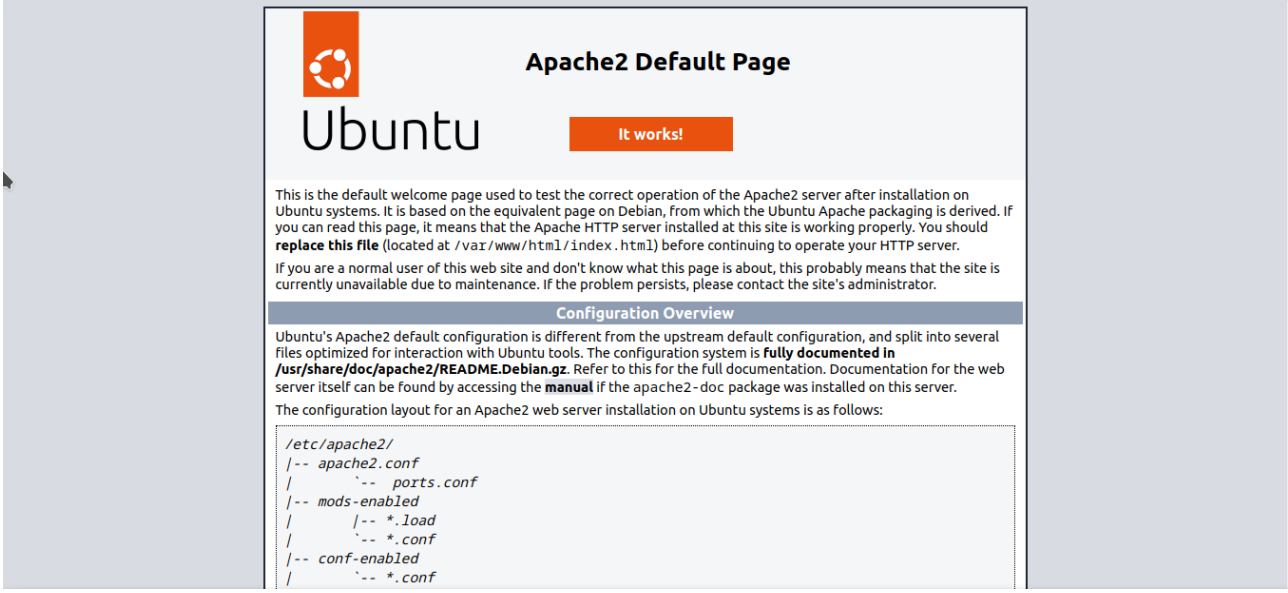
TP DE MISE EN PLACE D'UNE PETITE API D'AUTHENTIFICATION DE RIEN DU TOUT EN LOCAL (Android & PHP -Mysql)

Le système connexion/inscription sur Android est un scénario très courant. Nous allons configurer un serveur Web local et une base de données MySQL. Nous développerons une application de connexion et d'inscription Android. Nous utiliserons un script PHP pour nous connecter à la base de données MySQL pour l'occasion.

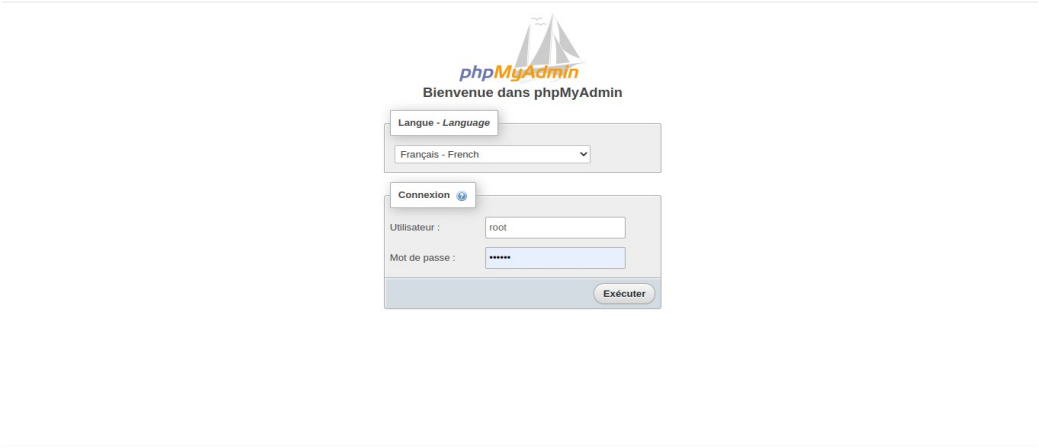
La première étape consiste à créer le serveur Web principal. Je travaille sur LINUX et LAMP peut être utilisé pour configurer rapidement un serveur Web Apache local et une base de données MySQL.

LAMP (ou WAMP) est un logiciel d'installation en un clic qui crée un environnement pour développer une application Web PHP, MySQL (que nous connecterons avec notre application Android). Téléchargez et installez LAMP ou WAMP au besoin.

Vous pouvez tester votre serveur en ouvrant http://localhost. L'écran suivant devrait apparaître.



Vous pouvez également vérifier phpMyAdmin en ouvrant http://localhost/phpmyadmin. Voyons ce que cela montre !



Dans phpmyadmin, créez une base de données de votre choix. Pour ce cas une BD **tp_api** a été créée. Ajoutons-y une table **users**. C'est-à-dire notre seule table qui stockera les infos d'authentification de nos utilisateurs.

Pour connecter un script PHP à la base de données MySQL, trois valeurs d'entrée sont requises. Voici les entrées et leurs valeurs par défaut pour un serveur LAMP ou WAMP :

Nom d'hôte : localhost
Nom d'utilisateur MySQL : root
Mot de passe MySQL : Il est vide. ""
Créons un script **test-connect.php** et ajoutons-le dans la racine du serveur local

```php
<?php
$servername = 'localhost';
$username = 'root';
$password = 'claire';
//On établit la connexion
```

```
$conn = new mysqli($servername, $username, $password);
//On vérifie la connexion
if($conn->connect_error){
die('Erreur : ' .$conn->connect_error);
}
echo 'Connexion réussie';
?>
```

Maintenant que nous avons discuté de la configuration de base de PHP et MySQL, passons à la partie application de connexion Android. Nous développerons une application de connexion/inscription. Pour rester bref et simple, nous vérifierons si le nom d'utilisateur et l'e-mail sont uniques lors de l'inscription. Avant de passer à la logique de l'application, travaillons sur les scripts PHP et la base de données MySQL.

Le script suivant contient toutes les fonctions principales de l'application. **user.php**

```
<?php
include_once 'db-connect.php';
class User{
private $db;
private $db_table = "users";
public function __construct(){
$this->db = new DbConnect();
}
public function isLoginExist($username, $password){
$query = "select * from ".$this->db_table." where username = '$username' AND password =
'$password' Limit 1";
$result = mysqli_query($this->db->getDb(), $query);
if(mysqli_num_rows($result) > 0){
mysqli_close($this->db->getDb());
return true;
}
```

```
mysqli_close($this->db->getDb());
return false;
}
public function isEmailUsernameExist($username, $email){
$query = "select * from ".$this->db_table." where username = '$username' AND email = '$email'";
$result = mysqli_query($this->db->getDb(), $query);
if(mysqli_num_rows($result) > 0){
mysqli_close($this->db->getDb());
return true;
}
return false;
}
public function isValidEmail($email){
return filter_var($email, FILTER_VALIDATE_EMAIL) !== false;
}
public function createNewRegisterUser($username, $password, $email){
$isExisting = $this->isEmailUsernameExist($username, $email);
if($isExisting){
$json['success'] = 0;
$json['message'] = "Erreur lors de l'inscription. Le nom d'utilisateur/mail existe probablement
déjà";
}
else{
$isValid = $this->isValidEmail($email);
if($isValid)
{
$query = "insert into ".$this->db_table." (username, password, email, created_at, updated_at)
values ('$username', '$password', '$email', NOW(), NOW())";
$inserted = mysqli_query($this->db->getDb(), $query);
if($inserted == 1){
$json['success'] = 1;
$json['message'] = "L'utilisateur a été enregistré avec succès";
}else{
$json['success'] = 0;
```

```
$json['message'] = "Erreur lors de l'inscription. Le nom d'utilisateur/mail existe probablement
déjà";
}
mysqli_close($this->db->getDb());
}
else{
$json['success'] = 0;
$json['message'] = "Erreur lors de l'inscription. L'adresse mail n'est pas valide";
}
}
return $json;
}
public function loginUsers($username, $password){
$json = array();
$canUserLogin = $this->isLoginExist($username, $password);
if($canUserLogin){
$json['success'] = 1;
$json['message'] = "Connexion réussie";
}else{
$json['success'] = 0;
$json['message'] = "Détails incorrects";
}
return $json;
}
}
?>
```

Dans le code ci-dessus, le **$json** contient les **JSONObjects** renvoyés. Le script PHP suivant est celui qui est appelé en premier depuis l'application. **index.php**

```
<?php
require_once 'user.php';
$username = "";
```

```php
$password = "";
$email = "";
if(isset($_POST['username'])){
$username = $_POST['username'];
}
if(isset($_POST['password'])){
$password = $_POST['password'];
}
if(isset($_POST['email'])){
$email = $_POST['email'];
}
$userObject = new User();
// Inscription
if(!empty($username) && !empty($password) && !empty($email)){
$hashed_password = md5($password);
$json_registration = $userObject->createNewRegisterUser($username, $hashed_password,
$email);
echo json_encode($json_registration);
}
// Connexion
if(!empty($username) && !empty($password) && empty($email)){
$hashed_password = md5($password);
$json_array = $userObject->loginUsers($username, $hashed_password);
echo json_encode($json_array);
}
?>
```

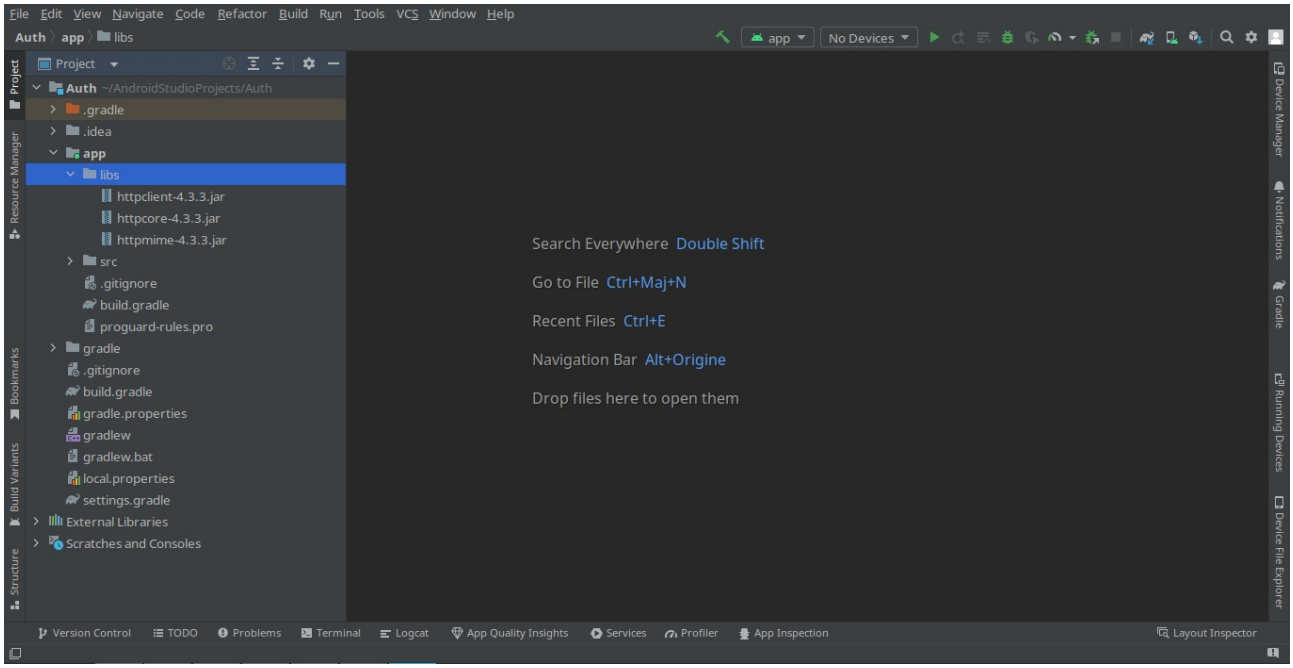
Dans le code ci-dessus, nous vérifions si le champ **email** est vide ou non. Si c'est le cas, nous appellerons la fonction de connexion dans le script PHP, sinon nous irons à la fonction d'enregistrement. La réponse **JSON** renvoie deux paramètres : **success(0 ou 1)** et le **message**.

La fonction **md5()** utilise l'algorithme **MD5 Message-Digest de RSA Data Security, Inc**. pour créer une chaîne de hachage du mot de passe.

Pour vérifier si l'adresse mail est valide, nous avons implémenté une méthode **isValidEmail()**. **FILTER_VALIDATE_EMAIL** fonctionne sur les versions PHP 5.2.0+

Partie ANDROID

Dans ce projet, nous avons utilisé trois bibliothèques pour implémenter les appels **HTTP** dans notre application. La classe **JSONParser** est utilisée pour effectuer les appels **POST** et **GET HTTP** vers l'hôte local et renvoyer la réponse sous la forme d'un **JSONObject**.



Voici le code de connexion android

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fillViewport="true">
```

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:paddingLeft="24dp"
        android:paddingRight="24dp"
        android:id="@+id/linearLayout">

        <EditText android:id="@+id/editName"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Nom d'utilisateur"
            android:textColor="#FF192133"
            android:textColorHint="#A0192133"
            android:fontFamily="sans-serif-light"
            android:focusable="true"
            android:focusableInTouchMode="true" />

        <EditText android:id="@+id/editPassword"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:inputType="textPassword"
            android:textColor="#FF192133"
            android:textColorHint="#A0192133"
            android:fontFamily="sans-serif-light"
            android:hint="Mot de passe"
            android:focusable="true"
            android:focusableInTouchMode="true" />

        <EditText android:id="@+id/editEmail"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:inputType="textEmailAddress"
            android:textColor="#FF192133"
            android:visibility="gone"
            android:textColorHint="#A0192133"
            android:fontFamily="sans-serif-light"
            android:hint="Email"
            android:focusable="true"
            android:focusableInTouchMode="true" />

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/btnSignIn"
            android:text="SE CONNECTER"
            android:textStyle="bold"
            />

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/btnRegister"
            android:text="S'INSCRIRE"
            android:textStyle="bold"
            />

    </LinearLayout>

  </RelativeLayout>

</ScrollView>
```

Le code du MainActivity ci-dessous :

```java
package com.example.auth;

import androidx.appcompat.app.AppCompatActivity;

import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import org.apache.http.NameValuePair;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    EditText editEmail, editPassword, editName;
    Button btnSignIn, btnRegister;

    String URL= "http://192.168.19.196/ins_con_android/index.php";

    JSONParser jsonParser=new JSONParser();

    int i=0;


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

```
editEmail=(EditText)findViewById(R.id.editEmail);
editName=(EditText)findViewById(R.id.editName);
editPassword=(EditText)findViewById(R.id.editPassword);

btnSignIn=(Button)findViewById(R.id.btnSignIn);
btnRegister=(Button)findViewById(R.id.btnRegister);

btnSignIn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        AttemptLogin attemptLogin= new AttemptLogin();
        attemptLogin.execute(editName.getText().toString(),editPassword.getText().toString(),"");
    }
});

btnRegister.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if(i==0)
        {
            i=1;
            editEmail.setVisibility(View.VISIBLE);
            btnSignIn.setVisibility(View.GONE);
            btnRegister.setText("CRÉER UN COMPTE");
        }
        else{

            btnRegister.setText("ENREGISTRER");
            editEmail.setVisibility(View.GONE);
            btnSignIn.setVisibility(View.VISIBLE);
            i=0;

            AttemptLogin attemptLogin= new AttemptLogin();
```

```
attemptLogin.execute(editName.getText().toString(),editPassword.getText().toString(),editEmail.getText
().toString());

            }

        }
    });
}

private class AttemptLogin extends AsyncTask<String, String, JSONObject> {

    @Override

    protected void onPreExecute() {

        super.onPreExecute();

    }

    @Override

    protected JSONObject doInBackground(String... args) {



        String email = args[2];
        String password = args[1];
        String name= args[0];

        ArrayList<NameValuePair> params = new ArrayList<NameValuePair>();
        params.add(new BasicNameValuePair("username", name));
        params.add(new BasicNameValuePair("password", password));
        if(email.length()>0)
            params.add(new BasicNameValuePair("email",email));
```

```
        JSONObject json = jsonParser.makeHttpRequest(URL, "POST", params);


        return json;

    }

    protected void onPostExecute(JSONObject result) {

        // dismiss the dialog once product deleted
        //Toast.makeText(getApplicationContext(),result,Toast.LENGTH_LONG).show();

        try {
            if (result != null) {

Toast.makeText(getApplicationContext(),result.getString("message"),Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(getApplicationContext(), "Impossible de récupérer les données du serveur",
Toast.LENGTH_LONG).show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }


    }

  }
}
```

Le code est assez gros ! Tirons les conclusions importantes du code ci-dessus.

http://191.168.19.196 est l'adresse de **routage localhost**. Cette adresse fonctionne exclusivement si vous avez configuré un réseau local. Elle pourrait être différente.
Lorsque le bouton **ENREGISTRER** est cliqué, nous masquons par programmation le bouton **CONNEXION** et affichons à la place le champ de saisie de l'adresse e-mail.
La classe **AttemptLogin** exécute les requêtes **HTTP** du réseau vers notre localhost en arrière-plan. Les paramètres de nom d'utilisateur, de mot de passe et d'e-mail sont ajoutés à une **ArrayList** transmise dans la méthode **makeHttpRequest(URL, "POST", params);** de la classe **JSONParser**.
Dans la méthode **onPostExecute**, nous affichons la chaîne de message renvoyée par le serveur dans un message Toast.

Le code de parsing des données `JSONParser.java`

```
package com.example.auth;

import android.util.Log;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.utils.URLEncodedUtils;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
```

```
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.ArrayList;

public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static JSONArray jArr = null;
    static String json = "";
    static String error = "";

    // constructeur
    public JSONParser() {

    }

    // fonction obtenir json à partir de l'url
    // en faisant HTTP POST ou GET mehtod
    public JSONObject makeHttpRequest(String url, String method,
                                      ArrayList<NameValuePair> params) {

        // Effectuer une requête http
        try {

            // vérifier la méthode de demande
            if(method.equals("POST")){
                // la méthode de requête est POST
                // defaultHttpClient
                HttpClient httpClient = new DefaultHttpClient();
                HttpPost httpPost = new HttpPost(url);
                httpPost.setEntity(new UrlEncodedFormEntity(params));
                try {
                    Log.e("API123", " " +convertStreamToString(httpPost.getEntity().getContent()));
                    Log.e("API123",httpPost.getURI().toString());
```

```
            } catch (Exception e) {
                e.printStackTrace();
            }

            HttpResponse httpResponse = httpClient.execute(httpPost);
            Log.e("API123",""+httpResponse.getStatusLine().getStatusCode());
            error= String.valueOf(httpResponse.getStatusLine().getStatusCode());
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        }else if(method.equals("GET")){
            // la méthode de requête est GET
            DefaultHttpClient httpClient = new DefaultHttpClient();
            String paramString = URLEncodedUtils.format(params, "utf-8");
            url += "?" + paramString;
            HttpGet httpGet = new HttpGet(url);

            HttpResponse httpResponse = httpClient.execute(httpGet);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();
        }

    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    } catch (ClientProtocolException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }

    try {
        BufferedReader reader = new BufferedReader(new InputStreamReader(
                is, "iso-8859-1"), 8);
        StringBuilder sb = new StringBuilder();
        String line = null;
```

```
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
            is.close();
            json = sb.toString();
            Log.d("API123",json);
        } catch (Exception e) {
            Log.e("Erreur de tampon", "Erreur lors de la conversion du résultat " + e.toString());
        }

        // try parse the string to a JSON object
        try {
            jObj = new JSONObject(json);
            jObj.put("error_code",error);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Erreur lors de l'analyse des données " + e.toString());
        }

        // return JSON String
        return jObj;

    }

    private String convertStreamToString(InputStream is) throws Exception {
        BufferedReader reader = new BufferedReader(new InputStreamReader(is));
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
            sb.append(line);
        }
        is.close();
        return sb.toString();
    }
}
```

Dans le code ci-dessus, nous appelons les classes respectives **HTTPPost** ou **HTTPG** et en fonction du deuxième paramètre passé dans la fonction **makeHttpRequest**.

Remarque : N'oubliez pas d'ajouter l'autorisation suivante dans votre fichier **AndroidManifest.xml**.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Veuillez noter que depuis Android 6.0 et supérieur, vous devez ajouter l'attribut suivant dans votre balise d'application dans le fichier Manifest.xml : **android:usesCleartextTraffic="true"**
Pourquoi ? Afin de permettre à la sécurité du réseau de l'émulateur/de l'appareil d'effectuer des appels http.