package com.mylink.bustrackingsystem;

import android.app.Activity;

import android.app.ProgressDialog;

import android.content.Intent;

import android.content.SharedPreferences;

import android.os.AsyncTask;

import android.os.Bundle;

import android.os.Handler;

import android.os.HandlerThread;

import android.preference.PreferenceManager;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import android.widget.Toast;

import org.json.JSONArray;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.URLEncoder;

import cz.msebera.android.httpclient.HttpEntity;

import cz.msebera.android.httpclient.HttpResponse;

import cz.msebera.android.httpclient.client.HttpClient;

import cz.msebera.android.httpclient.client.methods.HttpPost;

import cz.msebera.android.httpclient.impl.client.DefaultHttpClient;

public class DriverDashboard extends Activity {

Button b1,b3;

HandlerThread hThread ;

String currentlatitude,currentlongitude;

double latitude=0.0;

double longitude=0.0;

GPSTracker gps = null;

TextView tv1;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.driverdashboard);

b1=(Button)findViewById(R.id.startjourney);

b3=(Button)findViewById(R.id.logout);

tv1=(TextView)findViewById(R.id.location);

b1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i1= new Intent(DriverDashboard.this,MainActivity.class);

startActivity(i1);

}

});

b3.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i1= new Intent(DriverDashboard.this,MainActivity.class);

startActivity(i1);

}

});

gps= new GPSTracker(DriverDashboard.this);

hThread = new HandlerThread("HandlerThread");

hThread.start();

final Handler handler = new Handler(hThread.getLooper());

final long oneMinuteMs = 20 \* 500;

Runnable eachMinute = new Runnable() {

@Override

public void run() {

handler.postDelayed(this, oneMinuteMs);

handler.postDelayed(this, oneMinuteMs);

try{

latitude = gps.getLatitude();

longitude = gps.getLongitude();

Toast.makeText(DriverDashboard.this, "Your Location is - \nLat: "

+ latitude + "\nLong: " + longitude, Toast.LENGTH\_LONG).show();

tv1.setText("Lat : "+latitude+" Lang : "+longitude);

try{

SharedPreferences shr= PreferenceManager.getDefaultSharedPreferences(DriverDashboard.this);

String lat=shr.getString("uid","");

UpdateLoc gettrans=new UpdateLoc();

DbParameter host=new DbParameter();

String url=host.getHostpath();

url=url+"/UpdateLocation.php?uid="+ URLEncoder.encode(lat)+"&";

url=url+"&lat="+URLEncoder.encode(String.valueOf( latitude));

url=url+"&lang="+URLEncoder.encode(String.valueOf(longitude));

gettrans.execute(url);

}catch(Exception e){

Toast.makeText(DriverDashboard.this, ""+e, Toast.LENGTH\_LONG).show();

}

}catch(Exception e){

Toast.makeText(DriverDashboard.this, ""+e,Toast.LENGTH\_LONG).show();

}

}

};

handler.postDelayed(eachMinute, oneMinuteMs);

}

private class UpdateLoc extends AsyncTask<String, Integer, String> {

private ProgressDialog progress = null;

String out="";

@Override

protected String doInBackground(String... geturl) {

try{

// String url= ;

HttpClient http=new DefaultHttpClient();

HttpPost http\_get= new HttpPost(geturl[0]);

HttpResponse response=http.execute(http\_get);

HttpEntity http\_entity=response.getEntity();

BufferedReader br= new BufferedReader(new InputStreamReader(http\_entity.getContent()));

out = br.readLine();

}catch (Exception e){

out= e.toString();

}

return out;

}

@Override

protected void onPreExecute() {

// progress = ProgressDialog.show(DriverDashboard.this, null,

// "Loading Products...");

super.onPreExecute();

}

@Override

protected void onPostExecute(String result) {

// TODO Auto-generated method stub

Toast.makeText(DriverDashboard.this, " "+out, 500).show();

try{

}catch(Exception e){

//Toast.makeText(SearchByLocation.this,"Results Not Founds"+e, Toast.LENGTH\_LONG).show();

}

// progress.dismiss();

super.onPostExecute(result);

}

}

}

1. **Subclassing AsyncTask**: You create a subclass of **AsyncTask** and override its **doInBackground()**, **onPostExecute()**, and optionally **onPreExecute()** and **onProgressUpdate()** methods.
2. **Execution**: You instantiate your **AsyncTask** subclass and execute it by calling its **execute()** method. This starts the background task.
3. **Background Work**: The **doInBackground()** method runs on a background thread, allowing you to perform time-consuming tasks without blocking the UI thread. This is where you put your code that you want to run asynchronously.
4. **Publishing Results**: If needed, you can use the **publishProgress()** method within **doInBackground()** to publish intermediate results to the UI thread. This triggers a call to the **onProgressUpdate()** method.
5. **Updating UI**: The **onPostExecute()** method, which runs on the UI thread, is called after the background task completes. You can use it to update the UI with the results of the background operation.

It's worth noting that starting from Android API level 30, **AsyncTask** has been deprecated in favor of other concurrency utilities like **java.util.concurrent** or Kotlin's coroutines. This is mainly because **AsyncTask** has certain limitations, such as poor error handling and potential memory leaks if not used carefully. However, it's still widely used in existing projects and can be a simple solution for basic asynchronous tasks in Android development.

package com.mylink.bustrackingsystem;

import android.content.Context;

import com.google.android.gms.vision.MultiProcessor;

import com.google.android.gms.vision.Tracker;

import com.google.android.gms.vision.face.Face;

public class FaceTrackerDaemon implements MultiProcessor.Factory<Face> {

private Context context;

public FaceTrackerDaemon(Context context) {

this.context = context;

}

@Override

public Tracker<Face> create(Face face) {

return new EyesTracker(context);

}

}

package com.mylink.bustrackingsystem;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class Login extends Activity {

Button b1;

EditText uname,pass;

TextView tv1;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.login);

b1=(Button)findViewById(R.id.main\_login);

uname=(EditText)findViewById(R.id.main\_user\_name);

pass=(EditText)findViewById(R.id.main\_password);

tv1=(TextView)findViewById(R.id.signup);

tv1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent i1=new Intent(Login.this,DriverLogin.class);

startActivity(i1);

}

});

b1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View arg0) {

// TODO Auto-generated method stub

boolean status=true;

if(uname.getText().toString().contentEquals("")){

uname.setError("Enter Username");

uname.requestFocus();

status=false;

}

if(pass.getText().toString().contentEquals("")){

pass.setError("Enter Password");

pass.requestFocus();

status=false;

}

if(status){

if(pass.getText().toString().contentEquals("admin")&&uname.getText().toString().contentEquals("admin")){

Intent i1= new Intent(Login.this,AdminDashboard.class);

startActivity(i1);

}else{

Toast.makeText(Login.this, "Invalid Username or Password", Toast.LENGTH\_LONG).show();

}

}

}

});

}

}

Implementing a static admin login in an Android application typically involves creating a login screen where the user enters their credentials (username and password) and then verifying these credentials against hardcoded values for the admin account. Here's a basic outline of how you could do this:

1. **Layout**: Create an XML layout file for the login screen (**activity\_login.xml**). This layout should contain EditText fields for username and password, and a Button to trigger the login process.
2. **Activity**: Create a corresponding **LoginActivity** class that inflates the layout file and handles user interactions.
3. **Verify Credentials**: In the **LoginActivity**, when the user clicks the login button, retrieve the values entered in the username and password fields. Then, compare these values against hardcoded values for the admin account.
4. **Show Result**: If the entered credentials match the admin credentials, you can navigate to the admin dashboard or perform any other desired action. Otherwise, display an error message indicating that the login failed.

package com.mylink.bustrackingsystem;

import android.app.Activity;

import android.app.ProgressDialog;

import android.os.AsyncTask;

import android.os.Bundle;

import android.widget.Button;

import android.widget.ListView;

import android.widget.Spinner;

import android.widget.Toast;

import org.json.JSONArray;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.URLEncoder;

import cz.msebera.android.httpclient.HttpEntity;

import cz.msebera.android.httpclient.HttpResponse;

import cz.msebera.android.httpclient.client.HttpClient;

import cz.msebera.android.httpclient.client.methods.HttpPost;

import cz.msebera.android.httpclient.impl.client.DefaultHttpClient;

public class ViewDrowsiness extends Activity {

String vialocation[],time[],currentstatus[],uid[];

Button addbus;

Spinner from ,to;

ListView lst;

@Override

protected void onCreate(Bundle savedInstanceState) {

// TODO Auto-generated method stub

super.onCreate(savedInstanceState);

setContentView(R.layout.viewlocation);

lst=(ListView)findViewById(R.id.listView1);

try{

RegisterUser gettrans=new RegisterUser();

DbParameter host=new DbParameter();

String url=host.getHostpath();

url=url+"ViewAllBuses.php";

// url=url+"to="+ URLEncoder.encode(to.getSelectedItem().toString())+"&"

// url=url+"area="+URLEncoder.encode(user\_area)+"&";

gettrans.execute(url);

}catch(Exception e){

Toast.makeText(ViewDrowsiness.this, ""+e, Toast.LENGTH\_LONG).show();

}

}

public class RegisterUser extends AsyncTask<String, Integer, String> {

private ProgressDialog progress = null;

String out="";

@Override

protected String doInBackground(String... geturl) {

try{

// String url= ;

HttpClient http=new DefaultHttpClient();

HttpPost http\_get= new HttpPost(geturl[0]);

HttpResponse response=http.execute(http\_get);

HttpEntity http\_entity=response.getEntity();

BufferedReader br= new BufferedReader(new InputStreamReader(http\_entity.getContent()));

out = br.readLine();

}catch (Exception e){

out= e.toString();

}

return out;

}

@Override

protected void onPreExecute() {

progress = ProgressDialog.show(ViewDrowsiness.this, null,

"Registering...");

super.onPreExecute();

}

@Override

protected void onPostExecute(String result) {

// TODO Auto-generated method stub

//Toast.makeText(ViewDrowsiness.this,""+out,Toast.LENGTH\_LONG).show();

try{

JSONObject jsonResponse = new JSONObject(out);

JSONArray jsonMainNode = jsonResponse.optJSONArray("user\_info");

int arraylength=jsonMainNode.length();

vialocation =new String[arraylength];

time =new String[arraylength];

currentstatus=new String[arraylength];

uid=new String[arraylength];

//links =new String[arraylength];

String loc="r";

for (int i = 0; i < jsonMainNode.length(); i++) {

JSONObject jsonChildNode = jsonMainNode.getJSONObject(i);

uid[i]=jsonChildNode.optString("OwnerName");

vialocation[i]=" Driver Name : "+jsonChildNode.optString("OwnerName");

time[i]="Veh No : "+jsonChildNode.optString("BusNameNumber")+"| Via: "+jsonChildNode.optString("via");

currentstatus[i]="Lat :"+jsonChildNode.optString("lattitude")+"| Lang "+jsonChildNode.optString("longitude") ;

}

LevelAdapter lv1= new LevelAdapter(ViewDrowsiness.this,vialocation,time,currentstatus);

lst.setAdapter(lv1);

}catch(Exception e){

Toast.makeText( ViewDrowsiness.this, "No Records Found", Toast.LENGTH\_LONG).show();

}

progress.dismiss();

super.onPostExecute(result);

}

}

}

In Android development, a JSON object is a data structure used to store and exchange data in JSON (JavaScript Object Notation) format. JSON is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.

In Android, you can work with JSON objects using the built-in **JSONObject** class provided by the **org.json** package. This class allows you to create, manipulate, and parse JSON objects.