



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN

Đồ án thực hành
AN NINH MÁY TÍNH

BÁO CÁO

ĐỒ ÁN 1

Giảng viên hướng dẫn:

Lê Hà Minh
Lê Giang Thanh

Tên thành viên - MSSV:

Phạm Quốc Toàn - 19127585

Trần Hữu Trí - 19127596

I. Mục lục

Mục lục	1
I. Thông tin thành viên:	2
II. Phân công đánh giá:	2
III. Các chức năng đã thực hiện được:	3

I. Thông tin thành viên:

Họ tên	MSSV	Email
Phạm Quốc Toàn	19127585	19127585@student.hcmus.edu.vn
Trần Hữu Trí	19127596	19127596@student.hcmus.edu.vn

II. Phân công đánh giá:

Tên công việc	Người thực hiện	Tỉ lệ hoàn thành
Chức năng 1,2,3	Trần Hữu Trí	100%
Chức năng 4,5,6,7	Phạm Quốc Toàn	100%

III. Các chức năng đã thực hiện được:

1. Đăng ký tài khoản người dùng:

1.1 Ứng dụng cho phép người dùng đăng ký 1 tài khoản với các thông tin: email (dùng làm định danh tài khoản), họ tên, ngày sinh, điện thoại, địa chỉ, mật khẩu (passphrase).

Register

Name

Email

Day of Birth

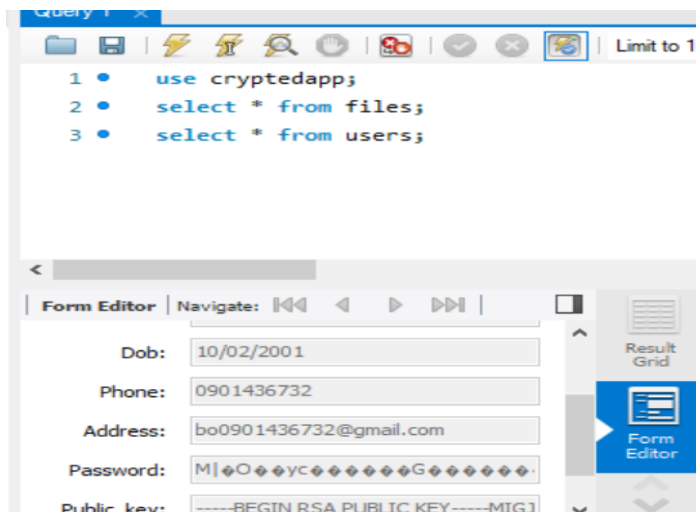
Phone number

Address

Password

Confirm Password

1.2 Mật khẩu cần được lưu trữ dưới dạng Hash có kết hợp với Salt. Thuật toán Hash là SHA256.



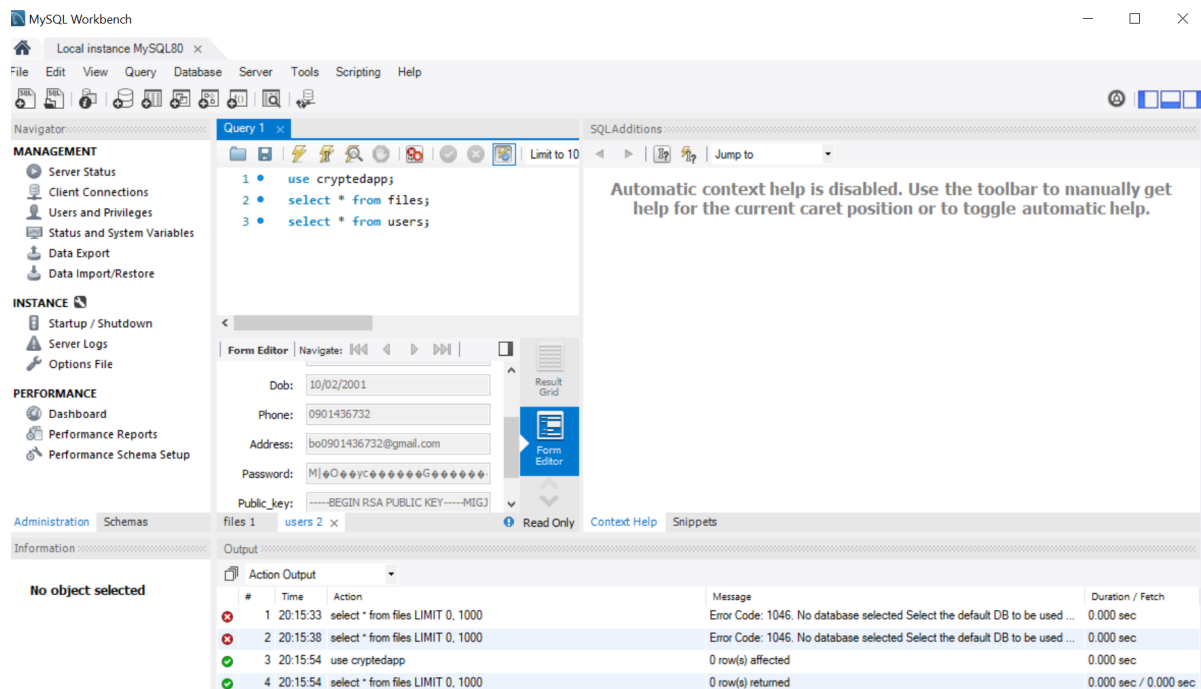
Đoạn code lưu trữ mật khẩu dưới dạng hash:

```
# User Register
@app.route('/register', methods=['GET', 'POST'])
def register():
    form = RegisterForm(request.form)
    if request.method == 'POST' and form.validate():
        name = form.name.data
        email = form.email.data
        dob = form.dob.data
        phone = form.phone.data
        address = form.address.data
        salt = os.urandom(32)

        password = hashlib.pbkdf2_hmac(
            'sha256', # The hash digest algorithm for HMAC
            form.password.data.encode('utf-8'), # Convert the password to bytes
            salt, # Provide the salt
            100000 # It is recommended to use at least 100,000 iterations of SHA-256
        )

        #storage both password and salt into database to decrypt later
        storage_password = salt+password
```

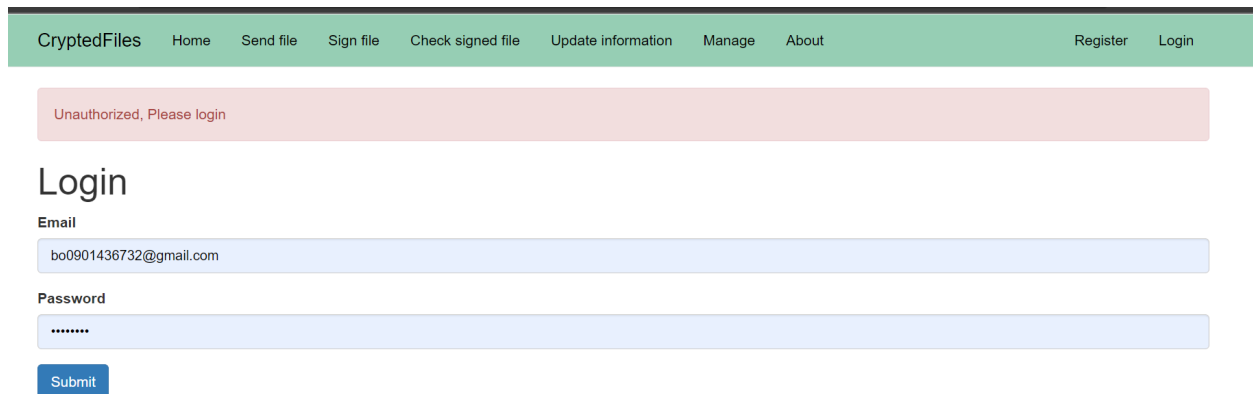
1.3 Ứng dụng có thể sử dụng CSDL SQL hoặc file XML, JSON để lưu trữ thông tin về người dùng



The screenshot displays the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar contains a Navigator pane with sections for MANAGEMENT (Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore), INSTANCE (Startup / Shutdown, Server Logs, Options File), and PERFORMANCE (Dashboard, Performance Reports, Performance Schema Setup). The main workspace is divided into three panes: a Query Editor on the left showing a query with three lines (using cryptedapp, selecting from files, and selecting from users), a SQL Additions pane on the right with a message about disabled context help, and a Form Editor in the center with fields for Dob (10/02/2001), Phone (0901436732), Address (bo0901436732@gmail.com), Password (masked), and Public_key (masked). Below the Form Editor is a tabbed interface with 'files 1' and 'users 2' tabs. The bottom pane is the Output window, showing an 'Action Output' table with columns for #, Time, Action, Message, and Duration / Fetch. The table contains four rows of log entries, with the first two showing 'Error Code: 1046. No database selected' and the last two showing '0 row(s) affected' and '0 row(s) returned'.

#	Time	Action	Message	Duration / Fetch
1	20:15:33	select * from files LIMIT 0, 1000	Error Code: 1046. No database selected Select the default DB to be used ...	0.000 sec
2	20:15:38	select * from files LIMIT 0, 1000	Error Code: 1046. No database selected Select the default DB to be used ...	0.000 sec
3	20:15:54	use cryptedapp	0 row(s) affected	0.000 sec
4	20:15:54	select * from files LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

1.4 Người dùng phải đăng nhập ứng dụng bằng email và passphrase trước khi sử dụng các tính năng tiếp theo sau đây.



CryptedFiles Home Send file Sign file Check signed file Update information Manage About Register Login

Unauthorized, Please login

Login

Email

bo0901436732@gmail.com

Password

Submit

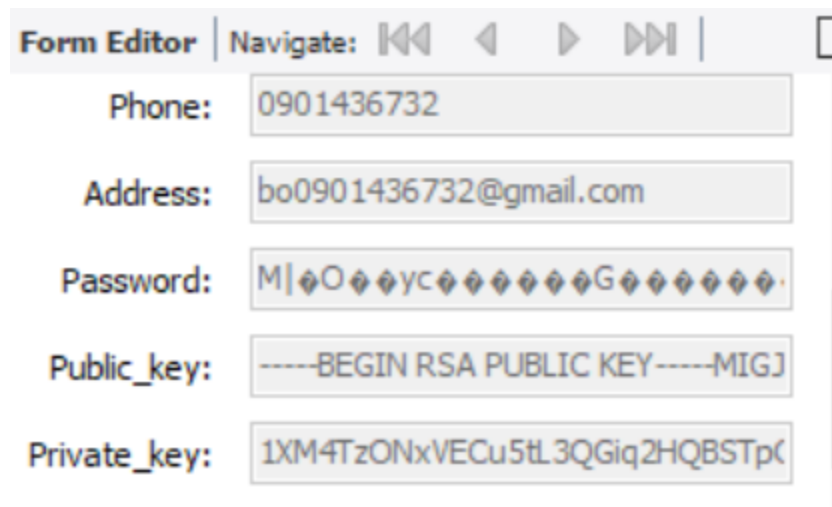
Khi người dùng chưa đăng nhập bằng email và passphrase thì bấm vào chức năng nào form cũng sẽ thông báo và chuyển hướng về trang login.

```
# Check if user logged in
def is_logged_in(f):
    @wraps(f)
    def wrap(*args, **kwargs):
        if 'logged_in' in session:
            return f(*args, **kwargs)
        else:
            flash('Unauthorized, Please login', 'danger')
            return redirect(url_for('login'))
    return wrap
```

2. Phát sinh cặp khoá bất đối xứng

2.1 Ứng dụng cho phép phát sinh một cặp khoá (K_{public} , $K_{private}$) có độ dài là 2048 bit cho thuật toán RSA tương ứng với mỗi người dùng.

Khi đăng nhập người dùng chỉ cần nhập các thông tin ở mục 1.1, sau khi đăng ký thành công thì khi lưu vào database chương trình sẽ tự động tạo Kprivate và Kpublic.



Form Editor | Navigate: ⏮ ⏪ ⏩ ⏭

Phone: 0901436732

Address: bo0901436732@gmail.com

Password: M|OycG

Public_key: -----BEGIN RSA PUBLIC KEY-----MIGJ

Private_key: 1XM4TzONxVECu5tL3QGiq2HQBSTpC

Đoạn mã phát sinh Kprivate và Kpublic:

```
# Create keys
(public_key1, private_key1) = rsa.newkeys(1024)
public_key = public_key1.save_pkcs1('PEM')
private_key = private_key1.save_pkcs1('PEM')

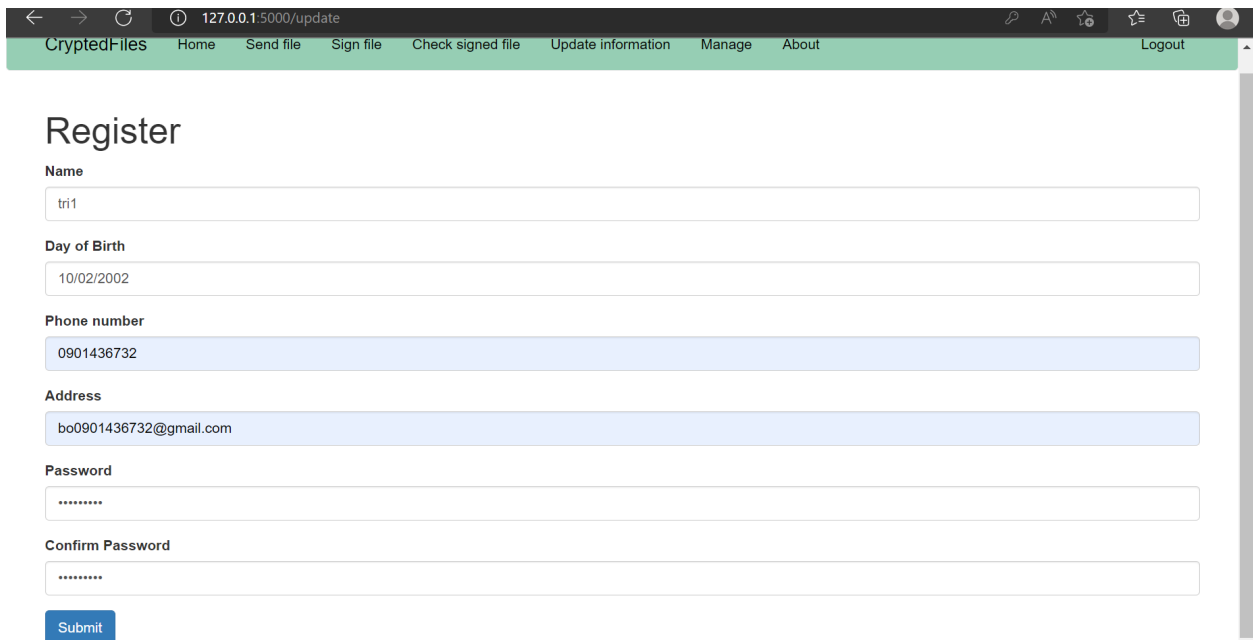
print("Register password: ", storage_password)
print("Register prikey: ", private_key)
```

2.2 Khoá riêng Kprivate cần được mã hoá bằng thuật toán AES. Passphrase của người dùng được sử dụng để phát sinh khoá bí mật Ksecret trong thuật toán AES. Khoá riêng Kprivate sau khi được mã hoá và khoá công cộng Kpublic được lưu trữ tương ứng với thông tin người dùng.

```
#encrypt private key by AES using password as secret key
secret_key = hashlib.sha256(password).digest()
raw = pad(private_key, BLOCK_SIZE)
iv = Random.new().read(AES.block_size)
cipher = AES.new(secret_key, AES.MODE_CBC, iv)
encrypted = base64.b64encode(iv + cipher.encrypt(raw))
```

3. Cập nhật thông tin tài khoản

3.1 Ứng dụng cho phép cập nhật thông tin tài khoản (họ tên, ngày sinh, điện thoại, địa chỉ, passphrase).



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/update". The browser's address bar and navigation buttons are visible. The page has a green header bar with the following navigation links: "Crypted-iles", "Home", "Send file", "Sign file", "Check signed file", "Update information", "Manage", "About", and "Logout". The main content area is titled "Register" and contains a form with the following fields:

- Name:** A text input field containing the value "tri1".
- Day of Birth:** A date input field containing the value "10/02/2002".
- Phone number:** A text input field containing the value "0901436732".
- Address:** A text input field containing the value "bo0901436732@gmail.com".
- Password:** A password input field with masked characters "*****".
- Confirm Password:** A password input field with masked characters "*****".

At the bottom of the form is a blue "Submit" button.

Sau khi nhấn vào mục Update Information, ta sẽ được bngr biểu để tự thay đổi thông tin bản thân.

3.2 Trường hợp đổi passphrase cần đảm bảo cặp khoá Kprivate, Kpublic không bị thay đổi. Tức là khoá Kprivate được mã hoá ở bước 2.2 với passphrase cũ, cần được mã hoá lại với passphrase mới.

Thông tin trước khi đổi mật khẩu, thông tin:

Phone:	0901436732
Address:	bo0901436732@gmail.com
Password:	M OoOycOOG
Public_key:	-----BEGIN RSA PUBLIC KEY-----MIGJ
Private_key:	1XM4TzONxVECu5tL3QGiq2HQBSTpC

Sau khi đổi thông tin:

Updated information!!

Welcome To CryptedFiles

Help you send files in a more secured way

Form Editor | Navigate: ⏮ ⏪ ⏩ ⏭

Phone: 0901436731

Address: bo0901436732@gmail.com

Password: >f-Kt6Oe

Public_key: -----BEGIN RSA PUBLIC KEY-----MIGJ

Private_key: Cg8U3jwsKrBxmhpt4x2y2bus/U0wHl

4. Mã hoá tập tin (người gửi mã khoá tập tin và gửi cho người nhận)

4.1 Ứng dụng cho phép người dùng chọn tập tin cần mã hoá và chọn người nhận (giả sử người nhận là 1 người dùng khác của ứng dụng).

Sau khi bấm vào mục Send File, ta sẽ được một giao diện như dưới đây:

CryptedFiles Home Send file Sign file Check signed file

Upload Files

Choose Files: Choose Files Capture.PNG

Share files to username

Username: bo@gmail.com SUBMIT

Kiểm tra bên tài khoản nhận:

CryptedFiles Home Send file Sign file Check signed file Update information Manage About

Files:

Capture.PNG

Trong mục manage thì file đã nhận được thành công.

4.2 Ứng dụng tự phát sinh một khoá bí mật Ksession (khoá phiên) cho thuật toán AES để mã hoá toàn bộ tập tin.

```
#search for receiver public key
query = "SELECT public_key FROM users WHERE email = %s"
cur.execute(query,[email])
data = cur.fetchall()
receiver_public_key_pem = ""
for i in data:
    receiver_public_key_pem = (i['public_key'])

#create Ksession key
k_session = secrets.token_urlsafe()
# print("K_session is: ",k_session )
# print("K_session type is: ",type(k_session) )
# print("Public key is: ", receiver_public_key_pem)
```

4.3 Ứng dụng sử dụng public key (Kpublic) của người nhận để mã hoá khoá Ksession bằng thuật toán RSA. Khoá Ksession sau khi được mã hoá thì sẽ được bổ sung vào tập tin đã mã hoá (sinh viên tự đề nghị cấu trúc tập tin này).

```
#encrypt receiver's public key
receiver_public_key = rsa.PublicKey.load_pkcs1(receiver_public_key_pem)
encrypted = rsa.encrypt(k_session.encode('ascii'), receiver_public_key)
# print("New k_session: ", encrypted)
```

Khoá Ksession được bổ sung vào tập tin đã mã hóa:

```

for file in files:
    #if file and allowed_file(file.filename):
    filename = secure_filename(file.filename)
    with open(UPLOAD_FOLDER + '/' + filename, 'wb') as out_file:
        encrypt(file, out_file, k_session)

    #Add information of files into database
    cur.execute("INSERT INTO files (file_name, uploaded_on, own_by, share_to, k_sessions) VALUES (%s, %s, %s, %s, %s)",(filename, n
    mysql.connection.commit()

#Close connection
cur.close()

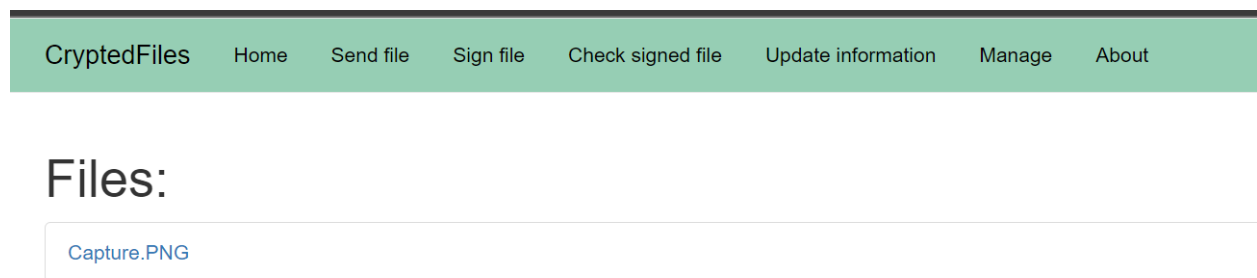
flash('File(s) successfully sent')

```

5. Giải mã tập tin (người nhận nhận tập tin và giải mã)

5.1 Ứng dụng cho phép người dùng chọn tập tin cần giải mã.

Sau khi nhận được file từ người dùng khác, sẽ được hiển thị ở mục Manage:



Đoạn code cập nhật dữ liệu trong manage sau khi nhận được file từ các người dùng khác:

```

#Manage
@app.route('/manage')
@is_logged_in
def manage_file():
    # Create cursor
    cur = mysql.connection.cursor()

    # Get file
    query = "SELECT file_name FROM files WHERE share_to = %s"
    param = ([session['email']])
    result = cur.execute(query,param)

    files_list = []
    data = cur.fetchall()
    for i in data:
        files_list.append(i['file_name'])

    if len(files_list) > 0:
        return render_template('manage.html', files_list=files_list)
    else:
        msg = 'No Files Found'
        return render_template('manage.html', msg=msg)
    # Close connection
    cur.close()

```

5.2 Ứng dụng dùng passphrase của người dùng (đã đăng nhập thành công) để giải mã thông tin private key (Kprivate) của mình đã được mã hoá bằng thuật toán AES ở bước 2.2

```
# Get user by email
cur.execute("SELECT * FROM users WHERE email = %s", [session['email']])

data = cur.fetchone()

password = data['password']
#get password and salt to check
salt_from_storage = password[:32] # 32 is the length of the salt
password_from_storage = password[32:]

#decrypt private key using password
enc = data['private_key']
secret_key = hashlib.sha256(password_from_storage).digest()
enc = base64.b64decode(enc)
iv = enc[:16]
cipher = AES.new(secret_key, AES.MODE_CBC, iv)
decrypted = unpad(cipher.decrypt(enc[16:]), BLOCK_SIZE)
```

5.3 Ứng dụng dùng private key (Kprivate) của mình để giải mã nội dung trong tập tin để có được khoá Ksession (giải mã cho bước 4.3).

```
# decrypt file's ksession key
receiver_private_key = rsa.PrivateKey.load_pkcs1(decrypted)
decrypted_session_key = rsa.decrypt(k_session, receiver_private_key).decode('ascii')
```

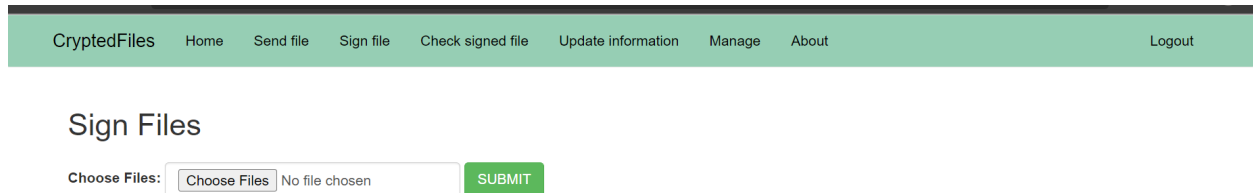
5.4 Ứng dụng dùng khoá Ksession để giải mã tập tin (giải mã cho bước 4.2)

```
# using decrypted session key to decrypt file and sent to user
with open(UPLOAD_FOLDER + '/' + filename, 'rb') as in_file, open(UPLOAD_FOLDER + '/' + filename + 'decrypted', 'wb') as out_file:
    decrypt(in_file, out_file, decrypted_session_key)

uploads = os.path.join(app.root_path, app.config['UPLOAD_FOLDER'])
return send_from_directory(uploads, filename + 'decrypted', filename=filename, as_attachment=True)
```

6. Ký trên tập tin:

6.1 Ứng dụng cho phép chọn tập tin cần ký



The screenshot shows a web application interface for signing files. At the top, there is a navigation bar with links: CryptdFiles, Home, Send file, Sign file, Check signed file, Update information, Manage, About, and Logout. Below the navigation bar, the page title is 'Sign Files'. Underneath, there is a section labeled 'Choose Files:' followed by a file selection interface. This interface includes a button labeled 'Choose Files', a text status 'No file chosen', and a green 'SUBMIT' button.

Giao diện trong mục Sign File.

6.2 Hash nội dung tập tin cần ký (dùng thuật toán SHA-256)

+ 6.3 Ký trên nội dung Hash sử dụng private key (Kprivate) của người dùng

```
# open file and hash the data
for file in files:
    filename = file.filename
    with open(UPLOAD_FOLDER + '/' + filename + '.sig', 'wb') as out_file:
        data = file.read().decode('utf-8') # read the data
        out_file.write(sign(data, receiver_private_key)) # write data into .sig file

    uploads = os.path.join(app.root_path, app.config['UPLOAD_FOLDER'])
    return send_from_directory(uploads, filename + '.sig', as_attachment=True)

flash('File(s) successfully signed')
```

6.4 Chữ ký lưu riêng thành 1 file .sig (ví dụ: chữ ký đi kèm với file sample.doc là file sample.doc.sig)

Sign Files

Choose Files:

Choose Files

New Text Document.txt

SUBMIT

▼ Today (1)



New Text Document.txt.sig

6/18/2022 6:48 PM

7. Xác nhận chữ ký trên tập tin

Signature is valid, own by bo@gmail.com

Check Sign Files

Choose Files:

Choose Files

New Text D...ment.txt.sig

SUBMIT

Sau khi chọn 1 file dạng sig, chương trình sẽ kiểm tra trong cơ sở dữ liệu xem có tài khoản nào là hợp lệ và sẽ báo lại nếu thành công.

```
# Check if there is a public key
# Create cursor
cur = mysql.connection.cursor()
# Get all the public key
cur.execute("SELECT * FROM users")

users_data = cur.fetchall()

msg = 'Cant identify the signature!!'

for user in users_data:
    if verify(data, sig, rsa.PublicKey.load_pkcs1(user['public_key'])):
        msg = 'Signature is valid, own by ' + user['email']

return render_template('checksign.html', msg=msg)
```

