**I3301**

**JMX**

**Joseph Maary: 50906**

**Maher Makhlouf: 51935**

****

Summary

* ***Introduction***
* ***Analysis***
  + ***Business Rules***
  + ***Use Case***
  + ***Class Diagram***
  + ***Activity Diagram***
  + ***Sequence Diagram***
* ***Implementation***
  + ***Technical environment used***
  + ***Screens description***
* ***Conclusion***

***Introduction:***

**The suggested subject for this project is an ‘OLX’ like mobile application called JMX. The name JMX for this application was inspired by taking our first name initials, J and M (which refers to ‘Joseph’ and ‘Maher’) and the letter X. The letter X means exchange, which gives us the name JMX.**

**The word ‘exchange’ alongside with the word ‘OLX’ might have given you an idea about the application that we developed, and what type of services does this application provide.**

**Ever wanted to sell things you don’t need but still in good shape and serves well?**

**Maybe even buying used things that still in a good condition as well?**

**JMX is and application where a person can find used objects, even fairly newly purchased things for sale. All you have to do is just register and then login as a user so you can search your desired items.**

**Designing JMX layout was fulfilled using ‘Android Studio’ Software for mobile applications. JMX’s database was set using ‘WAMP Server’ (PhpMyAdmin). Finally, the different diagrams (use case, class …) used to build our database were all implemented in ‘Rational Rose’ Software that provides UML models and architectures.**

***Analysis:***

***Business Rules:***

**Our main objective from this project was to give the users the freedom of sharing their personal stuff and put them online for sale.**

**By that, users are free to select from several categories and subcategories in order to seek their desired items. Categories and subcategories will therefore be 2 predefined tables in our class diagram. Of course, a user must be identified by his name, username, password, email etc.…**

**So in order to share a user’s stuff, an Ad must be created containing the Ad’s title, location, description with his phone number and a symbolic image of what the user is putting for sale. An ads table must then be added to the class diagram to maintain this information. In addition for an extra table holding the other details of the ad such as the date of its creation.**

**Similar to the predefined tables mentioned before (categories and subcategories), one more predefined table is inserted into the class diagram to list the countries.**

**And finally this class diagram will include some other classes that has to do with the control and boundaries dealt with in our application.**

***Use Case:***

******

* + - * + **A user can “Login or Register” to the application.**
        + **A user will display the welcome page as soon as he logs in into the application.**
        + **A user will search an Ad of his interests either by searching for it or browsing through categories and sub-categories.**
        + **A user will set up his ad before creating it.**
        + **A user will post his ad on the application.**
        + **A user will contact another user upon approval on a certain selected item.**
        + **A user can logout from the application when satisfied.**

**All these use cases contribute as main functionalities for a proper use of this application, to be more specific, for a user to surf, share his items or even consider buying somebody else’s.**

***Class Diagram:***

****

**The class diagram is formed with all the entities, java control classes and xml classes created in our project.**

****

**These are the classes made for our application consisting of: Users, Ads, AdsCat, AdsSubCat, AdsCreation, Countries.**

**These classes were implemented in a “phpMyAdmin” database using ”WampServer”**

**Every relation between these tables is established by an arrow defining the cardinality and the tables within this relation.**

****

**This used to be a ‘many to many’ relation between Users and Ads, therefore an extra table has to be created called AdsCreation to break the many cardinality on both sides.**

**1 user can have many AdsCreation identified by its idCreation, dateCreation and the duration.**

**1 Ad can be created many times by any user. We identify this creation also by AdsCreation table.**

****

**Although ternary relation are avoided by programmers, but here it is actually benefitial to imlplement a ternary relation between these 3 tables, knowing that an Ad can be identified either by its Category (AdCat) or by its Sub-category (AdsSubCat)**

**Many Ads can share one Category. Same goes for the Sub-category.**

**1 Category can have many Sub-categories. (**The cardinality is inversed in the image by mistake!)

**Last, we have this relation between Users and Countries.**

**Many Users can share one country.**





**These are the classes implemented in Android Studio.**

**.XML classes are responsible for the layout of our application. .JAVA classes ensure control on the .xml pages and control over our connection to the php server while inserting and retrieving data from various tables. More details will be given in the MVC explanation**

***Activity Diagram:***

******

******

**This is our activity diagram representing the flow of control in our application. In other words, the series of steps taken to fulfill certain tasks.**

**For example:**

****

**The series of steps taken in order to login, if the login was successful.**

**(**More details of the steps will be shown in a pdf or Rational Rose file.**)**

***Sequence Diagram:***

******

****





**These are our sequence diagram representing the interaction between objects. It shows in what order these objects operate with one another. These interactions are mainly methods formed in our app application.**

**They are respectively associated with the following use cases: login, register, Setting up Ad and Search Ad**

**More details about the order of steps will be given in the MVC explanation.**

***Implementation:***

***Technical environment used:***

**As it was said before, the main application for implementing the different component of ‘JMX’ application is Android Studio for mobile applications.**

**It is a platform where you get to choose between designing your app using XML codes or graphically (Drag and Drop) which will eventually be transformed to XML codes.**

**The control of the graphical components (Widgets) is made in a JAVA file associated with the corresponding XML file.**

**An MVC (or Model–view–controller**) **model is an architectural pattern used to divide the application into 3 interconnected parts.**

**Referring to our application, each XML file is considered as the ‘VIEW’ part of the MVC.**

**Several JAVA files are created in our application that contains only methods. These methods are responsible for the selection insertion or update action on the database. Each class is named after the aimed table on the database. For example if we are inserting on table “Users”, the JAVA file containing the methods is called: users.java. Each file of these files is considered a “MODEL"**

**Other JAVA files in the application are made to control the connection connection between the database and Android Studio. As well as controlling the execution of the queries, the link between the data on the view and the data in the database. Those files represent the “Controller” part of the MVC.**

***Screens description:***

**Our tree of Screens comes as follows:**

**Upon click on the application, user is being welcomed by a “welcome page”.**

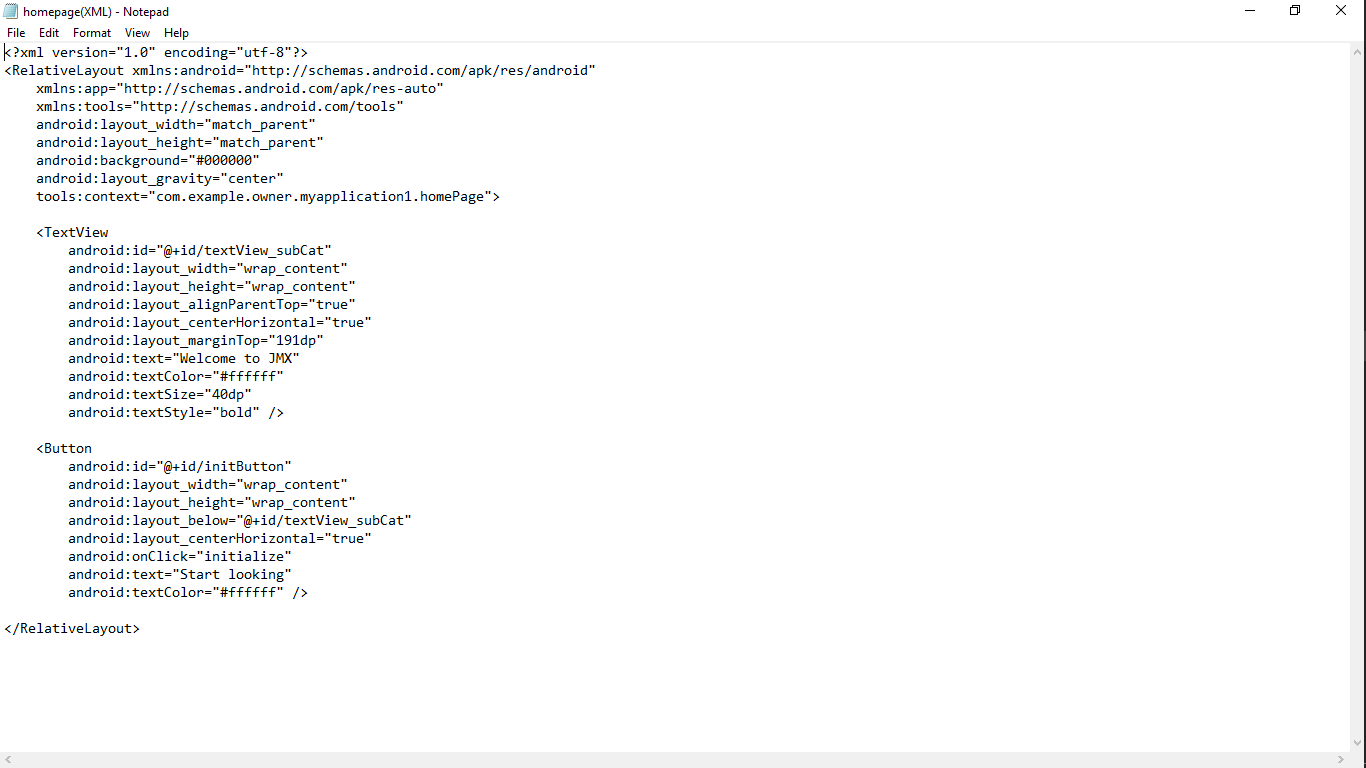


Figure HomePage.xml

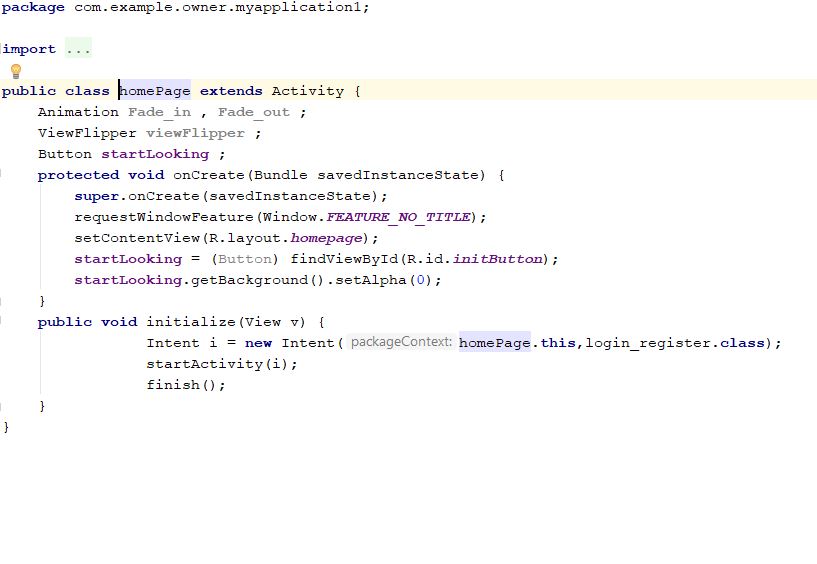
****

Figure Homepage.java



**When we click the “start Looking” Button from the Welcome Page,**

**the login/register Page is displayed.**

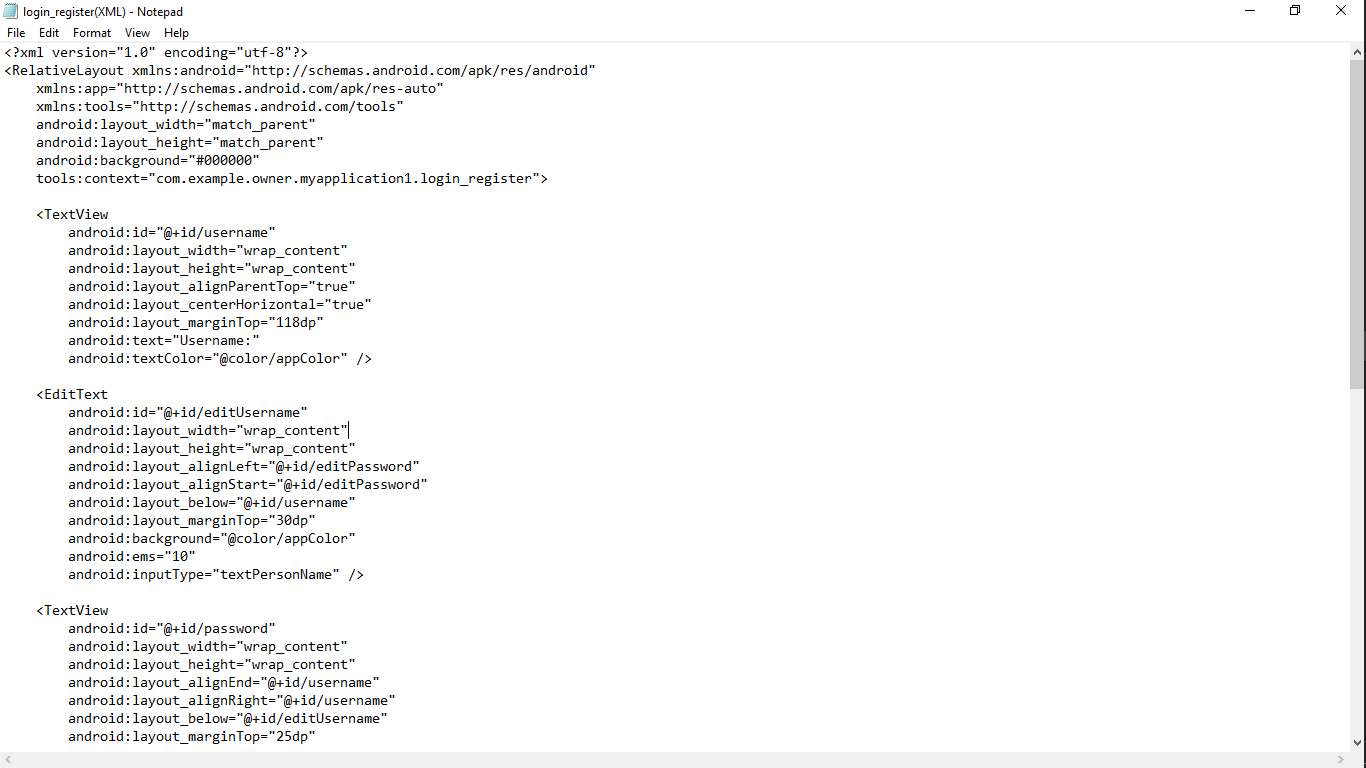
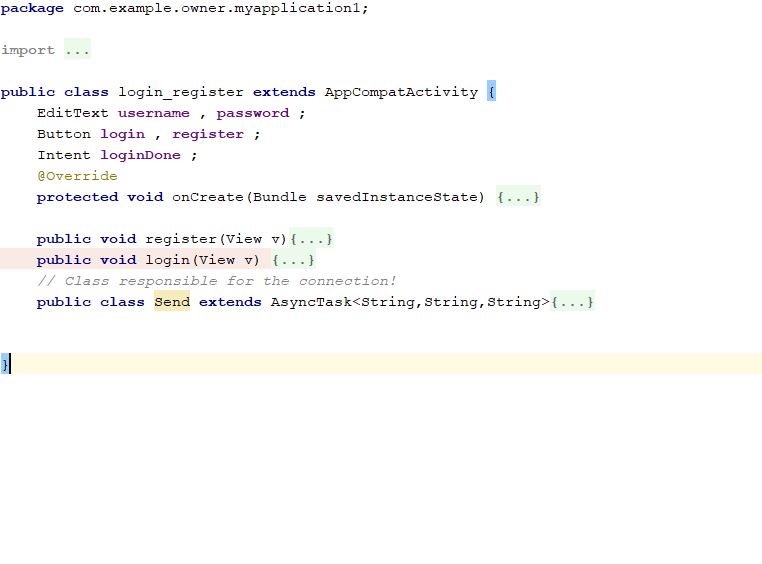


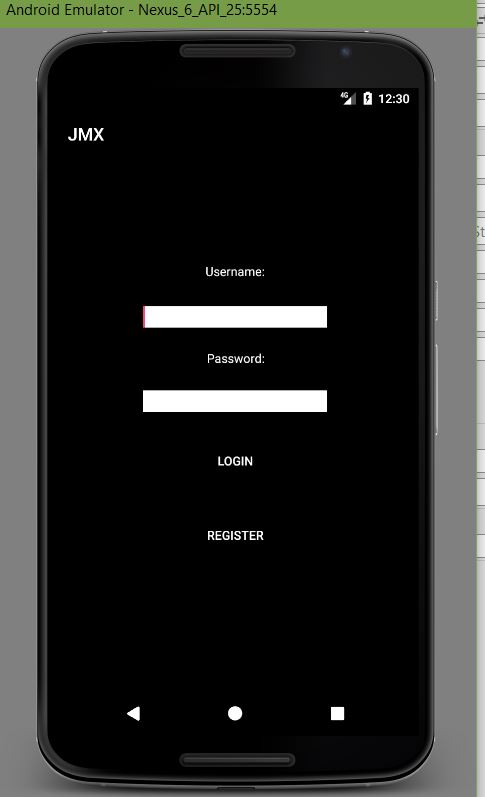
Figure Logn/Register.xml

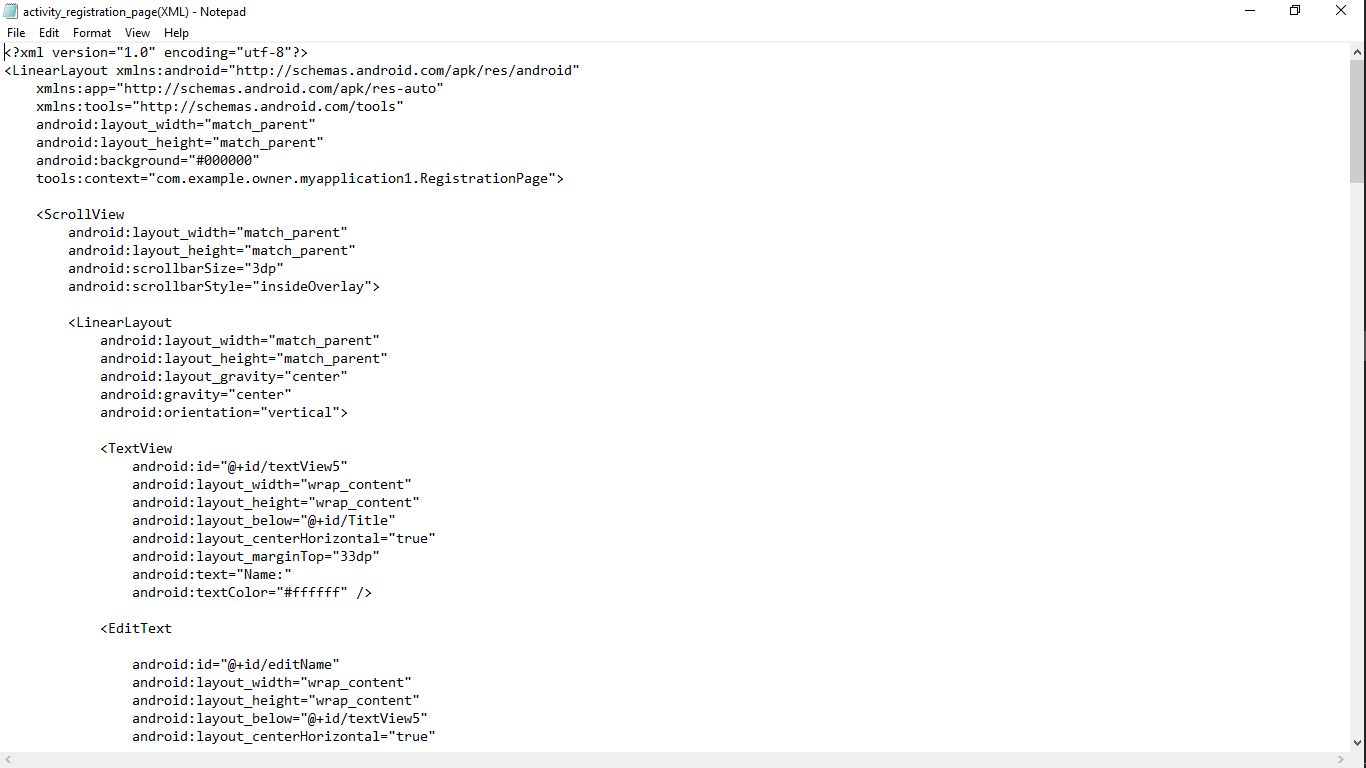
.

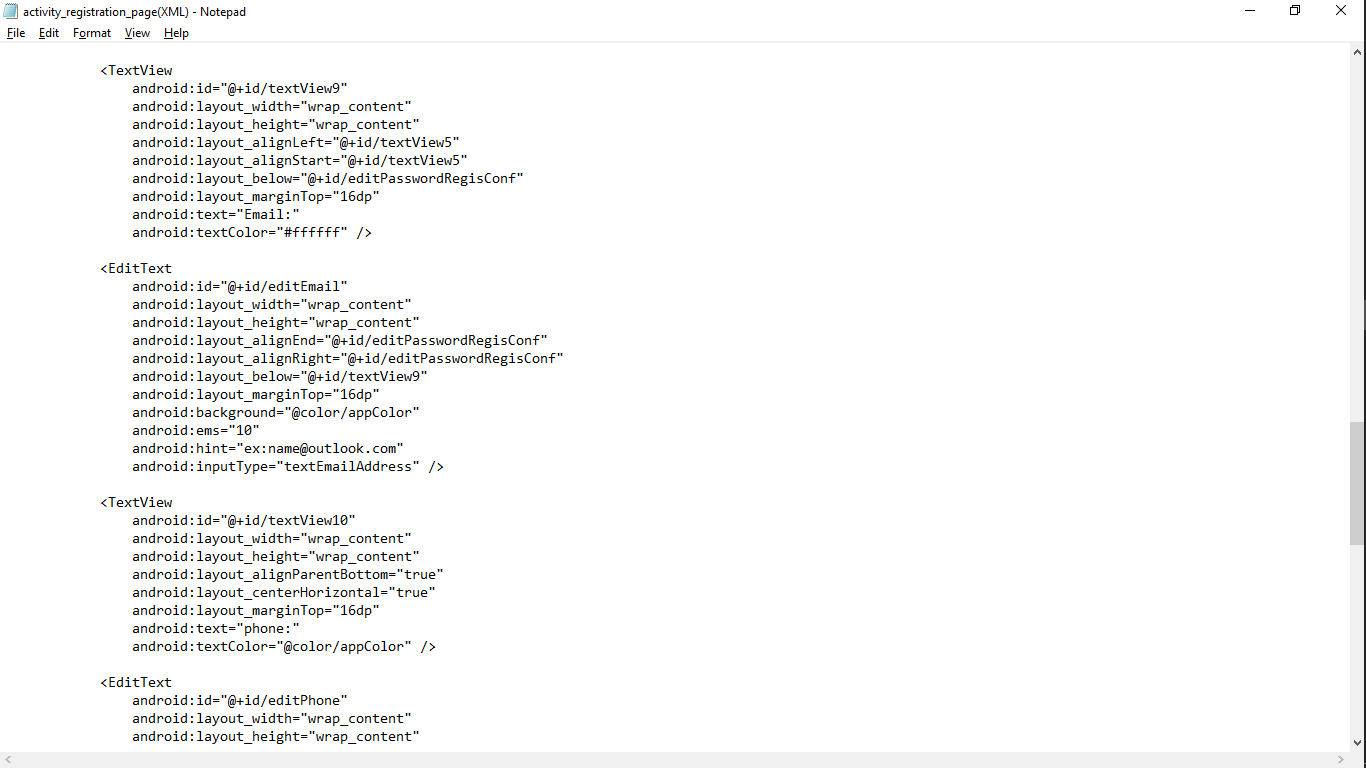
Registration page is opened.

If login successful, we move to the next page.

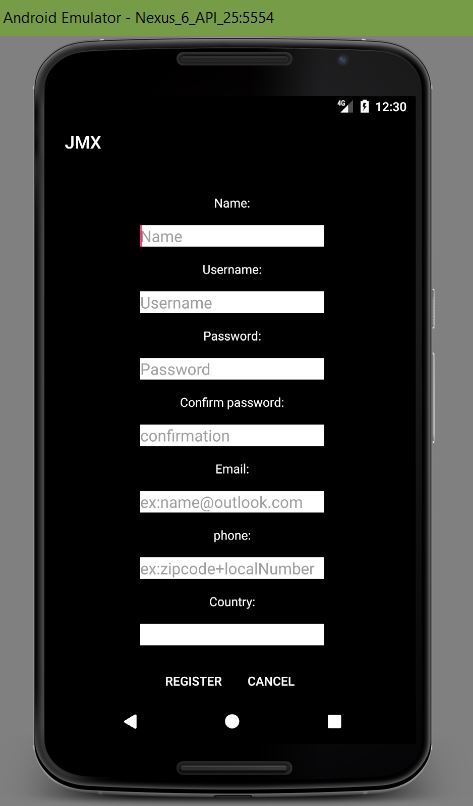
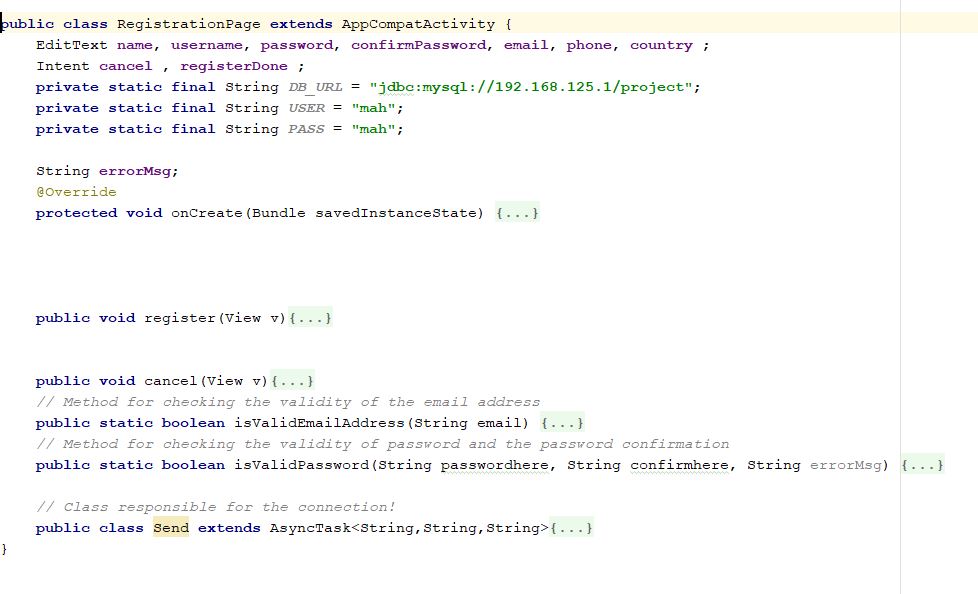
Figure Login/Register.java











One method to check whether the Password is entered correctly, matching the confirmation password. The other is to check if the email entered is valid.

Button cancel. Cancel registrstion

Registration button. Register a new user and move to the login page.

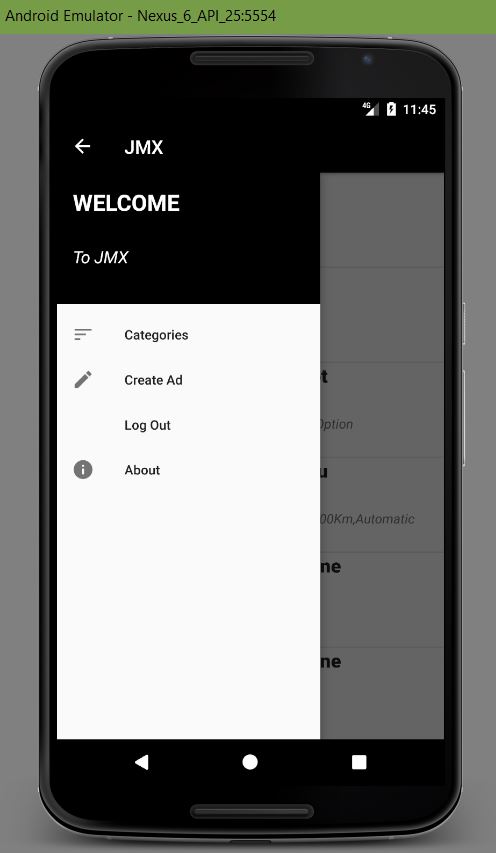
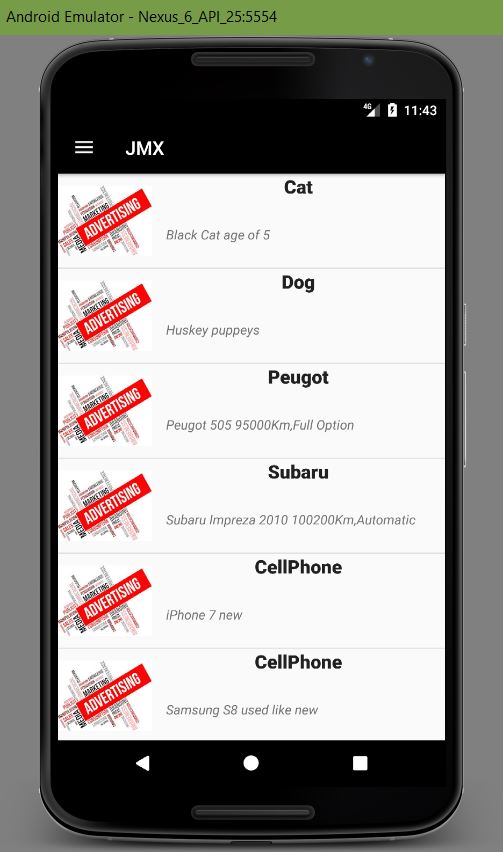


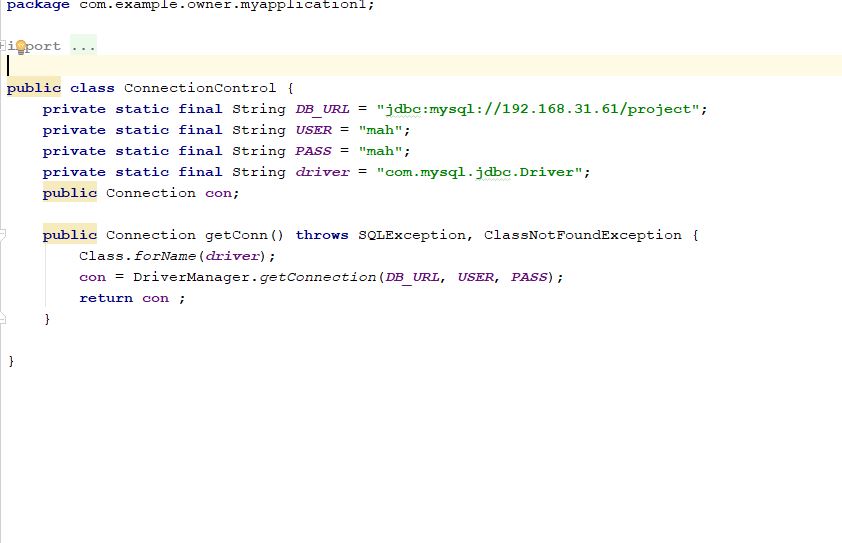
Xml code corresponding to the navigation bar in the explore page.

Xml code for showing the listView containing the title and the corresponding image of the item on the explore page.

Image and title are added n







Connection control class responsible for the connection on every class where we are importing or adding data to or from the database.

This class is called from these classes respectively.

**Conclusion**

**This project proved how indulging in group projects increase our experience in group work. Moreover, it provided us practical knowledge of not only programming in advance JAVA language and XML, Android Studio environment and its different features, but also about MVC architecture and how to implement this architecture while building our projects. It also provides beginner knowledge about connection between any platform we are working with and the database containing all the data. Hoping that this project gave us enough methodology in implementing future projects.**