SCHOOL OF SCIENCE AND ENGINEERING

***Final Report***

# MYRecipeBook APP

Capstone Design

April 2017

Asbat El Khairi

Supervised by: Dr. Bouchaib Falah.

# ACKNOWLEDGMENT

At the very beginning, I would like to express my gratitude for being supervised by Dr. Bouchaib Falah and being guided by his instructions all along the semester. In fact, his continuous supervision and tracking allows me to work on a regular basis and so I could so far learn a lot about the fascinating world of Mobile Application Development and Software Engineering.

## Abstract

There are a number of applications in Android store for Recipes Search but none of them support interface for creating, searching, saving, and sharing recipes all at once. RecipeBook is an Android application with image based UI for searching, sharing, creating and saving recipes. This app provides flexibility to user to search variety of recipes from available recipes in the forum. In addition, the app provides the users with two distinct features such as Shopping List where the users can select and add ingredients that need to be purchased. The second feature is about the Meal Planner, where the users can schedule meals. Also, This is very handy application, which every user can search for recipes, save recipe as favorite, share recipe with friends on social media Facebook. This app is time saver providing recipes in few clicks. Through title search, RecipeBook app makes finding recipes easy. With recipes being added daily there will always be something new for user to crave. The project has been implemented using Android Studio, PHP, and MySQL.

## *List of Figures:*

- Figure1: Screen shot of codes support Screen Compatibility ……………………………8
- Figure2: Use Case Diagram for RecipeBook system …………………………………10
- Figure3: Wamp Server Installation ………………………………………………...14
- Figure4: The Difference between Eclipse and Android Studio………………………..15
- Figure5: System Architecture 1………………………………………………………16
- Figure6: System Architecture 2………………………………………………………17
- Figure7: JSON Architecture…………………………………………………………..18
- Figure8: Screen shot of codes of Facebook API……………………………………19
- Figure9: Entity Relationship Diagram……………………………………………...20
- Figure10: Screen Shot of all the tables in MyPHPAdmin in Wamp Server…………..21
- Figure11: Activity Diagram…………………………………………………………..26
- Figure12: List of Activities in Android Studio Project………………………………26
- Figure13: Class Diagram for server…………………………………………………..29
- Figure14: Screen shots of code showing Shared Preferences ………………………..30
- Figure15: Screen shots of the application (Facebook API)…………………………...31
- Figure16: Screen Shots of Register Screen …………………………………………..32
- Figure17: Screen Shot of MD5 code for encryption and decryption…………………33

## *Table of Contents*

**I. Introduction:** 6

**II. Feasibility Study:** 7

**III. User and system requirement document:** 8

*1. Project Description:* 8

A. External Interface Requirements: 8

B. Functional Requirements: 9

C. Use Case Diagram: 10

D. Non Functional Requirements: 11

*2. Technology Enablers:* 13

A. Server Side: 13

B. Client Side: 14

C. Other technologies: 15

**IV. System design and architecture** 16

*1. System architecture* 16

*2. Entity Relationship Diagram:* 20

*3. Database Tables* 21

*4. Android Application Design:* 25

*5. Class Diagram for the server:* 29

**V. Implementation Phase:** 30

**VI. Testing:** 34

**VII. STEEPLE analysis** 34

**VIII. Future Work:** 35

**IX. Conclusion:** 36

**X. References:** 37

# I. Introduction:

Nowadays, it is safe to admit that the world has attained the form of global village, where everything is accessible through technology. The advent of mobile phone has shaped the life of many people. It is hard to pass a day by without checking and rechecking your social network accounts. This can merely assert that mobile applications has already made their ways to our lives. According to Butler (2011), "In recent years, the emergence of smart phones has changed the definition of mobile phones. Phone is no longer just a communication tool, but also an essential part of the people's communication and daily life. Various applications added unlimited fun for people's lives. It is certain that the future of the network will be the mobile terminal."(p, 3).

Whether people love to cook or just love to eat, they have a collection of dishes and recipes they'd like to try. Maybe they have a bunch handed down from a loved one. In either case, they certainly need a better method to keep them organized for the long haul than a bunch of index cards in a file folder, which old and tedious. Therefore cooking with your phone is a lot tastier when you have the right recipes.

The objective of this capstone is to develop a "MyRecipeBook" mobile application to be used to elevate user kitchen skills and streamline grocery shopping, instead of using index cards in a file folder. In other words, MyRecipeBook will turn your phone into a pocket sous chef.

MyRecipeBook application is a very useful app for people who love to cook and try out new recipes. It provides user flexibility to search, share, save recipes from a database with an additional capability to maintain personal cookbook for creating new recipe, deleting recipe that are no longer required.

This application is a time saver providing recipes in few clicks. The user is given choice to create personal cookbook, where user can create recipe, view recipe and delete recipe. The interface is clean and simple. It makes use of Android image button capability to display options on home screen with image icons. The user can search recipes, view added favorite recipe list and access personal cookbook all from home screen.

## II. Feasibility Study:

The feasibility study, as its name indicates, aims at assessing the practicality of the proposed capstone project. As stated in the Initial Specifications Report, The objective of this capstone is to develop a “MyRecipeBook” mobile application to be used to elevate user’s kitchen skills and streamline grocery shopping, instead of using index cards in a file folder.

The app development will consist of five parts. The first part will be devoted to data gathering and software requirements specification. Consequently, I will have a look at different mobile apps which target the same goal. They are plenty of Recipe Organizer apps. Each one has some various features. The second part will be dedicated to the design phase, including the app and the database. Also, in this phase, the software tools to be used will be specified. For example, the IDE, the database Server, the modelling language for the design, and finally the software testing tools. The third part will be the implementation phase, here, the design will be converted to code in order to develop the targeted app. The fourth step will be devoted to testing the app. In this phase, two testing methods will be used, namely: Black Box testing and White box testing. The last phase will be the deployment phase.

## III. User and system requirement document:

### *1. Project Description:*

The goal of this step is to ensure that the requirements are consistent, precise and complete to ensure that we meet the final outcome expectations. There are two types of requirements: functional and non-functional requirements. The functional requirements are the ones that describes the functions of the app; whereas, the non-functional requirements are the ones that present the app constraints and properties.

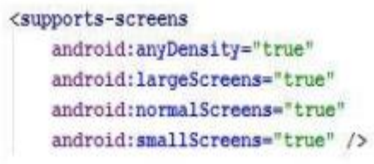
A. External Interface Requirements:

- ✓ **Hardware:**

The mobile app will be operating on Android Operating Apps.

- ✓ **Software:**

The mobile app will be compatible with the mobile and tabled (Android app) last versions.



```
<supports-screens
    android:anyDensity="true"
    android:largeScreens="true"
    android:normalScreens="true"
    android:smallScreens="true" />
```

*Figure 1: Screen shot of codes support Screen Compatibility*

- ✓ **Communication:**

The app will be dynamic. It will interact with the users instantly according to the inputs that it receives.

- ✓ **Users:**

Users will need to enter their personal details as name, address and phone to be recorded in the database and login to access to more elaborated functionalities as Create/View/Delete/Search a recipe. Also, if they want to keep a shopping list (the needed ingredients). These information will be kept in our database.

B. Functional Requirements:

- **Register:** A user needs to register in order to access the app. The user needs to enter his/her username, password, age, and country. Those information will be stored in the database and will serve as the profile info.
- **Login:** Any app operation requires the user to be registered. Login is the first step before performing an action.
- **View Profile:** After being successfully registered. The user can view his/her profile, where information of username, age, and country are displayed. Also, View Profile will enable you to view your created recipes as well as the recipes you have marked as favorite. In addition, View Profile will make you check your shopping list for the needed ingredients.
- **Edit Profile:** The registered user shall be able to edit his/her profile, such as editing the username and other information. Also, he/she shall be able to reset the password if needed.
- **View Recipe:** The registered user shall be able to see the recipes posted by other users. In other words, the user shall be able to access the Recipe Forum, where all the user's recipes are posted and ordered by date of creation.
- **Create Recipe:** Any registered user shall able to create his/her own recipe. The creation of recipe include entering the title, description, and other information that concerns the recipe, such as the steps and the ingredients.
- **Search for a Recipe:** Any registered user shall able to search for a recipe using the title as a criterion. The search functionality will provide the users with a shortcut to find their targeted recipes if posted before by the users.
- **Add Recipe as Favorite:** Once the user access the recipe information, he/she shall be able to mark a recipe as favorite.
- **View Ingredients:** The registered user of the app shall be able to view ingredients of a certain recipe. This option will allow him/her to mark those ingredients as needed, and ultimately find them in the shopping list.
- **View My Recipes:** The registered user shall be able to see all the recipes he/she already created.
- **Delete Recipe:** The registered user shall be able to delete his/her recipes, he/she already posted.
- **View My Shopping List:** The registered user shall be able to manage his/her shopping list based on the needed ingredients.

- **Delete Ingredient:** The user shall be able to delete the ingredients from the shopping list, if the shopping was done or for other reasons.
- **View My Favorite Recipes:** The user shall be able to access the recipes that he/she marked as favorite.
- **Plan a Meal:** The user shall be able to plan a meal/create events.
- **View My Planned Events:** The user shall be able to see the list of the events, he/she already created.
- **Follow/ Unfollow:** The user shall be able to follow a user. The follow feature will allow users to keep posted about the posted recipes by their following users. It is like another recipe forum, but this time, is specifically designed to list just the recipes posted by the users followed by the user. Further, the user shall be able to "unfollow" a certain user if necessary, consequently, that user will be dropped from the list of "following".

C. Use Case Diagram:

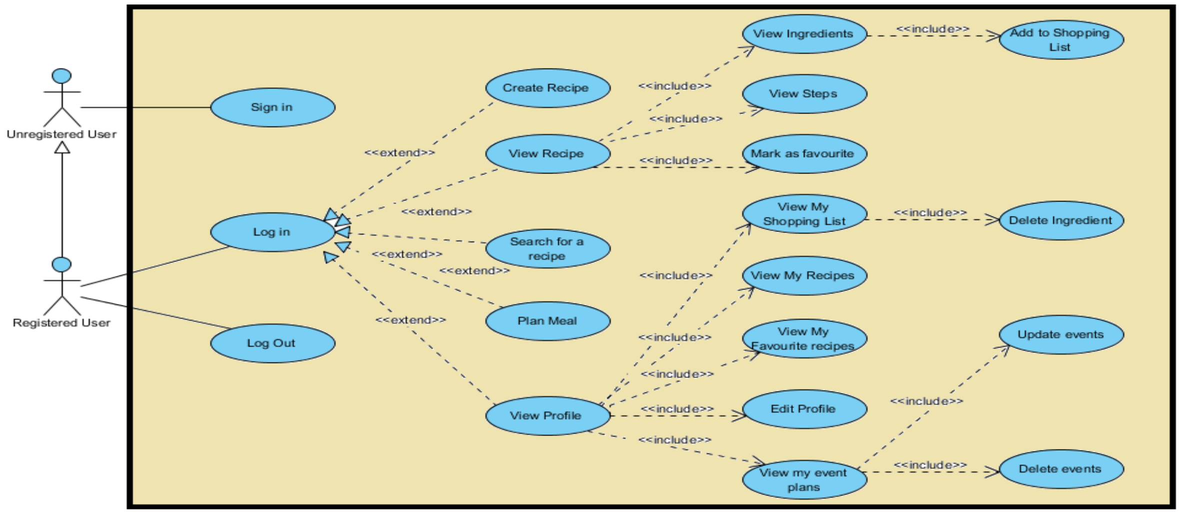


*Figure2: Use Case Diagram for MyRecipeBook system*

D. Non Functional Requirements:

✓ **Product Requirements**

▪ **Usability Requirements:**

-The application shall be easy to use and intuitive.

- The application shall have a user-friendly interface.

- GUI shall be simple and clear.

▪ **Performance Requirements:**

- The application shall be fast and robust when loading.
- The program shall not allow more than 10 min/year of failure.

▪ **Space Requirements:**

- The application shall have enough memory space in order to store high number of data.

▪ **Reliability Requirements:**

- The application shall not produce an incorrect output.

▪ **Portability Requirements:**

- The software shall work in all different platforms.

✓ **Organizational Requirements:**

▪ **Delivery Requirements:**

- The application shall be delivered no later than 17th, April.

▪ **Implementation Requirements:**

- The application shall be implemented using Android Studio 2.2.3, PHP, and MySQL.

▪ **Standards Requirements:**

- The application shall conform to ISO standards.

✓ **External Requirements:**

▪ **Inter-operability Requirements:**

- The application shall allow access to the different department of the application without altering its efficiency and consistency.

▪ **Ethical Requirements:**

- The application shall be license free.

- **Privacy Requirements:**

- Personal information of the registered user shall only be accessed by the administrator.
- The guest user shall not be able to order a medicine.

- **Safety Requirements:**

- The application shall be protected from any external danger or attacks.

## 2. *Technology Enablers:*

The choice of technology enablers that will be used for the development of the application is essential for its success. The technology enablers should provide a suitable way to fulfill the requirements stated before. Principles of enterprise class applications should be kept in mind during the choice of these technologies. The two main ones are that there is no best technology but instead suitable ones and that we shouldn't reinvent the wheel which means that we should take advantage on what was already implemented and offered to the community.

### A. Server Side:

The android application is no longer static, it is dynamic. As the information content grows, so does the need to make mobile applications more dynamic. PHP and SQL are two ways to make your application dynamic.

- **PHP**: is a robust, server-side, open source scripting language that is extremely flexible.
- **MySQL**: is the standard query language for interacting with databases. MySQL is an open source.

In order to test the php files locally and android "localhost" connection, WAMP server is needed.

- **WAMP:** (Windows, Apache, MySQL, PHP) is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. It also comes with PHPMyAdmin and SQLiteManager to easily manage your databases.

It is highly essential that we need to test our app for correct data before moving the database, PHP files and Android app to production. One way we can test for the same is, going live on the internet by hosting your files with hosting provider and it is bad idea to test your files live on the internet. Therefore, Rather than hosting your database and PHP files on the internet, we can test our app for PHP MySQL using localhost WAMP server. According to Ashwin (2012), "The main part for the server side, we used a free of cost, open-source, and easily adaptable stack of open source software: Apache, which will be present in any of the IDE's which are used for the development of the applications, MySQL and PHP, abbreviated as AMP technologies. Our solution was tested both on Linux (LAMP) and

Windows (WAMP) versions of AMP stack. Mostly used IDE's are Eclipse and NetBeans for Android Applications MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases.

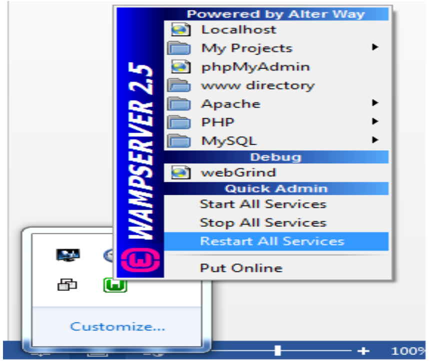


*Figure 3: Wamp Server Installation*

Successfully Running WAMP server on the computer indicates a green icon on system tray. The WAMP server enables the developer to put the php code in the www directory and then create and manage the databases using PHPmyAdmin.

B. Client Side:

- **ANDROID STUDIO**

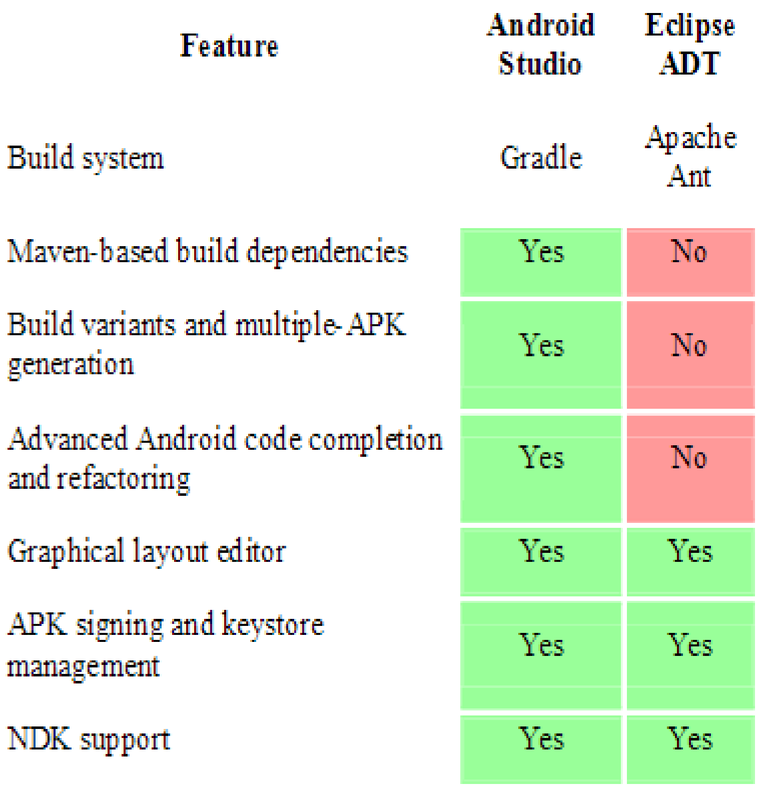
Android Studio is Android's official IDE. It is purpose built for Android to accelerate your development and help you build the highest-quality apps for every Android device. It offer tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools. I have selected Android Studio for the following reasons:

- **Instant Run**: When you click Run or Debug, Android Studio's Instant Run feature pushes code and resource changes to your running app. It intelligently understands the changes and often delivers them without restarting your app or rebuilding your APK, so you can see the effects immediately.

- **Intelligent code editor:** The code editor helps you write better code, work faster, and be more productive by offering advanced code completion, refactoring, and code analysis.

- **Optimized for all Android devices:** Android Studio provides a unified environment where you can build apps for Android phones, tablets, Android Wear, Android TV, and Android Auto. Structured code modules allow you to divide your project into units of functionality that you can independently build, test, and debug.

Here is an image that throws the lights on the main advantages of Android Studio over Eclipse:



| Feature | Android Studio | Eclipse ADT |
|---|---|---|
| Build system | Gradle | Apache Ant |
| Maven-based build dependencies | Yes | No |
| Build variants and multiple-APK generation | Yes | No |
| Advanced Android code completion and refactoring | Yes | No |
| Graphical layout editor | Yes | Yes |
| APK signing and keystore management | Yes | Yes |
| NDK support | Yes | Yes |

*Figure 4: Difference between Android Studio and Eclipse*

C. Other technologies:

- **GENY MOTION:**

Genymotion is a fast third-party emulator that can be used instead of the default Android emulator which may turn to be slow if the laptop has poor graphics. In some cases it's as good as or better than developing on actual devices.

- **Notepad++**

Notepad++ is a free source code editor and Notepad replacement that supports several languages.

# IV. System design and architecture

## 1. System architecture

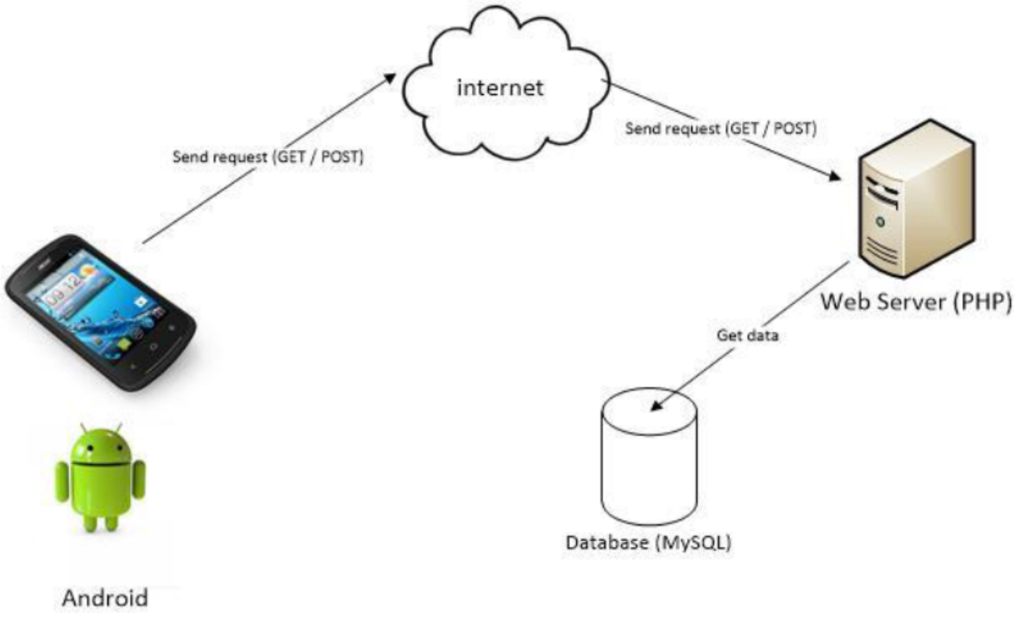


*Figure 5: System Architecture 1*

A system architecture describes the organization of the system in terms of structure and behaviors by representing the different components and the relationship between them. The architecture of this system is composed of a mobile client that allows the application's users to make use of the different functionalities of the system. Web service (PHP) which through php scripts, interacts with the DBMS that manage the raw data of the system. In other words, the user of the app will get the input using an Android Client and send it to the server via HTTP. A PHP script on the server then invokes the server side application to do its work. After that, the server will send the result back to the android device for display. So, PHP and MySQL are responsible for retrieving the information from the server. Also, the architecture used in this app design shows that the HTTP protocol is used from the android system. So, in a nutshell, if we talk about the client-server architecture, client is the android device and in server-side there is a combination of PHP Script and MySQL.

The image above shows the architecture in details:

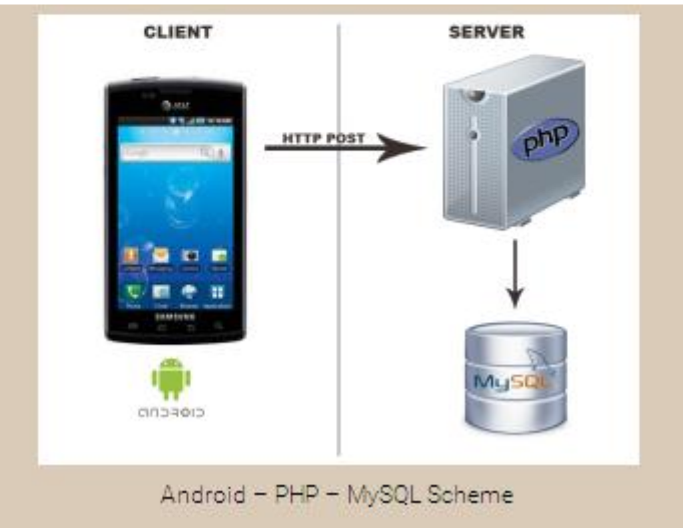
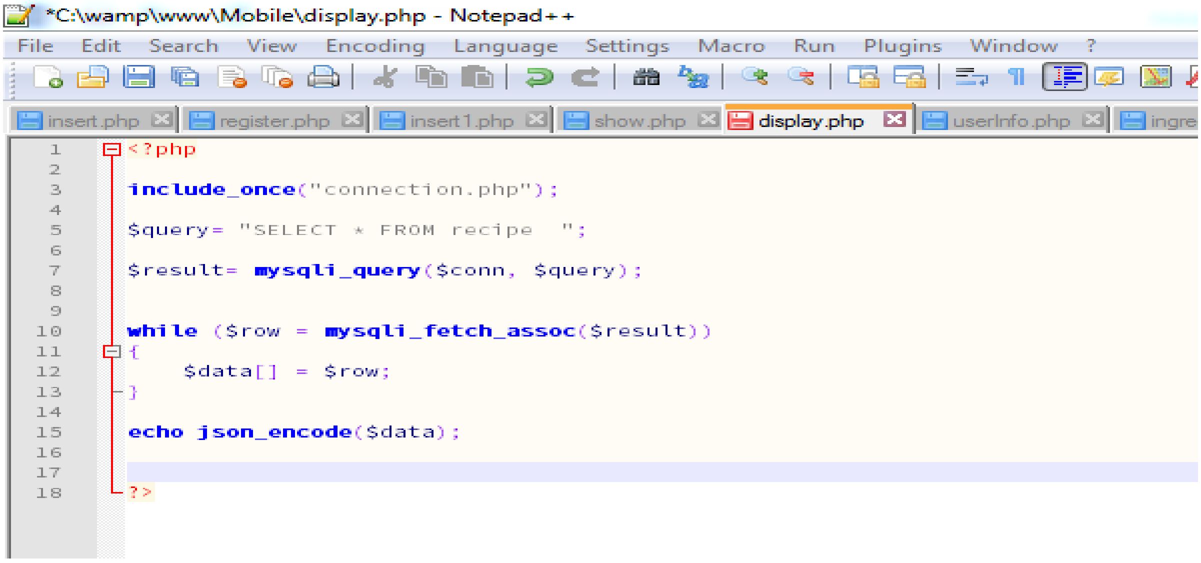


*Figure 6: System Architecture 2*

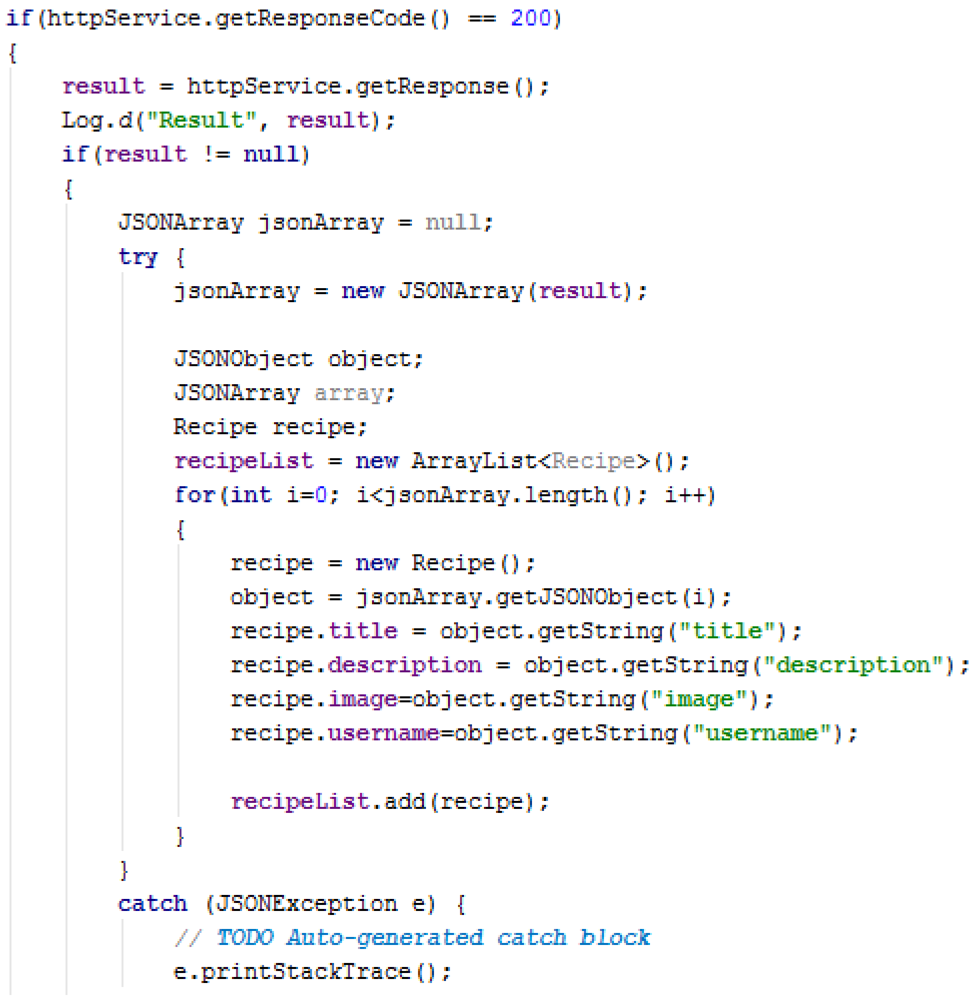
Also, I made use of JSON (JavaScript Object Notation), which is a lightweight text-based open standard designed for human-readable data interchange. Concerning, how JSON has been used in my application:

- When android application will execute, it will connect android device to PHP Script.
- PHP script will fetch data from the database, it will encode it to JSON Format and send it to the device.
- Android application will get these encoded data, it will parse the data and display it on android device.

The images are taken from my PHP scripts and Android Studio activity:



```php
<?php

include_once("connection.php");

$query= "SELECT * FROM recipe  ";

$result= mysqli_query($conn, $query);


while ($row = mysqli_fetch_assoc($result))
{
    $data[] = $row;
}

echo json_encode($data);


?>
```

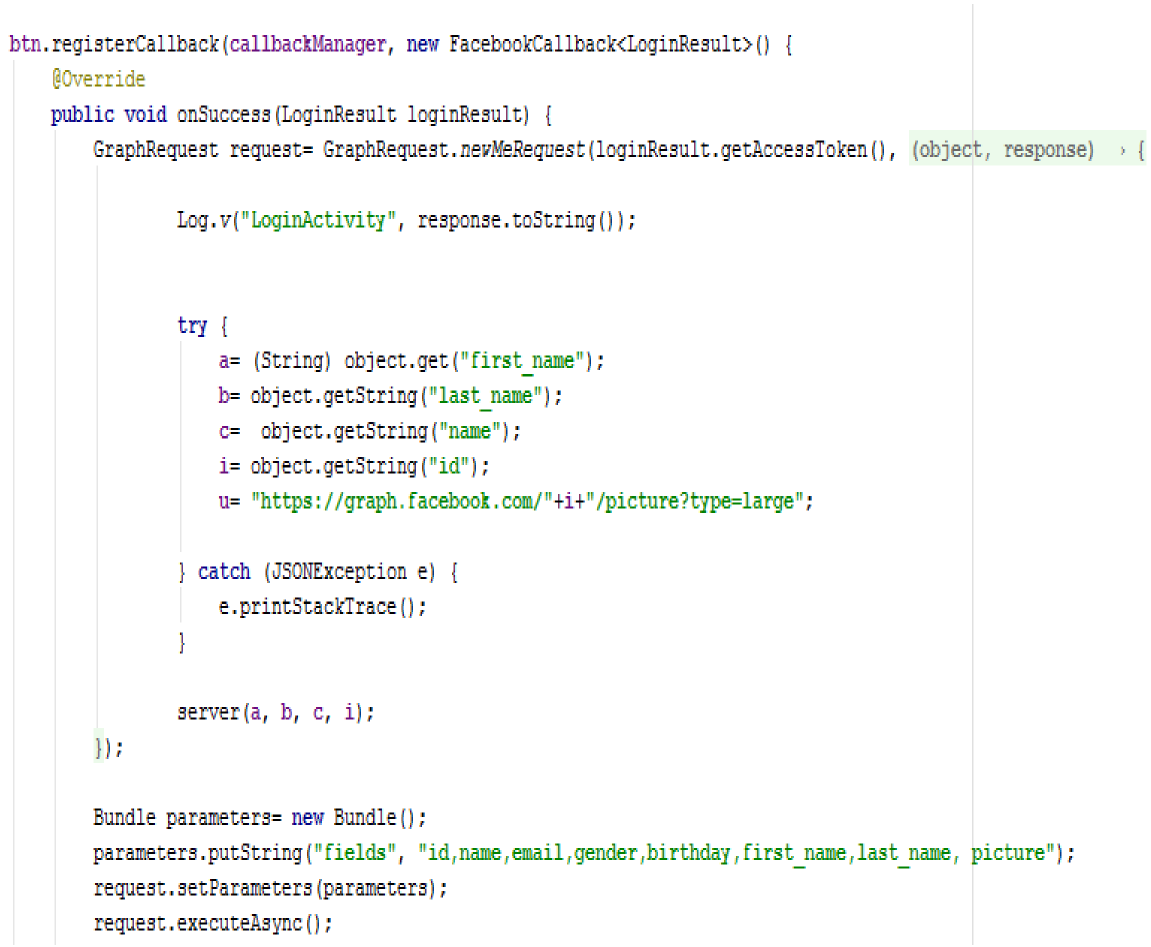


```java
if(httpService.getResponseCode() == 200)
{
    result = httpService.getResponse();
    Log.d("Result", result);
    if(result != null)
    {
        JSONArray jsonArray = null;
        try {
            jsonArray = new JSONArray(result);

            JSONObject object;
            JSONArray array;
            Recipe recipe;
            recipeList = new ArrayList<Recipe>();
            for(int i=0; i<jsonArray.length(); i++)
            {
                recipe = new Recipe();
                object = jsonArray.getJSONObject(i);
                recipe.title = object.getString("title");
                recipe.description = object.getString("description");
                recipe.image=object.getString("image");
                recipe.username=object.getString("username");

                recipeList.add(recipe);
            }
        }
        catch (JSONException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
```

*Figure 7: JSON Architecture*

Also, I have integrated my android app with Facebook through the Facebook SDK. This integration enables Facebook login, which authenticate people with their Facebook credentials, and Share and Send Dialogs, which enable sharing content from your app to Facebook. Therefore, it is safe to admit that the server side exposes a RESTFUL API that will be consumed by any REST Client. Facebook Graph API is used to query the essential data from Facebook for a given user and share his/her recipe on the social media. Personal information like first and last name, age, hometown, profile picture, and email address for example can be retrieved using this API. This information will be used to populate the application's database. Also, upon the creation of a recipe, this API may be used to share the recipes on Facebook in order to attract more users.

The figure below show the integration of my app with Facebook through Facebook SDK.



```
btn.registerCallback(callbackManager, new FacebookCallback<LoginResult>() {
    @Override
    public void onSuccess(LoginResult loginResult) {
        GraphRequest request= GraphRequest.newMeRequest(loginResult.getAccessToken(), (object, response) › {

                Log.v("LoginActivity", response.toString());


                try {
                    a= (String) object.get("first_name");
                    b= object.getString("last_name");
                    c=  object.getString("name");
                    i= object.getString("id");
                    u= "https://graph.facebook.com/"+i+"/picture?type=large";

                } catch (JSONException e) {
                    e.printStackTrace();
                }

                server(a, b, c, i);
        });

        Bundle parameters= new Bundle();
        parameters.putString("fields", "id,name,email,gender,birthday,first_name,last_name, picture");
        request.setParameters(parameters);
        request.executeAsync();
```

*Figure 8: Screen Shot of Facebook API*

## 2. *Entity Relationship Diagram:*

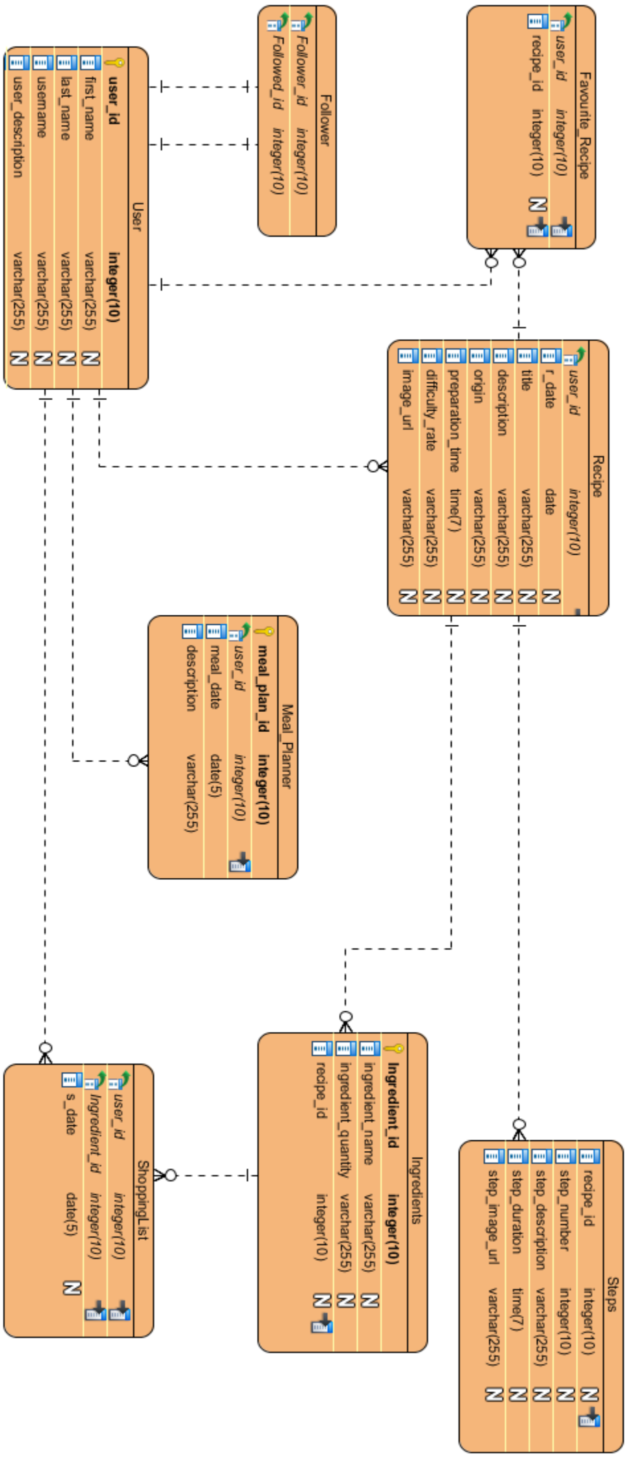
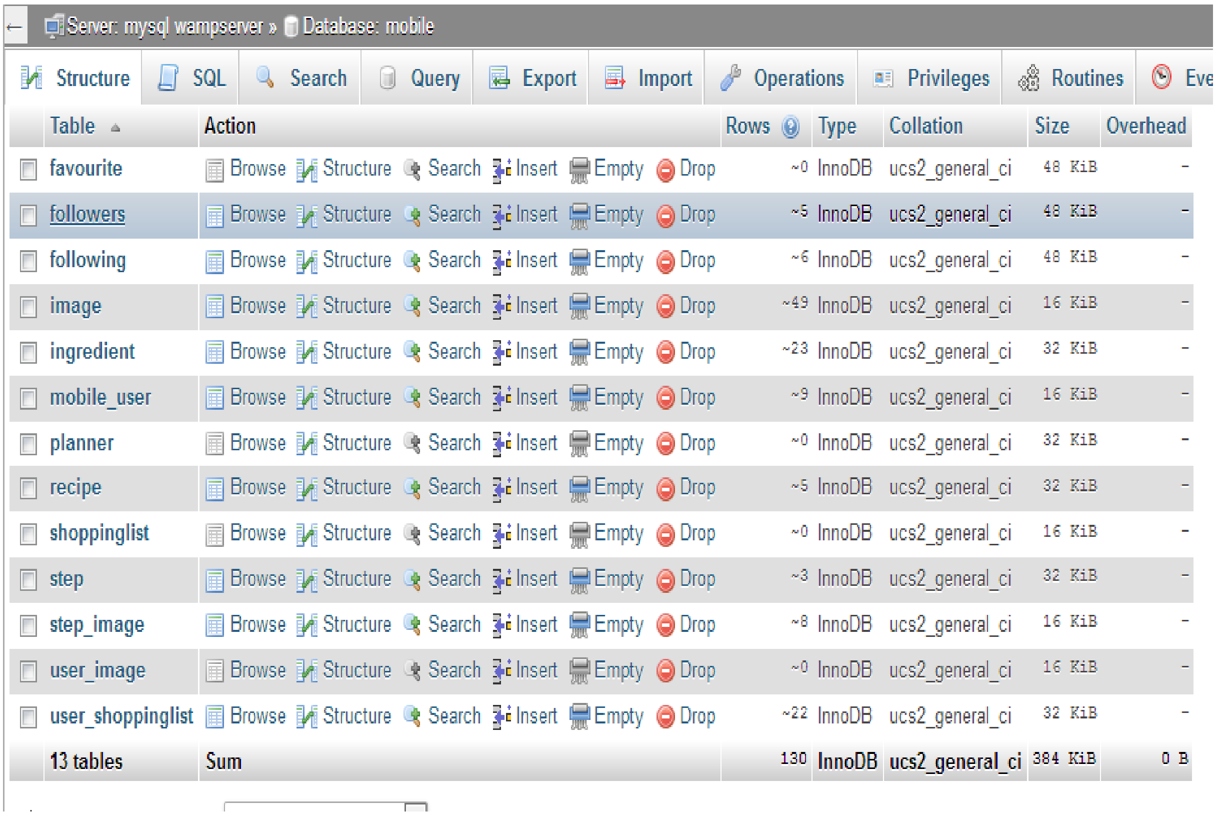


*Figure 9: MyRecipeBook Entity Relationship Diagram*



| Table | Action | Rows | Type | Collation | Size | Overhead |
|---|---|---|---|---|---|---|
| □ favourite | Browse Structure Search Insert Empty Drop | ~0 | InnoDB | ucs2_general_ci | 48 KiB | - |
| □ followers | Browse Structure Search Insert Empty Drop | ~5 | InnoDB | ucs2_general_ci | 48 KiB | - |
| □ following | Browse Structure Search Insert Empty Drop | ~6 | InnoDB | ucs2_general_ci | 48 KiB | - |
| □ image | Browse Structure Search Insert Empty Drop | ~49 | InnoDB | ucs2_general_ci | 16 KiB | - |
| □ ingredient | Browse Structure Search Insert Empty Drop | ~23 | InnoDB | ucs2_general_ci | 32 KiB | - |
| □ mobile_user | Browse Structure Search Insert Empty Drop | ~9 | InnoDB | ucs2_general_ci | 16 KiB | - |
| □ planner | Browse Structure Search Insert Empty Drop | ~0 | InnoDB | ucs2_general_ci | 32 KiB | - |
| □ recipe | Browse Structure Search Insert Empty Drop | ~5 | InnoDB | ucs2_general_ci | 32 KiB | - |
| □ shoppinglist | Browse Structure Search Insert Empty Drop | ~0 | InnoDB | ucs2_general_ci | 16 KiB | - |
| □ step | Browse Structure Search Insert Empty Drop | ~3 | InnoDB | ucs2_general_ci | 32 KiB | - |
| □ step_image | Browse Structure Search Insert Empty Drop | ~8 | InnoDB | ucs2_general_ci | 16 KiB | - |
| □ user_image | Browse Structure Search Insert Empty Drop | ~0 | InnoDB | ucs2_general_ci | 16 KiB | - |
| □ user_shoppinglist | Browse Structure Search Insert Empty Drop | ~22 | InnoDB | ucs2_general_ci | 32 KiB | - |
| 13 tables | Sum | 130 | InnoDB | ucs2_general_ci | 384 KiB | 0 B |

*Figure 10: MyRecipeBook database tables*

In order to persist data on the database, the entities should be defined. The following entities will be represented at the level of the server as models and persisted on the database as collections. The following table gives the detail about each entity including its name, its attributes and a description on what it is used for.

### 3. Database Tables

| Entity Name | Attributes | Attribute's Description | General Entity Description |
|---|---|---|---|
| Recipe | recipe_id | The ID of the recipe, which is unique for each recipe. (Primary Key) | This entity represents a single recipe. It stores different |
| | Title | The title of the recipe | |

| | | | |
|---|---|---|---|
| | description | Short description about the recipe, including its advantages, and other interesting information. | information about the single recipe. |
| | Origin | The origin of the recipe. | |
| | preparation_time | How much time to prepare/cook the recipe (field in minutes) | |
| | difficulty_rate | Either difficult, average, or easy. | |
| | image_url | The url of the image of the recipe | |
| | Category | The type of the recipe, whether it fallsfor Salad category or Meal, etc. | |
| | user_id | This is a foreign key form user table, which indicates who is the owner or creator of the recipe. | |
| Ingredients | ingredient_id | The ID of the ingredient, which shall be unique (Primary Key) | This entity represents a single ingredient. It stores different information about the single ingredient. |
| | ingredient_name | The name of the ingredient | |
| | ingredient_quantity | The quantity of the ingredient. | |
| | recipe_id | The recipe id, which is a foreign key from recipe table. | |

| | | | |
|---|---|---|---|
| Steps | step_id | The ID of the step, which shall be unique (Primary key). | This entity represents a single step. It stores different information about the single step. |
| | Step_number | The number of the step | |
| | Step_duration | The duration of the step, which is in minutes. | |
| | Step_description | The description of the step. | |
| | Step_image_url | Each step may have an image. This field is designed for the step image url | |
| | recipe_id | The recipe id, which is a foreign key from recipe table. | |
| Shopping_List | User_id | The user id is a foreign key from user_id. | Shopping List is a bridge table between Ingredient table and User table. One user may have selected many ingredients from ingredient_list to shopping list. Also, one ingredient may be selected by multiple users, which indicates that this table shall be a bridge table. |
| | Ingredient_id | It is a foreign key from Ingredient table | |
| | S_date | This is the date field. Once the ingredient is added to the shopping list, this field is designed to catch the date (now()) to display it on the shopping list. | |

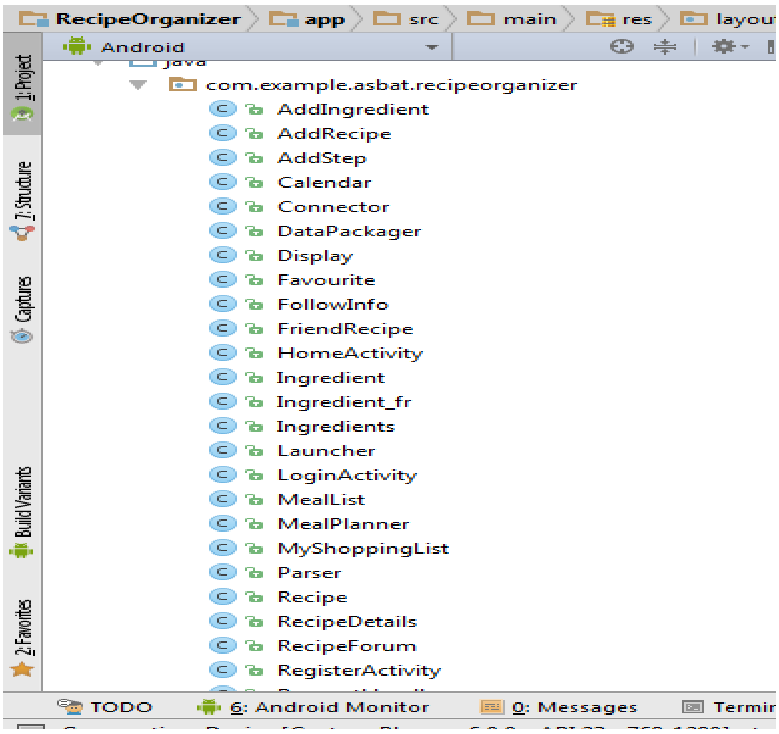
| | | | |
|---|---|---|---|
| Meal_Planner | Meal_plan_id | The id of the meal planned (primary key) | This table is designed for the meal planned by the user. |
| | meal_date | The date of the meal. | |
| | meal_description | A short description of the meal. | |
| | user_id | The id of the user who plans this meal/set the event. | |
| Favourite_Recipe | recipe_id | The id of the recipe, which is a foreign key from the Recipe table. | This table is a bridge table between Recipe and User tables. A user can have many favourite recipes. Also, a single recipe can be seen as favourite from many users. |
| | user_id | The id of the user which is a foreign key as well. | |
| Follower | Followed_id | Indexed field. This is the username, which is a foreign key from the User table. | This table has one to one relationship with the User table. A user can be a follower and at the same time following. |
| | Follower_id | Indexed field. This is the username, which is a foreign key from the User table. | |
| User | user_id | The ID of the user, which shall be unique. (Primary Key) | This entity represents a single user. It |

| | | | |
|---|---|---|---|
| | first_name | The first name of the user | stores different information about the single user. |
| | last_name | The last name of the user | |
| | Username | The username of the user | |
| | Description | Short description of the user, including his/her hobbies and personal status. | |
| | user_image_url | Each user has an image, this field is designed for getting the url of the image. | |

*Table 1*

### *4. Android Application Design:*

The Android’s user interface is built around Activities which are single focused thing that a user can do. They are directly related to the functional requirement defined for the application. For example, the functional requirement stating that a use should be able to login in the application, implies that there should be an activity that enables the user to login. Following are the derived, from the user requirements, activities that should be implemented to fulfill the functional requirements.



java
com.example.asbat.recipeorganizer
AddIngredient
AddRecipe
AddStep
Calendar
Connector
DataPackager
Display
Favourite
FollowInfo
FriendRecipe
HomeActivity
Ingredient
Ingredient_fr
Ingredients
Launcher
LoginActivity
MealList
MealPlanner
MyShoppingList
Parser
Recipe
RecipeDetails
RecipeForum
RegisterActivity

*Figure 11: MyRecipeBook Android Activities*

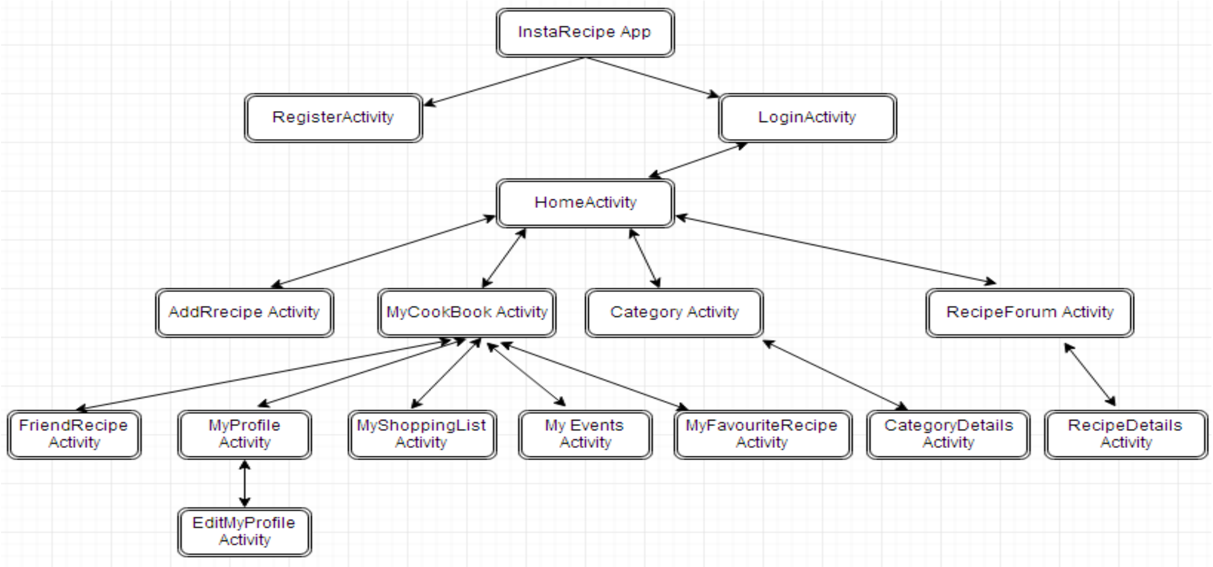


*Figure 12: Activity Diagram*

| Activity Name | Activity Description |
|---|---|
| InstaRecipe<br>Launcher Activity | It is a launcher that will last 4 seconds prior to the LoginActivity. |
| LoginActivity | This activity is the first activity that is opened in when the application is launched. It contains the buttons to login uses Facebook. When the login is successful, it opens the HomeActivity. |
| RegisterActivity | If the user selects the traditional way to register by filling lastname, firstname, username fields etc. The register activity is designed for that type of registration. |
| HomeActivity | The home activity is the core activity, where everything starts from. At this activity, there are many buttons that will lead you to the different activity this project has. |
| AddRecipe | This activity is designed to add the recipe, including ingredients and the steps. |
| MyCookBook | This activity will show the user the multiple option he/she want to enter, such as myfavouriteRecipes, myFriend's recipes. |
| Category | This activity is designed to select which category the user want to have access to. |
| RecipeForum | RecipeForum is the list of posted recipes by all the InstaRecipe users. |
| FriendRecipe | Each user has the possibility to follow a particular user if he gets bewitched by his/her posts. This activity is mainly designed for that. |
| MyProfile | This activity is designed to display the profile of the user, including all the information stored in the database. |
| MyShoppingList | This activity is designed to display the ingredients the user has added somewhen before. |
| MyEvents | This activity is designed to list all the events created by himself/herself. |

| | |
|---|---|
| MyFavouriteRecipe | This activity is designed to display all the recipes the user has marked as favourite. |
| CategoryDetails | This activity is designed to display the list of recipes that falls under the same category, the user has already selected. |
| RecipeDetails | This activity is designed to display the recipe selected in details, including the ingredients and the recipes. |
| EditMyProfile | This activity is designed to allow the user edit their profile by updating their profile picture or their status and name. |
| Follow | This activity is mainly designed to permit the user to follow a particular user that he/she posts interesting recipes. |

*Table 2*

## 5. Class Diagram for the server:

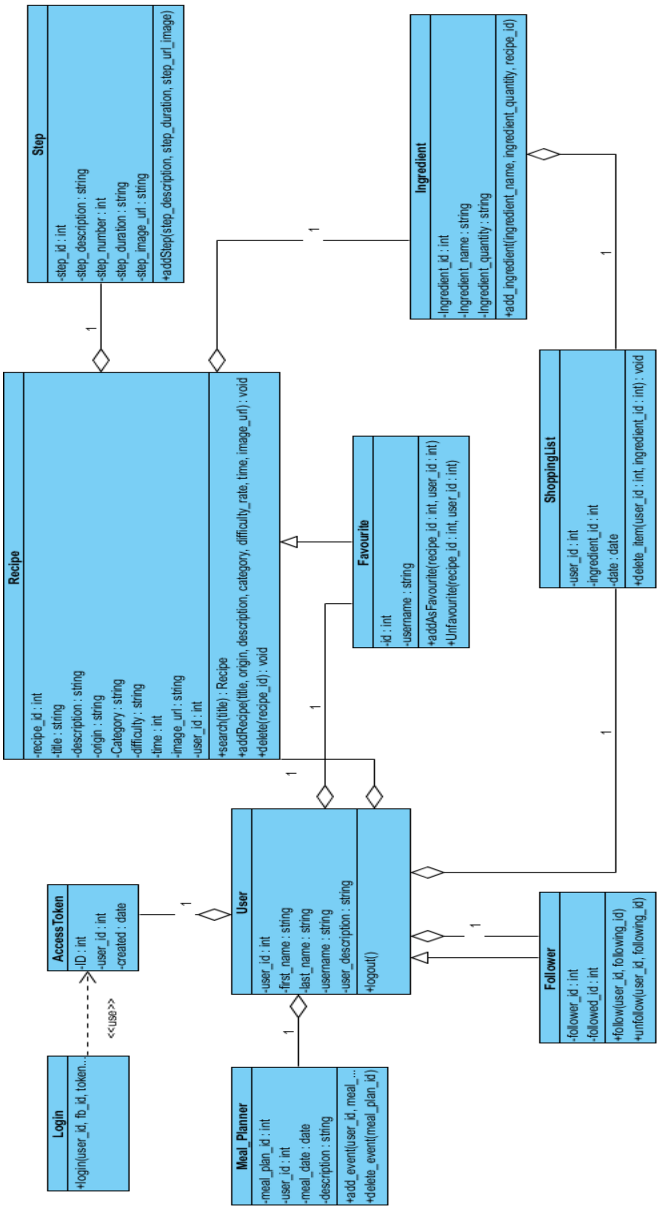
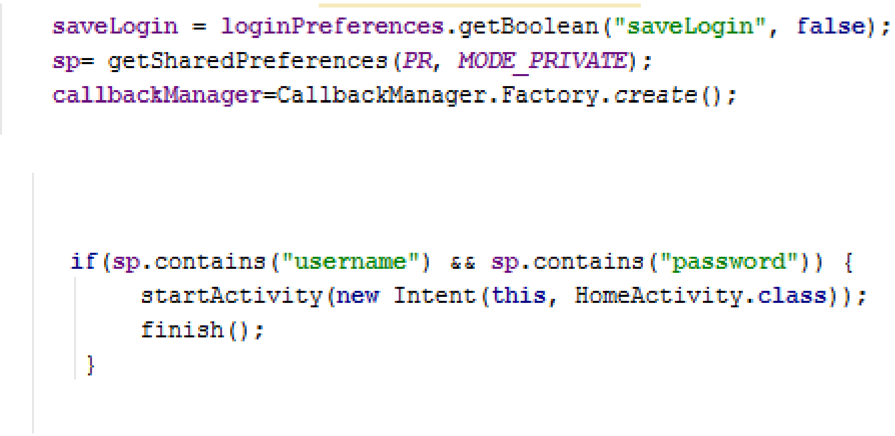


Figure 13: Class Diagram for the server

## V. Implementation Phase:

The implementation took roughly three months, and involved coding every single day to finally arrive at the final product which will be presented during the capstone defense. The implementation was about designing activities, which are similar to pages in web applications, creating the database inside my project and linking the graphical user interface with the functionalities and the database. Every time I implemented a functionality, I tested it directly on the GenyMotion emulator and my android phone so as to get a real life representation of what the application would look like on our clients' mobile phones. I also managed to make the mobile application compatible with phones supporting different versions. Further, the mobile application has many features:

- **User Session Management:** Session are useful when you want to store user data globally throughout the application. This can be done in two ways: one is storing them in a global variables and second is storing the data in shared preferences. The problem with storing data in global variable is the fact that the data will be lost once the user closes the application, but storing data in shared preferences will be persistent even though user closes the application. Therefore, I have used Shared Preferences.

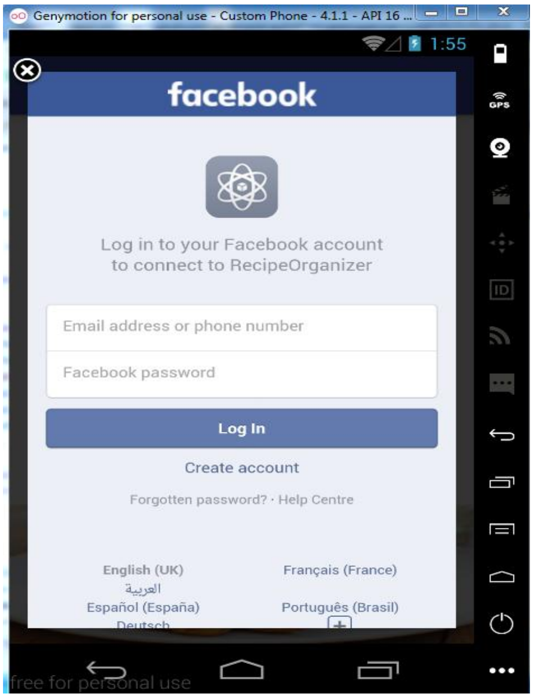


```
saveLogin = loginPreferences.getBoolean("saveLogin", false);
sp= getSharedPreferences(PR, MODE_PRIVATE);
callbackManager=CallbackManager.Factory.create();
```

```
if(sp.contains("username") && sp.contains("password")) {
    startActivity(new Intent(this, HomeActivity.class));
    finish();
}
```

*Figure 14:Session Management using Shared Preferences*

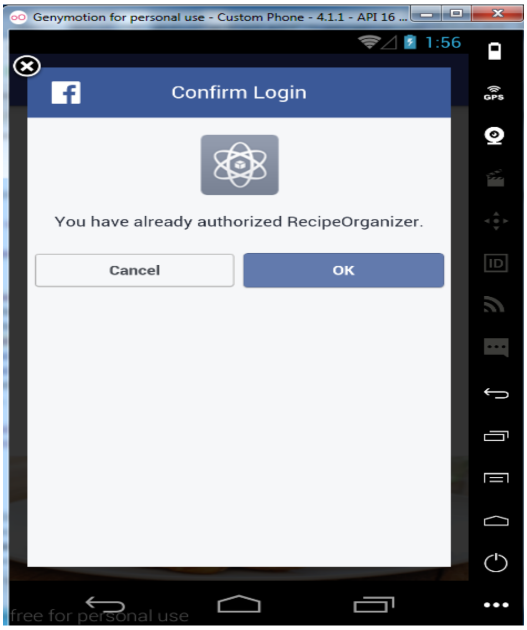
- **Single Sign-on:** Facebook has introduced single sign-on for mobile apps through Facebook SDK. In order to implement this feature, the application shall be integrated with Facebook SDK in order to take advantage from the login. I have integrated my app with the Facebook SDK.


Genymotion for personal use - Custom Phone - 4.1.1 - API 16 ...

Log in to your Facebook account
to connect to RecipeOrganizer

Email address or phone number

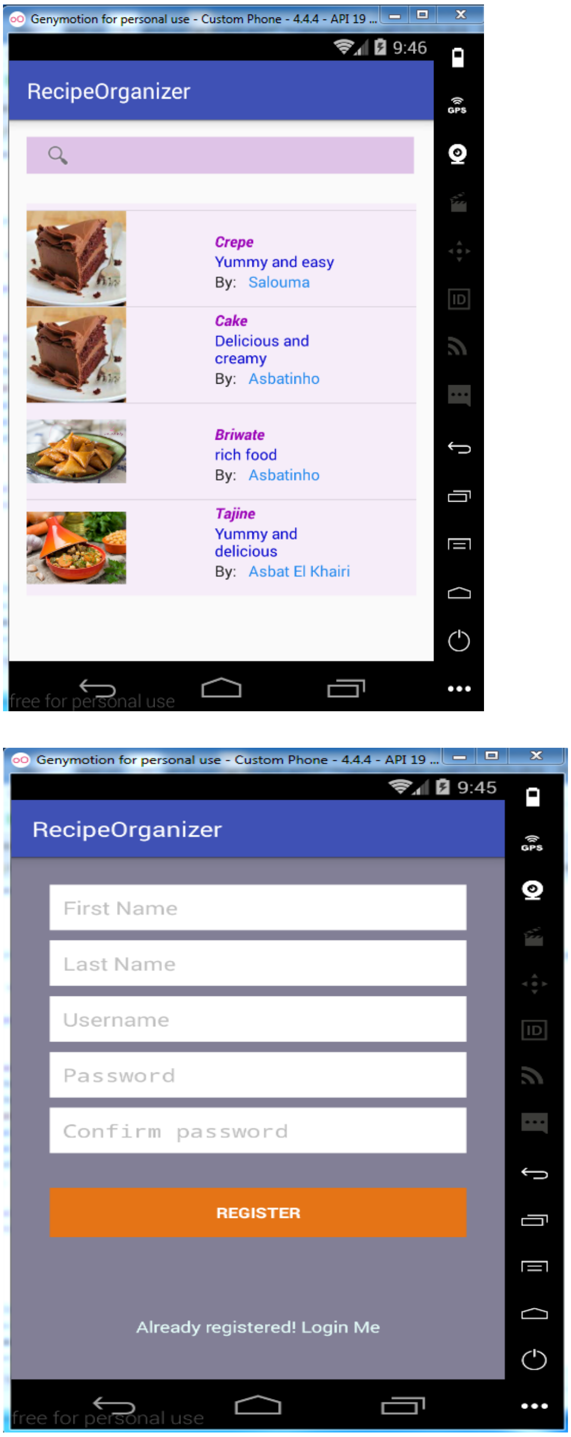
Facebook password


Genymotion for personal use - Custom Phone - 4.1.1 - API 16 ...

You have already authorized RecipeOrganizer.

*Figure 15: Facebook Integration*

- **Friendly and flexible GUI:** My app has a friendly and flexible Graphical User Interface. I have used Picasso as an image loader. Also, I made sure that every single layout is scrollable. I have used “PhotoShop” as a software to design the buttons, layouts and “textviews” backgrounds.



Genymotion for personal use - Custom Phone - 4.4.4 - API 19 ...

9:46

RecipeOrganizer

*Crepe*
Yummy and easy
By: Salouma

*Cake*
Delicious and creamy
By: Asbatinho

*Briwate*
rich food
By: Asbatinho

*Tajine*
Yummy and delicious
By: Asbat El Khairi

free for personal use

Genymotion for personal use - Custom Phone - 4.4.4 - API 19 ...

9:45

RecipeOrganizer

First Name

Last Name

Username

Password

Confirm password

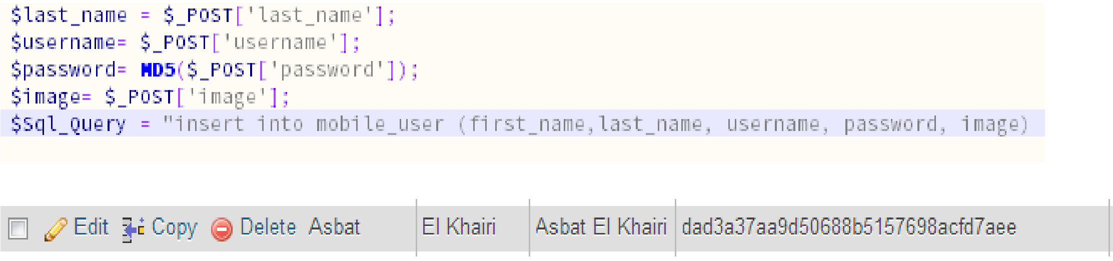
REGISTER

Already registered! Login Me

free for personal use

Figure 16: Screen shots from MyRecipeBook

- **Security:** I have used MD5 algorithm, which is widely used hash function that produces a 128-bit hash value. It has been used to hash the password, so it cannot be seen in the database.



```
$last_name = $_POST['last_name'];
$username= $_POST['username'];
$password= MD5($_POST['password']);
$image= $_POST['image'];
$sql_Query = "insert into mobile_user (first_name,last_name, username, password, image)
```

|  |  |  |  |
|---|---|---|---|
| □ Edit Copy Delete Asbat | El Khairi | Asbat El Khairi | dad3a37aa9d50688b5157698acfd7aee |

*Figure 17: Screen Shot of MD5 code for encryption and decryption*

## VI. Testing:

I have used two types of testing:

- **Black-Box testing**: is a functional testing that targets only the interface to ensure that they work as expected. I make sure that every AlertDialog is shown if an imput is missing using Toast in android studio.

- **White-Box testing:** is a clear-box one where the tester has access to the code. Since I have been the developer of my app, I have been practicing this type of testing on a daily basis.

## VII. STEEPLE analysis

- *Social*
  - Increase social interaction and solidarity
  - Meet new people and make new friends through the Recipe Forum
- Technology
  - Smartphone penetration is increasing day after day
  - The application is accessible from anywhere using a smartphone
  - Real time communication between actors
- Environmental
  - Increase of awareness regarding healthier foods and recipes.
- Economical
  - RecipeOrganizer will enable users to try good and less costly recipes
- Political
  - RecipeOrganizer may help the government in sharing good and useful information through ads.
- Legal
  - Insurances of users.
- Ethical
  - Client confidentiality should be kept: all information related to users should only be communicated to their respective users.

## VIII. Future Work:

As a future work, I am planning to persist in developing more mobile apps and entering deeply the world of Android development. RecipeOrganizer has helped me to gain a lot of development skills and enrich my background, as I spent the previous 4 months searching for every tiny detail that concerns the development of android application. Thankfully, I have built a good knowledge. Therefore, any upcoming project of mobile application development will undoubtedly be within my reach. In addition, as to the future of RecipeOrganizer, I will deploy it in Google PlayStore and update the app from time to time if necessary. I have already started the deployment phase by buying a host for my database in HostGator server. I will export the my database soon, and then buying an account in Google PlayStore to publish the app. Also, I will monetize my application using Admob.

## IX. Conclusion:

The RecipeOrganizer application meets with the enterprise class application principles. It is designed to be performing, scalable, extensible, and highly available. It also ensures the privacy of the users' data and secures its access. Given that it may be improved in many ways, the application is also easily maintainable.

This documents summarizes the work that has been done since the beginning of this semester. Indeed, it starts by giving an overview about the project specification and requirements. The document also states the methodology followed and which consists of 5 main parts: The first part will be devoted to data gathering and software requirements specification. Consequently, I will have a look at different mobile apps which target the same goal. They are plenty of Recipe Organizer apps. Each one has some various features. The second part will be dedicated to the design phase, including the app and the database. Also, in this phase, the software tools to be used will be specified. For example, the IDE, the database Server, the modelling language for the design, and finally the software testing tools. The third part will be the implementation phase, here, the design will be converted to code in order to develop the targeted app. The fourth step will be devoted to testing the app. In this phase, two testing methods will be used, namely: Black Box testing and White box testing. The last phase will be the deployment phase.

# X. References:

- "STEEPLE Definition | Marketing Dictionary | MBA Skool-Study.Learn.Share.", MBA Skool-Study.Learn.Share., 2016. [Online]. Available: http://www.mbaskool.com/businessconcepts/marketing-and-strategy-terms/6814-steeple.html.
- M. Butler, “Android: Changing the Mobile Landscape”, Pervasive Computing, (2011), pp. 4-7.
- Ashwin Kumar. Mobile Application for News and Interactive Services. *Department of Information Technology, JNTU, Hyderabad, India*. VOL. 2, NO. 1, January 2012.