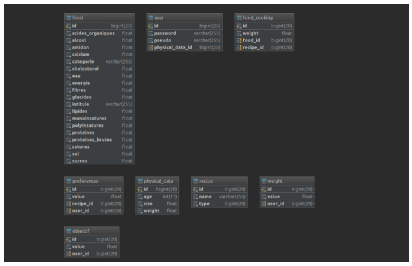

Figure 2: Database structure

2.3 Back-end

The heroku server contains all the AI algorithms and ensure the communication between the back and the front. All the code on the server are in Java using Spring framework [3] and Gradle package manager. The back-end was developed with Jetbrain's softwares, IntelliJ (Java IDE) and Datagrip (Databases Manager).

2.4 Front-end

The administrator front was created to feed the database because DataGouv database only had ingredients but we needed recipes. We created 68 recipes with this frontend. It was programmed in HTML and Javascript with a Vuejs framework.

The android front contains a log-in activity, a subscription form activity, a weight control graph activity, a food data activity, a meal maker activity and two setting activities. It was coded i Java and XML using Google's IDE, Android Studio. You will find some of the activities in Figure 3 at page 2.

3 AI Algorithms

In this section we will describe how we used AI to select the best recipes for the user.

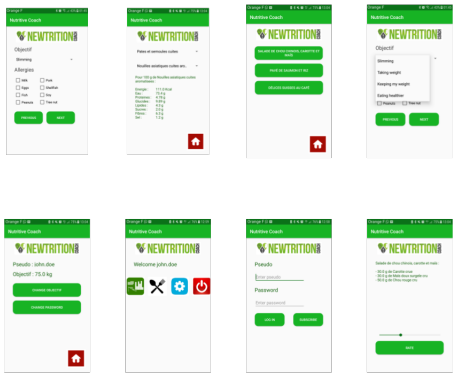

Figure 3: Application design

3.1 Knowledge and Reasoning algorithm

A first selection among the recipes is made with an expert system[1]. This expert system was implemented in Drools and followed the rules in Table 1 at page 2. Considering that the weight and the need in calories are two Gaussian curves. We apply an linear function to the mean of the current weight and the wanted weight.

Calories\Rate	0-4	4-7	7-10
0%-50%	No	No	No
50%-70%	No	No	Yes
70%-130%	No	Yes	Yes
130%-150%	No	No	Yes

Table 1: Expert system rules

3.2 Problem Solving Algorithm

We used to model the meal creation problem as Constraint Satisfaction Problem[1] (CSP). The constraint is to find a meal that give the right amount of calories (calculated as explained in last section) with a 10% tolerance. The variables

are an Entrance, a Main Dish and a Dessert and it is solved with a min conflict strategy.

3.3 Learning Algorithm

To predict the rate of a user we use a neural network[1] implemented in java. The input layer takes the following normalized data : user age, user weight, user size, recipe calories, recipe sugar quantity, recipe fiber quantity, recipe water quantity, recipe fat quantity. The output layer is a rate between 0 and 1 that is linearly set between 0 and 10 after the network. The two next layers are fully connected layers of 8 neurals and the last layer is a single neural fully connected to the third layer. You will find the structure in Figure 4 at page 3.

If we would commercialize this application this network would be trained at the beginning of the application with min squared loss and back-propagation optimizer and then used by the users. We could not trained the algorithm because of the lack of data.

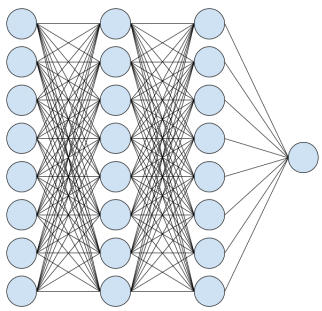

Figure 4: Neural network structure

4 Conclusion

The project result is an ergonomic and a scalable application with a lot of food's data. But the AI part would need more data to get really efficient.

References

[1] Russell, S.J. and Norvig, P., 2016. Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited,.

[2] Middleton, N. and Schneeman, R., 2013. Heroku: Up and Running: Effortless Application Deployment and Scaling. " O'Reilly Media, Inc.".

[3] Arthur, J. and Azadegan, S., 2005, May. Spring Framework for rapid open source J2EE Web Application Development: A case study. In Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005. Sixth International Conference on (pp. 90-95). IEEE.

[4] MySQL, A.B., 2001. MySQL.