**Project Architecture**

In this document we will describe how the project is built, which technologies we used to built it and how they interact.

**Flow & Structure**

**Vagrant file-** Defines the VM the app runs on. Works on ubuntu/bionic64 OS, triggers bootstrap.sh file.

**bootstrap.sh-** updates the OS (if update exists), defines the app environment & installs dependencies. Then runs app.py on localhost:5000

**app.py-** configures the app variables, configures the connection to the DB (based on config.py), runs all app routes (from routing.py) and defines a dictionary that will be used in the extractors scripts

**superx folder**

**information extractors-** two scripts which run on the relevant urls and extract data published by different food chains. branch\_extractor gets the data of the different food branches, and item\_extractor gets the products and their prices. The data is stored in the DB configured in app.py

**models-** contains \_\_init\_\_.py, a file which determines the different tables columns defined in the DB. Post & Get methods to the DB requires importing models folder

**routes-** home.py & signup.py, define the app's functionality from user sign in to searching and comparing products.   
- Rendering html files from templates dir.  
- Based on script.js & login\_script.js from static/js

**static-** images, CSS styling & JS scripts of the website

**templates-** html & jinja2 files of the app's different pages

**tests-** tests to the different functionality of the system. To run the tests-   
1. Activate the VM  
2. Activate superx venv  
3. Go to /vagrant/superx folder (not listed with dir command)  
4. Run "python -m pytest"

**.github/workflows**

Pylint.yml- CI for running all relevant tests whenever a PR is made

**More files**

**README.md**- description of the project's purpose, how it works and main contributors

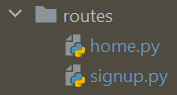
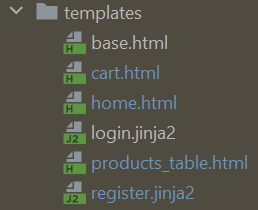
**DB Design.jpg**- the data structure of the project including relationships between tables

**Contributing.md-** contribution guidelines to the project

**.vagrant folder**- all relevant data to the VM

**.gitignore**- irrelevant files (github will ignore them)

**App run diagram**

**vagrant up**

shell script to create venv

Create DB on cloud

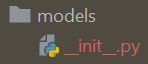
Cloud clouster DB



app routes

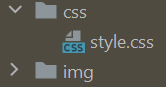
Use models and commit to DB

Define tables

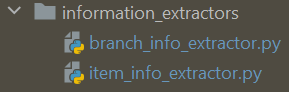


Pages functions

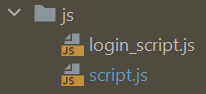
UI styling



render templates

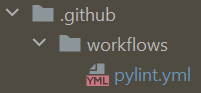


On click functions



Extract Data

**GitHub Actions**



Run on PR