

APPLICATION MOBILE

DOYEN ROMAIN, DUCHEMAN MARQUES, L3 informatique

1^{er} mai 2021

Dodo'Cuisto



Résumé

Vue globale et but de l'application

En tenant compte des différentes exigences requises et notées dans le document de CCTP, nous avons décidé de développer une application de cuisine dont le but principal serait de consulter des recettes de cuisines créoles que d'autres utilisateurs ont postés.

C'est un sujet qui nous a beaucoup inspirés car étant passionné de cuisine, nous voulions faire partager la cuisine réunionnaise à un grand nombre de personne. L'île de la Réunion étant une île quasiment isolée dans l'Océan Indien, nous voulions faire exporter les spécialités locales à travers le monde.

L'application permettra à un utilisateur de :

- **rechercher une recette** : Faire une recherche de recette disponible dans l'application
- **créer un compte** : Il faut obligatoirement s'inscrire pour pouvoir consulter, ajouter, modifier ou supprimer des recettes. Il peut également se déconnecter de son compte **Dodo'Cuisto**.
- **sauvegarder en favoris** : Permet de sauvegarder en favoris les recettes préférées de l'utilisateur.
- **consulter les recettes** : Permet de consulter les recettes affichées sur la page d'accueil
- **modifier les recettes** : Permet d'ajouter, modifier ou supprimer les recettes

Table des matières

1	Introduction	4
1.1	Technologies et outils utilisés	4
2	Présentation de l'application	4
2.1	Fonctionnalités et Interface graphique	4
2.2	Lecture des données avec DB Browser for SQLite	7
3	Les problèmes rencontrés	10
3.1	Quelques souci d'affichage	10
3.2	Problème avec certaines fonctionnalités	10
4	L'un des principaux challenges rencontrés	11
5	L'inconvénient d'un code en MVC	12
6	Quelques points délicats/intéressants	12
6.1	Le stockage des données avec SQLite	12
7	Conclusion	14
7.1	Les points à améliorer	14
8	Bibliographie	14

1 Introduction

Notre application sera réalisée sous les plateformes Android et iOS (la principale version se fera sous Android). Comme annoncé dans le Résumé, le but de l'application est de consulter des recettes de cuisine créoles disponibles dans l'application.

1.1 Technologies et outils utilisés

Nous avons utilisé respectivement les langages **Swift** et **Java** pour développer l'application mobile Dodo'Custo sous **iOS** et **Android**. Nous avons conçu le logo de l'application sur **Adobe Illustrator** et en complément nous avons utilisés **Photoshop**.

Pour satisfaire toutes les fonctionnalités de stockage des différentes informations dans l'application, nous avons décidé de travailler avec la base de données **SQLite**.

Nous avons utilisés la technologie **MVC** (Modèle, vue, contrôleur) pour développer l'application et pour mieux ainsi structurer le code. Nous avons principalement développé l'application sur Android, et pour iOS nous avons fait une simple démo.

2 Présentation de l'application

2.1 Fonctionnalités et Interface graphique

Nous détaillerons ici toute la partie utilisateur : Quelles seront les informations accessibles et comment l'utilisateur pourra y accéder. Tout d'abord, la première fois l'utilisateur se trouvera dans la page de connexion de l'application, il aura le choix entre deux modes : se connecter à un compte déjà existant, ou créer un nouveau compte.

L'utilisateur de **Dodo'Custo** se retrouvera ensuite sur une seconde page proposant différentes fonctionnalités s'il a choisi de créer un nouveau compte : Clic sur "*Créer un nouveau compte ici*" depuis la page de connexion : Des informations de connexion seront demandées à l'utilisateur pour pouvoir créer un compte tel que l'adresse email, mot de passe, nom d'utilisateur et le nom complet. Lorsque l'utilisateur aura donné tous ces renseignements il pourra cliquer sur "*Créer un compte*" et sera redirigé vers l'écran d'accueil de l'application. Si l'utilisateur possède déjà un compte, il pourra alors se connecter à un compte déjà existant via la rubrique "Connectez-vous à votre compte". Il suffira qu'il entre son adresse email et son mot de passe et il sera redirigé vers l'écran d'accueil principal de **Dodo'Custo**.

Sur la page d'accueil se trouve toutes les recettes disponibles que ce soit les plats ou les desserts, l'utilisateur pourra faire glisser à droite ou à gauche pour choisir quelle catégorie de recette il souhaite voir. Il peut également écrire directement dans la barre de recherche, le nom de la recette qu'il souhaite consulter.

Clic sur une recette par exemple "*Rougail Saucisse*" depuis la page d'accueil : Il aura alors le choix de la consulter, la modifier ou la supprimer. Si l'utilisateur décide de consulter la recette, il aura la possibilité de faire glisser à gauche ou à droite pour voir les ingrédients et les instructions de la recette.

Toujours depuis l'écran d'accueil, en cliquant sur les trois petits points en haut à droite de l'écran, l'utilisateur de **Dodo'Custo** pourra ajouter une nouvelle recette ou se déconnecter.

Clic sur "*Nouvelle recette*" depuis l'écran d'accueil : Des informations seront demandées à l'utilisateur pour pouvoir ajouter une nouvelle recette tel que le nom de la recette, une image illustrative et une courte description de la recette. Lorsque l'utilisateur aura donné tous ces renseignements il pourra cliquer sur "Suivant" et il sera redirigé vers une nouvelle page.

Sur cette nouvelle page, il devra ajouter les ingrédients de sa recette. L'utilisateur pourra alors ajouter autant d'ingrédients qu'il le souhaite en cliquant sur "*Ajouter*", et lorsqu'il aurait fini d'ajouter tous les ingrédients nécessaires à sa recette, il cliquera sur "Suivant" et il sera de nouveau redirigé vers une nouvelle page.

Dans cette nouvelle page finale, on doit écrire les instructions pour notre recette. Et comme précédemment, on peut alors ajouter autant d'instructions qu'on le souhaite en cliquant sur

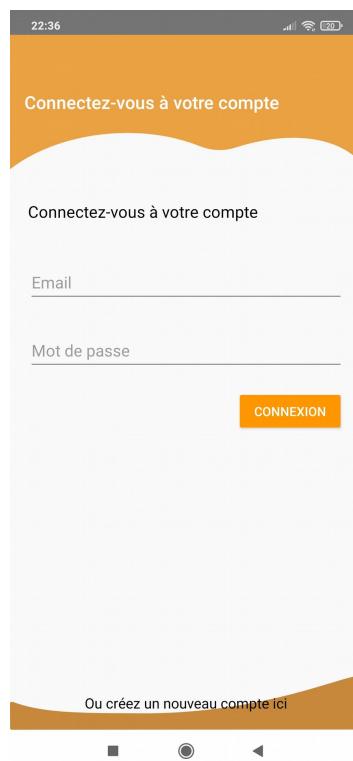
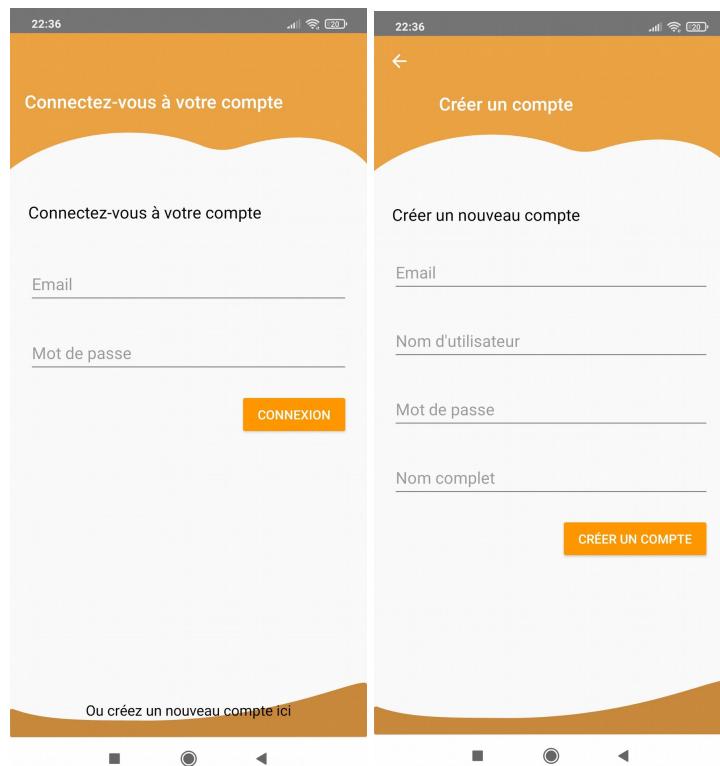
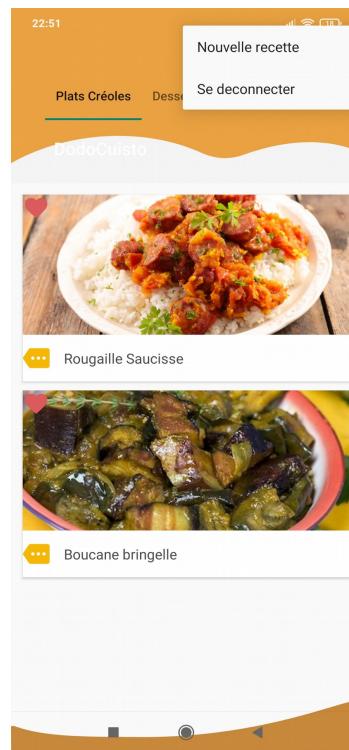
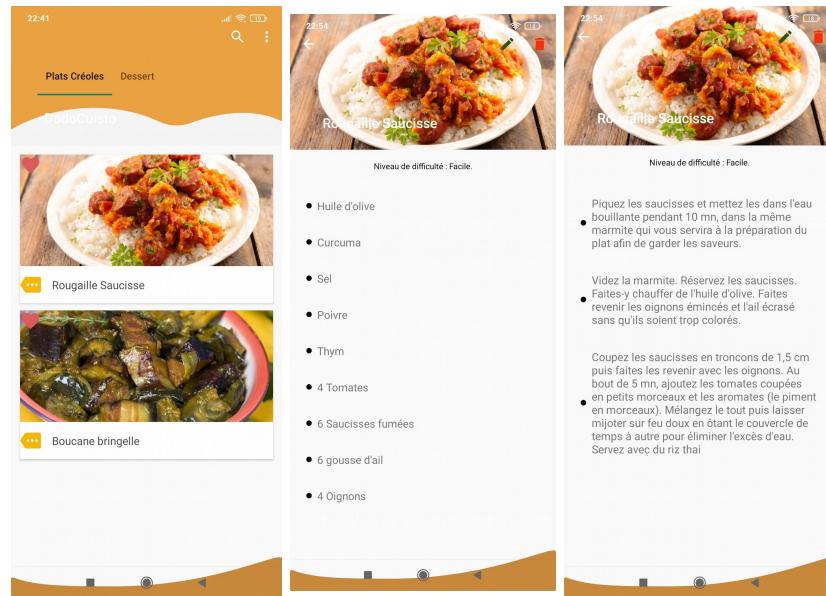
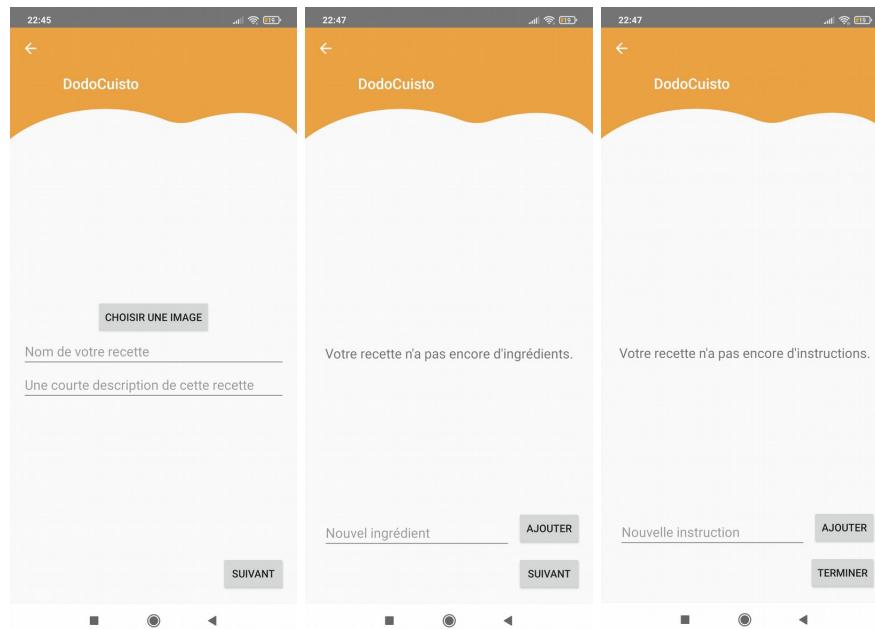


FIGURE 1 – (page de connexion)





"Ajouter", et lorsqu'on a fini d'écrire tout les instructions nécessaire à la réalisation de cette recette, on clique sur "Terminer" et on sera rediriger sur la page d'accueil.

2.2 Lecture des données avec DB Browser for SQLite

Nous avons utilisé le logiciel **DB Browser for SQLite** pour consulter la base de données locale et de s'assurer le bon fonctionnement de celle-ci. Ce logiciel gratuit et libre permet de créer et manipuler des bases de données sans avoir besoin d'un serveur.

Pour consulter notre base de données locale, tout d'abord, nous avons extrait le fichier "*recipesDatabase*", qui contient toute les données de la base de données locale, à partir d'Android Studio. Pour ce faire, sur Android Studio, on se place dans l'onglet "*Device File Explorer*" et dans le dossier "*database*" on récupère le fichier "*recipesDatabase*" et on lui ajoutera l'extension ".sqlite". Il ne reste plus qu'à ouvrir ce fichier avec le logiciel **DB Browser for SQLite**.

Nom	Type	Schéma
Tables (6)		
android_metadata		CREATE TABLE android_metadata (locale TEXT) ↳ locale TEXT "locale" TEXT
directions		CREATE TABLE directions (id INTEGER PRIMARY KEY AUTOINCREMENT, body TEXT NOT NULL, recipeld TEXT NOT NULL, FOREIGN KEY(recipeld) REFERENCES recipes(id)) ↳ id INTEGER "id" INTEGER ↳ body TEXT "body" TEXT NOT NULL ↳ recipeld TEXT "recipeld" TEXT NOT NULL
ingredients		CREATE TABLE ingredients (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT NOT NULL, recipeld TEXT NOT NULL, FOREIGN KEY(recipeld) REFERENCES recipes(id)) ↳ id INTEGER "id" INTEGER ↳ name TEXT "name" TEXT NOT NULL ↳ recipeld TEXT "recipeld" TEXT NOT NULL
recipes		CREATE TABLE recipes (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT NOT NULL, category TEXT NOT NULL, description TEXT NOT NULL, imagePath TEXT NOT NULL) ↳ id INTEGER "id" INTEGER ↳ name TEXT "name" TEXT NOT NULL ↳ category TEXT "category" TEXT NOT NULL ↳ description TEXT "description" TEXT NOT NULL ↳ imagePath TEXT "imagePath" TEXT NOT NULL
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq) ↳ name TEXT "name" TEXT ↳ seq INTEGER "seq" INTEGER
users		CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT, username TEXT NOT NULL, fullname TEXT NOT NULL, email TEXT NOT NULL, password TEXT NOT NULL) ↳ id INTEGER "id" INTEGER ↳ username TEXT "username" TEXT NOT NULL ↳ fullname TEXT "fullname" TEXT NOT NULL ↳ email TEXT "email" TEXT NOT NULL ↳ password TEXT "password" TEXT NOT NULL
Index (0)		
Vues (0)		
Déclencheurs (0)		

Dans l'onglet "*Parcourir les données*" du logiciel, on peut naviguer dans la liste déroulante "table" pour consulter les différentes tables de la base de données.

Table : ingredients		
id	name	recipeld
1	1 Huile d'olive	1
2	2 Curcuma	1
3	3 Sel	1
4	4 Poivre	1
5	5 Thym	1
6	6 4 Tomates	1
7	7 6 Saucisses fumées	1
8	8 6 gousse d'all	1
9	9 4 Oignons	1
10	10 Ingrédients : Pour 4 personnes.	2
11	11 1 brinelle (aubergine).	2
12	12 800 gr de boucané.	2
13	13 4 oignons.	2
14	14 4 tomates.	2
15	15 4 gousses d'all.	2
16	16 2 piments.	2
17	17 1 cuillère à café de curcuma.	2
18	18 1 petite branche de thym.	2
19	19 3 cuillères à soupe d'huile.	2
20	20 Sel.	2
21	21 1 kg de patates douces	3

Ici, nous avons la table "*ingredients*", qui contient tous les ingrédients de toutes les recettes disponibles dans notre application Dodo'Cuisto. On remarque que tous les ingrédients d'une même recette possèdent le même numéro d'identifiant "*recipeId*". Nous obtenons la même chose pour la table "*directions*", qui contient les instructions de recette.

Table : recipes

	id	name	category	description	imagePath
1	1	Rougaille Saucisse	Plats Cr��oles	Niveau de difficult�� : Facile.	/storage/emulated/0/...
2	2	Boucane bringelle	Plats Cr��oles	Niveau de difficult�� : Facile.	/storage/emulated/0/...
3	3	G��teau patate (g��teau de patates ...	Dessert	Niveau de difficult�� : Moyen	/storage/emulated/0/...
4	4	Bonbons Miel	Dessert	Niveau de difficult�� : Moyen	/storage/emulated/0/bonbonmiel.jpeg

Table : users

	id	username	fullname	email	password
1	1	dodocusto	dodo custo	dodocusto@gmail.com	password
2	2	rd	rde	romaindoyen39@gmail.com	1234
3	3	azerty	fg	romain.doyen97@gmail.com	13c6f3

FIGURE 2 – (Table recipes and Table users)

Table : directions

	id	body	recipeId
1	1	Piquez les saucisses et mettez les da...	1
2	2	Videz la marmite. R��servez les ...	1
3	3	Coupez les saucisses en troncons de ...	1
4	4	Pr��paration de la recette : Coupez le ...	2
5	5	Dans une marmite faites bouillir le ...	2
6	6	Pelez l ail, et p��lez avec une pinc��e d...	2
7	7	��mincez les oignons.	2
8	8	Mettez trois cuill��res  soupe d huile ...	2
9	9	Ajoutez le mlange pil�� (piments, ail...	2
10	10	Ajoutez les morceaux de bringelle (...	2
11	11	Servez chaud....	2
12	12	��pluchez les patates douces, coupez...	3
13	13	Pr��chauffez votre four  180°C, chale...	3
14	14	Ajoutez en plusieurs fois, les oeufs ...	3
15	15	Versez la farine puis m��langez jusqu'...	3
16	16	Incorporez finalement les graines de ...	3
17	17	Versez la p��te dans un moule  ...	3
18	18	Avec une fourchette, faites un dessin...	3
19	19	Enfournez pour 40  45 min.(le temp..	3
20	20	Laissez refroidir le g��teau avant de le...	3
21	21	Dans un saladier, m��langer au fouet ...	4
22	22	La p��te doit tre lisse, pas trop ...	4
23	23	Pendant ce temps, dans une casserole...	4
24	24	Versez la farine puis m��langez jusqu'...	4
25	25	Faire chauffer  feu doux et laisser ...	4
26	26	Remplir une seringue  pâtisserie et ...	4

Nous avons galement les tables "recipes" et "users" qui contiennent respectivement les recettes et les donn  es de l'utilisateur.

3 Les problèmes rencontrés

Nous n'avons pas pu trouver d'image libre de droit pour les recettes créoles dans des banques d'images libres de droits. Nous étions obligés de prendre les images sur Google image.

3.1 Quelques souci d'affichage

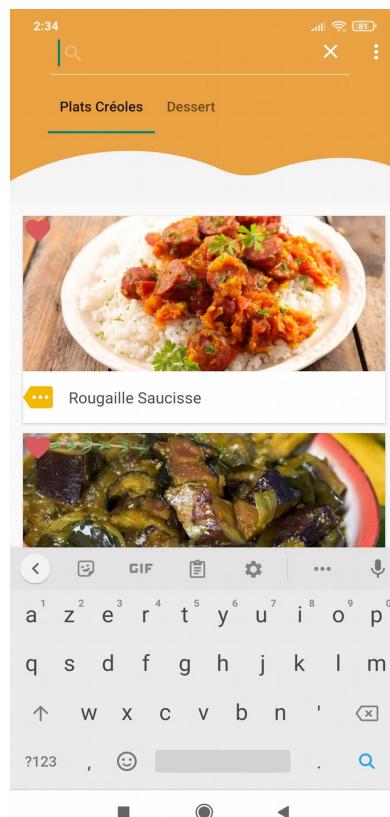
Nous avons eu quelques difficultés à faire afficher certaines fonctionnalités de l'application.

Quand on clique sur une recette, les fragments "*ingrédients*" et "*instructions (direction)*" n'apparaissent pas à l'écran. En effet, cela est une indication visuelle qui permet à l'utilisateur de mieux se repérer lorsqu'il navigue entre les ingrédients et les instructions de recettes, mais il n'empêche en aucun cas le bon usage de l'application.

3.2 Problème avec certaines fonctionnalités

Nous avons également eu quelques problème à faire fonctionner correctement certaines fonctionnalités de **Dodo'Custo**. La barre de recherche située dans la page d'accueil de l'application ne fonctionne pas correctement. En effet, la barre de recherche a beaucoup de difficulté à rechercher les recettes disponible dans la liste des recettes.

L'option "*favoris*" qui permet notamment de mettre en favoris les recettes préférés de l'utilisateur afin de mieux le répertorier dans la liste des recettes, rencontre également un souci de fonctionnement.



Lorsqu'on souhaite cliquer sur le "*coeur rouge*" situant en haut à gauche des recettes, symbolisant l'option "favoris", le "*coeur rouge*" reste figer et ne fonctionne pas.

4 L'un des principaux challenges rencontrés

L'un des principaux challenge est l'affichage des recettes. Pour cela nous avons eu besoin d'implémenter une **TabLayout** pour les ingrédients et à la fois pour les instructions (direction) **ViewRecetteActivity**

```
private ImageView mRecipeImage;
private TextView mRecipeDescription;
private ViewPager mViewPager;
private TabLayout mTabLayout;
private CollapsingToolbarLayout mCollapsingToolbarLayout;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_view_recipe);

    Window w = getWindow();
    w.setFlags(WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS, WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS);

    currentRecipe = getIntent().getParcelableExtra("name: \"recipe\"");
    databaseAdapter = DatabaseController.getInstance(this);
    findViewsById();

    setSupportActionBar(mToolbar);
    getSupportActionBar().setTitle(currentRecipe.getName());
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);
    getSupportActionBar().setDisplayShowHomeEnabled(true);

    mRecipeImage.setImageURI(Uri.fromFile(new File(currentRecipe.getImagePath())));
    mRecipeDescription.setText(currentRecipe.getDescription());
```

Ensuite il a fallut adapter l'affichage en fonction du **TabLayout** dans lequel on se trouve par l'intermédiaire **CollapsingToolbarLayout** dans le **MainActivity**, au début nous avons utilisé des conditions puis on s'est vite rendu compte que le **switch case** était beaucoup approprié.

```

mViewPager.setAdapter(mAdapter);
mTabLayout.setupWithViewPager(mViewPager);
mViewPager.addOnPageChangeListener(new TabLayout.TabLayoutOnPageChangeListener(mTabLayout));
mTabLayout.setTabsFromPagerAdapter(mAdapter);

mViewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener() {
    @Override
    public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) { }

    @Override
    public void onPageSelected(int position) {
        @DrawableRes int image = -1;
        switch (position) {
            case 0:
                image = R.drawable.ic_header;
                break;
            case 1:
                image = R.drawable.ic_header;
                break;
        }

        if (first.getVisibility() == View.VISIBLE) {
            second.setImageResource(image);
            mViewSwitcher.showNext();
        } else {
            first.setImageResource(image);
            mViewSwitcher.showPrevious();
        }
    }
}

@Override
public void onPageScrollStateChanged(int state) { }
});

```

5 L'inconvénient d'un code en MVC

Lors du développement de l'application nous avons utilisé la technologie **MVC** *comme dit plus haut* afin de séparer les tâches, la logique de l'interface mais cela implique une augmentation de la complexité de l'architecture. En effet nous avons rencontré été confronté à un problème de coordination niveau de l'interfacage des couches.

Aussi par l'énorme quantité de fichier à manipuler, en effet la séparation des différentes couches nécessite la création de plus de fichiers (3 fois pour les différentes couches).

Mais cela à permis la reutilisabilite de certaine couche

6 Quelques points délicats/intéressants

6.1 Le stockage des données avec SQLite

Nous avons utilisé la base de données **SQLite** pour notre application. **SQLite** est intégrée dans chaque appareil **Android**. Son utilisation sous **Android** ne nécessite pas de configuration ou d'administration de la base de données.

Pour créer et mettre à jour une base de données dans notre application **Android**, nous avons créé une classe de nom **SQLiteDatabaseHelper** qui hérite de **SQLiteOpenHelper**. Dans le constructeur de notre classe, nous avons appelé la méthode **super()** de **SQLiteOpenHelper**, en précisant le nom de la base de données et sa version actuelle.

Dans la classe **SQLiteDatabaseHelper**, nous avons redéfini les méthodes suivantes pour créer et mettre à jour notre base de données :

```
package com.example.dodocusto.database;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class SQLiteDatabaseHelper extends SQLiteOpenHelper {
    private static final String TAG = SQLiteDatabaseHelper.class.getSimpleName();

    public SQLiteDatabaseHelper(Context context, String databaseName, int databaseVersion) {
        super(context, databaseName, null, databaseVersion);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(UserDatabase.Config.CREATE_TABLE_STATEMENT);
        db.execSQL(RecetteDatabase.Config.CREATE_TABLE_STATEMENT);
        db.execSQL(DirectionDatabase.Config.CREATE_TABLE_STATEMENT);
        db.execSQL(IngredientDatabase.Config.CREATE_TABLE_STATEMENT);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        Log.w(TAG, "Upgrading database from version " + oldVersion + " to " + newVersion + ", which will destroy all old data.");
        db.execSQL("DROP TABLE IF EXISTS " + UserDatabase.Config.TABLE_NAME);
        db.execSQL("DROP TABLE IF EXISTS " + DirectionDatabase.Config.TABLE_NAME);
        db.execSQL("DROP TABLE IF EXISTS " + IngredientDatabase.Config.TABLE_NAME);
        db.execSQL("DROP TABLE IF EXISTS " + RecetteDatabase.Config.TABLE_NAME);
        onCreate(db);
    }
}
```

— **onCreate()** : Appelé lorsque la base de données est créée pour la première fois.

— **onUpgrade()** : Appelé lorsque la base de données doit être mise à jour.

La méthode **onCreate()** sera appelée par le framework pour accéder à une base de données qui n'est pas encore créée. Tandis que la méthode **onUpgrade()** sera appelée si la version de la base de données est augmentée dans le code de notre application. Cette méthode nous permet de mettre à jour un schéma de base de données existant ou de supprimer la base de données existante et la recréer par la méthode **onCreate()**.

Ces deux méthodes reçoivent en paramètre un objet **SQLiteDatabase** qui est la représentation Java de la base de données. **SQLiteDatabase** est la classe de base pour travailler avec une base de données **SQLite** sous **Android** et fournit des méthodes pour ouvrir, effectuer des requêtes, mettre à jour et fermer la base de données.

Plus précisément, **SQLiteDatabase** fournit les méthodes **insert()**, **update()** et **delete()**. En outre, elle fournit la méthode **execSQL()**, qui permet d'exécuter une instruction **SQL** directement.

L'utilisation de ces méthodes sont directement implémenté dans la class **DatabaseController**.

```

private DatabaseController(Context context) {
    mContext = context;
    dbHelper = new SQLiteDatabaseHelper(context, DATABASE_NAME, DATABASE_VERSION);
}

private DatabaseController open() {
    db = dbHelper.getWritableDatabase();
    userDatabase = new UserDatabase(db);
    recetteDatabase = new RecetteDatabase(db);
    return this;
}

public boolean signIn(String email, String password) {
    User currentUser = userDatabase.getUserByEmailAndPassword(email, password);
    UserPreferences.saveCurrentUser(mContext, currentUser);
    return currentUser != null;
}

public void addNewUser(User user) { userDatabase.insert(user); }

public long addNewRecipe(Recette recipe) { return recetteDatabase.insert(recipe); }

public void updateRecipe(Recette recipe) { recetteDatabase.update(recipe); }

public void deleteRecipe(long recipeId) { recetteDatabase.deleteById(recipeId); }

public List<Recette> getAllRecipesByCategory(String category) {
    return recetteDatabase.selectAllByCategory(category);
}

```

7 Conclusion

Pour conclure, avec son choix de sujets libre, ce projet nous a permis de faire preuve de créativité et d'imagination. En effet, grâce à **Android studio** et **Xcode** nous avons appris et découvrir les différentes étapes d'un processus de développement d'une application et surtout de pouvoir les réaliser soi-même. Nous sommes assez satisfait de notre travail, notre application est plutôt bien réussi.

7.1 Les points à améliorer

Nous avons réussi à développer notre application avec les fonctionnalités de base que nous attendions. Cependant, nous pouvons citer quelques points à améliorer comme par exemple :

- L'ergonomie de l'application : rendre **Dodo'Cuisto** encore plus efficace à utiliser
- Ajout d'un espace commentaire sur les recettes : donner son avis sur les recettes
- Possibilité de personnaliser son profil(mettre une photo de profil, ajouter un numéro de téléphone etc...)
- Une barre de recherche et option de favoris plus efficace et ergonomique

8 Bibliographie

[2] [4] [2] [3] [1] [5]

Nous nous sommes inspirés de plusieurs exemples de code sur des sites de tutoriel ou sur Youtube, de solutions trouvées sur StackOverflow, etc. . . qui dans la plupart des cas ont dû être adaptées afin de convenir à notre contexte.

Citées ci-dessous, quelques aides qui nous ont beaucoup aidé :

Références

- [1] Apprendre à créer des applications pour android. <https://openclassrooms.com/fr/courses/2023346-creez-des-applications-pour-android>, <https://www.youtube.com/channel/UCUIF5MImktJLDWDKe5oTdJQ/videos>.
- [2] La documentation officiel d'android. <https://developer.android.com/docs>.
- [3] Pour les images. <https://www.youtube.com/watch?v=eTDGOjCvi7Q>, <https://stackoverflow.com/questions/17546101/get-real-path-for-uri-android>,<https://developer.android.com/training/data-storage/use-cases>.
- [4] Manipulation d'une base de données sqlite. <https://koor.fr/Java/Android/SQLiteSample.wp>, <https://www.youtube.com/watch?v=vRaR3yLnHig&list=PLRR7wjtXb1cB-jibndUwqv7902KQkG6U&index=9>.
- [5] Tutoriel youtube android. https://www.youtube.com/watch?v=rUnuYTjaBoU&list=PLRR7wjtXb1cA_nZyku75L1ropfJgxrMMy.