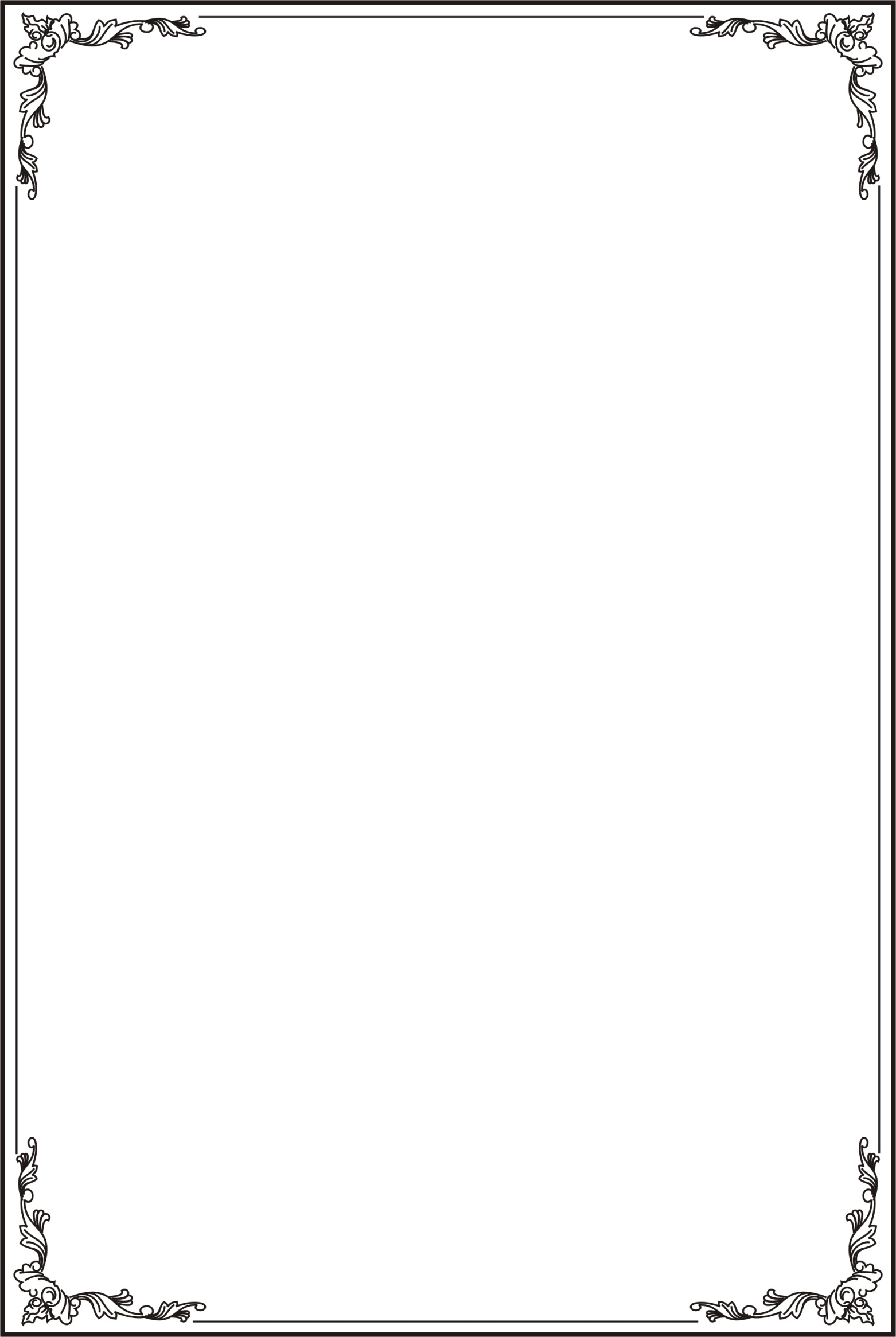
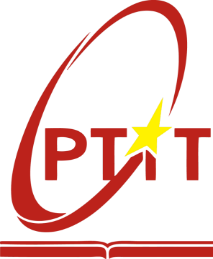
****BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

~~~~~~\*~~~~~~

**BÁO CÁO CHUYÊN ĐỀ  
AN TOÀN PHẦN MỀM**

***Đề tài:***

**XÂY DỰNG ỨNG DỤNG BẢO VỆ NỘI DUNG TIN NHẮN DỰA TRÊN MÃ HÓA**

**GVHD: TS. Huỳnh Trọng Thưa**

**Thực hiện: Nguyễn Hoàng Nam N14DCAT053**

**Cao Anh Hào N14DCAT080**

**Lớp: D14CQAT01-N**

**Khóa: 2014 – 2019**

**Hệ: Đại học chính quy**

**TPHCM, 11-2018**

**MỤC LỤC**

[**CHƯƠNG 1: ĐẶC TẢ CHƯƠNG TRÌNH 1**](#_Toc530660006)

[**1.1 Tổng quan về giao diện của ứng dụng 1**](#_Toc530660007)

[**CHƯƠNG 2: THIẾT KẾ CHƯƠNG TRÌNH 5**](#_Toc530660008)

[**2.1 Mô hình của ứng dụng 5**](#_Toc530660009)

[**2.2 Giải thuật 5**](#_Toc530660010)

[**CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH 7**](#_Toc530660011)

[**3.1 Chuẩn bị 7**](#_Toc530660012)

[**3.2 Code 7**](#_Toc530660013)

[**CHƯƠNG 4: KẾT QUẢ 15**](#_Toc530660014)

[**4.1 Kết quả đạt được. 15**](#_Toc530660015)

|  |  |
| --- | --- |
| **Nguyễn Hoàng Nam** | **Cao Anh Hào** |
| Xấy dựng danh sách tin nhắn mới nhất | Xấy dựng phần đăng nhập |
| Xấy dựng danh sách tin nhắn chi tiết | Xấy dựng phần đăng ký mật khẩu |
| Xấy dựng hàm mã hóa, giải mã | Xấy dựng phần xóa mật khẩu |
| Xấy dựng tiến trình lắng nghe tin nhắn đến và gửi tin nhắn | Xấy dựng hàm xử lý cơ sở dữ liệu và hàm băm |

**Bảng chia nhiệm vụ**

**Mục lục hình ảnh**

[Hình 1: Giao diện đăng nhập 1](#_Toc530658529)

[Hình 2: Giao diện xóa mật khẩu 2](#_Toc530658530)

[Hình 3: Giao diện tạo mật khẩu 2](#_Toc530658531)

[Hình 4: Giao diện tin nhắn gần nhất 3](#_Toc530658532)

[Hình 5: Giao diện tin nhắn mới 3](#_Toc530658533)

[Hình 6: Giao diện chi tiết tin nhắn 4](#_Toc530658534)

[Hình 7: Sơ đồ hoạt động của ứng dụng 5](#_Toc530658535)

[Hình 8: Danh sách tin nhắn của một số diện thoại 15](#_Toc530658536)

[Hình 9: Giãi mã tin từng tin nhắn 16](#_Toc530658537)

[Hình 10: Thông báo khi tin nhắn không thể giải mã 16](#_Toc530658538)

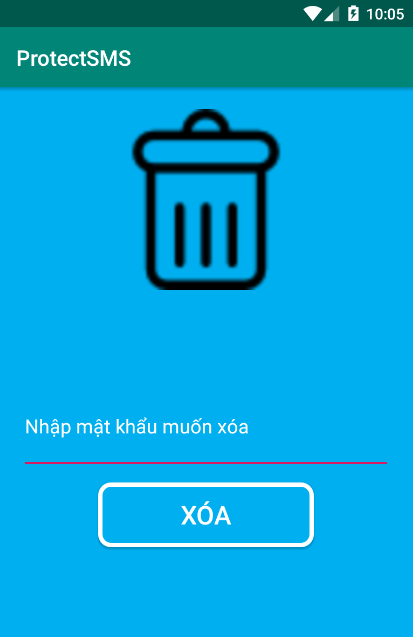
[Hình 11: Giải mã toàn bộ tin nhắn 17](#_Toc530658539)

# CHƯƠNG 1: ĐẶC TẢ CHƯƠNG TRÌNH

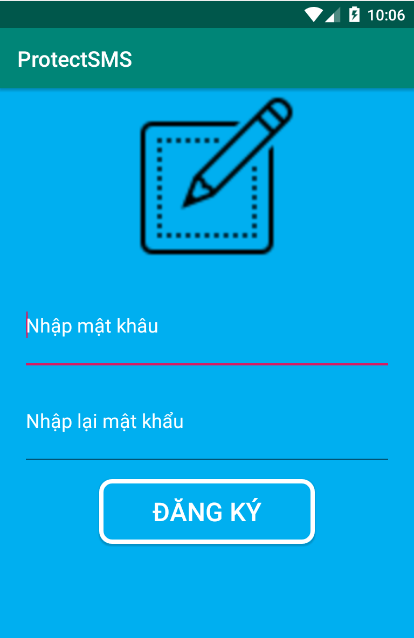
## **1.1 Tổng quan về giao diện của ứng dụng**



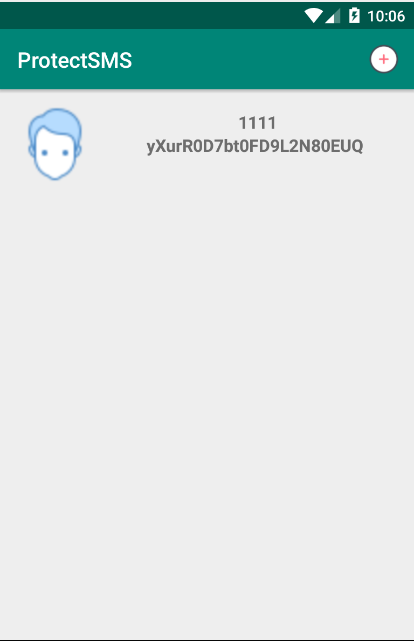
Hình 1: Giao diện đăng nhập



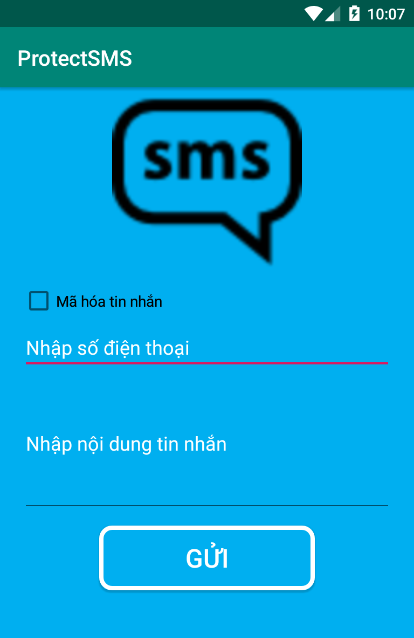
Hình 2: Giao diện xóa mật khẩu



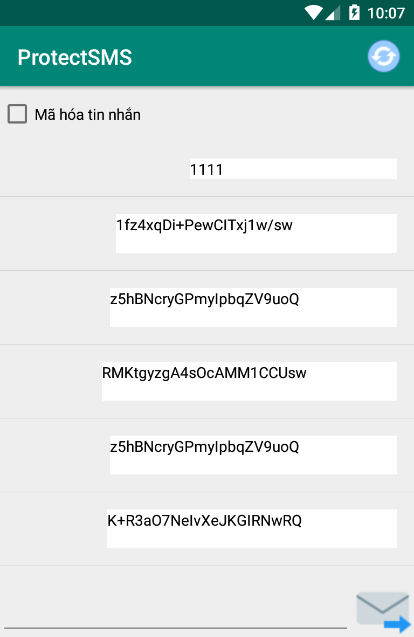
Hình 3: Giao diện tạo mật khẩu



Hình 4: Giao diện tin nhắn gần nhất



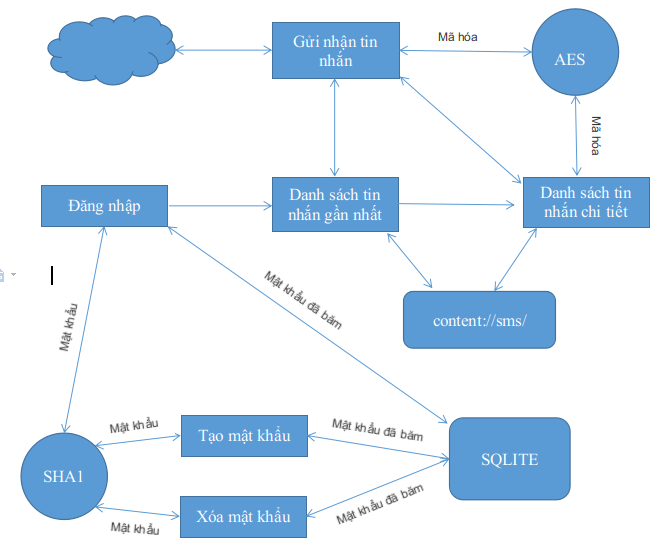
Hình 5: Giao diện tin nhắn mới



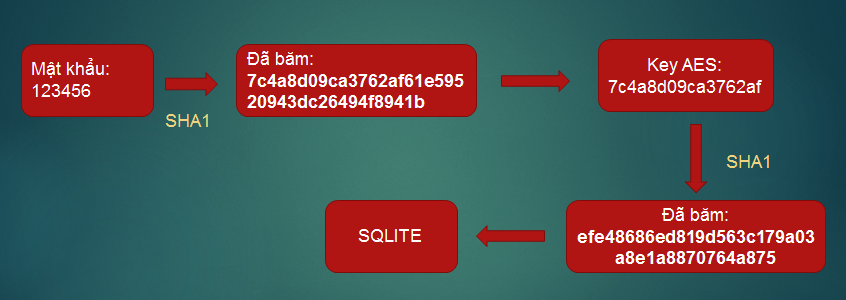
Hình 6: Giao diện chi tiết tin nhắn

# CHƯƠNG 2: THIẾT KẾ CHƯƠNG TRÌNH

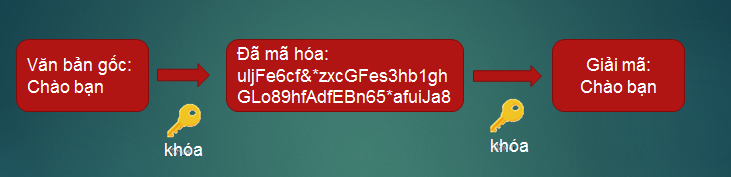
1. **Mô hình của ứng dụng**



Hình 7: Sơ đồ hoạt động của ứng dụng



*Hình 8: Sơ đồ tạo key mã hóa*



*Hình 9: Sơ đồ mã hóa*

1. **Giải thuật**

* Tạo mật khẩu:

Mật khẩu được băm bằng thuật toán SHA1 – Lấy 16 ký tự đầu băm thêm lần nữa – Lấy mã băm lưu vào database

* Xuất mật khẩu:

Mật khẩu được băm bằng thuật toán SHA1 – Lấy 16 ký tự đầu băm thêm lần nữa – Lấy mã băm để so sánh và xóa trong database

* Đăng nhập:

Mật khẩu được băm bằng thuật toán SHA1 - Lấy 16 ký tự đầu làm key cho thuật toán AES – Băm 16 ký tự trên và so sánnh với mã băm trong database – Nếu đúng chuyển đến Danh sách tin nhắn gần nhất

* Danh sách tin nhắn gần nhất:

Hàm lấy tin nhắn sẽ lọc tin nhắn gần nhất của từng số để hiển thị -- Nếu nhấn vào tin nhắn của số nào thi sẽ chuyển đến danh sách tin nhắn chi tiết của thằng số điên thoại -- Bên trên có thêm chức năng tạo tin nhắn mới.

* Danh sách tin nhắn chi tiết:

Hàm lấy tin nhắn sẽ lọc tất tin nhắn của số điện thoại được chọn – bấm vào từng tin nhắn để giải mã hoặc nhấn vào nút bên trên để giải mã toàn bộ -- Nhập tin nhắn muốn gửi ở bên dưới cùng với chọn phần tick phía trên để chọn gửi tin nhắn mã hóa hay không.

* Tạo tin nhắn mới:

Nhập tin nhắn muốn gửi cùng với chọn phần tick phía trên để chọn gửi tin nhắn mã hóa hay không.

# CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH

1. **Chuẩn bị**

Công cụ : Android studio 3.2.1

Ngôn ngữ : Java

1. **Code**

* Hàm mã hóa và giải mã theo thuật toán AES

public class AES {

public static String Encode(String pesan, String key){

try {

SecretKeySpec SK = new SecretKeySpec(key.getBytes(), "AES");

Cipher cipher = Cipher.getInstance("AES");

cipher.init(Cipher.ENCRYPT\_MODE, SK);

byte[] encrypted = cipher.doFinal(pesan.getBytes());

return Base64.encodeToString(encrypted, Base64.NO\_PADDING);

} catch (Exception e) {

return "ERROR:" + e.getMessage();

}

}

public static String Decode(String pesan, String key){

try {

SecretKeySpec SK = new SecretKeySpec(key.getBytes(), "AES");

Cipher cipher = Cipher.getInstance("AES");

cipher.init(Cipher.DECRYPT\_MODE, SK);

byte[] decrypted = cipher.doFinal(Base64.decode(pesan, Base64.NO\_PADDING));

return new String(decrypted);

} catch (Exception e) { return "ERROR"; }

}

}

* Hàm băm mật khẩu bằng thuật toán SHA1

public class GetHash{

public static String getSha1Hex(String clearString){

try{

MessageDigest messageDigest = MessageDigest.getInstance("SHA-1");

messageDigest.update(clearString.getBytes("UTF-8"));

byte[] bytes = messageDigest.digest();

StringBuilder buffer = new StringBuilder();

for (byte b : bytes) {

buffer.append(Integer.toString((b & 0xff) + 0x100, 16).substring(1));

}

return buffer.toString();

}

catch (Exception ignored){

ignored.printStackTrace();

return null;

}

}

public static String get16Characters(String hash){

String upToNCharacters = hash.substring(0, Math.min(hash.length(), 16));

return upToNCharacters;

}

}

}

* Xử lý đăng nhập

button\_confirm.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String password = edit\_password.getText().toString();

if(password.matches("")){

String mess = "Bạn chưa nhập đủ thông tin..!";

Error(context,mess);

}else{

String hash\_first = gethash.getSha1Hex(password);

String password\_real = gethash.get16Characters(hash\_first);

String hash = gethash.getSha1Hex(password\_real);

boolean check = db.CheckHash(hash);

if(check == false){

String mess = "Mật khẩu không đúng..!";

Error(context,mess);

edit\_password.setText("");

} else {

globals.setPASSWORD(password\_real);

globals.setCOUNT(0);

globals.setFLAG(false);

Intent intent = new Intent(MainActivity.this, ListSMSActivity.class);

startActivity(intent);

}

}

}

});}

* Xử lý tạo mật khẩu mới

button\_registration.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String password = edit\_password.getText().toString();

String retype = edit\_retype.getText().toString();

Context context = CreatePasswordActivity.this;

GetHash gethash = new GetHash();

if(password.matches("") || retype.matches("")){

//Toast.makeText(SendActivity.this, "Ban chua nhap du thong tin", Toast.LENGTH\_SHORT).show();

String mess = "Bạn chưa nhập đủ thông tin..!";

Error(context,mess);

}

else {

if(!password.matches(retype)){

String mess = "Mật khẩu nhập lại không trùng khớp..!";

Error(context,mess); }

else{

String hash\_first = gethash.getSha1Hex(password);

String password\_real = gethash.get16Characters(hash\_first);

String hash = gethash.getSha1Hex(password\_real);

db.addHash(hash);

Toast.makeText(CreatePasswordActivity.this, "Tạo mật khẩu thành công..!", Toast.LENGTH\_SHORT).show();

Intent intent = new Intent(CreatePasswordActivity.this, MainActivity.class);

startActivity(intent);

}

}

}

});

* Xử lý xóa mật khẩu

button\_delete.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String pass = edit\_pass.getText().toString();

GetHash gethash = new GetHash();

if(pass.matches("")) {

String mess = "Bạn chưa nhập đủ thông tin..!";

Error(DeleteActivity.this,mess);

}

else{

String hash\_first = gethash.getSha1Hex(pass);

String password\_real = gethash.get16Characters(hash\_first);

String hash = gethash.getSha1Hex(password\_real);

int check = db.deleteHash(hash);

if(check == 1){

Toast.makeText(getBaseContext(), "Mật khẩu đã được xóa..!", Toast.LENGTH\_LONG).show();

Intent intent = new Intent(DeleteActivity.this, MainActivity.class);

startActivity(intent);

}

else {

String mess = "Mật khẩu không tồn tại..!";

Error(DeleteActivity.this,mess);

}

}

}

});

* Hàm lấy tất cả tin nhắn sms

public void getAllSMS(Globals globals) {

synchronized (globals) {

ArrayList<Message> ListSMS = new ArrayList<>();

globals.RemoveListSMS(globals.ListSMS);

Uri uriSMSURI = Uri.parse("content://sms/");

Cursor cur = getBaseContext().getContentResolver().query(uriSMSURI, null, null, null, null);

while (cur != null && cur.moveToNext()) {

String number\_phone = cur.getString(cur.getColumnIndex("address"));

String content = cur.getString(cur.getColumnIndexOrThrow("body"));

int type = Integer.parseInt(cur.getString(cur.getColumnIndexOrThrow("type")));

Message message = new Message(number\_phone, content, type);

ListSMS.add(message);

}

globals.setListSMS(ListSMS);

if (cur != null) {

cur.close();

}

}

}

* Hàm gửi tin nhắn

public boolean kirimSMS(String noHP, String pesan){

String SENT = "SMS\_SENT";

String DELIVERED = "SMS\_DELIVERED";

PendingIntent sentPI = PendingIntent.getBroadcast(this, 0, new Intent(SENT), 0);

PendingIntent deliveredPI = PendingIntent.getBroadcast(this, 0, new Intent(DELIVERED), 0);

android.telephony.SmsManager sms = android.telephony.SmsManager.getDefault();

sms.sendTextMessage(noHP, null, pesan, sentPI, deliveredPI);

return true;

}

* Hàm lắng nghe tin nhắn nhận được

public class SmsListener extends BroadcastReceiver {

private SharedPreferences preferences;

@Override

public void onReceive(Context context, Intent intent) {

// TODO Auto-generated method stub

if(intent.getAction().equals("android.provider.Telephony.SMS\_RECEIVED")){

Bundle bundle = intent.getExtras(); //---get the SMS message passed in---

SmsMessage[] msgs = null;

String msg\_from;

if (bundle != null){

//---retrieve the SMS message received---

try{

Object[] pdus = (Object[]) bundle.get("pdus");

msgs = new SmsMessage[pdus.length];

for(int i=0; i<msgs.length; i++){

msgs[i] = SmsMessage.createFromPdu((byte[])pdus[i]);

msg\_from = msgs[i].getOriginatingAddress();

String msgBody = msgs[i].getMessageBody();

if(MainActivity.customMessAdapter != null && MainActivity.listShowMess != null) {

if(msg\_from.matches(MainActivity.listShowMess.get(0).getNumber\_phone())) {

Message message = new Message(msg\_from, msgBody, 1);

MainActivity.listShowMess.add(message);

MainActivity.customMessAdapter.notifyDataSetChanged(); }}

if(MainActivity.customAdapter != null && MainActivity.listShowSMS !=null){

boolean flag=false;

for(int j=0 ; j< MainActivity.listShowSMS.size();j++) {

if(msg\_from.matches(MainActivity.listShowSMS.get(j).getNumber\_phone())){

MainActivity.listShowSMS.get(j).setMessage(msgBody);

MainActivity.customAdapter.notifyDataSetChanged();

flag = true;}}

if(!flag){

SMS sms = new SMS(msg\_from,msgBody);

MainActivity.listShowSMS.add(0,sms);

MainActivity.customAdapter.notifyDataSetChanged(); }}}

}catch(Exception e){

// Log.d("Exception caught",e.getMessage())}

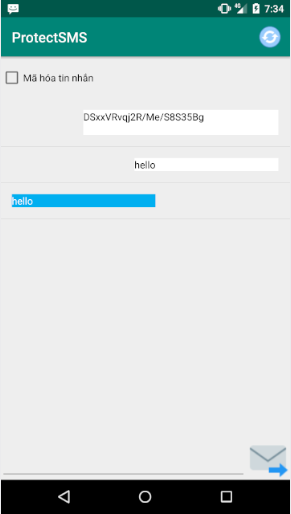
}

}

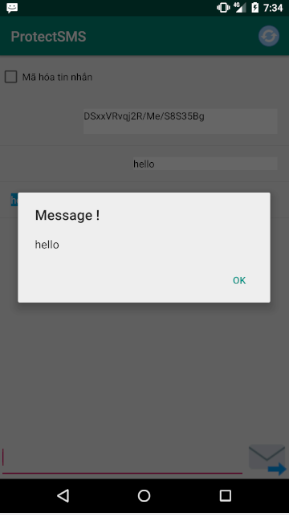
}

# CHƯƠNG 4: KẾT QUẢ

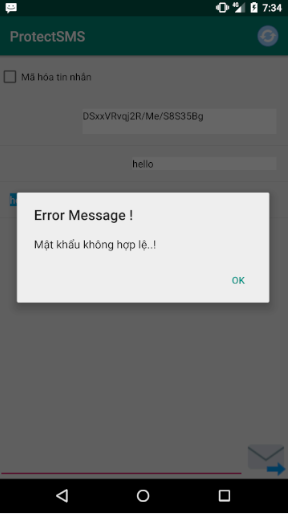
## **4.1 Kết quả đạt được.**



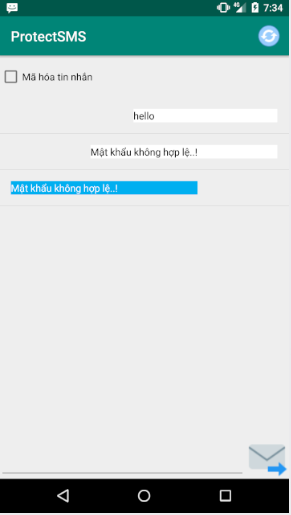
Hình 10: Danh sách tin nhắn của một số diện thoại



Hình 11: Giãi mã tin từng tin nhắn



Hình 12: Thông báo khi tin nhắn không thể giải mã



Hình 13: Giải mã toàn bộ tin nhắn