MiAM.

Projet mobile

HELB 2021-2022

Al haddaoui Marwane

RAPPORT D'ITÉRATION 4

INTRODUCTION :

Dans le cadre du cours de projet mobile, une application Android nous a été demandée d'être créé, vis-à-vis de la thématique de l'application mon choix c'est naturellement porté vers une application de Delivery Food (une application de livraison de nourriture), l'application se nomme miam ,elle a pour but de simuler une livraison de n'importe quel type de nourriture entre un client sur l'application et un restaurant ou un magasin. Elle proposera également une rubrique recette afin d'inciter l'utilisateur à l'achat des produits alimentaires autres que la nourriture fast fast-food(achat de légume, fruits, etc.)

FEEDBACK :

La dernière review du projet témoigne toujours de la régularité quant à la qualité des rapports remis, un conseil a cependant été émis quant à certaines formulations utilisées comme par exemple "90 % du projet est terminé" qui dévaluent la qualité du travail et altère le caractère sérieux du rapport.

DESCRIPTION DE L'ÉTAT ACTUEL DU PROJET :

Le projet est actuellement dans un stade avancé, toutes les fonctionnalités hypothétiques qui ont été énoncé aux prémices du projet on était créé ainsi que certaines fonctionnalités supplémentaires sont disponibles, la partie api qui été manquante à la dernière itération et présente et fonctionnelle, l'application est donc dans un stade de pré-version voire de bêta

MODIFICATION :

a) Implémentation de la partie api.

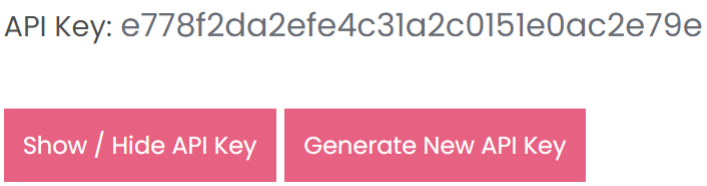
La partie api a été la branche principale de cette itération,pour rappel le parti api porté sur des recettes à utiliser au sein de l'application, dans cette section son fonctionnement ainsi que son utilisation sera vue de manière détaillée :

a.1)Fonctionnement :

L'api spooncular, génère un fichier json résultant d'une requête exprimée par le billet d'un lien web (URL) combiné avec un token de sécurité disponible suite à la création d'un compte sur le site de l'api, la requête diffère à l'aide de combinaisons possibles dans ce lien, la documentation est très explicite et propose également des requêtes complexes tels que la génération d'un nombre de recette de manière aléatoire.

Exemple d'utilisation :

1) *Token présent sur le site spooncular dans la partie profile*



2) Requete que l'on veut exécuté

Get Random Recipes

Find random (popular) recipes. If you need to filter recipes by diet, nutrition etc. you might want to consider using the complex recipe search endpoint and set the `sort` request parameter to `random`.

```
GET https://api.spoonacular.com/recipes/random
```

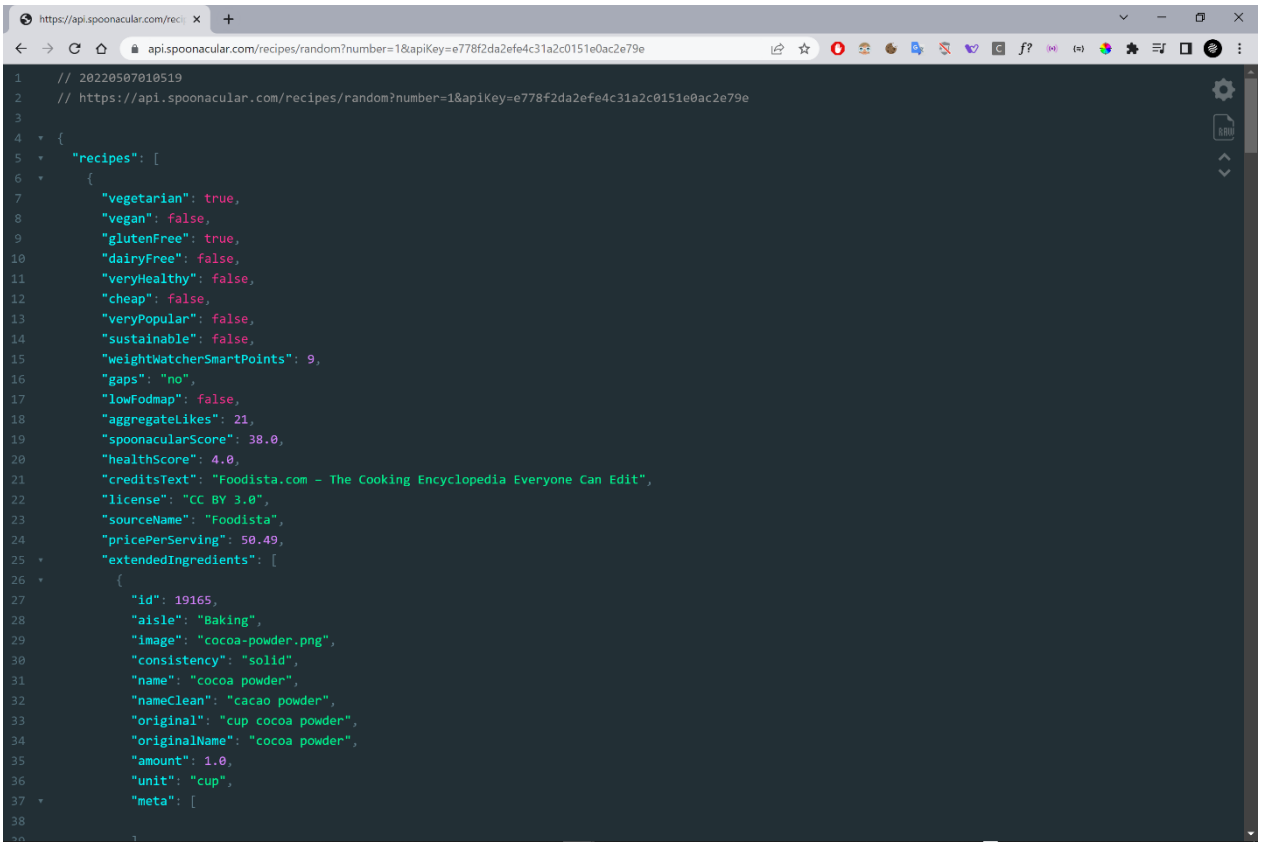
Note : Ici la requête utilisée génère une recette de manière aléatoire, toutes les requêtes disponibles ont présenté dans [la doc de l'api](#).

3) Concaténation de la requête et du keycode

```
Ici :
https://api.spoonacular.com/recipes/random?number=1&apiKey=e778f2da2efe4c31a2c0151e0ac2e79e
```

Note : L'ajout du paramètre ?number permet d'indiquer le nombre de recette à générer.

4) Exécution de la requête



Explication :

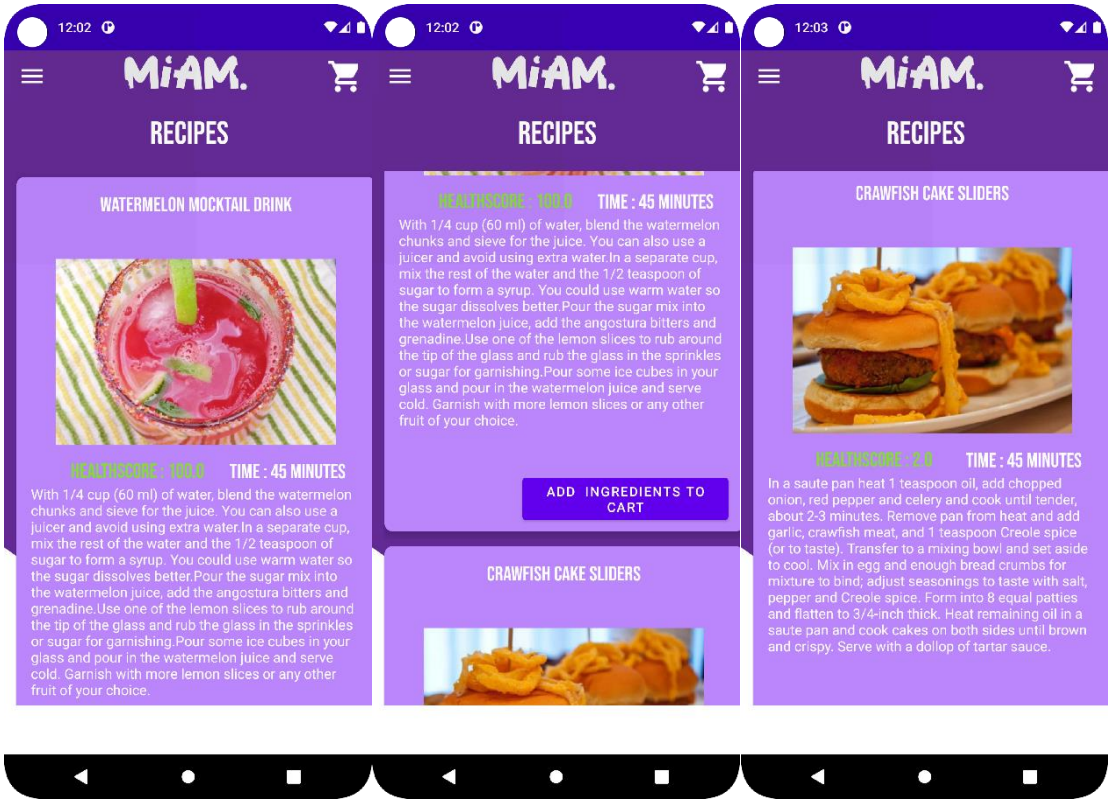
Un fichier json est généré comprenant des informations exploitables à la suite d'un traitement adéquat, ici la mise en forme de la page web est gérer par une extension google chrome ,[disponible ici](#).

Note : L'api permet 150 requêtes par jour dans sa version gratuite.

a.2) Manipulation de l'api dans l'application :

Suite à l'explication sur l'utilisation de l'api de façon traditionnelle, dans cette partie, une explication sera fournie quant à son utilisation au centre du projet suivis de sa représentation dans l'interface utilisateur.

1)Partie graphique :



Explication :

Lorsque l'utilisateur clique sur la partie recipes, il arrive sur une page comprenant une liste de 5 recettes générer de manière aléatoire, avec le nom de la recette,le temps de préparation et également le nutri-score (suite à votre demande lors de l'itération 1) a noté la présence d'un bouton add ingrédients To cart qui permet d'ajouter les ingrédients de la recette au panier.

2)Partie technique :

Afin d'utiliser l'api dans l'application,la requête qui génère le fichier json est stocké dans une variable,afin de faciliter sa modification elle est déclarée comme un attribut de la classe.

```
String url ="https://api.spoonacular.com/recipes/random?number=5&" +
                "&apiKey=e778f2da2efe4c31a2c0151e0ac2e79e";
```

Ensuite à l'aide de l'outil Volley,une requête Get combiné avec la variable qui stocke la requête de l'api est exécuté afin de récupérer l'objet Json,de le stocker dans une variable pour le manipuler (récupérer le nom de la recette, le nutri-score ,la recette et ses ingrédients).

```
RequestQueue queue= Volley.newRequestQueue(RecipesActivity.this);
JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.GET, url,
        null, new Response.Listener<JSONObject>() {
```

A posteriori , une reconversion de l'objet json en arrayjson est nécessaire pour récupérer toutes les recettes car elles sont dans un fichier json unique et non scindé.
La séparation faite,cette même liste est donc parcourue afin de récupérer chaque champ voulu ici extendedingredients pour la liste d'ingrédients de chaque recette, name pour les noms des ingrédients et est stockée dans une liste d'objet de type ingrédient.

```
@Override
public void onResponse(JSONObject response) {
    try {
        JSONArray recipes= response.getJSONArray("recipes");
        List<IngredientData> list= new ArrayList<>();
        for(int i=0;i<recipes.length();i++) {
            JSONObject currentrecipes = recipes.getJSONObject(i);
                JSONArray
ingredientsrecipes=currentrecipes.getJSONArray("extendedIngredients");
                list=new ArrayList<>();
                for(int j=0;j<ingredientsrecipes.length();j++ ){
                    Random r =new Random();
                    int price= r.nextInt(10);
                    IngredientData ingredient = new IngredientData("$",

ingredientsrecipes.getJSONObject(j).getString("name"),
                            String.valueOf(price),
                            "1");
                    list.add(ingredient);
                }
```

Note : cette partie est nécessaire afin d'ajouter les ingrédients dans la liste des objets a acheté, comme l'api ne comprenait pas des éléments tels que le prix ou l'unité des ingrédients, une génération aléatoire a été conclue pour le prix et pour la quantité elle est toujours de 1.

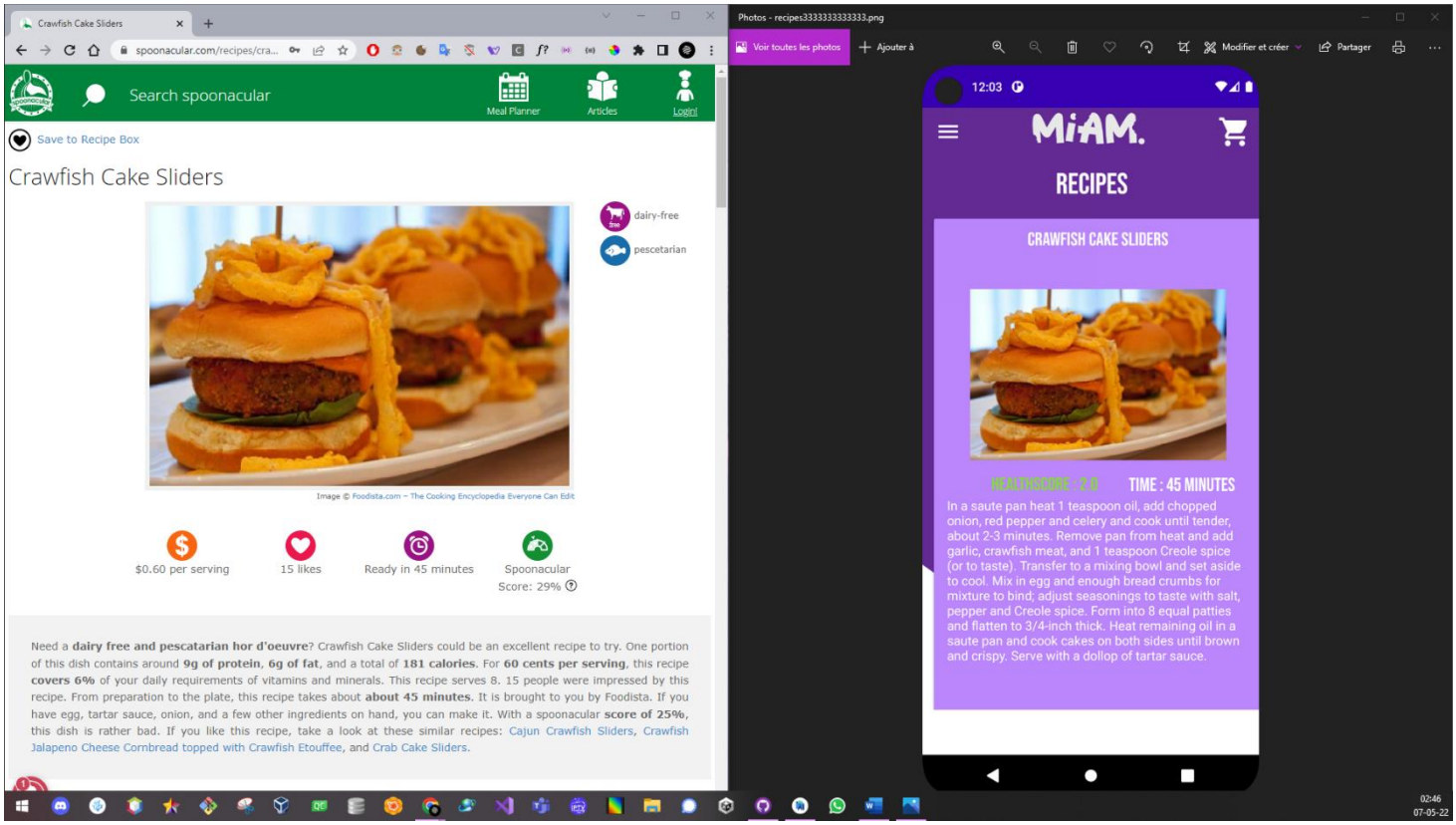
Finalement,le reste du contenu a affiché tel que le nom de la recette,son nutri-score ,le temps de préparations, ainsi que les instructions sont récupérées en suivant la même mécanique que le code présenter ci-dessus.
Dans le parcour de l'array Json ,les données adéquates sont récupéré en indiquant le champ du fichier json voulu,(tel que "titre" pour le titre de la recette, "healthscore" pour le nutri-score etc…),ces mêmes données sont stockées dans un objet de type recipesData comprenant toutes ses valeurs, objet qui va être stocké dans une liste d'objets afin de l'adapter dans une cardview et l'affiché.

Code

```
RecipesData r = new RecipesData(currentrecipes.getString("title"),
        "HealthScore : "+currentrecipes.getString("healthScore"),
        "Time : "+currentrecipes.getString("readyInMinutes")+" minutes",
        currentrecipes.getString("image"),
        Html.fromHtml(currentrecipes.getString("instructions")).toString(),
        list);


recipeslist.add(r);
recipesadaptater.notifyDataSetChanged();
```

Résultat



Problèmes rencontrés :

Texte issu de l'api comprenant des balises HTML(<p>,<HTML>, etc.)

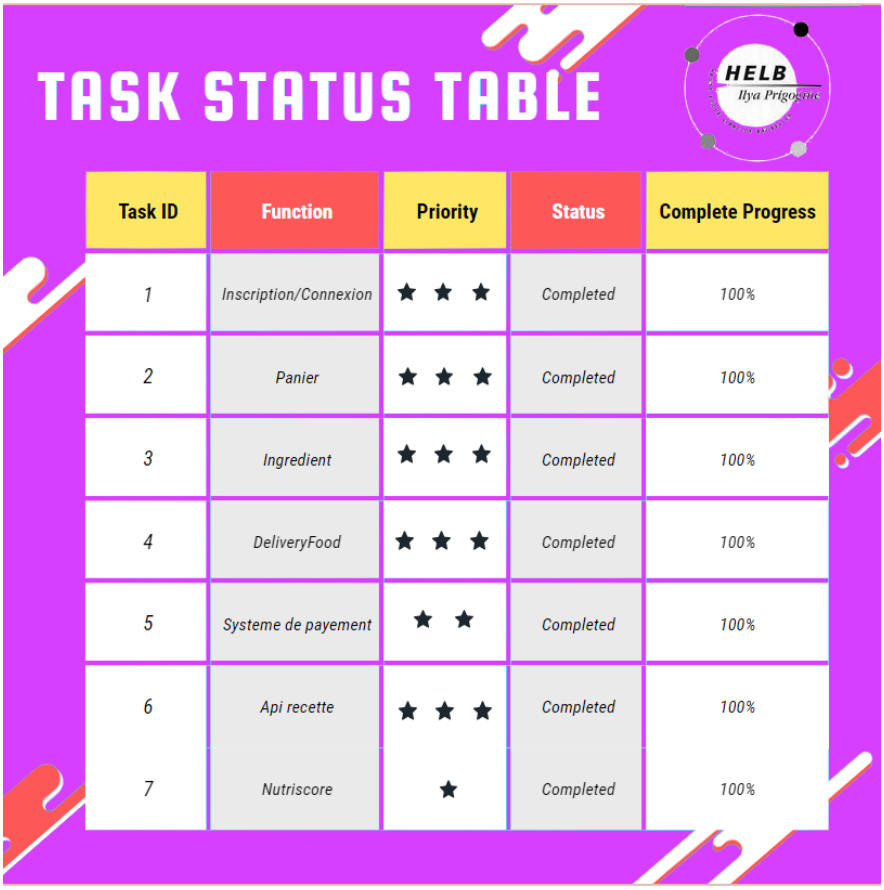
Solution : utilisation de la function Html.fromHtml.

Remarque personnelle :

L'ajout d'un bouton qui vide tout le panier avait été fait, il vidait tout le panier du client, seulement suite à un questionnement personnel quant à les bien faits de sa présence au sein de l'application, sa présence a été conclue comme contraire à la volonté de l'application, en effet au cours de ce projet un aspect de la programmation est apparue au sein de ma réflexion, , certaines fonctionnalités peuvent être très intéressantes d'un point de vue utilisateur mais elles peuvent également aller à l'encontre de la volonté de l'application, ici le but de l'application est de fair acheter le client en maximisant son expérience utilisateur sans pour antant proposer une facilité à ne pas acheter.

Afin d'illustrer mon propos, l'exemple de Facebook me semble pertinent, si l'onglet supprimé mon compte Facebook était facile d'accès, plus d'utilisateurs supprimeraient leur compte.

Liste des taches



Task ID	Function	Priority	Status	Complete Progress
1	Inscription/Connexion	★ ★ ★	Completed	100%
2	Panier	★ ★ ★	Completed	100%
3	Ingredient	★ ★ ★	Completed	100%
4	DeliveryFood	★ ★ ★	Completed	100%
5	Systeme de payement	★ ★	Completed	100%
6	Api recette	★ ★ ★	Completed	100%
7	Nutriscore	★	Completed	100%

CONCLUSION :

Toutes les fonctionnalités de l'application ont été implémenté ainsi que celle-ci demandée dans le cadre du projet , l'application est dans un stade de bêta, une utilisation prolongée de l'application ainsi qu'une création du cas de test est actuellement en cours afin de pallier tous les problèmes éventuelle , malgré un temps de travail surchargé l'application respecte les contraintes établies au début du projet.

Lien git de l'application : Git