

Projet Resto.fr – Itération 1

Affichage de la liste des restaurants

User story : En tant qu'utilisateur, je souhaite pouvoir consulter la liste des restaurants (nom et ville) afin de pouvoir en sélectionner un ensuite.

(Les tickets ne sont pas forcément dans un ordre logique)

TICKET 01 : REMPLISSAGE DU README (MODE OPERATOIRE DE TEST, ...)	2
TICKET 03 : SCHEMATISATION DES INTERFACES	3
TICKET 04 : CREATION DE L'INTERFACE --> LISTE DES RESTAURANTS	4
TICKET 05 : MISE EN PLACE DE LA BDD ET DU DAO	6
TICKET 06 : MODELE METIER	7
TICKET 07 : CONSTRUCTEUR --> LISTE DES RESTAURANT	9
RESULTAT :	12

TICKET 01 : REMPLISSAGE DU README (MODE OPERATOIRE DE TEST, ...)

Le ticket a pour objectif de rédiger un fichier README professionnel pour l'application Android de réservation de restaurants. Ce fichier décrit en détail le projet, en présentant ses fonctionnalités principales (consultation de restaurants, affichage des détails et réservation), le mode opératoire des tests et la documentation technique (architecture MVC, API REST, base de données).

Voici une capture d'écran du README :

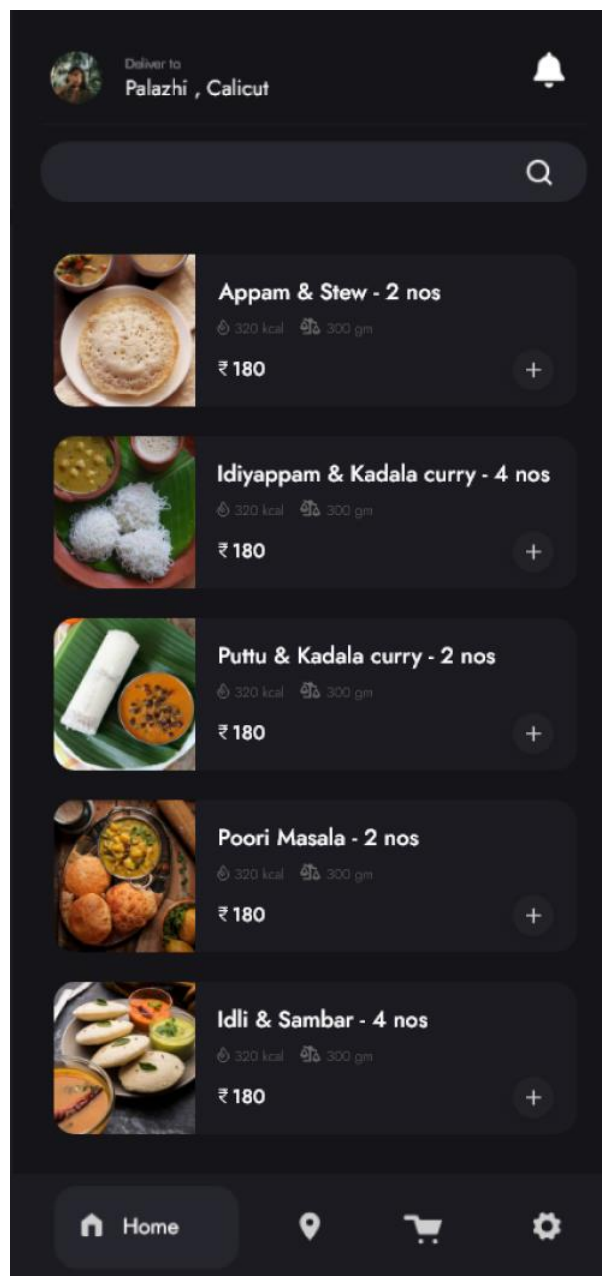
```

.Compte-Rendu_RestoAndroid.Link  README.md M X
# Application Android de Réservation de Restaurants > ## Mode opératoire de test > ### Installation de l'application > #### Étapes d'installation :
1  # **Application Android de Réservation de Restaurants**
2
3  ---
4
5  ## **Description**
6  Cette application Android permet aux utilisateurs de :
7  - Consulter une liste de restaurants (nom et ville).
8  - Voir les détails d'un restaurant (photo et informations).
9  - Réserver une table en précisant un nom, un téléphone, une date, une heure et un nombre de personnes.
10
11  L'application s'appuie sur une API REST et une base de données relationnelle (MySQL/MariaDB). Elle est conçue en respectant le modèle architectural **MVC**.
12  ---
13
14  ## **Mode opératoire de test**
15
16  ### **Installation de l'application**
17  #### **Pré-requis :**
18  - Un terminal Android version **10** ou supérieure.
19  - Une connexion internet pour accéder à l'API REST.
20  - Une base de données MySQL/MariaDB préconfigurée et accessible.
21
22  #### **Étapes d'installation :**
23  1. **Téléchargement :**
24  - Téléchargez le fichier `.apk` depuis le dépôt ou une URL fournie.
25  2. **Installation :**
26  - Transférez le fichier sur votre terminal Android.
27  - Activez les **sources inconnues** dans les paramètres.
28  - Cliquez sur le fichier `.apk` pour l'installer.
29  3. **Configuration :**
30  - Lors du premier lancement, configurez l'URL de l'API et les identifiants SGDB.
31  ---
32
33  ## **Scénarios de test**
34
35  ### **1. Affichage de la liste des restaurants**
36  - **Objectif :** Vérifier que les restaurants sont affichés (nom et ville).
37  - **Étapes :**
38  1. Lancez l'application.
39  2. Vérifiez que la liste des restaurants s'affiche correctement.
40  - **Critères de réussite :** Les noms et villes des restaurants sont visibles sans erreurs.
41
42  ### **2. Affichage des détails d'un restaurant**
43  - **Objectif :** Vérifier l'affichage des informations détaillées d'un restaurant.
44  - **Étapes :
```

TICKET 03 : SCHEMATISATION DES INTERFACES

Dans ce ticket, je réalise une maquette de l'interface dédiée à la liste des restaurants, conformément aux exigences de l'itération 1. Cette maquette illustre la disposition des éléments clés tels que les menus, le profil, en respectant une certaine ergonomie et clarté visuelle. Elle sert de guide pour le développement de l'application.

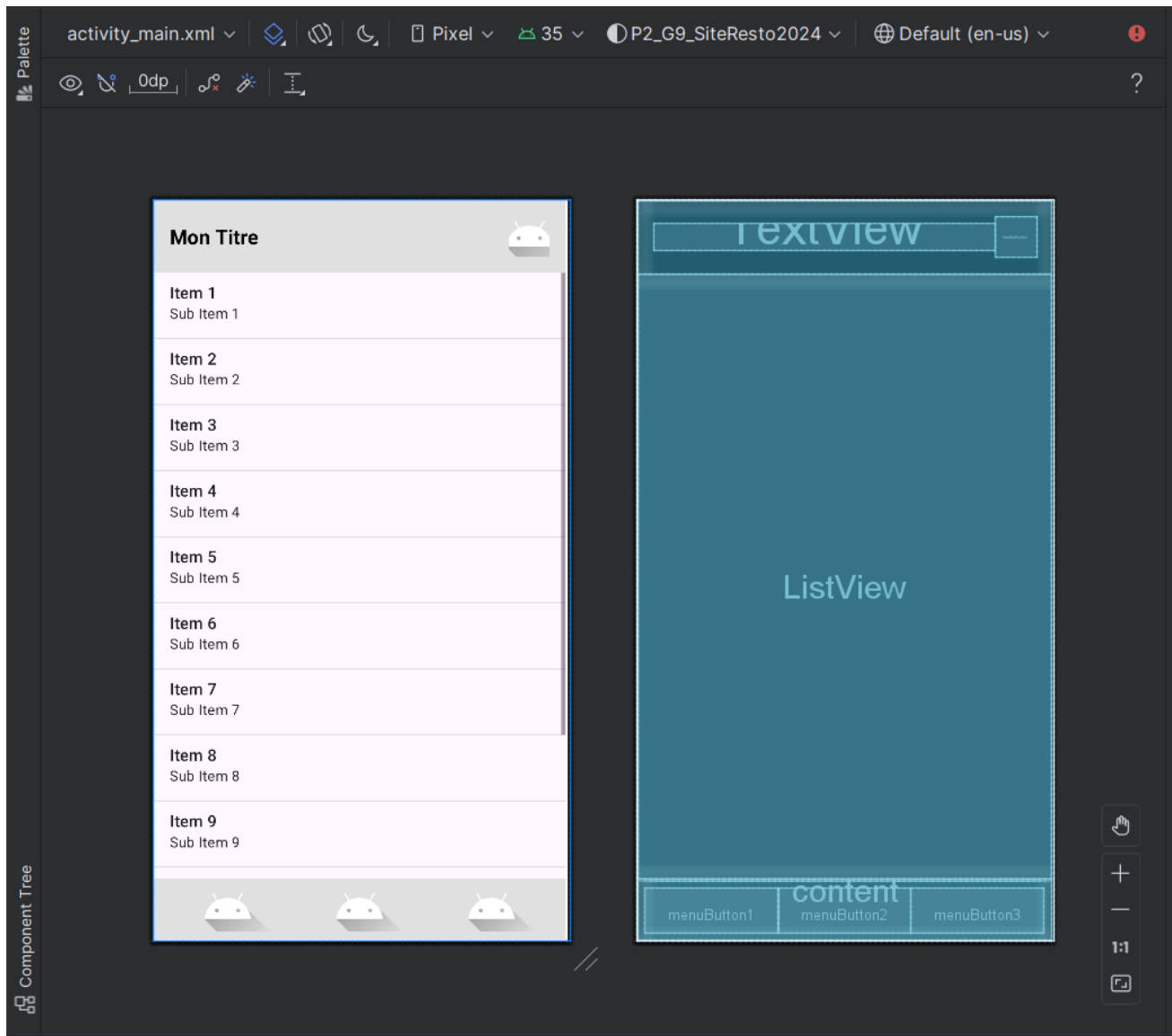
Voici la maquette de l'interface dédiée à la liste des restaurants :



TICKET 04 : CREATION DE L'INTERFACE --> LISTE DES RESTAURANTS

Dans ce ticket, je réalise une maquette de l'interface dédiée à la liste des restaurants, conformément aux exigences de l'itération 1 et de la maquette (ticket 03). Pour cela j'ai dû modifier le fichier XML : "activity_main.xml". J'ai divisé l'écran en 3 : un header / une section avec la listView / un footer avec les boutons de navigation

Voici l'interface dédiée à la liste des restaurants :



Et voici une partie du code de “activity_main.xml” :

```
</> activity_main.xml ×
2      <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas
10      <LinearLayout
56      <RelativeLayout
57          xmlns:android="http://schemas.android.com/apk/res/android"
58          android:layout_width="match_parent"
59          android:layout_height="match_parent">
60      <FrameLayout
61          android:id="@+id/content"
62          android:layout_width="match_parent"
63          android:layout_height="match_parent"
64          android:layout_above="@+id/bottomMenu" />
65      <LinearLayout
66          android:id="@+id/bottomMenu"
67          android:layout_width="match_parent"
68          android:layout_height="60dp"
69          android:layout_alignParentBottom="true"
70          android:orientation="horizontal"
71          android:gravity="center"
72          android:background="#E0E0E0"
73          android:padding="8dp">
74      <ImageButton
75          android:id="@+id/menuButton1"
76          android:layout_width="0dp"
77          android:layout_height="match_parent"
78          android:layout_weight="1"
79          android:src="@drawable/ic_launcher_foreground"
80          android:background="@null"
81          android:contentDescription="Menu 1" />
82      <ImageButton
83          android:id="@+id/menuButton2"
84          android:layout_width="0dp"
85          android:layout_height="match_parent"
86          android:layout_weight="1"
87          android:src="@drawable/ic_launcher_foreground"
88          android:background="@null"
89          android:contentDescription="Menu 2" />
90      <ImageButton
91          android:id="@+id/menuButton3"
92          android:layout_width="0dp"
93          android:layout_height="match_parent"
94          android:layout_weight="1"
95          android:src="@drawable/ic_launcher_foreground"
96          android:background="@null"
97          android:contentDescription="Menu 3" />
98      </LinearLayout>
99      </FrameLayout>
100  </LinearLayout>
101  </RelativeLayout>
```

TICKET 05 : MISE EN PLACE DE LA BDD ET DU DAO

Dans ce Ticket le but Ajout des fichier .sql dans le projet Créé le script SQL pour faire la connexion à la bdd Externe Créé le modèle DAO en PHP qu'on utilisera sous Android.

Ajout du SQL dans le projet

Table	Action	Lignes	Type	Interclassement	Taille	Perte
<input type="checkbox"/> aimer	Parcourir Structure Rechercher Insérer Vider Supprimer	16	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
<input type="checkbox"/> critiquer	Parcourir Structure Rechercher Insérer Vider Supprimer	19	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
<input type="checkbox"/> photo	Parcourir Structure Rechercher Insérer Vider Supprimer	14	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
<input type="checkbox"/> resto	Parcourir Structure Rechercher Insérer Vider Supprimer	11	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> utilisateur	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
5 tables	Somme	67	InnoDB	utf8mb4_unicode_ci	144,0 kio	0 o

Création du script SQL pour la connexion a la BDD

```

C: > Users > pierr > OneDrive > Bureau > Resto > connexion.php
1  <?php
2  $host = 'localhost';
3  $dbname = 'resto2';
4  $username = 'resto_util';
5  $password = 'secret';
6
7  try {
8      $pdo = new PDO(dsn: "mysql:host=$host;dbname=$dbname", username: $username, password: $password);
9      $pdo->setAttribute(attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
10 } catch (PDOException $e) {
11     echo "Erreur de connexion : " . $e->getMessage();
12     die();
13 }
14 ?>

```

et la récupération des informations dans la bdd pour Android studio

```

1  <?php
2  header('Content-Type: application/json');
3
4  require 'connexion.php';
5
6  try {
7      $stmt = $pdo->query("SELECT idR, nomR, numAdrR, voieAdrR, cpR, villeR, latitudeDegR, longitudeDegR, descR, horairesR FROM resto");
8      $lacs = $stmt->fetchAll(PDO::FETCH_ASSOC);
9      echo json_encode($lacs);
10 } catch (PDOException $e) {
11     echo json_encode(['error' => 'Erreur de récupération des données : ' . $e->getMessage()]);
12 }
13 ?>

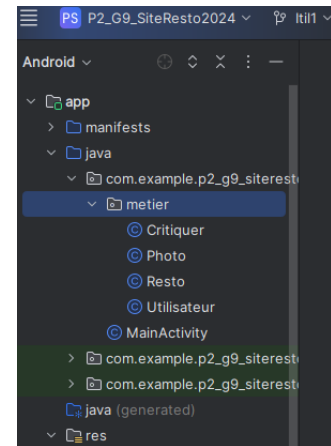
```

TICKET 06 : MODELE METIER

Dans ce ticket, le but était de créer tout le modèle métier de notre app avec ses classes complétées.

Pour cela il fallait savoir dans un premier temps lequel allait être utile, nous avons donc besoin d'uniquement celle avec des informations contenues dedans, soit la classe resto, utilisateur, critique et photo. La classe aimer est une simple table composite donc inutile, elle ne contient aucune information.

Tout d'abord il faut que l'on crée le package metier :



Ensuite j'ai commencé à créer la classe Photo :

```

1 package com.example.p2_g9_siteresto2024.metier;
2
3 import java.util.List;
4
5 public class Utilisateur {
6     private int idU;
7     private String mailU;
8     private String mdpU;
9     private String pseudoU;
10    private List<Resto> lesRestosAimes;
11
12    public Utilisateur(int idU, String pseudoU, String mdpU, String mailU, List<Resto> lesRestosAimes) {
13        this.idU = idU;
14        this.pseudoU = pseudoU;
15        this.mdpU = mdpU;
16        this.mailU = mailU;
17        this.lesRestosAimes = lesRestosAimes;
18    }
19
20    public int getIdU() { return idU; }
21    public void setIdU(int idU) { this.idU = idU; }
22    public String getPseudoU() { return pseudoU; }
23    public void setPseudoU(String pseudoU) { this.pseudoU = pseudoU; }
24    public String getMdpU() { return mdpU; }
25    public void setMdpU(String mdpU) { this.mdpU = mdpU; }
26    public String getMailU() { return mailU; }
27    public void setMailU(String mailU) { this.mailU = mailU; }
28    public List<Resto> getLesRestosAimes() { return lesRestosAimes; }
29

```

J'ai ensuite poursuivi sur la classe Utilisateur qui n'a pas aussi besoin d'utiliser les autres classes java :

```

1 package com.example.p2_g9_siteresto2024.metier;
2
3 public class Photo {
4     private int idP;
5     private String cheminP;
6
7     public Photo(String cheminP, int idP) {
8         this.cheminP = cheminP;
9         this.idP = idP;
10    }
11
12    public int getIdP() { return idP; }
13    public void setIdP(int idP) { this.idP = idP; }
14    public String getCheminP() { return cheminP; }
15    public void setCheminP(String cheminP) { this.cheminP = cheminP; }
16
17 }

```

Après j'ai ajouté la classe Critiquer qui elle utilise un utilisateur dedans pour définir l'auteur du commentaire :

```

1 package com.example.p2_g9_siteresto2024.metier;
2
3 public class Critiquer {
4     private int note; //note sur 5 3 usages
5     private String commentaire; 3 usages
6     private Utilisateur unUtilisateur; 3 usages
7
8     public Critiquer(int note, Utilisateur unUtilisateur, String commentaire) {
9         this.note = note;
10        this.unUtilisateur = unUtilisateur; //Auteur du commentaire
11        this.commentaire = commentaire;
12    }
13
14    public int getNote() {
15        return note;
16    }
17
18    public void setNote(int note) { this.note = note; }
19
20    public String getCommentaire() { return commentaire; }
21
22    public void setCommentaire(String commentaire) { this.commentaire = commentaire; }
23
24    public Utilisateur getUnUtilisateur() { return unUtilisateur; }
25
26    public void setUnUtilisateur(Utilisateur unUtilisateur) { this.unUtilisateur = unUtilisateur; }
27 }

```

Enfin j'ai ajouté la table principal resto à la fin, car celle-ci utilise deux listes (class) une liste de photo du restaurant et la liste des critiques du restaurant :

```

1 package com.example.p2_g9_siteresto2024.metier;
2
3 import java.util.List;
4
5 public class Resto {
6     private int idR; 3 usages
7     private String nomR; 3 usages
8     private String numAdr; 3 usages
9     private String voieAdr; 3 usages
10    private String cpR; 3 usages
11    private String villeR; 3 usages
12    private float latitudeDegR; //Latitude en degrés 3 usages
13    private float longitudeDegR; //Longitude en degrés 3 usages
14    private String descR; 3 usages
15    private String horairesR; //Horaire d'ouverture 3 usages
16    private List<Photo> lesPhotos; 3 usages
17    private List<Critiquer> lesCritiques; 3 usages
18
19    public Resto(int idR, String nomR, String numAdr, String voieAdr, String cpR, String villeR, float latitudeDegR, float longitudeDegR, String descR, String horairesR, List<Photo> lesPhotos, List<Critiquer> lesCritiques) {
20        this.idR = idR;
21        this.nomR = nomR;
22        this.numAdr = numAdr;
23        this.voieAdr = voieAdr;
24        this.cpR = cpR;
25        this.villeR = villeR;
26        this.latitudeDegR = latitudeDegR;
27        this.longitudeDegR = longitudeDegR;
28        this.descR = descR;
29        this.horairesR = horairesR;
30        lesPhotos = lesPhotos;
31        lesCritiques = lesCritiques;
32    }
33
34    public int getIdR() { return idR; }
35
36    public void setIdR(int idR) { this.idR = idR; }
37
38    public String getNomR() { return nomR; }
39
40    public void setNomR(String nomR) { this.nomR = nomR; }
41
42    public String getNumAdr() { return numAdr; }
43
44    public void setNumAdr(String numAdr) { this.numAdr = numAdr; }
45
46    public String getVoieAdr() { return voieAdr; }
47
48    public void setVoieAdr(String voieAdr) { this.voieAdr = voieAdr; }
49
50    public String getCpR() { return cpR; }
51
52    public void setCpR(String cpR) { this.cpR = cpR; }
53
54    public String getVilleR() { return villeR; }
55
56    public void setVilleR(String villeR) { this.villeR = villeR; }
57
58    public float getLatitudeDegR() { return latitudeDegR; }
59
60    public void setLatitudeDegR(float latitudeDegR) { this.latitudeDegR = latitudeDegR; }
61
62    public float getLongitudeDegR() { return longitudeDegR; }
63
64    public void setLongitudeDegR(float longitudeDegR) { this.longitudeDegR = longitudeDegR; }
65
66    public String getDescR() { return descR; }
67
68    public void setDescR(String descR) { this.descR = descR; }
69
70    public String getHorairesR() { return horairesR; }
71
72    public void setHorairesR(String horairesR) { this.horairesR = horairesR; }
73
74    public List<Photo> getLesPhotos() { return lesPhotos; }
75
76    public void setLesPhotos(List<Photo> lesPhotos) { this.lesPhotos = lesPhotos; }
77
78    public List<Critiquer> getLesCritiques() { return lesCritiques; }
79
80    public void setLesCritiques(List<Critiquer> lesCritiques) { this.lesCritiques = lesCritiques; }
81
82    public void setIdR(int idR) { this.idR = idR; }
83
84    public void setNomR(String nomR) { this.nomR = nomR; }
85
86    public void setNumAdr(String numAdr) { this.numAdr = numAdr; }
87
88    public void setVoieAdr(String voieAdr) { this.voieAdr = voieAdr; }
89
90    public void setCpR(String cpR) { this.cpR = cpR; }
91
92    public void setVilleR(String villeR) { this.villeR = villeR; }
93
94    public void setLatitudeDegR(float latitudeDegR) { this.latitudeDegR = latitudeDegR; }
95
96    public void setLongitudeDegR(float longitudeDegR) { this.longitudeDegR = longitudeDegR; }
97
98    public void setDescR(String descR) { this.descR = descR; }
99
100   public void setHorairesR(String horairesR) { this.horairesR = horairesR; }
101
102   public void setLesPhotos(List<Photo> lesPhotos) { this.lesPhotos = lesPhotos; }
103
104   public void setLesCritiques(List<Critiquer> lesCritiques) { this.lesCritiques = lesCritiques; }
105 }

```

Bien évidemment sur chaque class, tous les construct ont été défini, ainsi que tous les getter et setter aucune autre méthode n'a été défini à part celle-ci.

TICKET 07 : CONSTRUCTEUR --> LISTE DES RESTAURANT

Dans ce ticket, le but était de modifier le constructeur de notre interface qui affiche la liste des restaurants.

Pour cela, il a fallu autoriser la connexion à internet et créer le client http pour que l'on puisse accéder à notre bdd externe. Qui récupère les informations des restaurants à l'aide de la fonction DAO et les mettre dans des objet métier resto.

Tout d'abord la première chose que j'ai donc faite, c'est toute la partie client http et autoriser la connexion. Pour le client http, il fallait implémenter okhttp dans la library avec la version :

app/build.gradle.kts		gradle/libs.versions.toml	
...	... @@ -37,6 +37,7 @@ dependencies { @@ -16,6 +16,7 @@ appcompat = { group = "androidx.appcompat", name = "appcompat", version.ref = "a
37	37 implementation(libs.material)	16	16 material = { group = "com.google.android.material", name = "material", version.ref = "material" }
38	38 implementation(libs.activity)	17	17 activity = { group = "androidx.activity", name = "activity", version.ref = "activity" }
39	39 implementation(libs.constraintlayout)	18	18 constraintlayout = { group = "androidx.constraintlayout", name = "constraintlayout", version.ref = "constraintlayout" }
40	40 + implementation(libs.okhttp)	19	19 + okhttp = "com.squareup.okhttp3:okhttp:4.10.0"
41	41 testImplementation(libs.junit)	20	20
42	42 androidTestImplementation(libs.ext.junit)	21	21 [plugins]
43	43 androidTestImplementation(libs.espresso.core)	22	22 android-application = { id = "com.android.application", version.ref = "app" }
...

Ensuite il fallait que j'autorise notre app à accéder à Internet :

app/src/main/AndroidManifest.xml	
1	1 <?xml version="1.0" encoding="utf-8"?>
2	2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3	3 xmlns:tools="http://schemas.android.com/tools">
4	4 + <uses-permission android:name="android.permission.INTERNET" />
5	5 + <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
6	6 + <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
7	7
8	8 <application
9	9 android:allowBackup="true"
10	10
11	11 android:roundIcon="@mipmap/ic_launcher_round"
12	12 android:supportsRtl="true"
13	13 android:theme="@style/Theme.P2_G9_SiteResto2024"
14	14 + android:usesCleartextTraffic="true"
15	15 tools:targetApi="31">
16	16 <activity
17	17 android:name=".MainActivity"
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28

Ensuite, je me suis attaqué à la MainActivity.java qui correspond au constructeur de notre layout, j'ai d'abord commencé à faire les imports qui allait me servir :

app/src/main/java/com/example/p2_g9_siteresto2024/MainActivity.java	
1	1 package com.example.p2_g9_siteresto2024;
2	2
3	3 import android.os.Bundle;
4	4 + import android.util.Log;
5	5 + import android.widget.ArrayAdapter;
6	6 + import android.widget.ListView;
7	7
8	8 import androidx.activity.EdgeToEdge;
9	9 import androidx.appcompat.app.AppCompatActivity;
10	10
11	11 @@ -8,6 +11,21 @@ import androidx.core.graphics.Insets;
12	12 import androidx.core.view.ViewCompat;
13	13 import androidx.core.view.WindowInsetsCompat;
14	14 + import com.example.p2_g9_siteresto2024.metier.Resto;
15	15 +
16	16 + import org.json.JSONArray;
17	17 + import org.json.JSONException;
18	18 + import org.json.JSONObject;
19	19 +
20	20 + import java.io.IOException;
21	21 + import java.util.ArrayList;
22	22 +
23	23 + import okhttp3.Call;
24	24 + import okhttp3.Callback;
25	25 + import okhttp3.OkHttpClient;
26	26 + import okhttp3.Request;
27	27 + import okhttp3.Response;
28	28 +

Et par la suite, j'ai récupéré le listView du layout pour afficher les restaurants :

```
// Référence de la ListView pour afficher la liste des restaurants
ListView listViewResto = (ListView) findViewById(R.id.listViewResto);
ArrayList<Resto> lesResto = new ArrayList<>();
```

Après j'ai envoyé ma requête pour récupérer les restaurants :

```
// Création de la requête HTTP vers le fichier PHP (Voir DAO pour les détails)
Request requestClients = new Request.Builder().url("http://10.15.13.176/Resto/get_restos.php").build();
OkHttpClient httpClient = new OkHttpClient();

// Envoi de la requête HTTP en arrière-plan
httpClient.newCall(requestClients).enqueue(new Callback() {
```

Puis j'ai traité la réponse ou non, en la parcourant dans un tableau pour placé les resto un par un dans objet resto qui sont ajouté dans une ArrayList

```
MainActivity.java
30 public class MainActivity extends AppCompatActivity {
31     protected void onCreate(Bundle savedInstanceState) {
32         httpClient.newCall(requestClients).enqueue(new Callback() {
33             @Override 4 usages
34             public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {
35                 // Traitement de la réponse HTTP reçue
36                 final String myResponse = response.body().string();
37                 MainActivity.this.runOnUiThread() -> {
38                     try {
39                         // Conversion de la réponse en un objet JSON principal
40                         JSONObject jsonObjectLesClients = new JSONObject(myResponse);
41
42                         // Extraction du tableau JSON contenant les restaurants
43                         JSONArray jsonArray = jsonObjectLesClients.optJSONArray("clients");
44
45                         // Nettoyage de la liste existante avant d'ajouter les nouveaux éléments
46                         lesResto.clear();
47
48                         // Parcours du tableau JSON pour transformer chaque objet en instance de Resto
49                         for (int i = 0; i < jsonArray.length(); i++) {
50                             JSONObject jsonObject = jsonArray.getJSONObject(i);
51
52                             // Récupération des champs nécessaires pour créer un objet Resto
53                             int idR = jsonObject.getInt("idR");
54                             String nomR = jsonObject.getString("nomR");
55                             String numAdr = jsonObject.getString("numAdr");
56                             String voieAdr = jsonObject.getString("voieAdr");
57                             String cpR = jsonObject.getString("cpR");
58                             String villeR = jsonObject.getString("villeR");
59                             float latitudeDegR = (float) jsonObject.getDouble("latitudeDegR");
60                             float longitudeDegR = (float) jsonObject.getDouble("longitudeDegR");
61                             String descR = jsonObject.getString("descR");
62                             String horairesR = jsonObject.getString("horairesR");
63
64                             // Création de l'objet Resto et ajout à la liste
65                             Resto unResto = new Resto(idR, nomR, numAdr, voieAdr, cpR, villeR, latitudeDegR, longitudeDegR, descR, horairesR);
66                             lesResto.add(unResto);
67                         }
68                     }
69                 }
70             }
71         });
72     }
73 }
```

jet_SiteRestoAndroid > app > src > main > java > com > example > p2_g9_siteresto2024 > MainActivity > onCreate

À savoir que j'ai temporairement peut-être, créer un construct resto qui n'avais pas besoin de photo et des critiques car pour l'instant ça m'était inutile dans mon affichage.

```
public Resto(int idR, String nomR, String numAdr, String voieAdr, String cpR, String villeR, float latitudeDegR, float longitudeDegR, String descR, String horairesR) {
    this.idR = idR;
    this.nomR = nomR;
    this.numAdr = numAdr;
    this.voieAdr = voieAdr;
    this.cpR = cpR;
    this.villeR = villeR;
    this.latitudeDegR = latitudeDegR;
    this.longitudeDegR = longitudeDegR;
    this.descR = descR;
    this.horairesR = horairesR;
}
```

Pour finir j'ai simplement mis à jour la listview dans le layout et j'ai également préparer l'app en cas d'erreur sur la requête http :

```
91 // Mise à jour de l'adaptateur pour afficher les restaurants dans la ListView
92 ArrayAdapter<Resto> dataAdapter = new ArrayAdapter<>(context, MainActivity.this, android.R.layout.simple_list_item_1, lesResto);
93 listViewResto.setAdapter(dataAdapter);
94
95 } catch (JSONException e) {
96     // Gestion des erreurs liées au traitement du JSON
97     Log.e(tag, "JSONError", msg, "Erreur lors de l'analyse du JSON", e);
98 }
99
100 };
101
102
103 @Override
104 public void onFailure(@NonNull Call call, @NonNull IOException e) {
105     // Gestion des erreurs de connexion ou d'exécution de la requête HTTP
106     e.printStackTrace();
107     Log.e(tag, "HttpRequestError", msg, "Erreur lors de la requête vers get_restos.php", e); // Voir DAO
108 }
109
110 }
```

RESULTAT :

Nous rencontrons actuellement plusieurs problèmes avec Android Studio et GitLab, ce qui nous empêche de tester correctement le programme :

- **De mon côté (Thibault)** : Lorsque je lance le programme, j'obtiens une page blanche sans aucun message d'erreur, ce qui complique le diagnostic.
- **Pour Pierre** : Lorsqu'il clone le projet, celui-ci n'apparaît pas dans ses documents.
- **Pour Thomas** : Le "Run" d'Android Studio ne fonctionne plus depuis un certain temps. Même après avoir réinstallé l'application, le problème persiste.

Nous espérons que le programme fonctionnera si vous parvenez à le tester.

Quelques points importants à noter pour le test :

- Assurez-vous d'avoir bien configuré la base de données dans votre environnement MySQL. Les scripts nécessaires se trouvent dans le dossier **SQL**.
- Copiez également le dossier **dao_resto** dans le répertoire **xampp/htdocs** (ou **wamp/www**, selon votre environnement).