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 los modelos de datos que utilizó para aplicación son súper importantes ya que sobre estos trabaja la aplicaion

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:

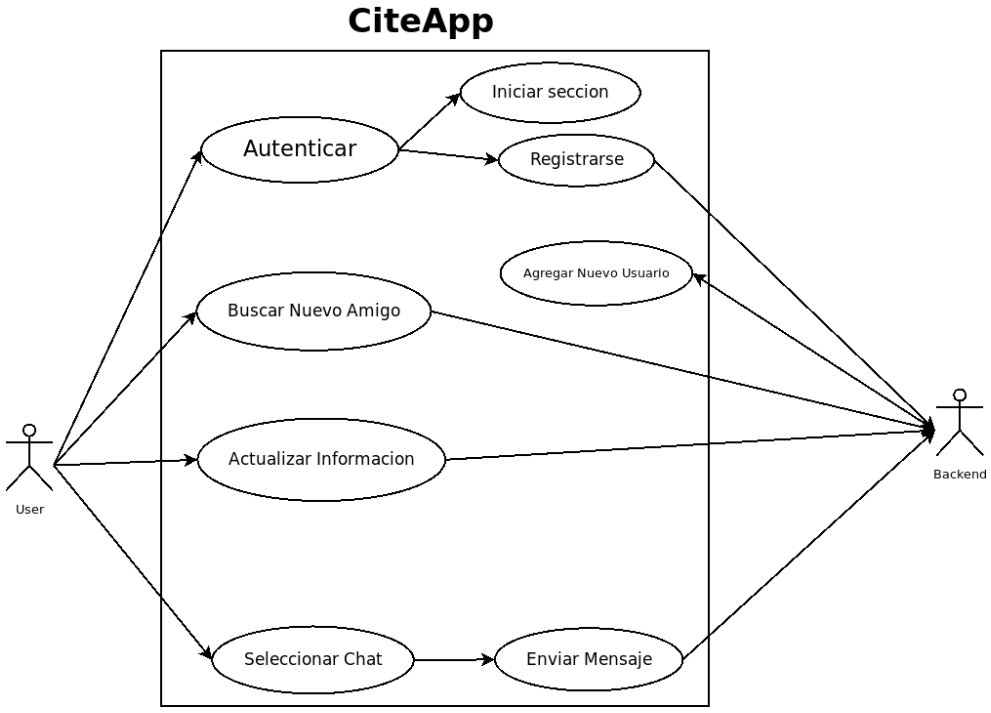
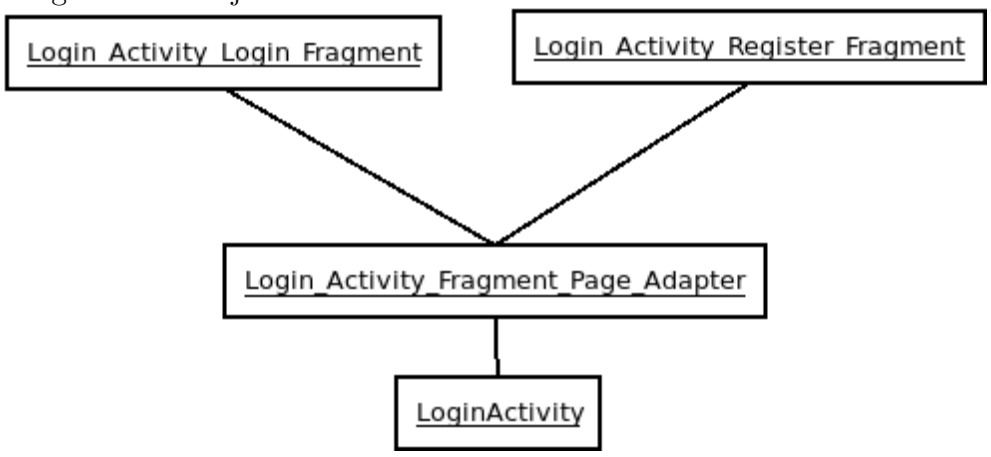
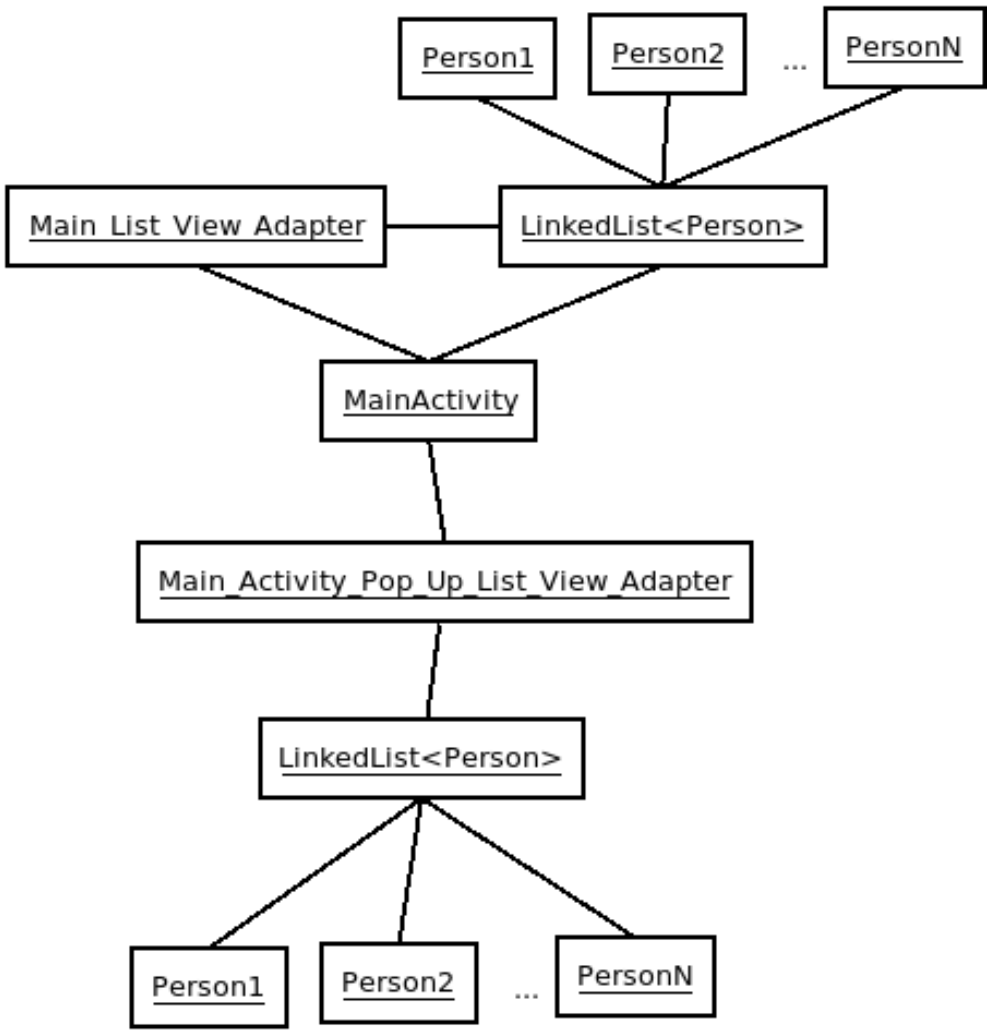


Diagrama de Objetos:




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

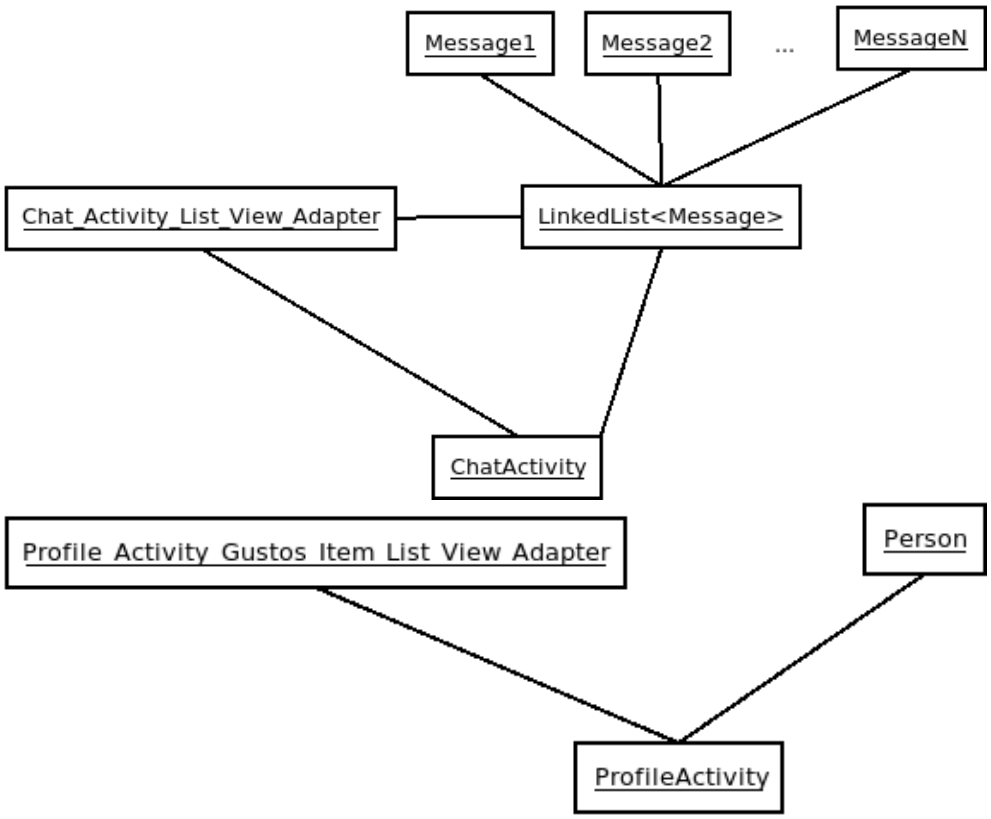


Diagrama de clases:

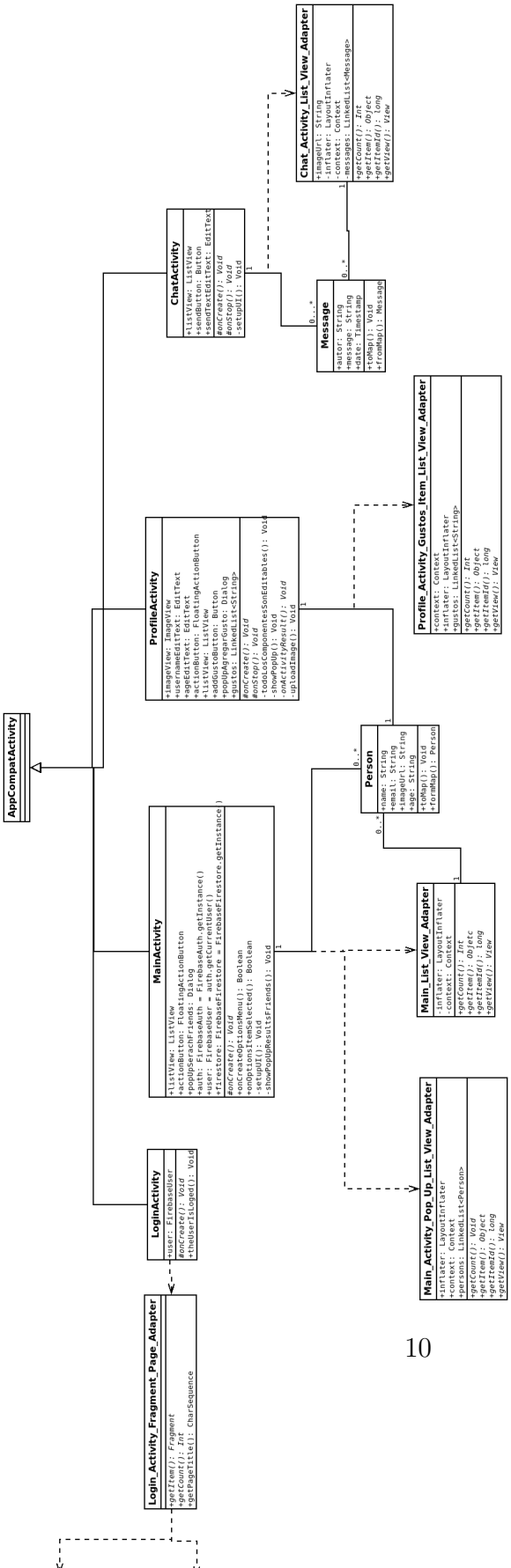

AppCompatActivity
LoginActivity
-user: FirebaseUser
#onCreate(): Void
+theUserIsLoged(): Void
MainActivity
+listView: ListView
+actionButton: FloatingActionButton
+popUpSeracnFriends: Dialog
+auth: FirebaseAuth = FirebaseAuth.getInstance()
+user: FirebaseUser = auth.getCurrentUser()
+access: FirebaseFirestore = FirebaseFirestore.getInstance()
#onCreate(): Void
+onCreateOptionsMenu(): Boolean
+onOptionsItemSelected(): Boolean
-setupUI(): Void
-showPopUpResultsFriends(): 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:

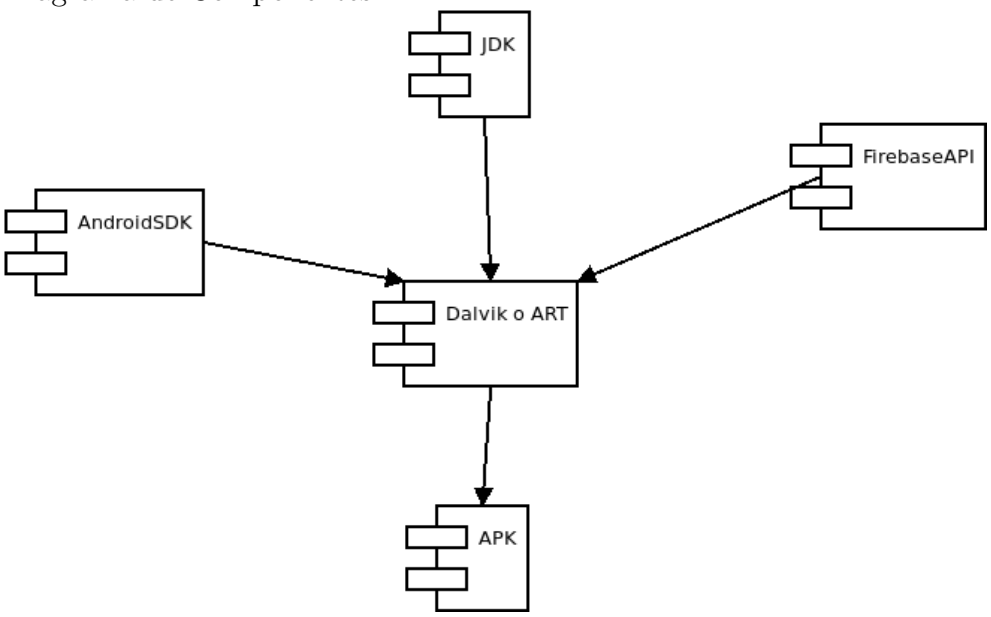
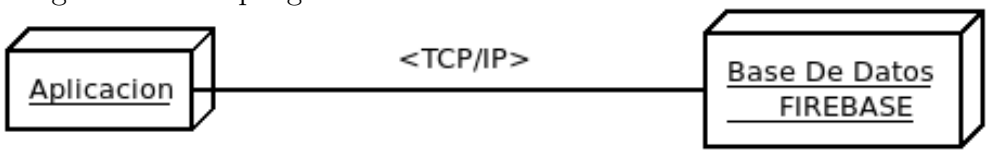
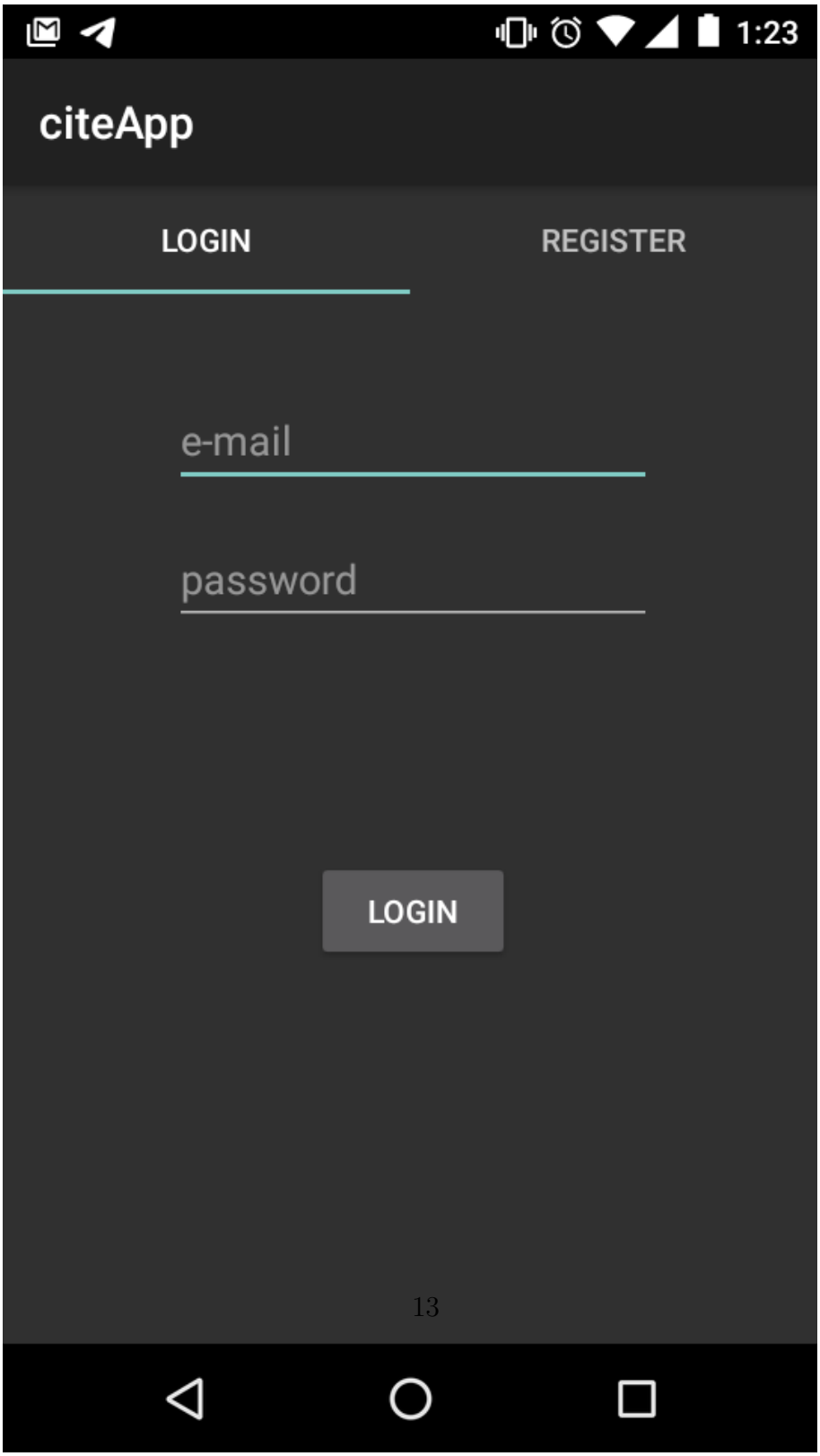
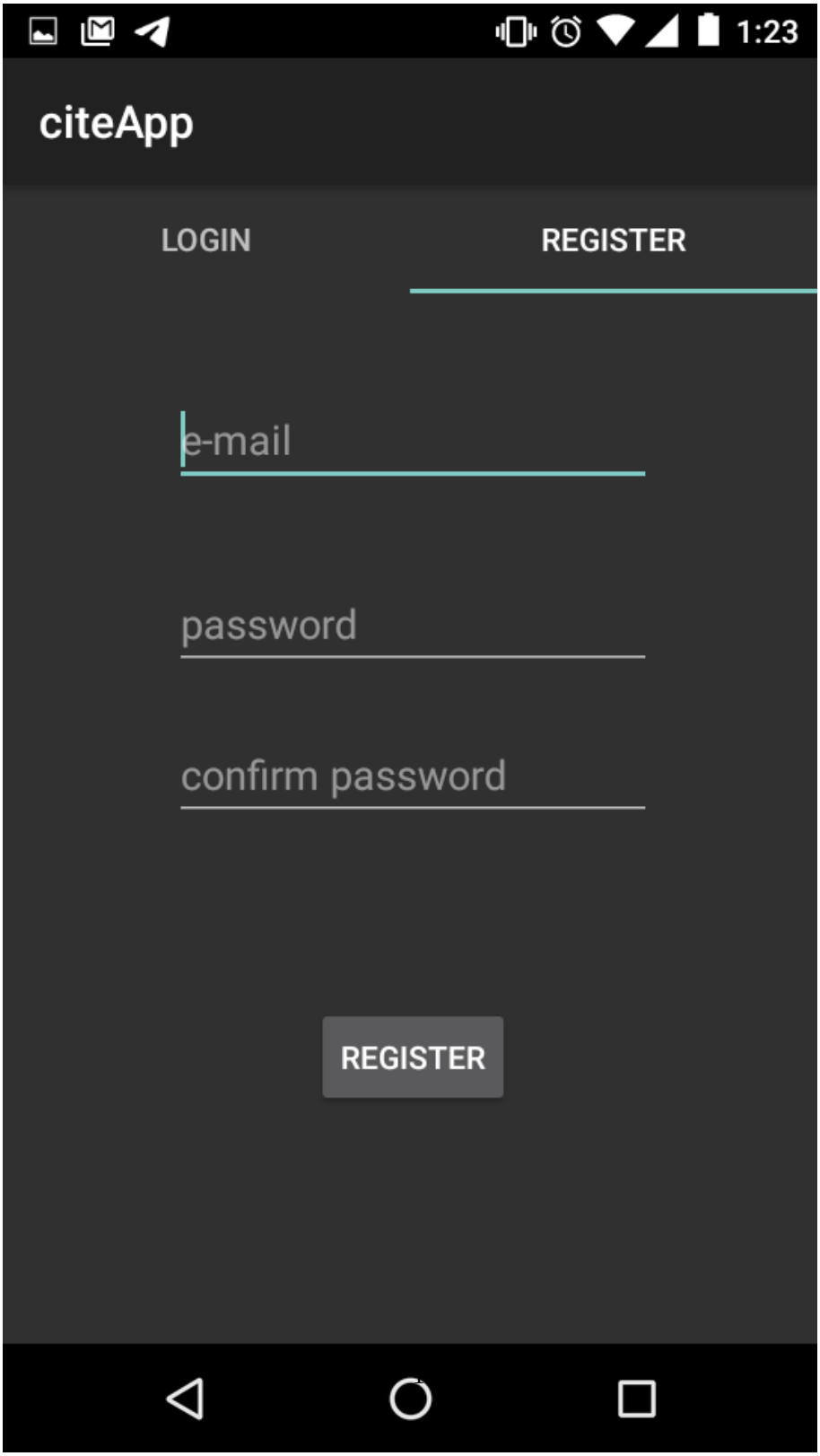


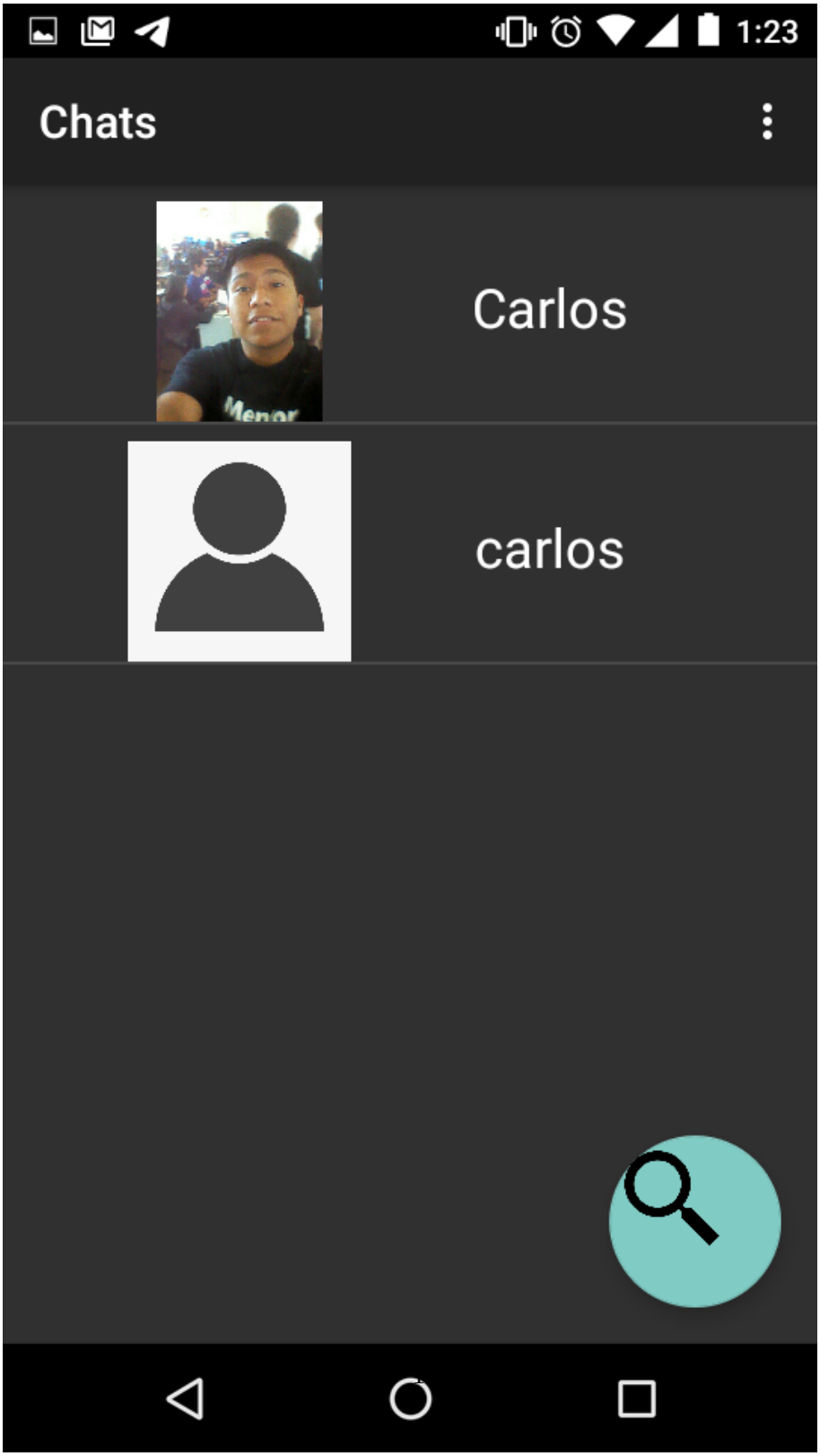
Diagrama de Despliegue:

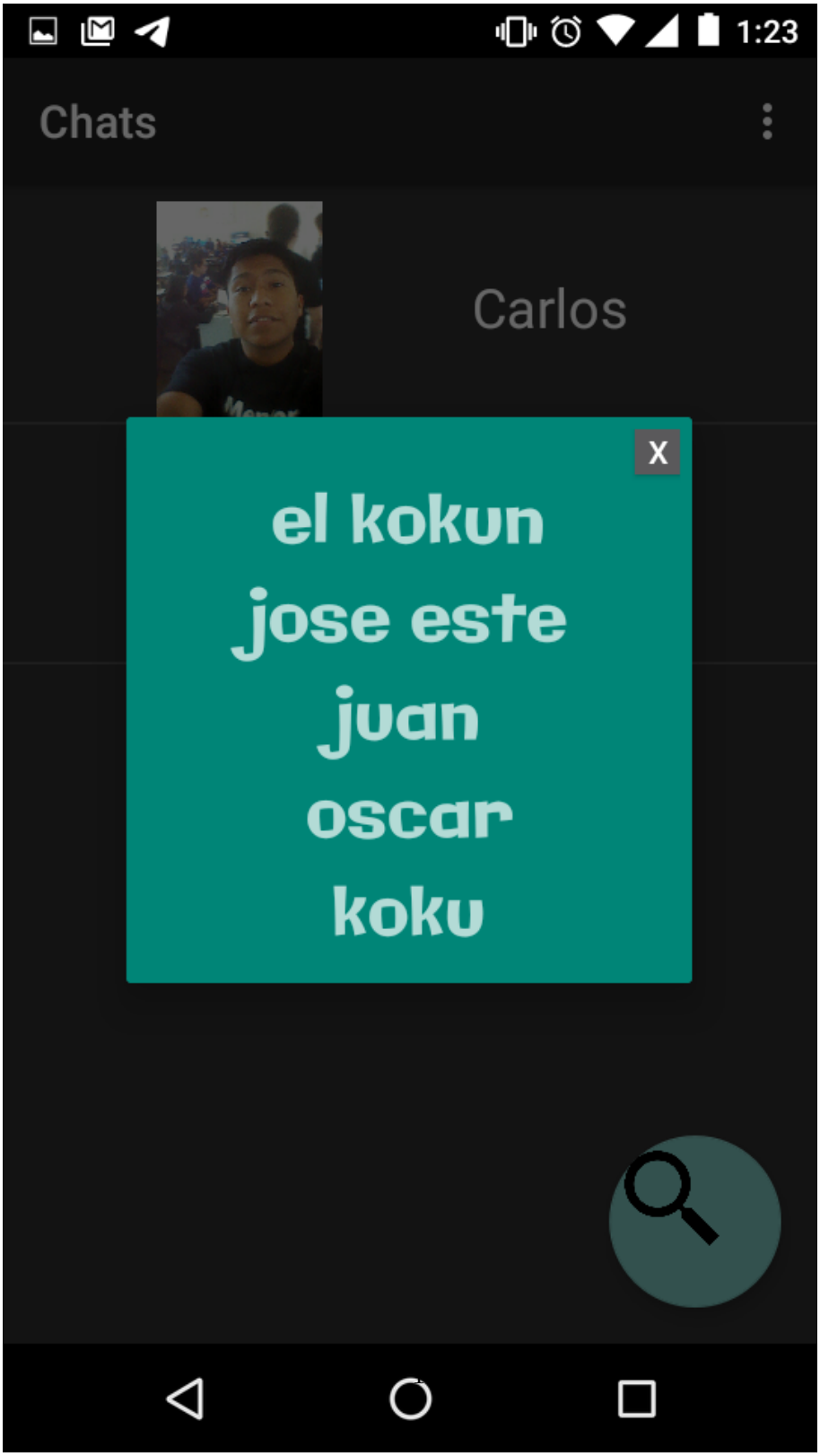


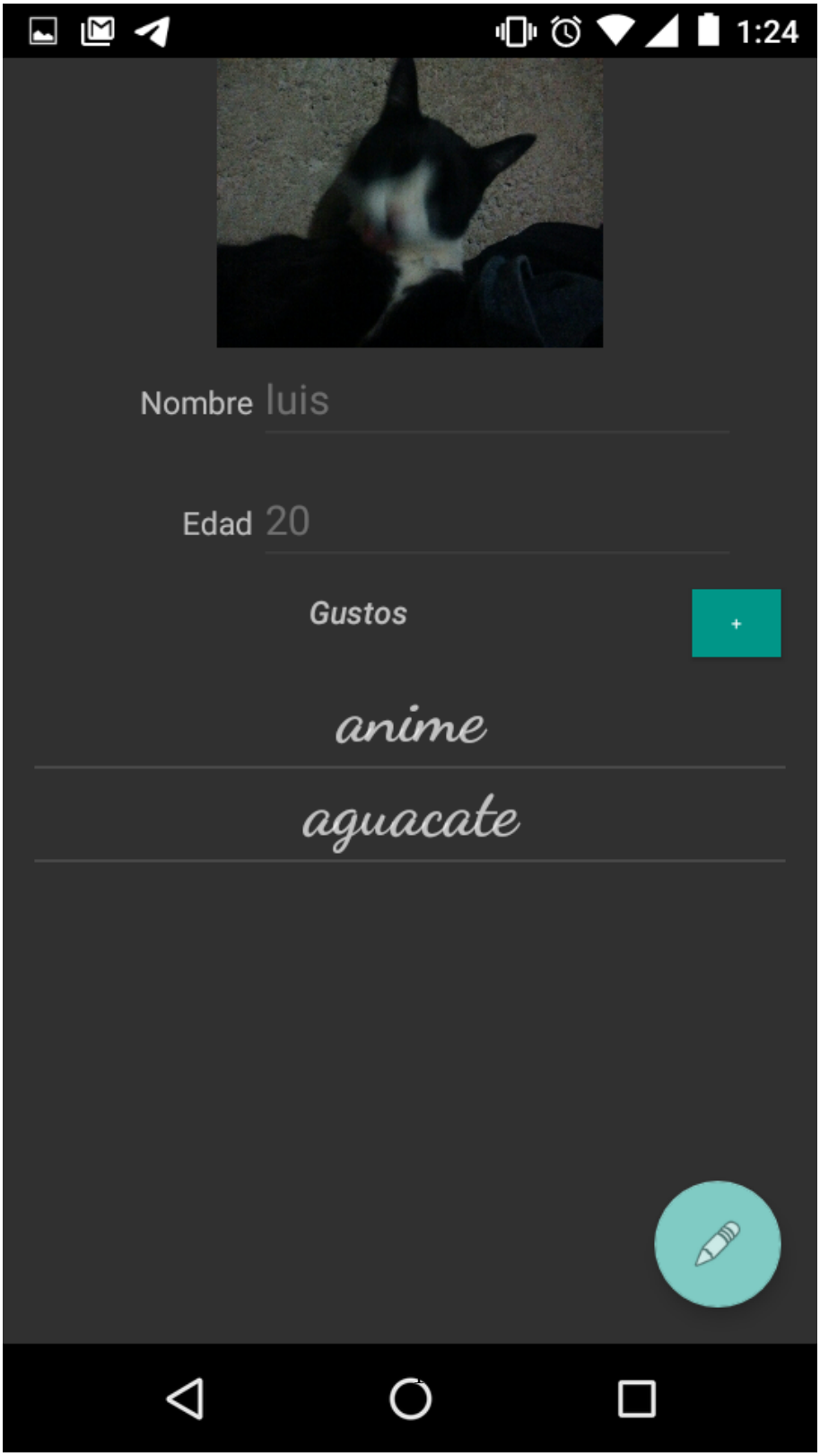
5. Capturas de Pantalla

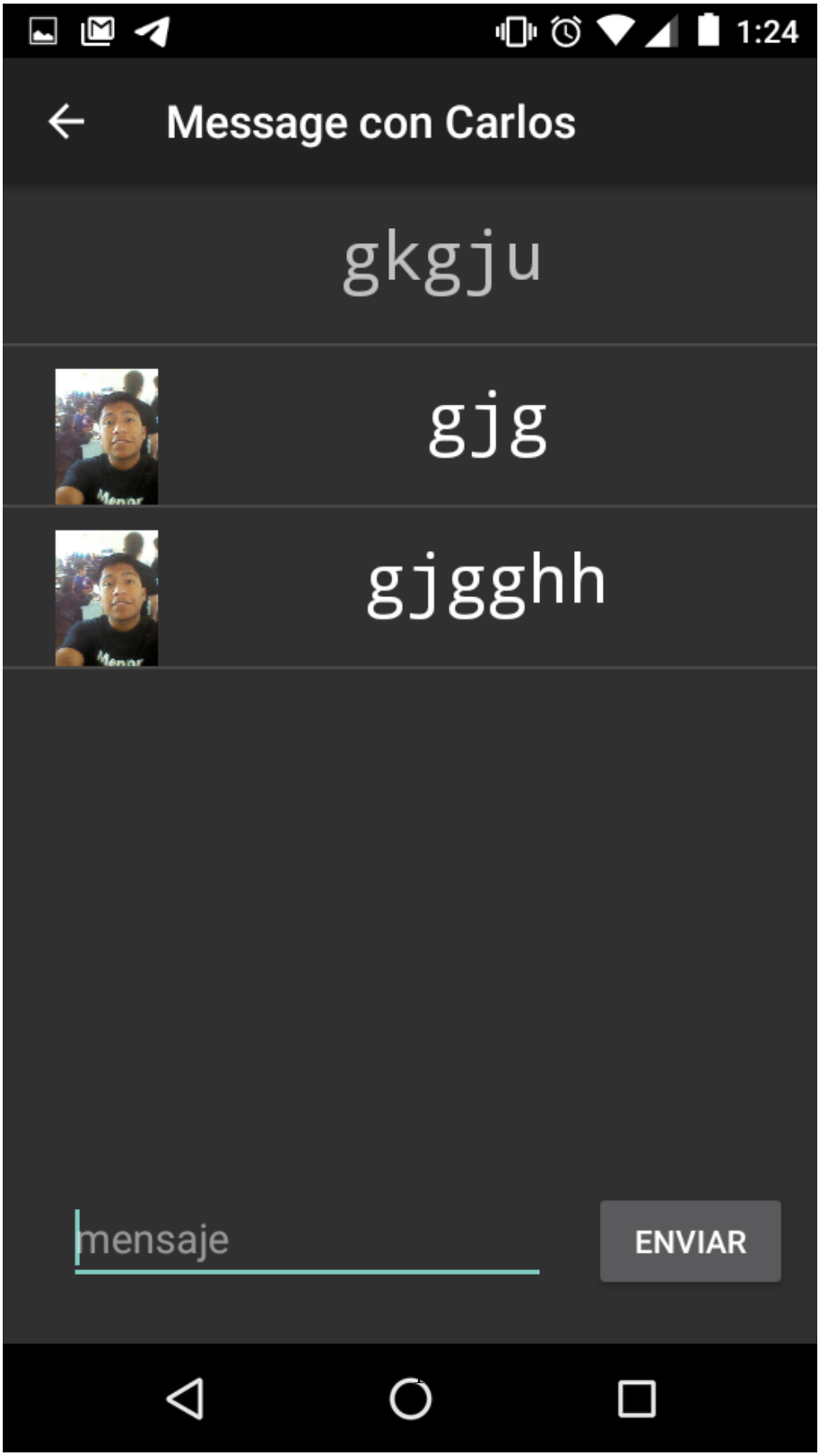



1:23
citeApp
LOGIN
REGISTER
e-mail
password
confirm password
REGISTER


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