

Cette fonctionnalité fait partie des améliorations entreprises et développées au cours de mon travail personnel. Elle offre la possibilité aux utilisateurs de réserver une table auprès du restaurant de leur choix. La réservation se fait à partir de l'affichage des détails du restaurant, un bouton permet de réserver le restaurant courant en redirigeant vers une page demandant de sélectionner un créneau, une fois le créneau validé, une confirmation s'affiche avec un récapitulatif et une redirection vers la première page.

1 – Les contrôleurs

Pour cette fonctionnalité, nous avons besoins de deux contrôleurs, ils interagissent avec leur layout associé, mais aussi en l'occurrence, avec les autres activités et la base de données au travers d'un fichier PHP.

Commençons par le contrôleur permettant de saisir les informations de réservation : « ReservationActivity ».

1.a - ReservationActivity

Tout d'abord, on importe les classes et on déclare le nécessaire.

```
import ...

7 usages
public class ReservationActivity extends AppCompatActivity {

    4 usages
    private DatePicker datePicker;
    3 usages
    private TimePicker timePicker;
    2 usages
    private Button validateButton;

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

        // Modifier le titre de la barre d'action
        if (getSupportActionBar() != null) {
            getSupportActionBar().setDisplayHomeAsUpEnabled(true);
            getSupportActionBar().setTitle("R3STO.FR");
        }

        datePicker = findViewById(R.id.datePicker);
        timePicker = findViewById(R.id.timePicker);
        validateButton = findViewById(R.id.validateButton);
    }
}
```

Pour réserver le restaurant, nous avons besoin de plusieurs données :

1. le nom du restaurant ;
2. la date (jour, heure et année) ;
3. le pseudo de l'utilisateur.

1. On récupère le nom du resto par un intent. On l'envoie à partir de « DetailsRestoActivity ».

```
@Override
public void onClick(View v) {
    // Vérification de l'état de connexion avant de permettre la réservation
    if (checkLoginStatus()) {
        // L'utilisateur est connecté, démarrage de l'activité de réservation
        Intent intent = new Intent( packageContext: DetailsRestoActivity.this, ReservationActivity.class);
        String nomResto = extras.getString( key: "nomResto",   defaultValue: "Nom inconnu");
        intent.putExtra( name: "nomResto", nomResto);
        startActivity(intent);
    } else {
        // L'utilisateur n'est pas connecté, affichage d'un message toast
        Toast.makeText( context: DetailsRestoActivity.this, text: "Veuillez vous connecter pour réserver.",
    }
}
```

Notez que seul un utilisateur connecté accédera à la page de réservation.

Puis on le récupère dans « ReservationActivity ».

```
// Récupérer les données de l'intent
Bundle extras = getIntent().getExtras();

String nomResto = extras.getString( key: "nomResto",   defaultValue: "Nom inconnu");
```

2. la date est obtenu à l'aide d'un datePicker et d'un timePicker contenu dans la vue « activity_reservation ».

Réserveation

1970

Dim. 4 janv.

<	Janvier 1970							>
L	M	M	J	V	S	D		
			1	2	3	4		
5	6	7	8	9	10	11		
12	13	14	15	16	17	18		
19	20	21	22	23	24	25		
26	27	28	29	30	31			

2

21

AM

3

:

22

PM

4

23

VALIDER

Que l'on récupère lors du clique sur le bouton valider à l'aide d'un Listener, puis que l'on converti en une variable.

```
validateButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        int day = datePicker.getDayOfMonth();
        int month = datePicker.getMonth() + 1; // Month is 0-based, add 1
        int year = datePicker.getYear();

        Calendar calendar = Calendar.getInstance();
        calendar.set(year, month, day);

        SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "yyyy-MM-dd", Locale.getDefault());
        String date = dateFormat.format(calendar.getTime());

        int hour = timePicker.getHour();
        int minute = timePicker.getMinute();
        String heure = String.format(Locale.getDefault(), format: "%02d:%02d:00", hour, minute);

        String nomResto = extras.getString( key: "nomResto", defaultValue: "Nom inconnu");

        String pseudoU = LoginActivity.getUsername( context: ReservationActivity.this);

        sendReservationData(pseudoU, nomResto, String.valueOf(date), heure);
    }
});
```

3. Enfin, on récupère le pseudo à l'aide de la méthode « getUsername » de « LoginActivity », pour rappel il est stocké en mémoire à l'aide d'un « SharedPreferences ».

```
String pseudoU = LoginActivity.getUsername( context: ReservationActivity.this);
```

Ensuite, on envoie les données à une méthode qui les transmettra au fichier « insertReservation.php » afin d'insérer une nouvelle réservation dans la table « reservation » de notre base de données.

```
        sendReservationData(pseudoU, nomResto, String.valueOf(date), heure);
    }
});
}

1 usage
private void sendReservationData(String pseudoU, String resto, String date, String heure) {
    OkHttpClient client = new OkHttpClient();
    RequestBody formBody = new FormBody.Builder()
        .add( name: "pseudoU", pseudoU)
        .add( name: "resto", resto)
        .add( name: "date", date)
        .add( name: "heure", heure)
        .build();

    Request request = new Request.Builder()
        .url("http://10.15.13.167/android/appResto/insertReservation.php")
        .post(formBody)
        .build();
```

Arrêtons-nous un instant sur ce fichier PHP.

```
<?php
try {
    // Connexion à la base de données MySQL
    $db="resto_test";
    $dbhost="localhost";
    $dbport=3306;
    $dbuser="root";
    $dbpasswd="joliverie";

    $connexion = new PDO('mysql:host='.$dbhost.';port='.$dbport.';dbname='.$db.'', $dbuser, $dbpasswd);
    $connexion->exec("SET CHARACTER SET utf8");

    // Récupération des données POST
    $pseudoU = $_POST['pseudoU'];
    $resto = $_POST['resto'];
    $date = $_POST['date'];
    $heure = $_POST['heure'];

    $reponse=$connexion->prepare("INSERT INTO reservation (pseudoU, resto, date, heure) VALUES (?, ?, ?, ?);");
    $reponse->execute([$pseudoU, $resto, $date, $heure]);

    echo json_encode(["success" => true]);
}
catch (Exception $e) {
    echo json_encode(["success" => false, "message" => $e->getMessage()]);
}
?>
```

Il contient une requête préparée (INSERT), une gestion des erreurs, et retourne un booléen (échec ou réussite) afin de répondre de manière spécifique dans notre contrôleur au résultat de la requête.

Pour finir, si la requête est un succès, on lance la page récapitulative de la réservation tout en lui transmettant les données à afficher.

Dans le cas contraire, un message d'erreur sera affiché.

```
client.newCall(request).enqueue(new Callback() {
    @Override
    public void onResponse(@NotNull Call call, @NotNull Response response) throws IOException {
        if (response.isSuccessful()) {
            runOnUiThread() -> {
                Intent intent = new Intent( packageContext, ReservationActivity.this, RecapitulatifActivity.class);
                intent.putExtra( name: "nomResto", resto);
                intent.putExtra( name: "date", date);
                intent.putExtra( name: "heure", heure);
                startActivity(intent);
            };
        } else {
            runOnUiThread() -> Toast.makeText( context: ReservationActivity.this, text: "Erreur lors de la réservation, veuillez essayer ultérieurement.", Toast.LENGTH_SHORT).show();
        }
    }

    @Override
    public void onFailure(@NotNull Call call, @NotNull IOException e) {
        Toast.makeText( context: ReservationActivity.this, text: "Erreur lors de la réservation, veuillez essayer ultérieurement.", Toast.LENGTH_SHORT).show();
    }
});
```

Passons maintenant au contrôleur « RecapitulatifActivity » permettant d'afficher un récapitulatif et une confirmation de la réservation.

2.a - RecapitulatifActivity

On importe et déclare le nécessaire.

```
package com.example.projet_android.controleur;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.example.projet_android.R;

2 usages
public class RecapitulatifActivity extends AppCompatActivity {

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

        if (getSupportActionBar() != null) {
            getSupportActionBar().setDisplayHomeAsUpEnabled(true);
            getSupportActionBar().setTitle("R3STO.FR");
        }

        TextView tvRecap = findViewById(R.id.tvRecap);
    }
}
```

On récupère les variables à l'aide d'un intent, on les concatène et on les affiche, le tout avec une gestion d'erreurs.

```
Intent intent = getIntent();
String nomResto = intent.getStringExtra( name: "nomResto");
String date = intent.getStringExtra( name: "date");
String heure = intent.getStringExtra( name: "heure");

// Vérification que les données ne sont pas nulles
if (nomResto != null && date != null && heure != null) {
    String recapText = "Restaurant : " + nomResto + "\nDate: " + date + "\nHeure: " + heure;
    tvRecap.setText(recapText);
} else {
    tvRecap.setText("Informations de réservation manquantes");
}
}
```

Pour terminer, on développe les deux boutons (retour à l'accueil et voir mes réservations), qui opèrent tous deux une redirection.

```
Button btnGoHome = findViewById(R.id.btnGoHome);
Button btnViewReservations = findViewById(R.id.btnViewReservations);

btnGoHome.setOnClickListener(v -> {
    Intent homeIntent = new Intent( packageContext: this, MainActivity.class);
    homeIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
    startActivity(homeIntent);
});

btnViewReservations.setOnClickListener(v -> {
    Intent profileIntent = new Intent( packageContext: this, ProfilActivity.class);
    startActivity(profileIntent);
});
}
```

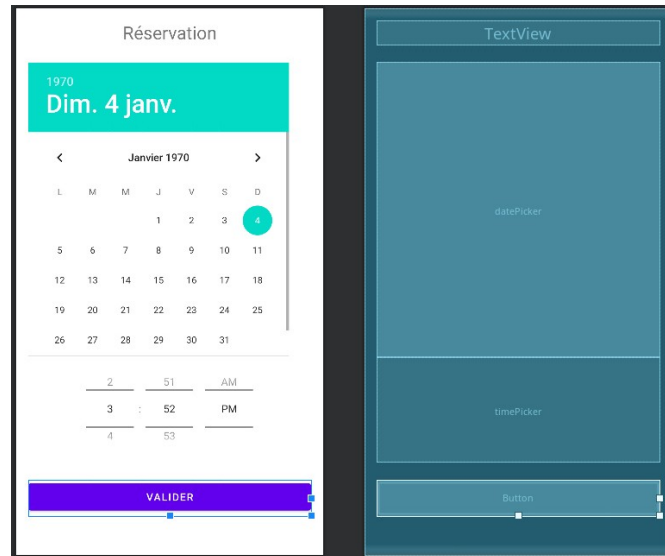
2 – Les vues

Pour cette étape on dispose de deux vues (une par contrôleur).

Commençons par la page de réservation :

2.a - « activity_reservation »

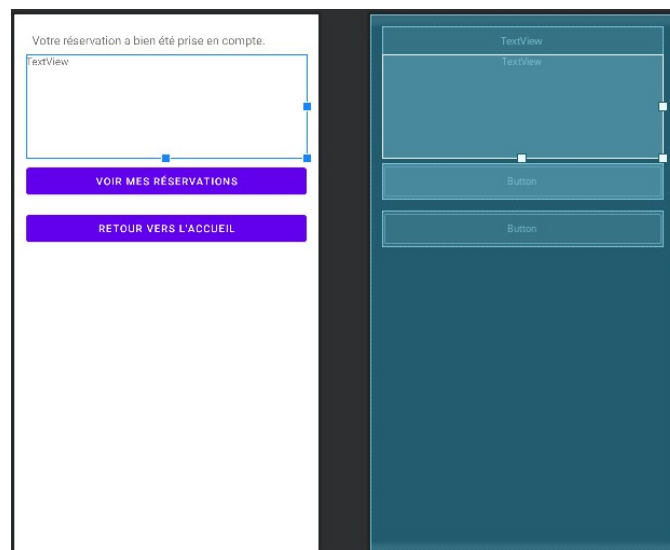
On l'a vu antérieurement, la vue est composé d'un titre, d'un calendrier, d'une horloge et d'un bouton « valider ».



2.a - « activity_recapitulatif »

Ce layout est composé :


- d'un TextView affichant la confirmation ;
- d'un TextView affichant les détails de la réservation ;
- de deux boutons : « voir mes réservation » et « retour vers l'accueil ».




3 – Jeu d’essai

Dans ce jeu d’essai, il conviendra de procéder à la réservation d’un restaurant en bonne et due forme.

Commençons par tester la prise de réservation en étant déconnecté.

R3STO.FR 

21 Quai Amiral Dubourdieu, 64100
Bayonne



Un restaurant moderne proposant
des bières artisanales et des burgers
gastronomiques dans une ambiance
décontractée.


Ouverture	Semaine	Week-end
Midi	de 11h45 à 14h30	de 11h45 à 15h00
Soir	de 18h45 à 22h30	de 18h45 à 1h
À emporter	de 11h30 à 23h	de 11h30 à 2h

Veuillez vous connecter pour réserver.

RÉSERVER

R  it  rons mais connect   cette fois-ci.

R3STO.FR



R  servation

2024

Mon, May 13

<

May 2024

>

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

3

08

AM

4

:

09

PM

5

10

VALIDER

Bien, cela fonctionne.

Ajoutons une r  servation, le 27/06/2024    13:30 par exemple.

R3STO.FR



Votre r  servation a bien   t   prise en compte.

Restaurant : La table de POTTOKA

Date: 2024-07-27

Heure: 13:30:00

VOIR MES R  SERVATIONS

RETOUR VERS L'ACCUEIL

Cliquons sur « Voir mes réservations » pour s'assurer qu'elle s'affiche correctement sur la page du profil.

R3STO.FR

Profil

Bonjour Testeur SIO !

Vos réservations

Sapporo - Date : 2024-06-10, Heure : 12:32:00

Agadir - Date : 2024-06-10, Heure : 13:21:00

Sapporo - Date : 2024-06-15, Heure : 16:17:00

la petite auberge - Date : 2024-06-13, Heure : 00:31:00

La table de POTTOKA - Date : 2024-07-27, Heure : 13:30:00

DÉCONNEXION

On obtient le résultat escompté.

Pour rappel, si vous souhaitez obtenir de plus amples détails concernant la base de données vous trouverez un pdf consacré à ce sujet sur le GitHub.