Programación Orientada a Objetos

Proyecto Final

26 de mayo de 2019

Proyecto Final POO

Alumno
Carlos Daniel
Hernandez Chauteco

Profesora
Ing. Guadalupe
Lizeth Parrales
Romay

1. Justificación Del Proyecto

1.1. Resumen

El proyecto nace de una pequeña idea que tenia la cual es hacer una aplicación que busque entre muchas personas con tu mismos gustos con demás personas así nació la idea de la aplicación aunque al principio se suponía que era una aplicación de citas tengo pensado que si la llego a escalar a mas esta puede ser una aplicación para conocer gente con los mismos gustos y que no solo se limite a citas como en un principio se pensaba el principal problema para esta aplicación fue el backend que era inexistente y no sabia en concreto como lo iba hacer hasta que recordé el backend de Firebase que funciona como plataforma como servicio el cual me proporciona un backend con ciertas restricciones para las cuentas gratuitas pero que me soluciona el problema rápido y también a futuro cuando aprenda alguna tecnología de backend podria hacer el servidor.

2. Descripción

La aplicación esta hecha en java con el entorno de desarrollo de android studio para facilitar el desarrollo de la aplicación esta también esta escrita con mucho código XML que es el que define la interfaz gráfica del las aplicaciones con android la aplicación por el momento solo ha sido hecha para android así que solo esta en esta plataforma posiblemente se desarrolle su contra parte en ios

3. Código Fuente

Como la aplicación contiene mucho código decidí que este apartado solo contuviera el código de las clases que me parecieran importantes que son las mas 'importantes' para la aplicación a MainActitity no lo pogo por que no cabe en el documento

Login Activity:

```
package com.example.citeapp;

import android.content.Intent;
import android.os.Handler;
import android.os.Looper;
import android.support.design.widget.TabLayout;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
/**
 * Esta clase es el controlador del activity de
    Login
 * @author Carlos Daniel Hernandez Chauteco
 */
public class LoginActivity extends
   AppCompatActivity {

    FirebaseUser user =
       FirebaseAuth.getInstance().getCurrentUser();

    /**
     * Este metodo se mada a llamar cada que se
        crea una intancia del activity
     */
    @Override
    protected void onCreate(Bundle
       savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        //verificando que no se haya iniciado
           session
```

```
        if(user != null){
            theUserIsLoged();
        }

        Login_Activity_Fragment_Page_Adapter
           pageAdapter = new
           Login_Activity_Fragment_Page_Adapter(getSupportFragmentMa

        ViewPager viewPager = (ViewPager)
           findViewById(R.id.login_activity_view_pager);
        viewPager.setAdapter(pageAdapter);

        TabLayout tabLayout = (TabLayout)
           findViewById(R.id.login_activity_tabs);
        tabLayout.setupWithViewPager(viewPager);
    }

    /**
     * Verifica si el usuario antes se habia logeado
     */
    private void theUserIsLoged(){
        Intent intent = new
           Intent(this,MainActivity.class);
        startActivity(intent);
        finish();//elimina el activity
    }
}
```

Chat Activity:

```
package com.example.citeapp;

import android.content.Intent;
import android.graphics.Bitmap;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Adapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;

import
   com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.Timestamp;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import
   com.google.firebase.firestore.CollectionReference;
import com.google.firebase.firestore.DocumentChange;
import
   com.google.firebase.firestore.DocumentReference;
import
   com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.EventListener;
import
   com.google.firebase.firestore.FirebaseFirestore;
import
   com.google.firebase.firestore.FirebaseFirestoreException;
import
   com.google.firebase.firestore.ListenerRegistration;
import com.google.firebase.firestore.Query;
```

```
import
   com.google.firebase.firestore.QueryDocumentSnapshot;
import com.google.firebase.firestore.QuerySnapshot;
import com.squareup.picasso.Picasso;

import java.io.IOException;
import java.util.Date;
import java.util.LinkedList;

import javax.annotation.Nullable;

/**
 * Esta clase pertenece al activity principal de la
    vista del chat
 * @author Carlos Daniel Hernandez Chauteco
 */


public class ChatActivity extends AppCompatActivity
   {

    ListView listView;
    Button sendButton;
    EditText sendTextEditText;

    ListenerRegistration listener;
    LinkedList<Message> messages = new
       LinkedList<>();


    FirebaseUser user =
       FirebaseAuth.getInstance().getCurrentUser();

    /**
     * Este metodo se manda a llamar cada que una
        instancia de esta vista es creada
     */
```

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

    setupUI();

    //traer todo el layout

    listView = (ListView)
       findViewById(R.id.chat_activity_list_view);
    sendButton = (Button)
       findViewById(R.id.chat_activity_send_button);
    sendTextEditText = (EditText)
       findViewById(R.id.chat_activity_edit_text);

    Intent intent = getIntent();
    String chatId =
       intent.getStringExtra("chatId");
    String userName =
       intent.getStringExtra("userName");
    String imageUrl =
       intent.getStringExtra("imageUrl");

    final Chat_Activity_List_View_Adapter
       adapter = new
       Chat_Activity_List_View_Adapter(messages,getApplicationCo
    listView.setAdapter(adapter);

    setTitle("Message con " + userName);

    final EventListener eventListener = new
       EventListener<QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable
           QuerySnapshot snapshots, @Nullable
           FirebaseFirestoreException exeption)
```

```
        {
          if(exeption != null){
              //error
              return;
          }

          if(!snapshots.isEmpty()){
              for(DocumentChange
                 documentChange :
                 snapshots.getDocumentChanges()){
                  QueryDocumentSnapshot
                     document =
                     documentChange.getDocument();
                  messages.add(Message.fromMap(document.getDat
                  listView.smoothScrollToPosition(messages.siz
                  adapter.notifyDataSetChanged();
              }
          }else{
              //error
          }
      }
};

//cargamos los mensages
FirebaseFirestore firestore =
   FirebaseFirestore.getInstance();//obtenemos
   la referencia
final CollectionReference reference =
   firestore.collection("chats").document(chatId).collection

listener = reference.orderBy("date",
   Query.Direction.ASCENDING).addSnapshotListener(eventListe



//accion para el boton de enviar

sendButton.setOnClickListener(new
```

```
        View.OnClickListener() {
         @Override
         public void onClick(View v) {
             final String messageText =
                sendTextEditText.getText().toString();
             if(!TextUtils.isEmpty(messageText)){
                 Message newMessage = new
                    Message(user.getUid(),messageText,
                    Timestamp.now());
                 reference.document().set(newMessage.toMap());
                 listView.smoothScrollToPosition(adapter.getCount
             }
         }
     });

}

/**
 * Este metodo se llama cada que la vista se
    destuye y en este caso destruimos el lister
    que solo
 * es un socket que nos ayuda a que cada que
    llegue un nuevo mensaje este lo agrege a la
    tabla
 */

@Override
protected void onStop() {
    if(listener != null){
        listener.remove();
    }
    super.onStop();
}

/**
 * Este metodo sirve para algunas cosas de UI
 */
private void setupUI(){
```

```
        View rootView = getWindow().getDecorView();
           //obtenemos la vista principal
    }
}
```

login Activity:

```
package com.example.citeapp;

import android.content.Intent;
import android.os.Handler;
import android.os.Looper;
import android.support.design.widget.TabLayout;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
/**
 * Esta clase es el controlador del activity de
    Login
 * @author Carlos Daniel Hernandez Chauteco
 */
public class LoginActivity extends
   AppCompatActivity {

    FirebaseUser user =
       FirebaseAuth.getInstance().getCurrentUser();

    /**
     * Este metodo se mada a llamar cada que se
        crea una intancia del activity
     */
    @Override
    protected void onCreate(Bundle
       savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        //verificando que no se haya iniciado
           session
```

```
        if(user != null){
            theUserIsLoged();
        }

        Login_Activity_Fragment_Page_Adapter
           pageAdapter = new
           Login_Activity_Fragment_Page_Adapter(getSupportFragmentMa

        ViewPager viewPager = (ViewPager)
           findViewById(R.id.login_activity_view_pager);
        viewPager.setAdapter(pageAdapter);

        TabLayout tabLayout = (TabLayout)
           findViewById(R.id.login_activity_tabs);
        tabLayout.setupWithViewPager(viewPager);
    }

    /**
     * Verifica si el usuario antes se habia logeado
     */
    private void theUserIsLoged(){
        Intent intent = new
           Intent(this,MainActivity.class);
        startActivity(intent);
        finish();//elimina el activity
    }
}
```

Profile Activity:

```
package com.example.citeapp;

import android.app.Activity;
import android.app.Dialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import
   android.support.design.widget.FloatingActionButton;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.ListView;

import
   com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import
   com.google.firebase.firestore.CollectionReference;
import com.google.firebase.firestore.DocumentChange;
import
   com.google.firebase.firestore.DocumentReference;
import
   com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.EventListener;
import
   com.google.firebase.firestore.FirebaseFirestore;
import
```

```
   com.google.firebase.firestore.FirebaseFirestoreException;
import
   com.google.firebase.firestore.ListenerRegistration;
import com.google.firebase.firestore.Query;
import com.google.firebase.firestore.QuerySnapshot;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.UploadTask;
import com.squareup.picasso.Picasso;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.Map;
import java.util.Set;

import javax.annotation.Nullable;

public class ProfileActivity extends Activity {

    ImageView imageView;
    EditText usernameEditText;
    EditText ageEditText;
    FloatingActionButton actionButton;
    ListView listView;
    Button addGustoButton;
    Dialog popUpAgregarGusto;

    ListenerRegistration userDocListener;
    ListenerRegistration
       collectionGustosUserListener;

    LinkedList<String> gustos = new LinkedList<>();
    Profile_Activity_Gustos_Item_List_View_Adapter
       listViewAdapter;

    FirebaseFirestore firestore =
       FirebaseFirestore.getInstance();
```

```
FirebaseUser user =
   FirebaseAuth.getInstance().getCurrentUser();
FirebaseStorage storage =
   FirebaseStorage.getInstance();

/**
 * Se llama la crear en cada intacia de la
    clase y este agrega la logica de los
    widgets y de la base de daros
 */
@Override
protected void onCreate(Bundle
   savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_profile);

    //get all viewa
    imageView = (ImageView)
       findViewById(R.id.activity_profile_image_user);
    usernameEditText = (EditText)
       findViewById(R.id.activity_profile_nombre_edit_text);
    ageEditText = (EditText)
       findViewById(R.id.activity_profile_edad_user);
    actionButton = (FloatingActionButton)
       findViewById(R.id.activity_profile_float_button);
    listView = (ListView)
       findViewById(R.id.activity_profile_list_view);
    addGustoButton = (Button)
       findViewById(R.id.activity_profile_add_gusto);
    //creando el adapter para que se vean los
       gustos en el list view

    listViewAdapter = new
       Profile_Activity_Gustos_Item_List_View_Adapter(this,gusto

    listView.setAdapter(listViewAdapter);

    final DocumentReference userDocRef =
```

```
    firestore.collection("users").document(user.getUid());
//para registrar los eventos del documento
    del ussuario
userDocListener =
    userDocRef.addSnapshotListener(new
    com.google.firebase.firestore.EventListener<DocumentSnapsh
    {
     @Override
     public void onEvent(@Nullable
        DocumentSnapshot documentSnapshot,
        @Nullable FirebaseFirestoreException
        exeption) {
         if(exeption != null){
             return;
         }

         if(documentSnapshot != null &&
            documentSnapshot.exists()){

             Map<String, Object> userMap =
                documentSnapshot.getData();

             final String userName =
                (String)
                userMap.get(Person.NAME);
             final String ageUser = (String)
                userMap.get(Person.AGE);
             final String imageUrl =
                (String)
                userMap.get(Person.IMAGE_PATH);

             if(!imageUrl.equals("")){
                 Picasso.get().load(imageUrl).into(imageView)
             }
             usernameEditText.setText(userName);
             ageEditText.setText(ageUser);

             todoLosComponentesSonEditables(false);
```

```
            }else{
                return;
            }
        }
});

//listener para ver lso gustos del usuario
   en real time
final CollectionReference
   gustosCollectionReference =
   userDocRef.collection("gustos");

collectionGustosUserListener =
   gustosCollectionReference.orderBy("field",
   Query.Direction.DESCENDING).addSnapshotListener(new
   EventListener<QuerySnapshot>() {
    @Override
    public void onEvent(@Nullable
       QuerySnapshot snapshots, @Nullable
       FirebaseFirestoreException exeption)
       {
        if(exeption != null){
            return;
        }

        if(snapshots!= null &&
           !snapshots.isEmpty()){
            for(DocumentChange
               documentChange :
               snapshots.getDocumentChanges()){
                Map<String, Object>
                   gustoMap =
                   documentChange.getDocument().getData();

                final String gusto =
                   (String)
                   gustoMap.get("field");
```

```
                    gustos.add(gusto);

                    listViewAdapter.notifyDataSetChanged();
                }
            }else{
                return;
            }
        }
    });

    //acccion para el floatButton
    actionButton.setOnClickListener(new
       View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if(ageEditText.isEnabled() &&
               usernameEditText.isEnabled()){
                final String newAge =
                   ageEditText.getText().toString();
                final String newName =
                   usernameEditText.getText().toString();
                userDocRef.update(Person.NAME,newName);
                userDocRef.update(Person.AGE,newAge);

                todoLosComponentesSonEditables(false);
            }else{
                todoLosComponentesSonEditables(true);
            }
        }
    });

    //mostrar pop up
    popUpAgregarGusto = new Dialog(this);

    addGustoButton.setOnClickListener(new
       View.OnClickListener() {
        @Override
        public void onClick(View v) {
```

```
                showPopUp();
            }
        });

        //request cargar imagen
        imageView.setOnClickListener(new
           View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new
                   Intent(Intent.ACTION_PICK,
                   MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
                startActivityForResult(intent,200);
            }
        });

    }

    /**
     * Este metodo se llama cuando el activity se
        pausa y de paso elimino el lintener de
        gustos
     * para que no siga escuchando los cambios y
        guardamos los gustos en memoria secundaria
     */
    @Override
    protected void onStop() {
        if(userDocListener != null){
            userDocListener.remove();
        }
        if(collectionGustosUserListener != null){
            collectionGustosUserListener.remove();
        }

        //guardamos los datos de los gustos

        SharedPreferences preferences =
           getSharedPreferences("utils",Context.MODE_PRIVATE);
```

```
        SharedPreferences.Editor editor =
           preferences.edit();

        final Set<String> setOfGustos = new
           HashSet<String>();
        for(String gusto : gustos){
            setOfGustos.add(gusto);
        }

        editor.putStringSet("gustos",setOfGustos);
        editor.commit();

        super.onStop();
    }

    /**
     * Este metodo solo cambia el estado de los
        elementos de que se cambiaran el la base de
        datos
     * @param condicion
     */
    private void
       todoLosComponentesSonEditables(boolean
       condicion){
        usernameEditText.setEnabled(condicion);
        ageEditText.setEnabled(condicion);
    }

    /**
     * Este metodo hace aparecer el pop up para
        agregar una un nuevo gusto
     */
    private void showPopUp(){
        popUpAgregarGusto.setContentView(R.layout.activity_profile_p

        final EditText newGustoEditText =
           (EditText)
           popUpAgregarGusto.findViewById(R.id.activity_profile_pop_
```

```
        Button addNewGustoButton = (Button)
           popUpAgregarGusto.findViewById(R.id.activity_profile_pop_
        Button closePopUp = (Button)
           popUpAgregarGusto.findViewById(R.id.main_activity_pop_up_

        addNewGustoButton.setOnClickListener(new
           View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final String gusto =
                   newGustoEditText.getText().toString().replace("
                   ","_").toLowerCase();
                FirestoreUtils.instance.setContext(getApplicationCon
                FirestoreUtils.instance.addNewGusto(gusto,listViewAd
                popUpAgregarGusto.dismiss();
            }
        });

        closePopUp.setOnClickListener(new
           View.OnClickListener() {
            @Override
            public void onClick(View v) {
                popUpAgregarGusto.dismiss();
            }
        });

        popUpAgregarGusto.show();
    }

    /**
     * Se llama cada que que se selecciona una
        imagen de la galeria del usuario
     * @param data los datos de la imagen
        seleccionada
     */
    @Override
    protected void onActivityResult(int
       requestCode, int resultCode, Intent data) {
```

```
        super.onActivityResult(requestCode,
           resultCode, data);
        if(requestCode == 200){
            Uri selectedImage = data.getData();
            Bitmap imageBitmap = null;
            try {
                imageBitmap =
                   MediaStore.Images.Media.getBitmap(this.getContentI
            } catch (IOException e) {
                e.printStackTrace();
            }
            uploadImage(imageBitmap);
            imageView.setImageBitmap(imageBitmap);
        }
    }

    /**
     * Sube la imagen del usuario en la base de
        datos
     * @param image El bitmap de la imagen a subir
     */
    private void uploadImage(Bitmap image){
        ByteArrayOutputStream baos = new
           ByteArrayOutputStream();
        image.compress(Bitmap.CompressFormat.JPEG,100,baos);
        byte[] data = baos.toByteArray();

        storage.getReference().child("users").child(user.getUid()).p
           OnSuccessListener<UploadTask.TaskSnapshot>()
           {
            @Override
            public void
               onSuccess(UploadTask.TaskSnapshot
               taskSnapshot) {
                taskSnapshot.getMetadata().getReference().getDownloa
                   OnSuccessListener<Uri>() {
                    @Override
                    public void onSuccess(Uri uri) {
```

```
                        firestore.collection("users").document(user.
                    }
                });
            }
        });
    }
}
```

Por supuesto los modelos de datos que utilize para apliacion son super importantes

Person:

```
package com.example.citeapp;


import java.util.HashMap;
import java.util.Map;

/**
 * Esta clase es el modelo de datos de una persona
 * @author Carlos Daniel Hernandez Chauteco
 */
public class Person {
    //constantes
    public static final String NAME = "name";
    public static final String EMAIL = "email";
    public static final String IMAGE_PATH =
       "imagePath";
    public static final String AGE = "age";


    String name;
    String email;
    String imageUrl;
    String age;

    //variblesUtiles en ciertos casos
    String chatId = null;
    String userId = null;

    /**
     * @param name nombre de la persona
     * @param email email de la persona
     * @param imageUrl url de la imagen del usuario
     * @param age edad de la persona
     */
```

```
public Person(String name, String email, String
   imageUrl, String age){
    this.name = name;
    this.email = email;
    this.imageUrl = imageUrl;
    this.age = age;
}

/**
 * Contructor que sirve para hacer una persona
    solo con el nombre
 * @param name Nombre de la persona
 */
public Person(String name){
    this.name = name;
    this.email = "";
    this.imageUrl = "";
    this.age = "-1";
}

/**
 * Convierte el objeto en un Map para subirlo a
    la base de datos
 */
public HashMap<String,Object> toHashMap(){
    HashMap<String, Object> returnValue = new
       HashMap<>();
    returnValue.put(NAME,name);
    returnValue.put(EMAIL,email);
    returnValue.put(IMAGE_PATH,imageUrl);
    returnValue.put(AGE,age);
    return returnValue;
}

/**
 * Convierte un map en un objeto del tipo
    persona funciona para la base de datos
 * @param hashMap El map que se convertira
```

```
     * @return Regresa la instancia de la persona
     */
    public static Person
       fromhMap(Map<String,Object> hashMap){
        String name = (String) hashMap.get(NAME);
        String email = (String) hashMap.get(EMAIL);
        String imagePath = (String)
           hashMap.get(IMAGE_PATH);
        String age = (String) hashMap.get(AGE);

        return new Person(name,email,imagePath,age);
    }
}
```

Message:

```
package com.example.citeapp;

import com.google.firebase.Timestamp;

import java.util.HashMap;
import java.util.Map;

/**
 * Este es el modelo de datos de un mensaje
 * @author Carlos Daniel Hernandez Chauteco
 */
public class Message {
    //constantes para la base de datos
    public static final String AUTOR = "autor";
    public static final String MESSAGE = "message";
    public static final String DATE = "date";


    String autor;
    String message;
    Timestamp date;

    /**
     * @param autor Autor del mensaje
     * @param message Contenido del mensaje
     * @param date Fecha del mensaje
     */
    public Message(String autor, String message,
       Timestamp date){
        this.autor = autor;
        this.message = message;
        this.date = date;
    }

    /**
     * Este metodo convierte la instancia en un
```

```
        mapa este funciona por que de esta manera
        lo enviamos a la
     * base de datos
     * @return El mapa del objeto
     */
    public Map<String,Object> toMap(){
        Map<String,Object> returnValue = new
           HashMap<>();

        returnValue.put(AUTOR,this.autor);
        returnValue.put(MESSAGE,this.message);
        returnValue.put(DATE,this.date);

        return returnValue;
    }

    /**
     * Convierte un Map en una instancia del objeto
        mensaje este por eso es estatico este metodo
     * al final esto sirve por que la base de datos
        regresas mapas
     * @param map el mapa que se convertira en una
        instancia del tipo Mensaje
     * @return La instacia del mensaje como objeto
     */
    public static Message fromMap(Map<String,
       Object> map){
        String autor = (String) map.get(AUTOR);
        String message = (String) map.get(MESSAGE);
        Timestamp date = (Timestamp) map.get(DATE);
        return new Message(autor,message,date);
    }
}
```

4. Diagramas UML

Diagrama de caso de uso de la aplicación:

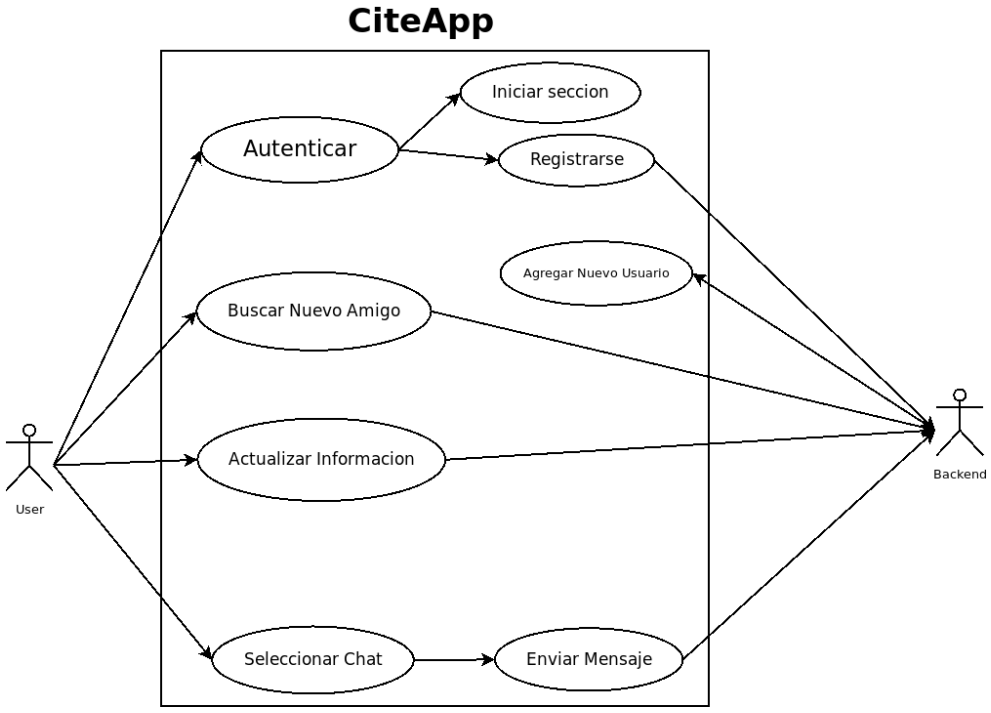
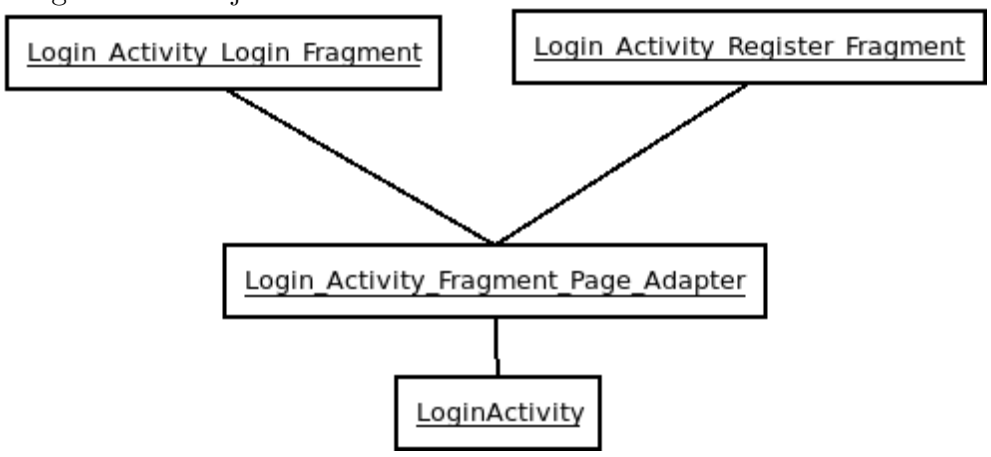
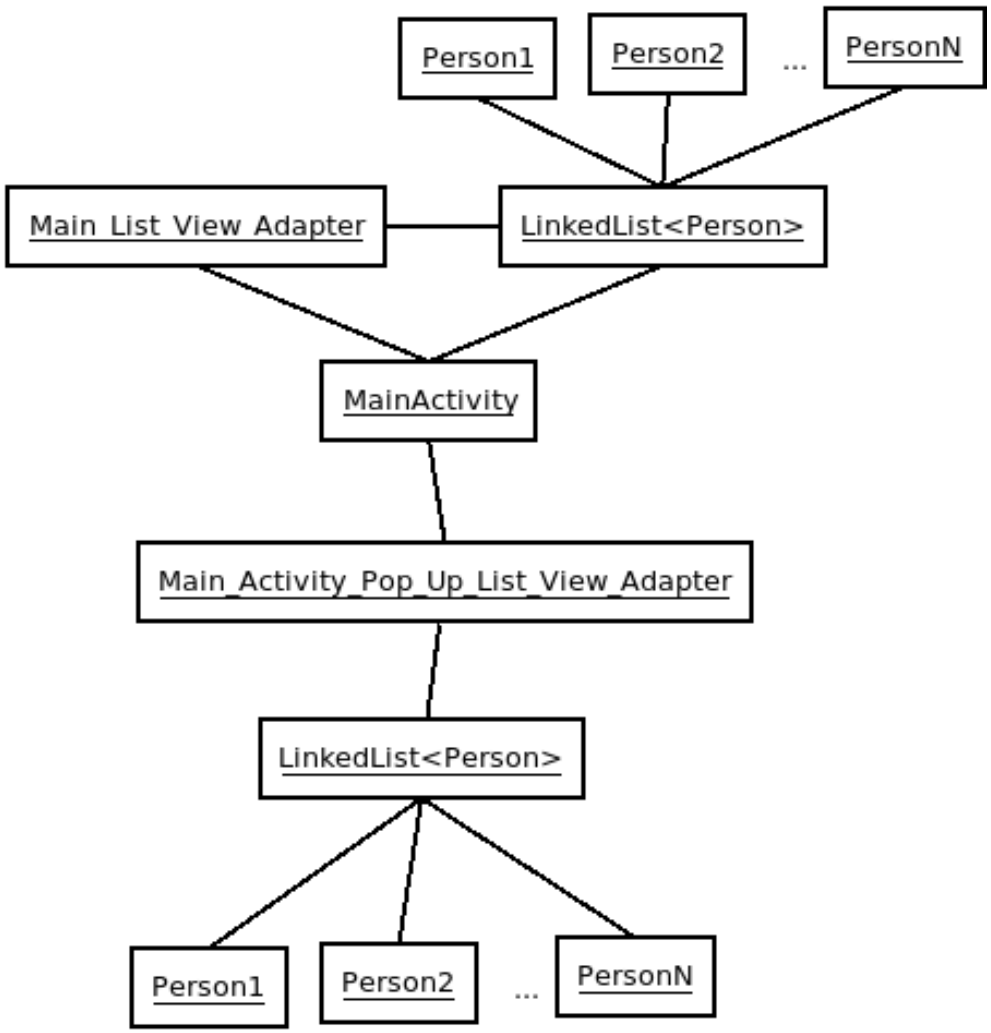


Diagrama de Objetos:




Person1
Person2
...
PersonN
Main_List_View_Adapter
LinkedList<Person>
MainActivity
Main_Activity_Pop_Up_List_View_Adapter
LinkedList<Person>
Person1
Person2
...
PersonN

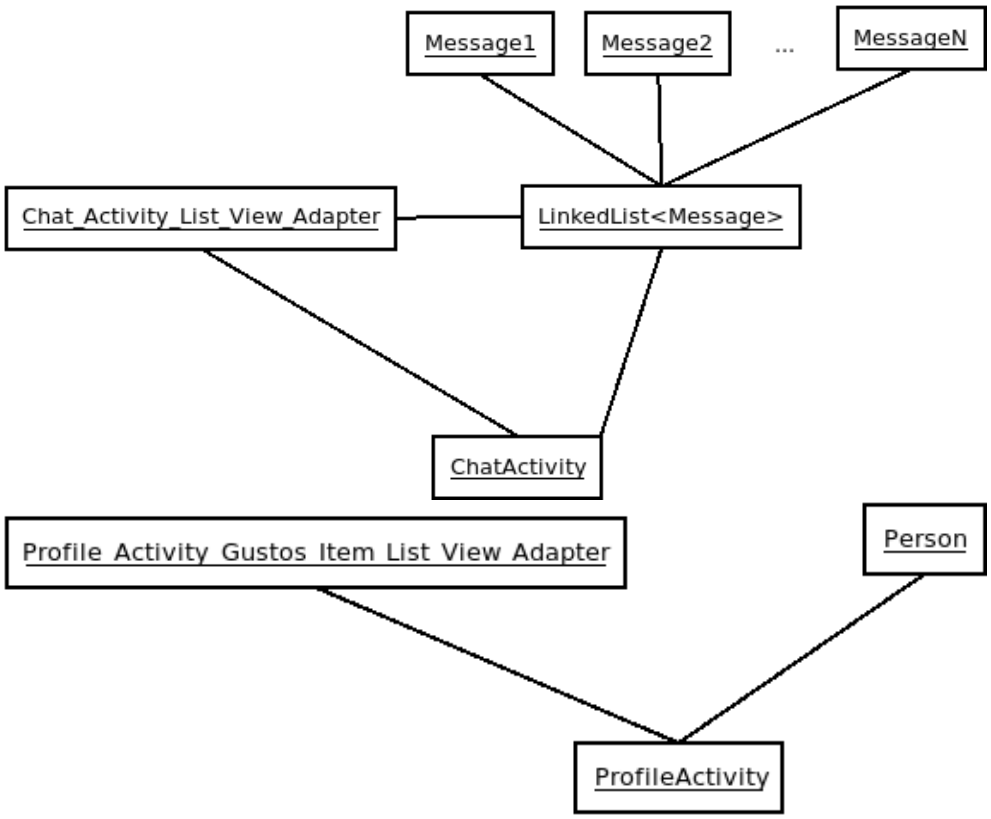


Diagrama de clases:

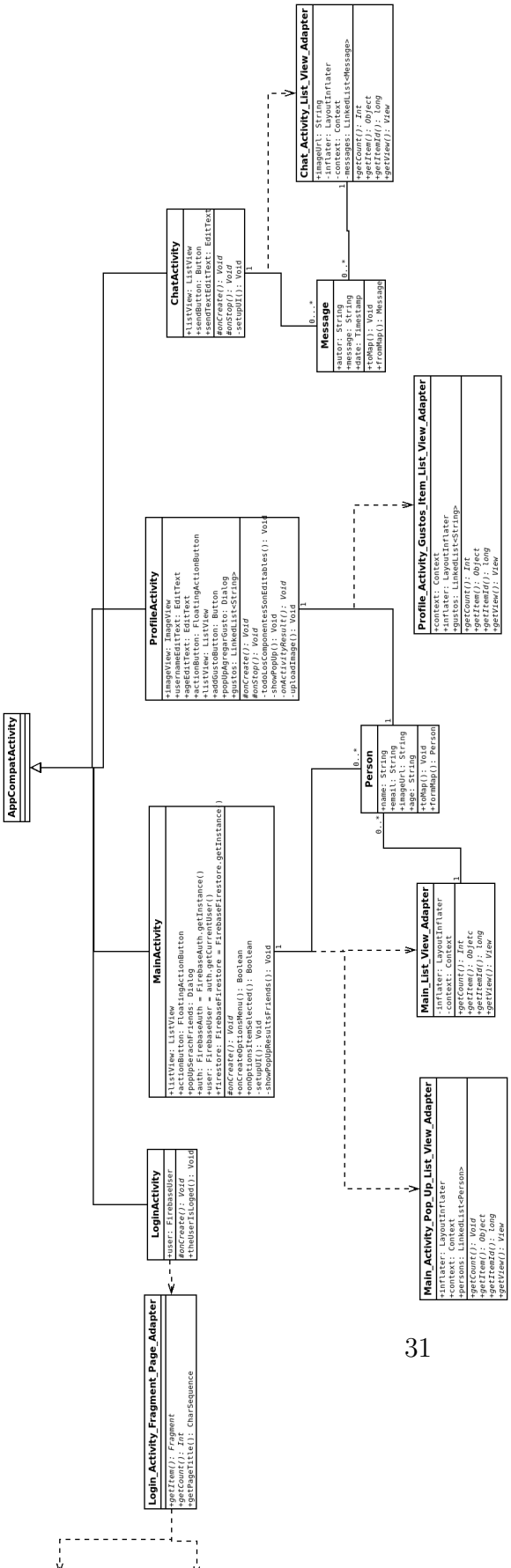

AppCompatActivity
LoginActivity
-user: FirebaseUser
#onCreate(): Void
+theUserIsLoged(): Void
MainActivity
+listView: ListView
+actionButton: FloatingActionButton
+popUpSeracnFriends: Dialog
+auth: FirebaseAuth = FirebaseAuth.getInstance()
+user: FirebaseUser = auth.getCurrentUser()
+access: FirebaseFirestore = FirebaseFirestore.getInstance()
#onCreate(): Void
+onCreateOptionsMenu(): Boolean
+onOptionsItemSelected(): Boolean
-setupUI(): Void
-showPopUpResultFriends(): Void
ProfileActivity
+imageView: ImageView
+usernameEditText: EditText
+ageEditText: EditText
+actionButton: FloatingActionButton
+listView: ListView
+addGustoButton: Button
+popUpAgregarGusto: Dialog
+gustos: LinkedList<String>
#onCreate(): Void
#onStart(): Void
-todoLosComponentesSonEditables(): Void
-showPopUp(): Void
-onActivityResult(): Void
-uploadImage(): Void
ChatActivity
+listView: ListView
+sendButton: Button
+sendTextEditText: EditText
#onCreate(): Void
#onStop(): Void
-setupUI(): Void
Chat_Activity_List_View_Adapter
+inputStr: String
-inflater: LayoutInflater
-context: Context
-messages: LinkedList<Message>
+getCount(): Int
+getItem(): Object
+getItemId(): Long
+getView(): View
Message
+autor: String
+message: String
+date: Timestamp
+toMap(): Void
+fromMap(): Message
Person
+name: String
+email: String
+imageUrl: String
+age: String
+toMap(): Void
+formMap(): Person
Profile_Activity_Gustos_Item_List_View_Adapter
+context: Context
+inflater: LayoutInflater
+gustos: LinkedList<String>
+getCount(): Int
+getItem(): Object
+getItemId(): long
+getView(): View
Main_List_View_Adapter
-inflater: LayoutInflater
-context: Context
+getCount(): Int
+getItem(): Objetc
+getItemId(): long
+getView(): View
Main_Activity_Pop_Up_List_View_Adapter
+inflater: LayoutInflater
-context: Context
+persons: LinkedList<Person>
+getCount(): Void
+getItem(): Object
+getItemId(): long
+getView(): View
Login_Activity_Fragment_Page_Adapter
+getItem(): Fragment
+getCount(): Int
+getPageTitle(): CharSequence
1
0..*

Diagrama de Componentes:

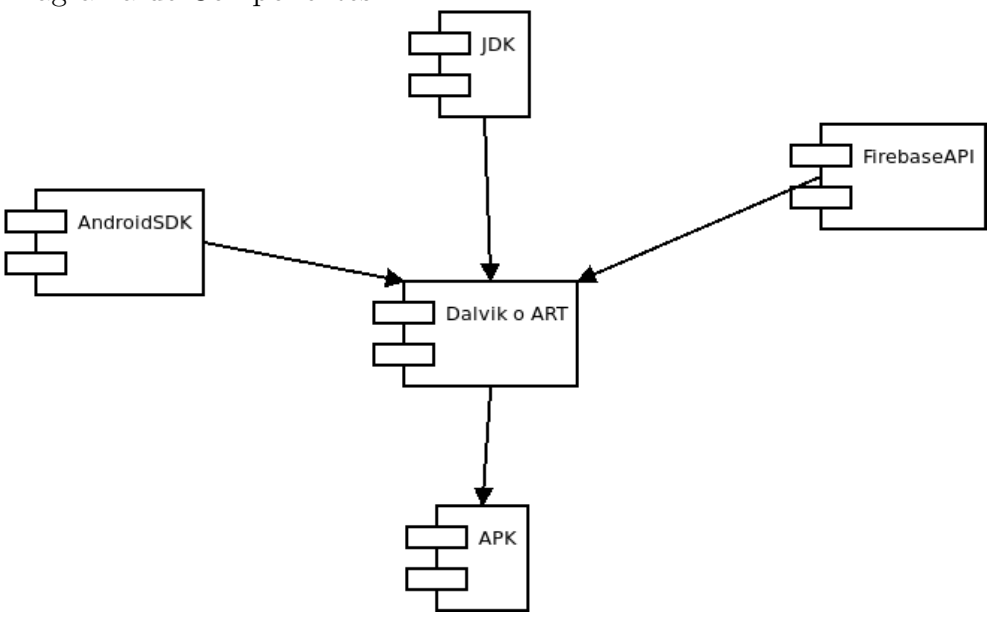
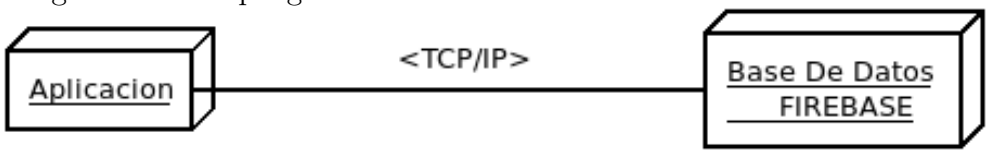
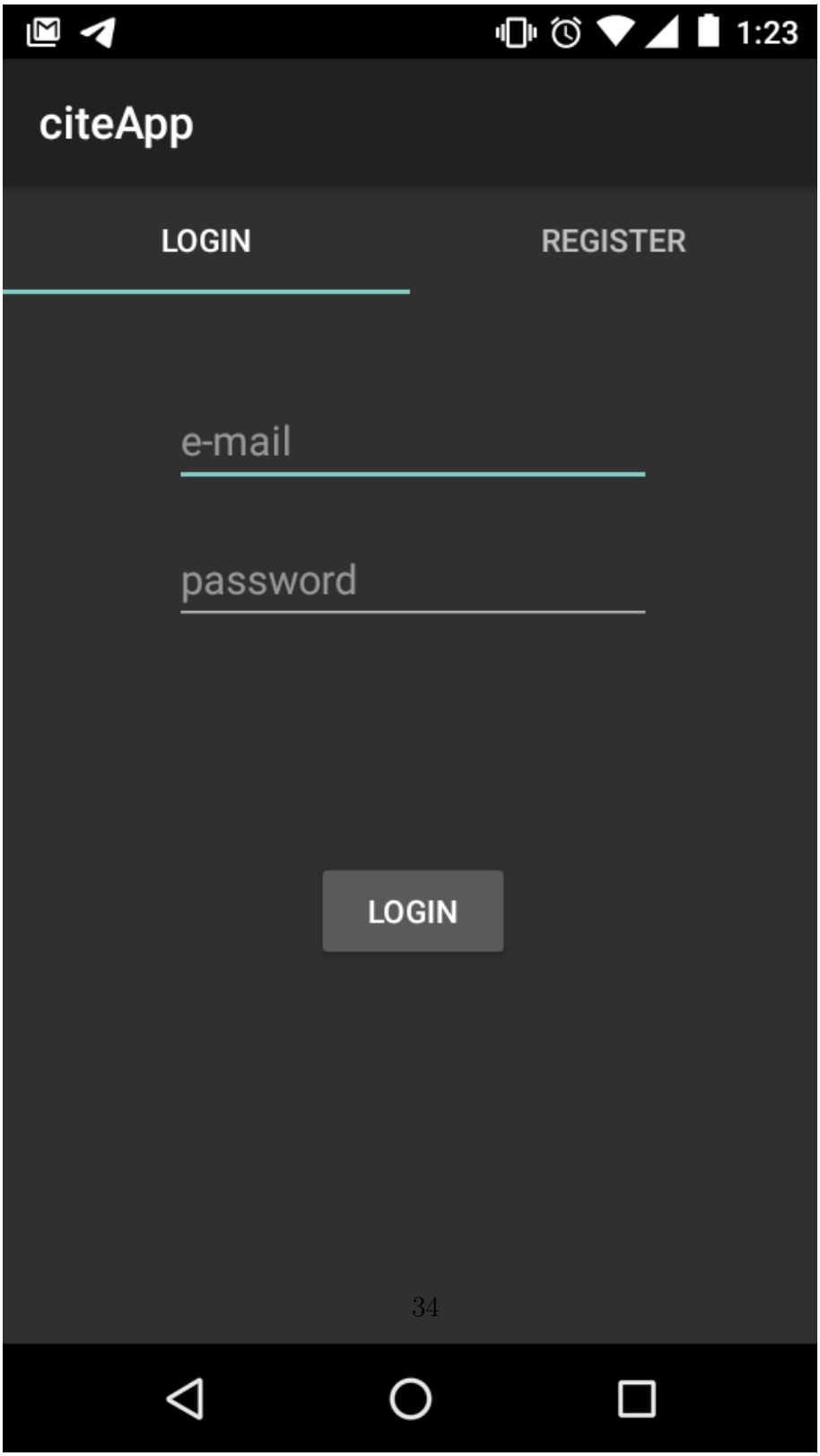
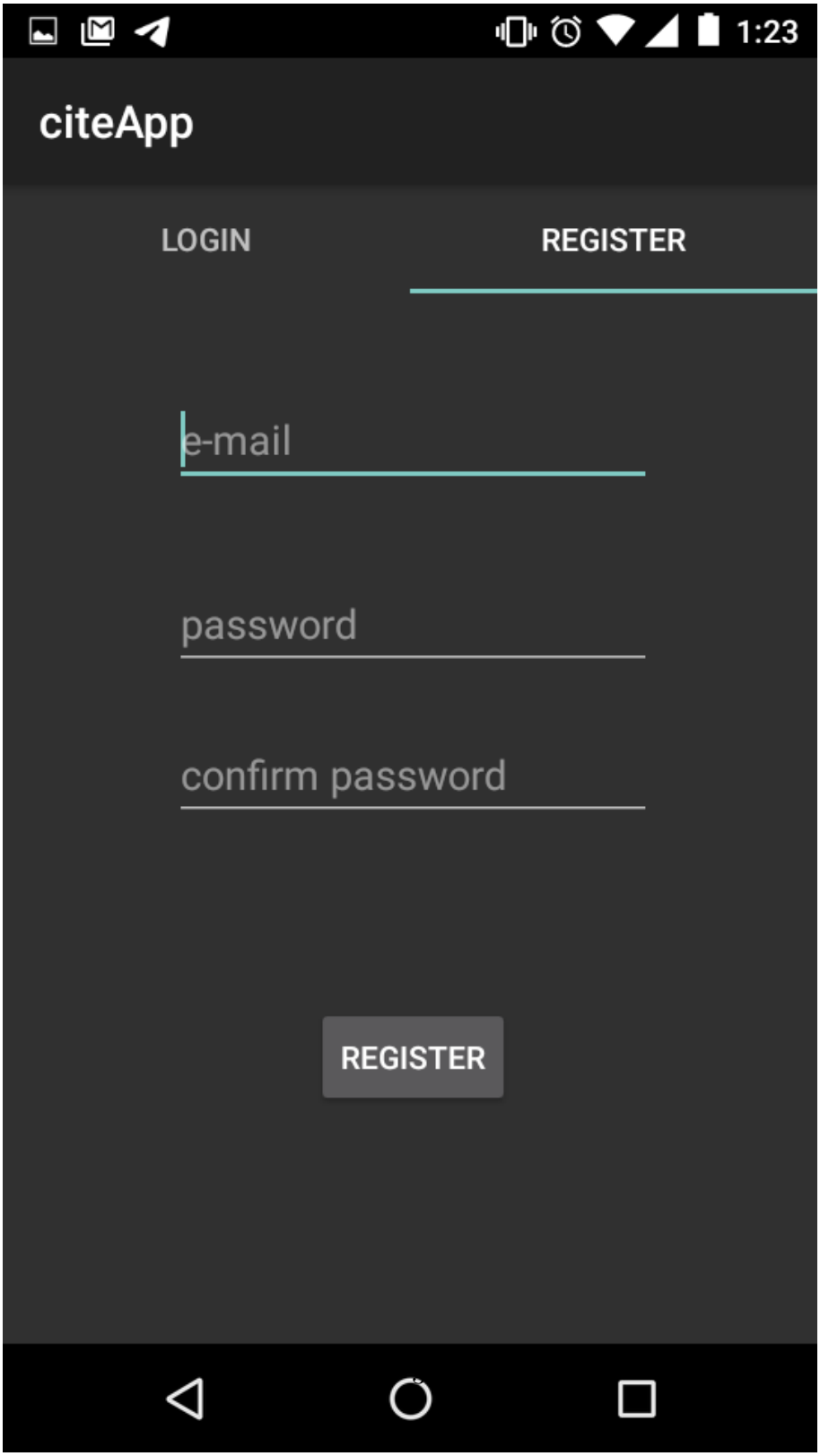


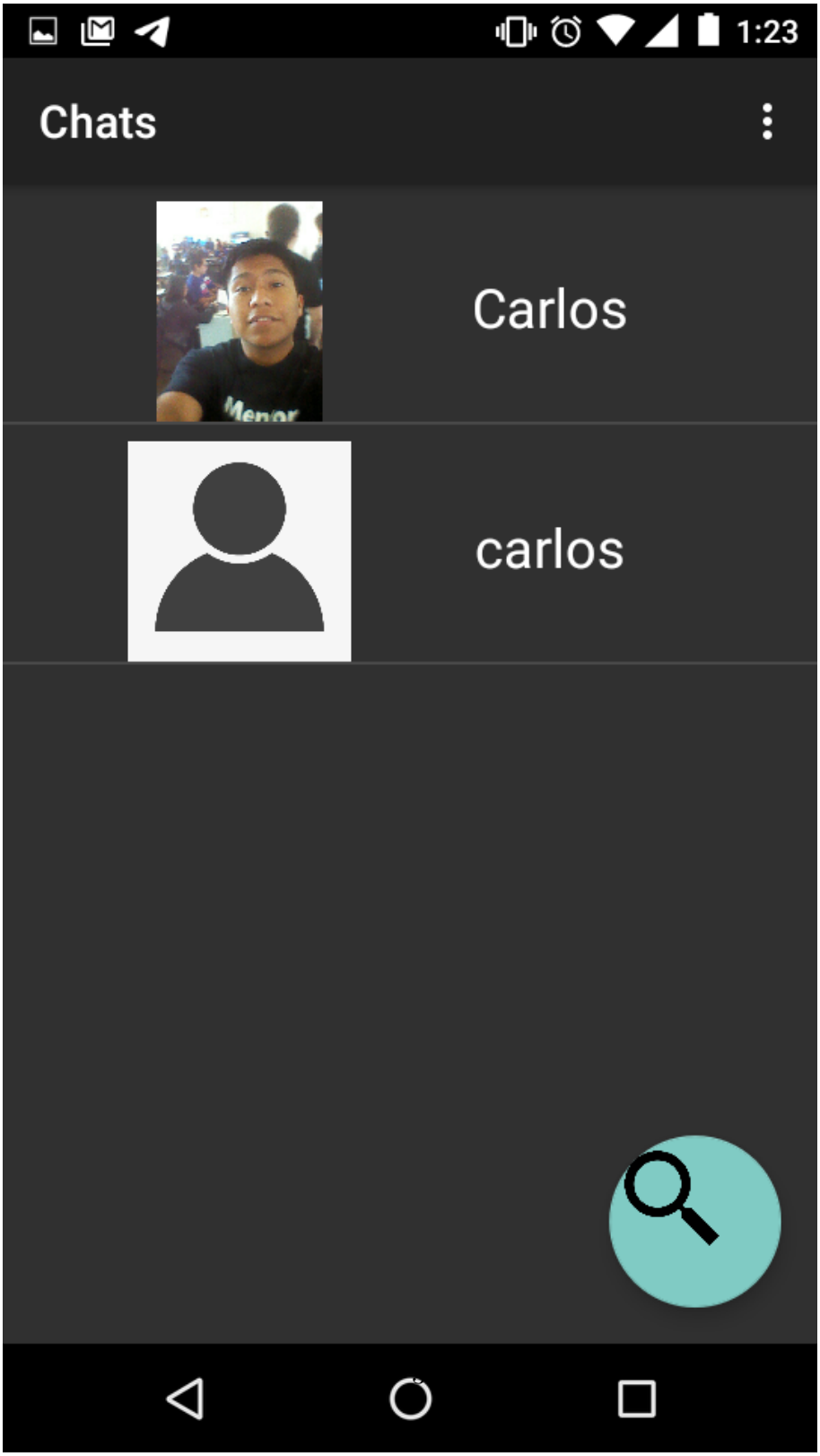
Diagrama de Despliegue:

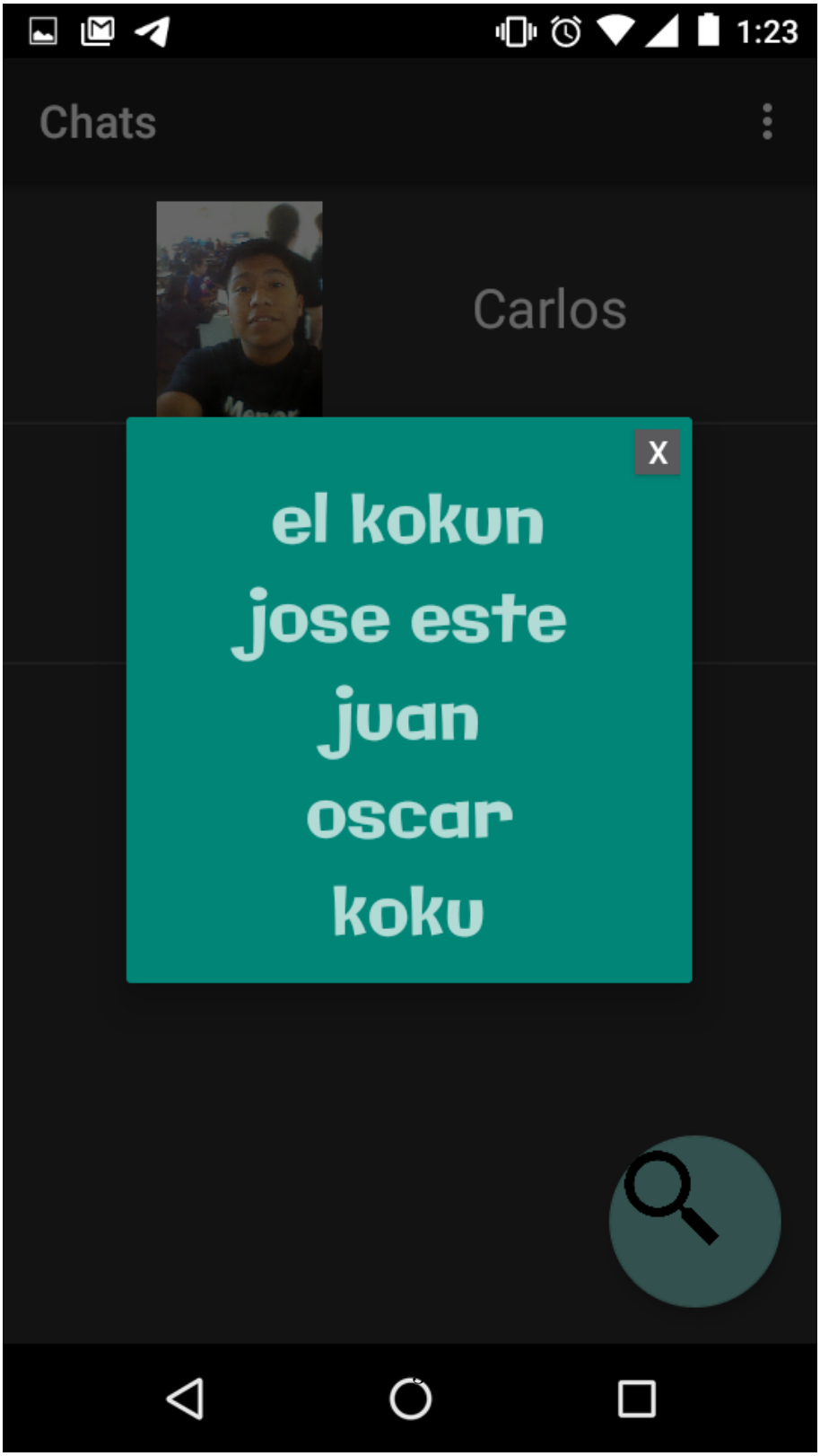


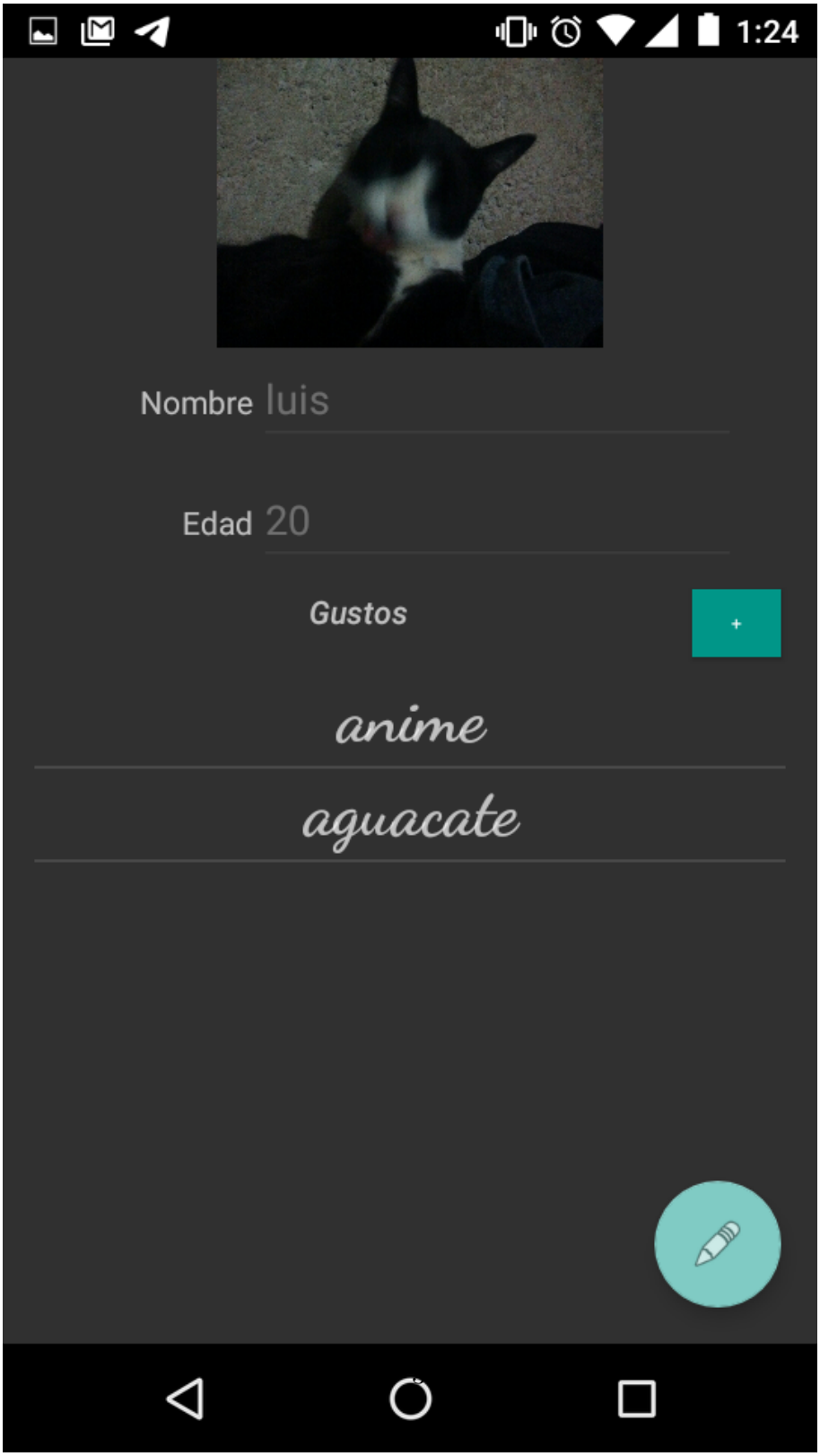
5. Capturas de Pantalla

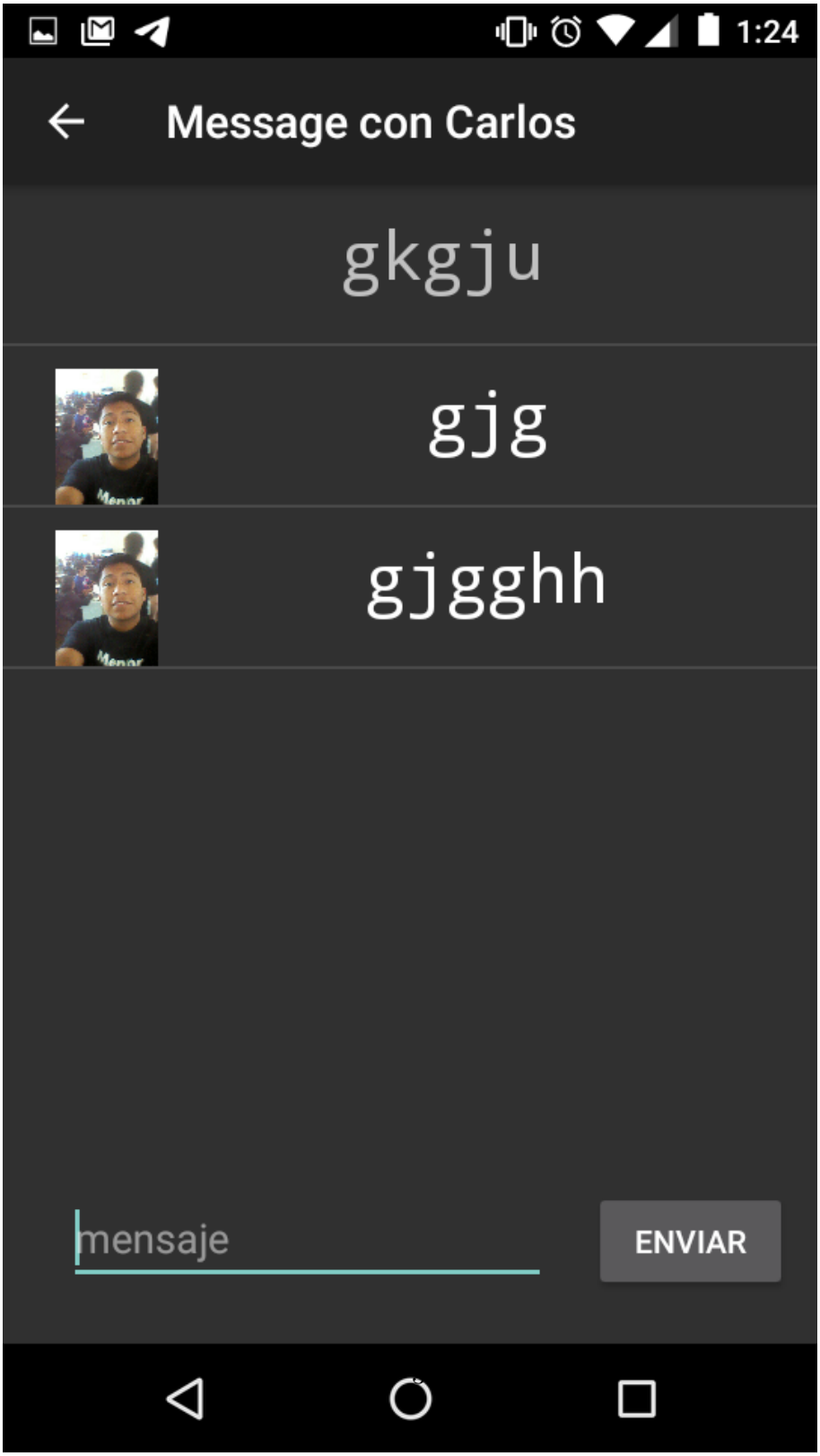



1:23
citeApp
LOGIN
REGISTER
e-mail
password
confirm password
REGISTER


1:23
Chats
Carlos
carlos


1:23
Chats
Carlos
X
el kokun
jose este
juan
oscar
koku


1:24
Nombre luis
Edad 20
Gustos
+
anime
aguacate


1:24
Message con Carlos
gkgju
gjg
gjgghh
mensaje
ENVIAR