Report
Coursework Assignment #2
Advanced Web Technologies (SET09103)

Student : Garance Récalde
Matriculation Number : 40219045
Deadline of submission : 29st November 2015
Module Leader : Simon Wells

Introduction

Description of the web-app

The web app I worked on for this second coursework was the one I started on the first coursework : **an outline for an extensive lifestyle related blog**. However for the coursework, I decided to only only develop the *Recipes* part. So it is still a work in Progress.
This time, I tried to implement **more complex features** (like the Login) and also some **details to provide a better User experience (Display of Flash messages).**

The **3 main features** I have been adding are :

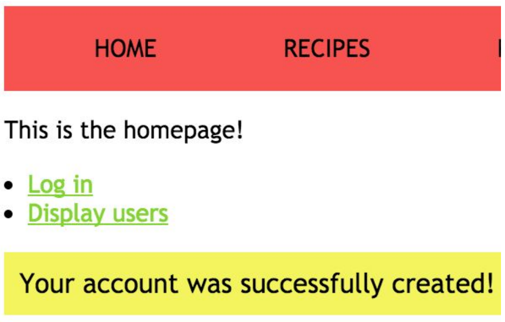
1. Using database with **sqlite3**, and the way to **display the data** in the **HTML template** by getting it from the **database** (see *recipes/breakfast/pancakes/1)* instead of using **Collections** in python like I did for the coursework #1.

```
@app.route('/recipes/breakfast/pancakes/')
@app.route('/recipes/breakfast/pancakes/<int:id>')
def pancakes(id=None):
  #ingredients = ['2 eggs', '1 cup oats', '200g yoghurt']
  #toppings = ['Banana','Blueberries','Fig','Raspberries','Honey','Chocolate','Nuts']
  recipe = get_recipe(id)
  return render_template('pancakes.html', recipe = recipe)
  #return render_template('pancakes.html',ingredients=ingredients,
  #toppings=toppings, recipe=recipe)
```

```
<h2>Ingredients</h2>
 {% if recipe %}
 <ul>
   <li>
     {{ recipe.ingredients_recipe }}
   </li>
 </ul>
 {% endif %}
```

```
<h1>
  {% block title_recipe %}
    {% if recipe %}
      {{recipe.title_recipe }}
    {% endif %}
  {% endblock title_recipe %}
</h1>
```

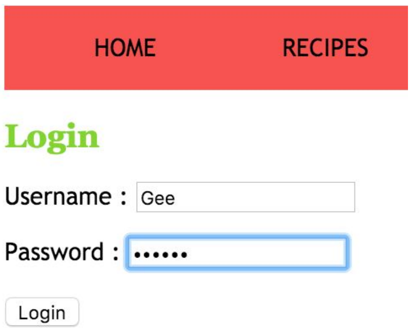
2. **Create an account** + **Login** feature for users (implemented using **Flask-Login**)



This is the homepage!

- Log in
- Display users

Your account was successfully created!

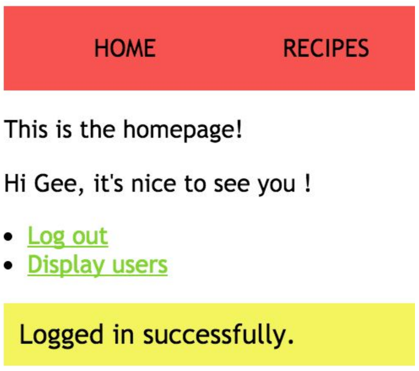


Login

Username : Gee

Password : ••••••

Login



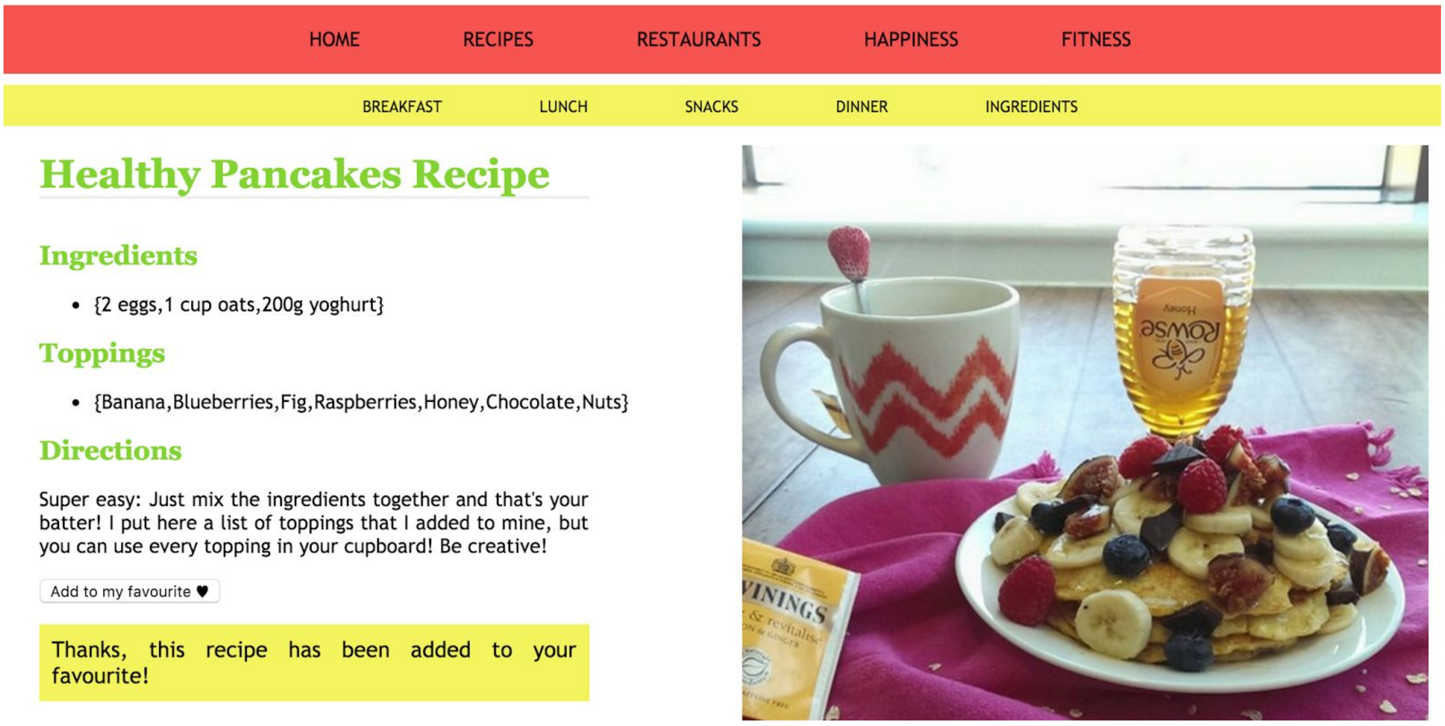
This is the homepage!

Hi Gee, it's nice to see you !

- Log out
- Display users

Logged in successfully.

3. Adding recipes to a “favourite” list


HOME RECIPES RESTAURANTS HAPPINESS FITNESS

BREAKFAST LUNCH SNACKS DINNER INGREDIENTS

Healthy Pancakes Recipe

Ingredients

- {2 eggs,1 cup oats,200g yoghurt}

Toppings

- {Banana,Blueberries,Fig,Raspberries,Honey,Chocolate,Nuts}

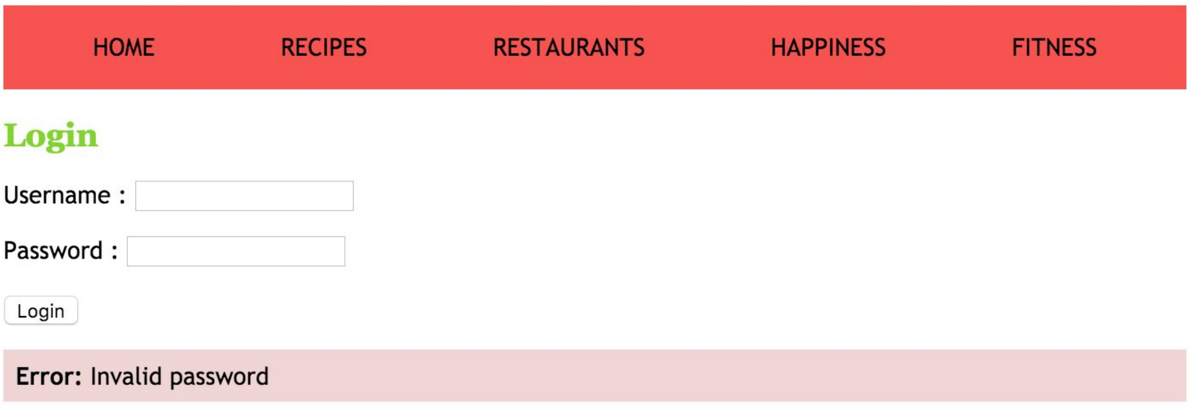
Directions

Super easy: Just mix the ingredients together and that's your batter! I put here a list of toppings that I added to mine, but you can use every topping in your cupboard! Be creative!

Add to my favourite ♥

Thanks, this recipe has been added to your favourite!

Some **appropriate error messages** by **Login**:


HOME RECIPES RESTAURANTS HAPPINESS FITNESS

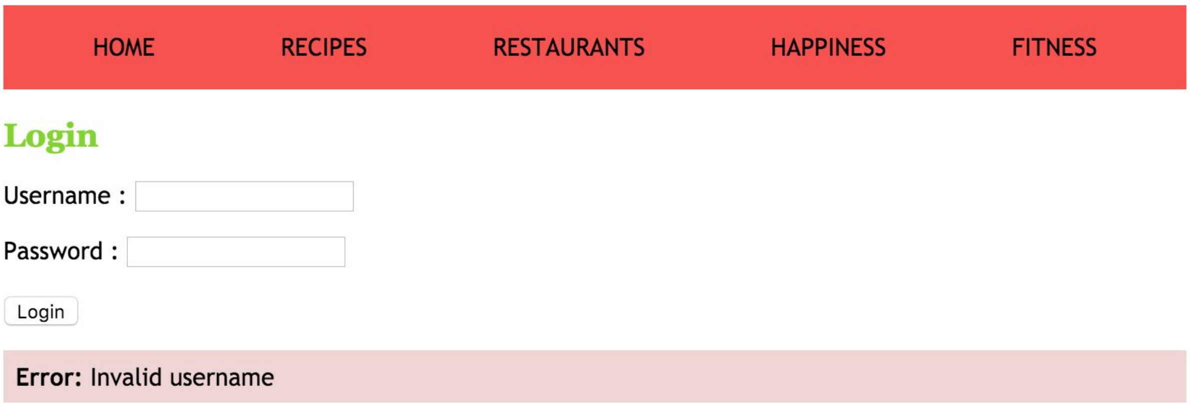
Login

Username :

Password :

Login

Error: Invalid password


HOME RECIPES RESTAURANTS HAPPINESS FITNESS

Login

Username :

Password :

Login

Error: Invalid username

Design

How I architected my web-app

I have architected my **web app** in different **categories** :

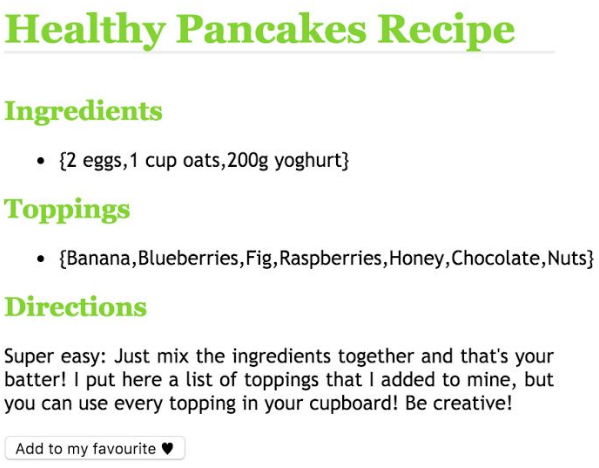
1. The first menu includes categories such as : *Home, Recipes, Restaurants, Happiness, Fitness*. Those aim to be categories where a collection of related articles could be found.

2. Under the *Recipes* category, the second menu is classed by **type of meal** (Breakfast, Lunch, Snacks, Dinner) but I thought it would also be interesting to have a section “**Ingredients**”

where food would be classed by their food group. The 5 different **food groups** are : *Carbs, Veggies, Fruits, Dairy, Protein* and then you have *Healthy Fats* as well.

3. In the *Restaurants* category, I thought I would class things by **cities**, so I have architected it like so with the cities I have tested restaurants in so far / or where I am planning to go very soon : *Bordeaux, Berlin, NYC, Edinburgh, Amsterdam*. However, like I said before I haven’t develop this part.

This is then how a **typical recipe** look like :



Healthy Pancakes Recipe

Ingredients

- {2 eggs,1 cup oats,200g yoghurt}

Toppings

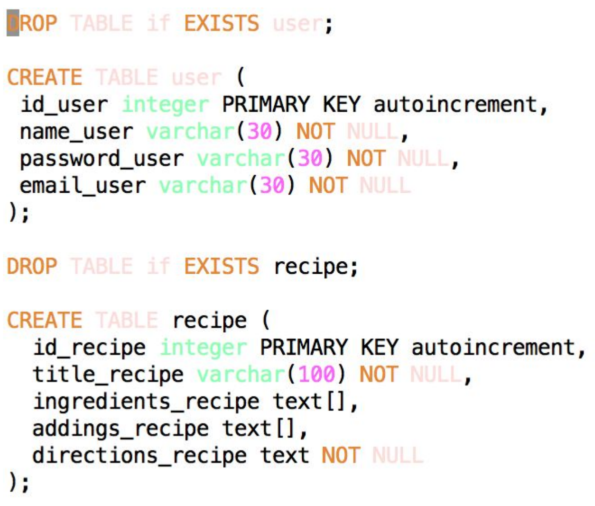
- {Banana,Blueberries,Fig,Raspberries,Honey,Chocolate,Nuts}

Directions

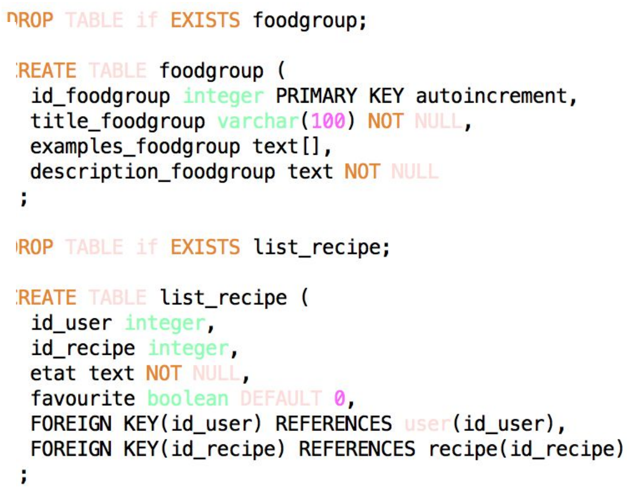
Super easy: Just mix the ingredients together and that's your batter! I put here a list of toppings that I added to mine, but you can use every topping in your cupboard! Be creative!

See the **coursework#1 report** for **other routes** not directly related to the web app and not included in the visible menu.

Database

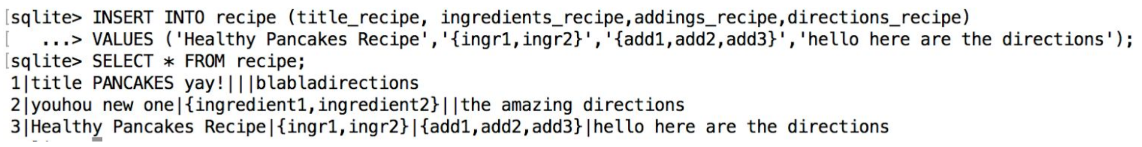


```
DROP TABLE if EXISTS user;

CREATE TABLE user (
 id_user integer PRIMARY KEY autoincrement,
 name_user varchar(30) NOT NULL,
 password_user varchar(30) NOT NULL,
 email_user varchar(30) NOT NULL
);

DROP TABLE if EXISTS recipe;

CREATE TABLE recipe (
  id_recipe integer PRIMARY KEY autoincrement,
  title_recipe varchar(100) NOT NULL,
  ingredients_recipe text[],
  addings_recipe text[],
  directions_recipe text NOT NULL
);
```

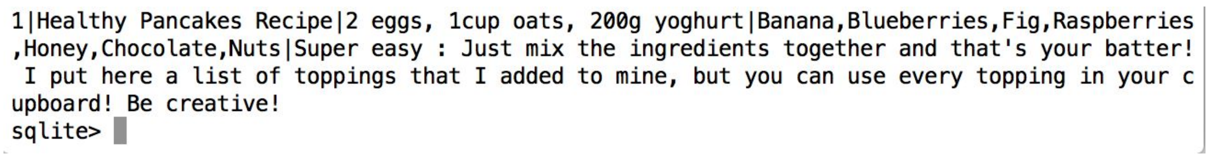


```
DROP TABLE if EXISTS foodgroup;

REATE TABLE foodgroup (
 id_foodgroup integer PRIMARY KEY autoincrement,
 title_foodgroup varchar(100) NOT NULL,
 examples_foodgroup text[],
 description_foodgroup text NOT NULL
;

ROP TABLE if EXISTS list_recipe;

REATE TABLE list_recipe (
 id_user integer,
 id_recipe integer,
 etat text NOT NULL,
 favourite boolean DEFAULT 0,
 FOREIGN KEY(id_user) REFERENCES user(id_user),
 FOREIGN KEY(id_recipe) REFERENCES recipe(id_recipe)
;
```

Database Schema

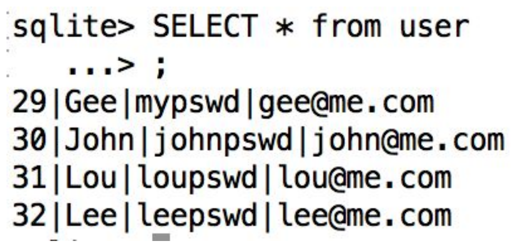


```
sqlite> INSERT INTO recipe (title_recipe, ingredients_recipe,addings_recipe,directions_recipe)
   ...> VALUES ('Healthy Pancakes Recipe','{ingr1,ingr2}','{add1,add2,add3}','hello here are the directions');
sqlite> SELECT * FROM recipe;
1|title PANCAKES yay!|||blabladirections
2|youhou new one|{ingredient1,ingredient2}||the amazing directions
3|Healthy Pancakes Recipe|{ingr1,ingr2}|{add1,add2,add3}|hello here are the directions
```



```
1|Healthy Pancakes Recipe|2 eggs, 1cup oats, 200g yoghurt|Banana,Blueberries,Fig,Raspberries
,Honey,Chocolate,Nuts|Super easy : Just mix the ingredients together and that's your batter!
 I put here a list of toppings that I added to mine, but you can use every topping in your c
upboard! Be creative!
sqlite> 
```

Content of the recipe table



```
sqlite> SELECT * from user
   ...> ;
29|Gee|mypswd|gee@me.com
30|John|johnpswd|john@me.com
31|Lou|loupswd|lou@me.com
32|Lee|leepswd|lee@me.com
```

Content of the user table

Enhancements

Features I would add or improve :

See **errors not resolved** (Displaying content in a certain way, Getting content into the database by scripts ...) (⟶ Critical evaluation)

From Login Feature:

- Add the **Remember Me** feature
- Link "Forgot your password?" to be displayed if people fail at typing their password + actually implement that feature
- Adding everything related to **encryption** of the password
- Check **Cookies** and **Sessions** more closely
- Checking if the password and the confirmation password field or correct by the Create an account feature
- Implement properly the upload of a file by Creating an account + allocate an appropriate name to the file
- Have something displayed for example in the top-left corner with the name and the picture of the current user
- Create a "My account section" where the user could be able to see his details, the recipes he has in his list, update his profile picture ...
- Make the forms look nicer (buttons ...)
- Get a proper redirect and appropriate error message when trying to access a page only accessible when logged it (to customize with Flask-Login?)

Database:

- Extend the database schema to Restaurants ... (+ list restaurants)
- Add a table with text fields state for the items in the lists (e.g. "To Try", "Love it", ...)
- Add feature "Remove from my favourite" ⟶ DELETE from the table in the database
- + Change the aspect of the button "Add to my favourite" once the recipe has been added
- Try to implement SQLAlchemy? compatible? (seemed to be used a lot with Flask-Login)
- Try to retrieve the current/previous-page from the URL? (instead of passing it manually from the route python to the template)

From Coursework #1:

- First, I think I would like to take more time to **make things look nicer** : work more with **CSS** and **Javascript** to add some **nice effects** and provide a **better User Experience** e.g things like a nice **hover** with a **fade effect**, highlighting the active menu items ...
- Make the design **responsive** : maybe see what could be available with **Bootstrap** ?
- *Beauty is in the details* : Add a **logo**, a **favicon**, some **nice icons** for the bullet points (strawberries?)
- Better home page with an **introduction text**, and some **links** to the **other categories**
- Implement an **Instagram API** to have the Instagram feed included to the website
- Add a **footer** to the main template (would contain links to **social networks**), which would be reused in all other pages
- Add a cliquable **breadcrumb**

- Add way **more content** and write **real articles** ⟶ being able to display the latest articles on the home page
- Add a **search bar**
- Organize the static folder properly
- HTML pages content for different cities from “Restaurants” category
- Develop the **other categories** as well
- Figure out a way to dynamically provide the title of the page using the defined URL route, maybe with URL Variables? ⟶ using database
- Use a **database** for images?
- Add more **appropriate error responses** and design nice / funny pages for them with links to **redirect** users to other pages of the website (e.g. “Sorry, darling, the page you requested couldn’t be found. But you might want to check out this awesome carrot cake recipe or go back to the home page.”)
- Really implement this **Newsletter** subscription and link it to a **Database** to store the **users’ names** and their **e-mails** (w/ MailChimp and SQL/phpmyadmin)
- **Sessions** for users : Add possibility for visitors to create an **account** and to **log in** ⟶ which would enable them to **save recipes**, add restaurants to list e.g. “To Try”, “Love it”, able to write comments …
- Add more GET & **POST requests** + investigate how to generate **dynamic file names** or let the user choose the name of the picture he is posting
- Make a **Contact page** enabling visitors to send a message through the website
- Make my report look nicer

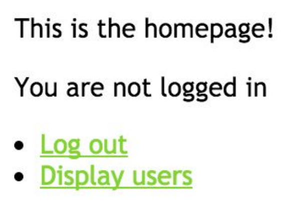
Critical Evaluation

Critical evaluation of my web app

I think I made a more appropriate use of **requests**, **redirects** and **responses** by adding the feature a **login** with for example allowing the person to upload a picture of themselves…
I think that I have **well designed** the **URL hierarchy** and the **associated routes**. For the category I decided to develop here (*Recipes*), I provided for each route an **appropriate response** with a **template HTML page**.
I think I have used the **template inheritance** properly with for example *pancakes.html* who extends *recipes.html*.
To make the web app more complete, I could have provided **appropriate HTML template pages** for each of the route that are present in the app (e.g. for the categories *Restaurants, Happiness, Fitness* ...).
The aim would be to first create the tables in the **database** for those categories (as I did for the *Recipes* part) and then be able to **display** all the **content** in the **templates** by getting all the **data** from the **database** (as I did in *recipes/breakfast/pancakes/1*).
I think I well extended the webapp by adding everything related to **database** as it is a more **flexible** and **efficient method for handling data**.

Some **specific examples** of problems I have been caught with (and haven't resolved):

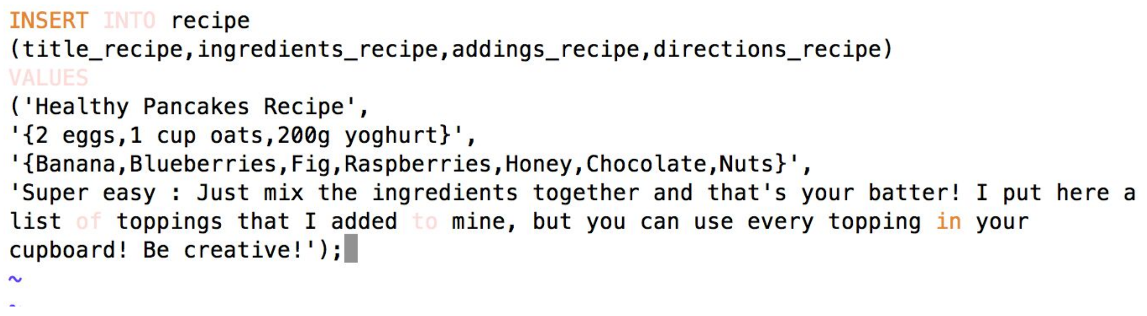
- A problem by getting the **current_user of the session** : I am still wondering if I implemented the **session** feature right or if my website is (partly) remembering me when I log in just by the cache available in my browser ?
  See non-sense "You are not logged in" / "Log out", both being displayed if a specific condition of **session** / being logged in or not is fulfilled.


This is the homepage!

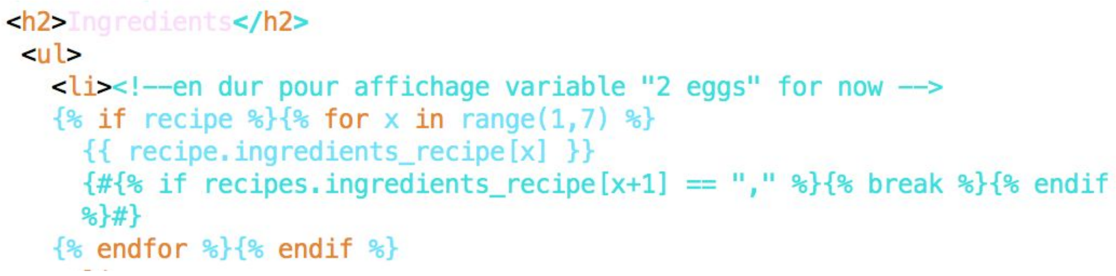
You are not logged in

- Log out
- Display users

- Problem with **importing script in sqlite3** : Mainly the reason why I haven't implemented more content. I got **many errors** (e.g.: *"expected 5 columns but found 1 - filling the rest with NULL"*) by trying to import scripts I had been writing to fill my database with some content. (And I still haven't fix the problem)

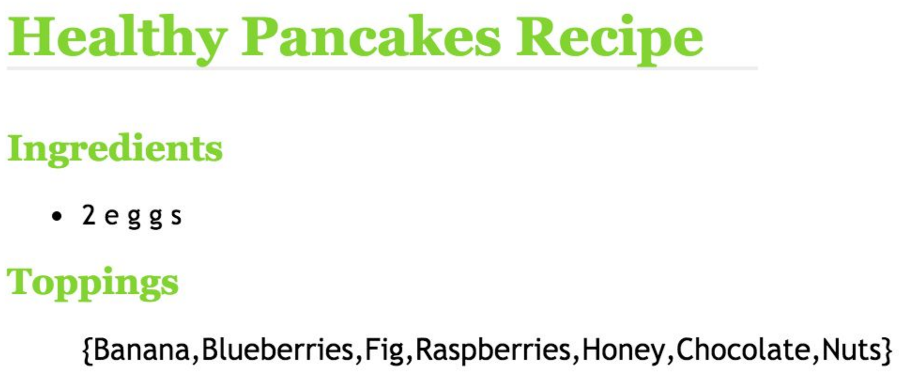

```
INSERT INTO recipe
(title_recipe,ingredients_recipe,addings_recipe,directions_recipe)
VALUES
('Healthy Pancakes Recipe',
'{2 eggs,1 cup oats,200g yoghurt}',
'{Banana,Blueberries,Fig,Raspberries,Honey,Chocolate,Nuts}',
'Super easy : Just mix the ingredients together and that's your batter! I put here a
list of toppings that I added to mine, but you can use every topping in your
cupboard! Be creative!');
~
```

- Display the **ingredients** of a **recipe** as a **list** : I put a **ARRAY** type as I thought it would be more suitable than a text value for the column *ingredients* of a *recipe.* However, I soon realized that an element of an array is a character, not a word so I have been having some trouble with that and experiencing different things to try to get it work. (JInja2, loops, break ...) After spending a few hours on hours, I ended up deciding it wasn't that bad after all if it wasn't displayed as a list :D. However, a better solution would be: having other tables in the database schema, called "ingredients", "quantity", "unity" et get the recipe's ingredients by linking those tables together with foreign keys. It would definitely be more suitable if the website grows and if many recipes are using the same ingredients. And one single ingredient should then be more handy to display.



```
<h2>Ingredients</h2>
 <ul>
   <li><!--en dur pour affichage variable "2 eggs" for now -->
   {% if recipe %}{% for x in range(1,7) %}
     {{ recipe.ingredients_recipe[x] }}
     {#{% if recipes.ingredients_recipe[x+1] == "," %}{% break %}{% endif
     %}#}
   {% endfor %}{% endif %}
```

A try : when encounter a comma please go out of the function and start another item of the list (Trying to have a regular list of ingredients, like I used to have with the Collection passing in python).



Healthy Pancakes Recipe

Ingredients

- 2 e g g s

Toppings

{Banana,Blueberries,Fig,Raspberries,Honey,Chocolate,Nuts}

Personal Evaluation

What I learned, the challenges I faced and the methods I used to overcome them

I definitely kept on **improving** my **skills** in using **Python Flask**, **Levinux** and all the tools and **Environment** we had to use to continue **building the web-app**. I am getting more **familiar** with it by **practicing**. I am now used to **shortcuts** that are really useful to me (e.g the Copy and Paste, within Levinux, as well as when the copy comes from an external source).

I have also become more familiar with **Github**, that I have been using to keep track of my code and I like to work that way, enabling to work from different places with different computers and still have the updated working version! Plus it is also really interesting when other people (like the teacher for example...) who wants to help you finding bugs or what's wrong in your code, then you can share your work easily by simply sharing your **repository URL**.

I must admit I had more **problems** than I had for the 1st coursework which consisted in building the web app :
Multiple little mistakes as well as other **more consequent errors** which came by adding **more features to the web app**.
Methods I used to overcome these problems :

- Review the code, read the **error messages,** make the most of the **debug mode** (when it's properly working...), correct the syntax and indentation errors
- Ask help to another student more advanced
- Ask help to *Google*, find help with *Stackoverflow*
- Ask help to the teacher

However, I definitely feel **more confident** about coding and I know that I am actually capable of **building nice things**, I just have to **keep working on it**! 💪

Resources / References

Summary of Resources and List of References

Food related

- GOOGLE :D for some images
- My account on Instagram, @garancerecalde, https://instagram.com/garancerecalde/
- Deliciouslyella.com
- Blog of Kaylaitsines.com
- Pinterest.com
- *Carrot and Butternut squash soup*, Goodtoknow.co.uk http://www.goodtoknow.co.uk/recipes/536447/carrot-and-butternut-squash-soup
- *Coconut mint matcha cake and a matcha gift box giveaway*, Talesofakitchen.com, http://talesofakitchen.com/desserts/coconut-mint-matcha-cake-and-a-matcha-gift-box-giveaway/

Code related

Flask-Login implementation
Sqlite3
From Coursework #1

- Example of a menu, Dairy of Daybook @siwells github, https://github.com/siwells/teaching_set09103/blob/master/case.study/daybook/src/templates/diary.html
- *Raccourcis clavier Vim*, Commentcamarche.net, http://www.commentcamarche.net/faq/8400-raccourcis-clavier-vi-m
- *Un menu horizontal*, Openclassroom.com, https://openclassrooms.com/courses/un-menu-horizontal
- *Les templates*, Créez vos applications web avec Flask, Openclassroom.com, https://openclassrooms.com/courses/creez-vos-applications-web-avec-flask/les-templates-4
- *Vim, l'éditeur de texte du programmeur*, Openclassroom.com, https://openclassrooms.com/courses/reprenez-le-controle-a-l-aide-de-linux/vim-l-editeur-de-texte-du-programmeur
- *[VIM] Coller un texte depuis l'extérieur*, Forum Debian-fr.org, https://www.debian-fr.org/coller-un-texte-copie-depuis-l-exterieur-t40920.html
- *Template Designer Documentation*, Jinja.pocoo.org, http://jinja.pocoo.org/docs/dev/templates/
- *Jinja2 if statement in vs equals on dict*, Stackoverflow, http://stackoverflow.com/questions/22238265/jinja2-if-statement-in-vs-equals-on-dict