04/04/2021

-Report Project Ping Pong-

Android Studio Application

Victor Quidet & Théophile Tarbé
ING4 SI GR01

Mobile Programming   K. KHOURY

I - List of functions implemented

1- We created an ergonomic interface using fragments (for a newGame, newPoint, Maps, Photo, listOfGames, Statistics …) and a BottomNavigation in the HomeActivity in order to switch between fragments (newGame, newPoint, Maps, Photo).

2- The application is available in 2 languages : English (default) and French.

3- The interface is adapted to landscape and portrait view.

4- The fragment Map allows the user to get a Google Map view.

5- The user can take pictures. Each picture is first displayed on the screen and the user can then decide to save it locally inside the picture directory of the emulator.

6- Statistics of each game are first saved locally in an SQLite database (for the 5 first games).

7- Full statistics are saved in a real external database (MySQL)

8- The user can then access the list of past games and their statistics through the fragment listGames and Statistics.

II - List of functions not implemented

Everything asked in the subject has been implemented except the precise localisation of the user on the Google Map.

III - Possible improvements

We put the functional aspect of the application first, which is why the front-end is less well developed. One major improvement could be to get a better design aspect.

The main improvement could be the implementation of the Google Map in order to make the geolocalisation work with the position of the user, and the possibility to research a new location for a match. We first succeeded, but when we got the localisation the map wasn’t visible anymore. So we chose to let the map visible but the marker point is not working since the app doesn't get the right location. To put it in a nutshell, we didn’t succeed in showing the map plus the user localisation.

The statistics part could be improved by displaying charts about the games (the chart on the application is not dynamically related to the real statistics).

IV - Application architecture : implementation and screenshots

A) Front-End Architecture

The implementation of the front-end is explained through the diagram (Figure A). We used some activities (Home & History) in order to access the 2 main parts of the application and be able to register games and see their statistics by using fragments (NewGame, NewMatch, Map, Photo, ListOfGames, Statistics).

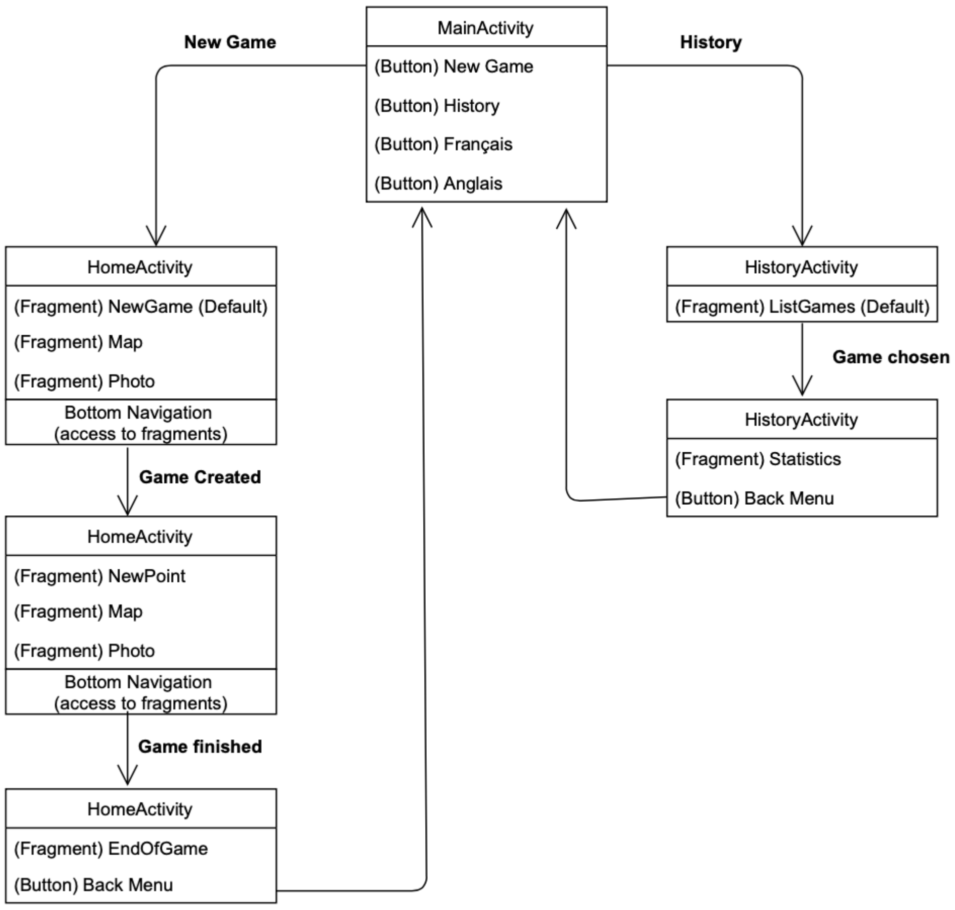
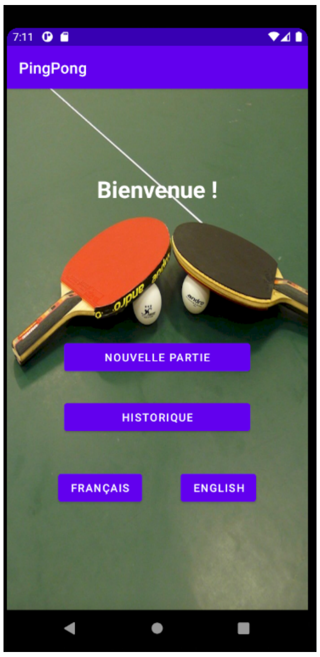
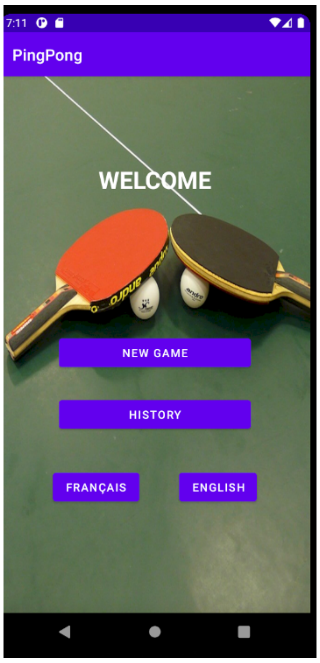


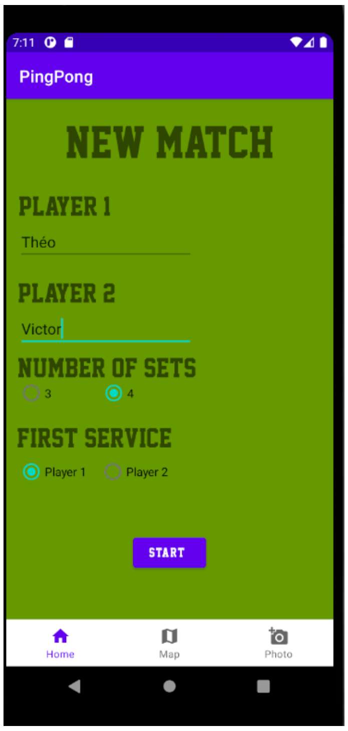
Figure A

Here are some screenshots of the Front-End of the application:





Menus in French/English



NEW MATCH

PLAYER 1

Théo

PLAYER 2

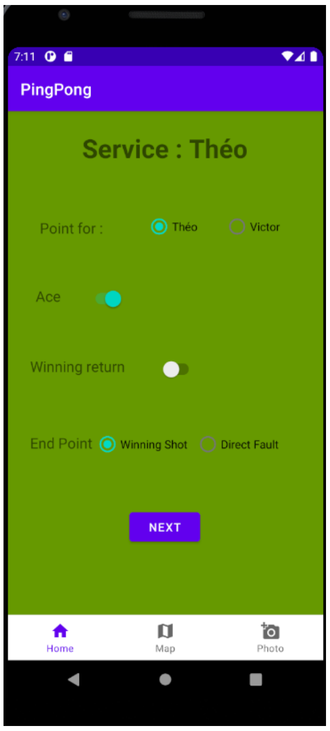
Victor

NUMBER OF SETS

3 [✓ 4]

FIRST SERVICE

[✓ Player 1] Player 2



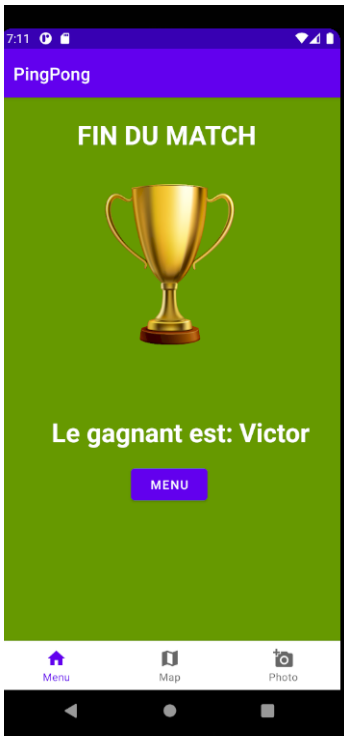
Service : Théo

Point for : [✓ Théo] Victor

Ace [✓]

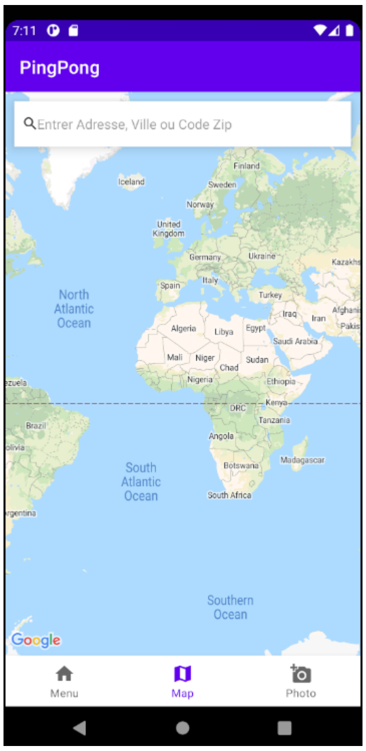
Winning return

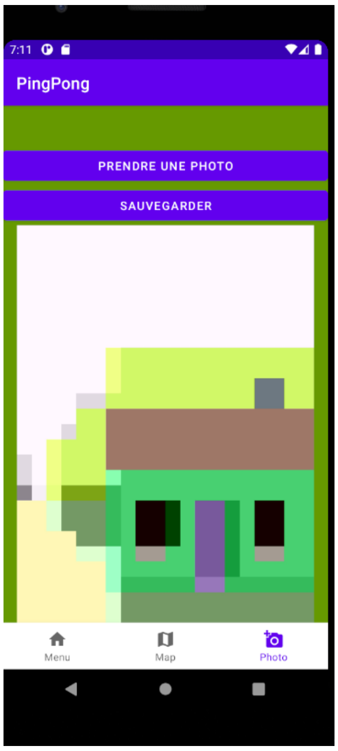
End Point [✓ Winning Shot] Direct Fault

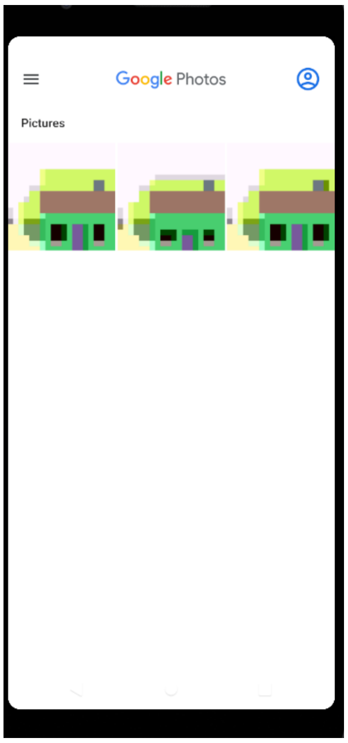


Le gagnant est: Victor

Create New Match - New Points - EndGame (Name of the winner)







Map - Take a picture and display on the app - Saving picture locally

B) Back-End Architecture

The implementation of the front-end is explained through the diagram (figure B).

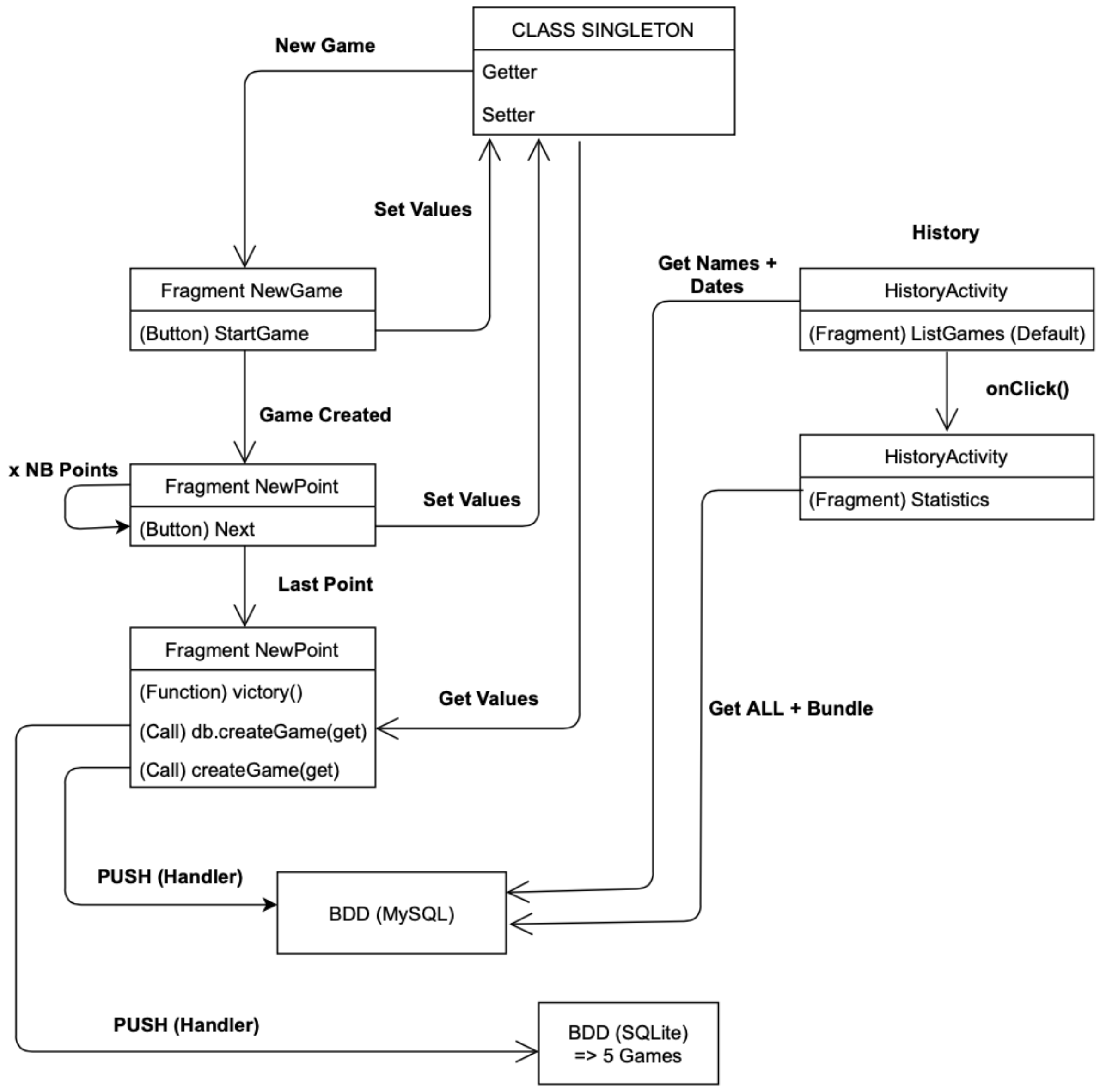
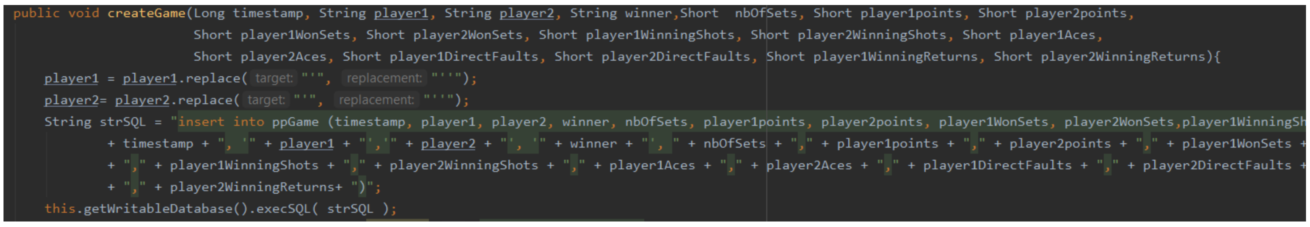


Figure B

Regarding the Back-End, we used a Singleton class which is a unique instance of an object accessible through the whole app with the getters and setters. This class allowed us to easily manage our data through the all app.

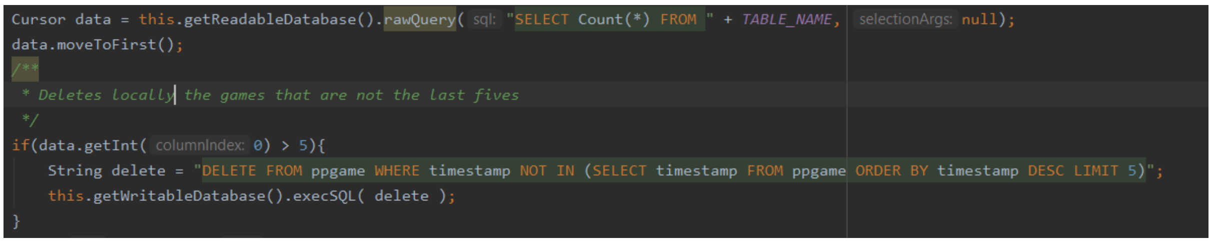
We have also implemented the 2 different ways of storing data in our application :

- **Internal** : SQLite with a class named MySQLiteGameHelper which allowed us to stock the data of <u>the 5 last games</u> locally in a database.



```java
public void createGame(Long timestamp, String player1, String player2, String winner,Short  nbOfSets, Short player1points, Short player2points,
                        Short player1WonSets, Short player2WonSets, Short player1WinningShots, Short player2WinningShots, Short player1Aces,
                        Short player2Aces, Short player1DirectFaults, Short player2DirectFaults, Short player1WinningReturns, Short player2WinningReturns){
    player1 = player1.replace( target: "'", replacement: "''");
    player2= player2.replace( target: "'", replacement: "''");
    String strSQL = "insert into ppGame (timestamp, player1, player2, winner, nbOfSets, player1points, player2points, player1WonSets, player2WonSets,player1WinningSh
            + timestamp + ", '" + player1 + "','" + player2 + "', '" + winner + "', " + nbOfSets + "," + player1points + "," + player2points + "," + player1WonSets +
            + "," + player1WinningShots + "," + player2WinningShots + "," + player1Aces + "," + player2Aces + "," + player1DirectFaults + "," + player2DirectFaults +
            + "," + player2WinningReturns+ ")";
    this.getWritableDatabase().execSQL( strSQL );
```

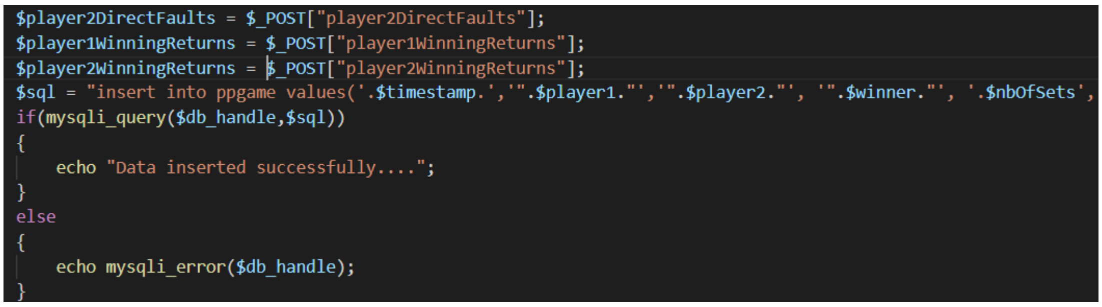
<u>Insert into Database MySQLite</u>



```java
Cursor data = this.getReadableDatabase().rawQuery( sql: "SELECT Count(*) FROM " + TABLE_NAME, selectionArgs: null);
data.moveToFirst();
/**
 * Deletes locally the games that are not the last fives
 */
if(data.getInt( columnIndex: 0) > 5){
    String delete = "DELETE FROM ppgame WHERE timestamp NOT IN (SELECT timestamp FROM ppgame ORDER BY timestamp DESC LIMIT 5)";
    this.getWritableDatabase().execSQL( delete );
}
```

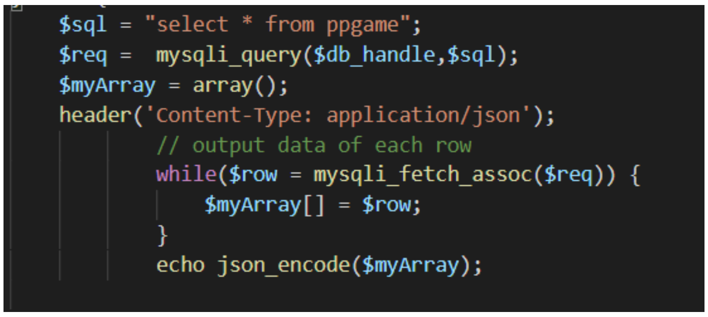
<u>Delete matches that are not the 5 last</u>

- **External** : the rest of the games are stored in a external database using MySQL Database and a Web interface. We used for POST requests. We used a handler to insert the data and an AsyncTask to retrieve the data.



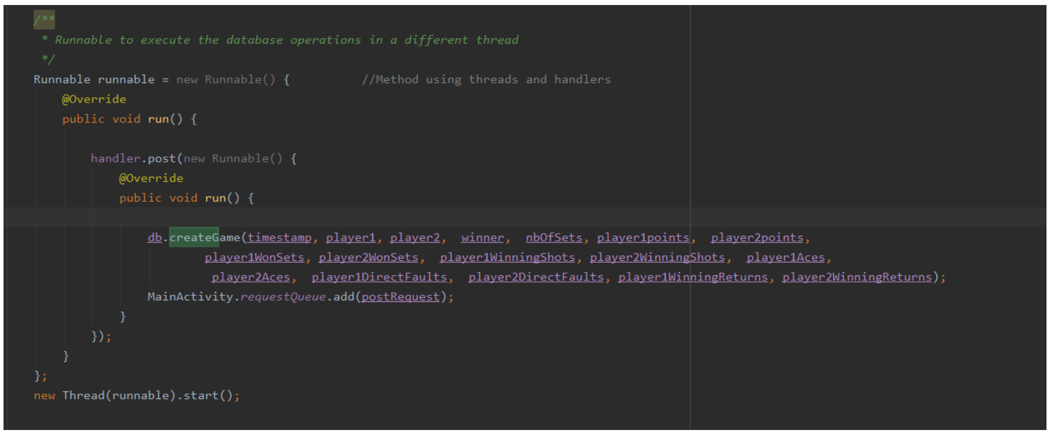
```php
$player2DirectFaults = $_POST["player2DirectFaults"];
$player1WinningReturns = $_POST["player1WinningReturns"];
$player2WinningReturns = $_POST["player2WinningReturns"];
$sql = "insert into ppgame values('.$timestamp.','".$player1."','".$player2."', '".$winner."', '.$nbOfSets',
if(mysqli_query($db_handle,$sql))
{
    echo "Data inserted successfully....";
}
else
{
    echo mysqli_error($db_handle);
}
```

<u>Insert into database MySQL</u>



```php
    $sql = "select * from ppgame";
    $req =  mysqli_query($db_handle,$sql);
    $myArray = array();
    header('Content-Type: application/json');
            // output data of each row
            while($row = mysqli_fetch_assoc($req)) {
                $myArray[] = $row;
            }
            echo json_encode($myArray);
```

<u>Retrieve all the data from the BDD in a JSON array</u>



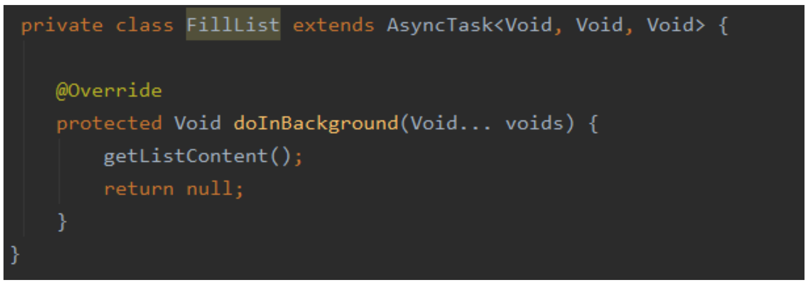
```java
/**
 * Runnable to execute the database operations in a different thread
 */
Runnable runnable = new Runnable() {          //Method using threads and handlers
    @Override
    public void run() {

        handler.post(new Runnable() {
            @Override
            public void run() {

                db.createGame(timestamp, player1, player2,  winner,  nbOfSets, player1points,  player2points,
                        player1WonSets, player2WonSets,  player1WinningShots, player2WinningShots,  player1Aces,
                         player2Aces,  player1DirectFaults,  player2DirectFaults, player1WinningReturns, player2WinningReturns);
                MainActivity.requestQueue.add(postRequest);
            }
        });
    }
};
new Thread(runnable).start();
```

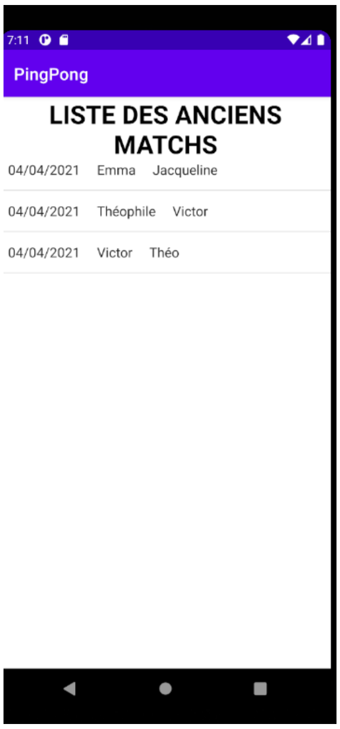
Data pushed to the BDD in a different thread

Here we can see some fragments of the app where the data from the database is displayed (in fragments ListOfGames and Statistics). For the example we took very simple data but all the calculation are made within AsyncTask :



```java
private class FillList extends AsyncTask<Void, Void, Void> {

    @Override
    protected Void doInBackground(Void... voids) {
        getListContent();
        return null;
    }
}
```

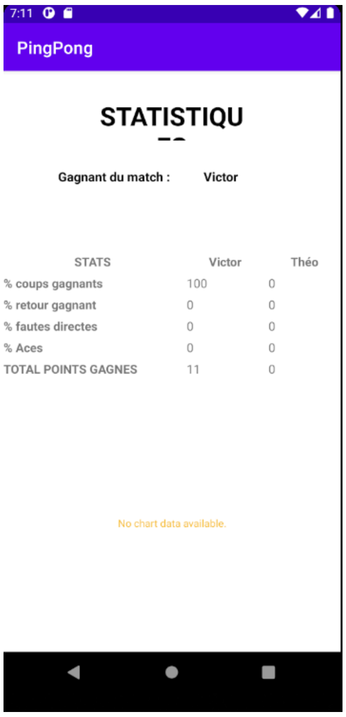


7:11

PingPong

LISTE DES ANCIENS MATCHS

04/04/2021	Emma	Jacqueline
04/04/2021	Théophile	Victor
04/04/2021	Victor	Théo



7:11

PingPong

STATISTIQU

Gagnant du match : Victor

STATS	Victor	Théo
% coups gagnants	100	0
% retour gagnant	0	0
% fautes directes	0	0
% Aces	0	0
TOTAL POINTS GAGNES	11	0

No chart data available.

AsyncTask - ListOfGames - Statistics