E-AUTHENTICATION SYSTEM USING QR CODE AND OTP.

Socially Relevant Project Report

by

M.Bhavya Sri - 19BQ1A05E5

R.Jyothirmai - 19BQ1A05J3

P.Charitha Sri - 19BQ1A05H4

M.Mano Teja - 19BQ1A05E4

N.Vamsi Krishna - 20BQ5A0518

Under the guidance of

Mr.M.China Saheb

Asst. Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

(B.Tech Program is Accredited by NBA)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified

NAMBUR(V), PEDAKAKANI(M), GUNTUR-522 508

Tel no: 0863-2118036, url:www.vvitguntur.com

June 2021

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
Permanently Affiliated to JNTU Kakinada, Approved by AICTE
Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified
Nambur, Pedakakani (M), Guntur (Dt) - 522508
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
B.Tech Program is Accredited by NBA

CERTIFICATE

This is to certify that this Socially Relevant Project Report is the bonafide work of **……, ……, ……..…… ,**bearing **Reg. No. ……, ……, …….** who had carried out the project entitled "……..." under our supervision.

Project Guide **Head of the Department**

_____ _____

Submitted for Viva voce Examination held on _____

Internal Examiner **External Examiner**

ABSTRACT

With the rapid evolution of wireless communication technology, user authentication is important in order to ensure the security of wireless communication technology. Passwords play an important role in the process of authentication. In the process of authentication, the password entered by the user will be transmitted along the traffic to the authentication server in order to allow the server to grant access to the authorised user. The attackers will use the chance to attempt to sniff another person's password in order to perform some illegal activities by using another's identity to keep them safe from trouble. Due to the issues, there are many solutions proposed to improve the security of wireless communication technology.The solution adopted is the one time password(OTP) and QR Code authentication.

The objective of the system outcome is to enhance the current login authentication system. It provides solutions for making password breaking more difficult as well as convinces users to choose and set hard-to-break passwords.

TABLE OF CONTENTS

List of Tables :

List of Figures

CHAPTER-1

INTRODUCTION

In the password scheme, the user can easily and efficiently login into the system. If we analyze the security and usability of the login through password, there is a possibility for hacking of login credentials, shoulder surfing and accidental login. The shoulder surfing attack can be performed by the adversary to obtain the user's password by watching over the user's shoulder as he enters his password.

Authentication is an activity to prove the person's credential who wishes to perform the activity. In authentication, the password entered by the user will be transmitted along with the traffic to the authentication server to allow the server to grant access to the authorized user. When the password is transmitted, the attackers will try to sniff into the network to obtain data that include the user's password.

It is important to protect our own account because our credit is priceless. It is hard to trace the attackers in the cyber world. The secure login system is needed to ensure cybersafety. Therefore, this project would like to provide alternative ways to log in to a system because the current login system is not secure enough.

- ➢ The main objective is to implement a secure login authentication system using otp and qr code
- ➢ Another objective is to ensure the login password .

CHAPTER 2

CONCEPTS AND METHODS

2.1 *PROBLEM DESCRIPTION :*

we have come up with a secure system schemes with different degrees of resistance to shoulder surfing has been proposed.
Therefore, this project would like to provide alternative ways:

1 . Login through OTP

2.Login through QR Code

A one-time password is a password that is valid for only one login session or transaction, on a computer system or other digital device. The main idea of OTP authentication is to create a random password every time when the user tries to log in which improves the security of the system.

A **QR code** (abbreviated from **Quick Response code**) is a type of matrix barcode.A barcode is a machine-readable optical label that contains information about the item to which it is attached. In practice, QR codes often contain data for a locator, identifier, or tracker that points to a website or application.A QR code consists of black squares arranged in a square grid on a white background, which can be read by an imaging device such as a camera, and processed.

2.2 PROPOSED SOLUTION

In order to use this authentication system,

1. Users need to first register themselves into this system by filling up the basic registration details.
2. Registration fails,
   if username is already used by some other user,
   if the format of phone Number(10 digits),email(@gmail.com) is wrongly entered.
   The user will be prompted with respective of their mistakes to modify them.
3. After a successful registration, the user can access the login module where he/she needs to first authenticate the account by entering the username and password which was entered while registration.
4. Once the username and password is authenticated, the user may proceed with the next authentication section where he/she needs to select the type of authentication as QR (Quick Response) Code or OTP (One Time Password).
5. Once the user selects the authentication type as QR Code, then the system will generate a QR Code and send it to the user's mail id over the internet.
6. If the user selects OTP, then OTP will be sent to his mail ID provided while registering.
7. If the user passes the authentication, then the system will redirect to the main page.
8. User main page displays the profile of the user.
9. The QR Code and OTP are randomly generated by the system at the time of login.
10. The user can change the password ,if he/she forgets password.To change user must provide correct user name .An email with link to change password will be sent user’s mail id .There user can change password.
11. For ensuring the security of the password stored,the password is added with some random string and then hashed and stored in the database.Even the system admin cannot know the password.
12. user can delete their account,but user have to login first and confirm to delete their account permanently

COMPARISON BETWEEN PROPOSED SOLUTION :

proposed solution	Strength	Weakness
login using username and password	easy login	data breach can occur
One Time Password(OTP)	● Not vulnerable to replay attack ● Can be time limited	● Require internet or phone connection to complete the process ● User will feel annoyed to wait for the message to reach if the connection is slow
QR Code	● can be time limited ● difficult to hack	● Requires webcam/scanner ● Require internet or phone connection to complete the process ● User will feel annoyed to wait for the message to reach if the connection is slow

Table 2.1

2.3 SYSTEM REQUIREMENTS

Needs and Requirements:

1. Mysql server
2. Python virtual environment
3. Pre-defined modules
4. Flask Framework
5. Any code editor.
6. Browser(Chrome,Firefox,Microsoft Edge)

OS Environment: Linux, Windows, MAC.

Hardware:

- Processor: Intel dual-core or above
- wifi network
- RAM: 1 GB or above

solution type : web based

Flow Char:

CHAPTER 3

IMPLEMENTATION

3.1 TOOLS USED

1.Smart Phone :

It is used to show QR code to the laptop webcam ,so that QR Code can be scanned.

2. Laptop :

The system also requires a server to perform authentication service. Instead of using a real server to set up the system, the laptop is used to be a virtual server. The laptop also will be used to surf the site built to log in

3.Text Editor/IDE : for writing the essential codes.

Flask Framework:

The route() function of the Flask class is a decorator, which tells the application which URL should call the associated function.

app.route(rule, options)

- The rule parameter represents URL binding with the function.
- The options are a list of parameters to be forwarded to the underlying Rule object.

The data from a client's web page is sent to the server as a global request object. In order to process the request data, it should be imported from the Flask module.

Important attributes of request object:

- Form − It is a dictionary object containing key and value pairs of form parameters and their values.
- args − parsed contents of the query string which is part of the URL after the question mark (?).
- Cookies − dictionary object holding Cookie names and values.
- files − data pertaining to uploaded files.
- method − current request method.

A Flask application is started by calling the run() method. The Debug mode is enabled by setting the debug property of the application object to True before running or passing the debug parameter to the run() method.

```
app.run(debug =True)
```

Languages Used:

- Python flask for running the app.
- Flask_mail for sending emails
- Html and CSS for designing register and login and homepages.
- Mysql for creating databases and for storing registered account details.

SOME MODULES USED :

bcrypt : We use a slow hash algorithm, preventing the hacker from using brute force. bcrypt is a slow hash algorithm that can be made slower as processors get faster.

gensalt() is used which returns a randomly generated "salt" string and hashpw(plain, hashed/salt) which hashes the plaintext password, plain, along with the salt pulled from the second arg, and returns the hash string.

As bcrypt requires inputs to be bytes not string, hence encode() to convert strings to bytes and decode() to convert bytes to strings.

Flask-Mail:

Flask-mail module is used for sending the email.Flask-Mail is configured as per following settings

```
app.config['MAIL_SERVER']='smtp.gmail.com'

app.config['MAIL_PORT'] = 465

app.config['MAIL_USERNAME'] = 'yourId@gmail.com'

app.config['MAIL_PASSWORD'] = '*****'

app.config['MAIL_USE_TLS'] = False

app.config['MAIL_USE_SSL'] = True
```

Next, an Instance of Mail is created (mail=Mail(app))

Next, Message class is used for creating the mail (through this class we can add subject,body,recipients etc for a mail) .

Next, send() from Mail class is used for sending the email

qrcode:

1. qr.add_data() is used to add data .
2. qrcode.make() creates a PilImage object.
3. qr.make_image(fill_color='black', back_color='white') is used to generate the QRcode and is then stored in your system at some specific location.

For sending the mail again Flask-mail module is used. attach() is used to attach the QR code image with the mail and by using send() function an email is sent.

cv2:

cv2 module is used for verification of qrcode:

1. cv2.VideoCapture(-1) is used for capturing the qr code
2. cv2.QRCodeDetector() detects the QR code.
3. read() function is used to read the captured image.
4. detector.detectAndDecode(img) is used to extract the data from the image.

flask_mysqldb:

Flask-MySQL is a Flask extension that allows you to access a MySQL database

3.2 PSEUDO CODES/ALGORITHMS

Create database and table :

create database authentication;

use authentication ;(to change database)

create table :

```
create table if not exists `accounts`(
`id` int(11) NOT NULL AUTO_INCREMENT,
`username` varchar(255) NOT NULL,
-> `first_name` varchar(255) NOT NULL,
-> `last_name` varchar(255) NOT NULL,
-> `phone_number` varchar(40) NOT NULL,
-> `email` varchar(255) NOT NULL,
-> `password` varchar(255) NOT NULL,
-> PRIMARY KEY(`id`)
-> )ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
```

Connect to database and retrieve data:

```
cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
cursor.execute('SELECT * FROM accounts WHERE username = % s', (username, ))
account = cursor.fetchone()
```

Encrypting password :

```
password = request.form['password']
hashed = bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())
hashed_str = hashed.decode('utf-8')
```

Decrypting Password :

```
hashed = account['password']
hashed2 = bcrypt.hashpw(password.encode('utf-8'),hashed.encode('utf-8'))
hashed2_str = hashed2.decode('utf-8')
```

Registration:

From the registration form, this module takes the user details like the first name, last name, phone number, email address, username, password.

We store a hash of the password, rather than the plaintext password. Next, we connect to the Mysql database and information is then stored in the database created. If the username is already present, a message is displayed to change the username. Regular Expressions are used to check whether the details are in a valid format or not. If not, a message is displayed to change the respective invalid information.

```
if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'Username must contain only characters and numbers !'
        elif not re.match(r'[A-Za-z]+', first_name):
            msg = 'First name must contain only characters...!'
        elif not re.match(r'[A-Za-z]+', last_name):
            msg = 'Last name must contain only characters...!'
        elif not re.match(r'[0-9]+', phone_number):
            msg = 'Phone number must contain only digits.....!'
        elif not username or not password or not email or not first_name or not last_name:
            msg = 'Please fill out the form !'
        else:
            cursor.execute('INSERT INTO accounts VALUES (NULL, % s, % s, % s, %s, %s,
%s)', (username,first_name,last_name,phone_number, email,hashed_str,))
            mysql.connection.commit()
            msg = 'registered successfully.. !'
```

Login:

The username and password of the user which is already stored in the database are requested by the login module. Mysql server is connected and used to retrieve the details corresponding to the username. If the username is not present, a message is displayed asking for the correct username. The password given by the user in the login form is hashed again using the bcrypt module and is compared with hashed password retrieved from the database. If passwords are matched then the user will be given a choice to log in either through OTP or QR Code according to his/her wish.

OTP Verification:

If a user chooses to log in with OTP, an email is sent to the mail ID given by the user during registration.

- A six digit OTP is generated using randint() function .
- users will be prompted to enter OTP they received.
- Finally,If the user enters the correct OTP sent to him/her, the user is directed to the homepage, else they have to login again.

```
user_otp = request.form['otp']
        if otp == int(user_otp):
                session['loggedin'] = True
                msg="Email  verification is  successful"
                #return render_template('index.html',msg=msg)
                return redirect(url_for('home'))
        else:
                msg = "failure, OTP does not match"
                return render_template('login.html',msg=msg)
```

QR Code Verification:

If a user chooses to log in through QR Code, they will get a mail with the QR Code attached to it.

Users have to scan the QR Code in front of the webcam.

If the data extracted is the same as the data present in the qr code generated before then the main page is accessed, else if the wrong QRcode is scanned then the user has to login again. In case if the user fails to scan QRcode below 5 minutes then the time limit is exceeded and the user has to log in again.

the user can use esc character from the keyboard to stop videocapture if he/she does not want to login through QR code.

```
while(true) :
        webcam activated
        user scans qr code and data is extracted from qrcode
        d = data.split()
        if (d[0]==username and d[1]==password and d[2]==str(otp)):
                        cam.release()
                        cv2.destroyAllWindows()
                        return redirect(url_for('home'))
        else:
                        cam.release()
                        cv2.destroyAllWindows()
                        return login page;
        if timelimit exceeded then
                login unsuccessful,return to login page
```

Reset Password:

If a user forgets his/her password then he can choose the forget password option from the login form.

First user will be asked to give his/her username.

For a user to change his password, a confirmation link will be sent to the user's mail account which the user provided while registering .

For this, itsdangerous module is used. Confirmation link is a token generated using dumps() and loads() functions from URLSafeTimedSerializer class. The confirmation link will be active only for 1 minute.

Once the user clicks the confirmation link sent to his email, a web page to change the user's password will be displayed.

The user has to enter the username, password and confirm the password. If a user enters a wrong username/invalid username then he/she will not be allowed to change the password.

If the password and confirm passwords are the same then the user will be updated with a new password.

```
if request.method == 'POST' and 'username' in request.form and 'password' in request.form
and 'confirmPassword' in request.form:
        username = request.form['username']
        password = request.form['password']
        confirmPassword = request.form['confirmPassword']
        if (password!=confirmPassword):
                return render_template('reset.html',msg="both password fields must be
entered same")
        else:
                hashed = bcrypt.hashpw(password.encode('utf-8'),bcrypt.gensalt())
                hashed_str = hashed.decode('utf-8')
                print(password, type(password), hashed, hashed_str)
                cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
                cursor.execute('SELECT * FROM accounts WHERE username = % s',
(username, ))
                account = cursor.fetchone()
                if account:
                        session['id'] = account['id']
                        session['username'] = account['username']
                        cursor.execute('UPDATE accounts SET password = %s where
username= %s ',(hashed_str,session['username'], ))
                        mysql.connection.commit()
                        return render_template('login.html',msg='password successfully
changed')
                else :
                        msg='Account with given username is not present\n,If you did not
register before,Please register..!'
                        return render_template('login.html',msg=msg)
```

Main Page access:

If a user logins successfully through OTP or QRCode then he/she will be redirected to this main page. It contains a profile that simply shows details of the user. Users can click logout to log out from the website.

```
if 'loggedin' in session:
    # We need all the account info for the user so we can display it on the profile page
    cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
    cursor.execute('SELECT * FROM accounts WHERE id = %s', (session['id'],))
    account = cursor.fetchone()
    # Show the profile page with account info
    return render_template('profile.html', account=account)
  # User is not loggedin redirect to login page
  return redirect(url_for('login'))
```

Delete Account:

If a user wishes to delete their account,he can click on the delete button available on the home page.

Users will be asked for confirmation whether he/she really wants to delete their account.

If the user clicks yes then the database is connected again and the tuple containing the details of the user will be permanently deleted.login form will be displayed again showing message your account was deleted.

```
def remove():
        if 'loggedin' in session:
                if request.method == 'POST' :
                        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
                        cursor.execute('SELECT * FROM accounts where
id=%s',(session['id'],))
                        account = cursor.fetchone()
                        if account :
                                session['id']=account['id']
                                session['username'] = account['username']
                                if request.form["LoginBtn1"]=="Yes":
                                        cursor.execute('DELETE from accounts where
username= %s and id = %s',(session['username'],session['id'], ))
                                        mysql.connection.commit()
                                        return render_template('register.html',msg='account
deleted')
                                elif request.form["LoginBtn1"]=="No":
                                        return render_template('home.html')
                        else :
                                return redirect(url_for('login'))
        return redirect(url_for('login'))
```

html page for Register :

```
<html>
        <head>
                <meta charset="UTF-8">
                <title> Register </title>
                <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">

        </head>
        <body></br></br></br></br></br>
                <div align="center">
                <div align="center" class="border">
                        <div class="header">
                                <h1 class="word">Register</h1>
                        </div></br></br></br>
                        <h2 class="word">
                                <form action="{{ url_for('register') }}" method="post">
                                <div class="msg">{{ msg }}</div>
                                        <input id="username" name="username" type="text"
placeholder="Enter Your Username" class="textbox"/></br></br>
                                        <input id="first_name" name="first_name"
type="text" placeholder="Enter Your First Name" class="textbox"/></br></br>
                                        <input id="last_name" name="last_name"
type="text" placeholder="Enter Your Last Name" class="textbox"/></br></br>
                                        <input id="phone_number" name="phone_number"
type="tel" placeholder="Enter Your mobile number" class="textbox"/></br></br>
                                        <input id="email" name="email" type="text"
placeholder="Enter Your Email ID" class="textbox"/></br></br>
                                        <input id="password" name="password"
type="password" placeholder="Enter Your Password" class="textbox"/></br></br>
                                        <input type="submit" class="btn" value="Sign
Up"></br>
                                </form>
                        </h2>
                        <p class="bottom">Already have an account? <a class="bottom"
href="{{url_for('login')}}"> Sign In here</a></p>
                </div>
                </div>
        </body>
</html>
```

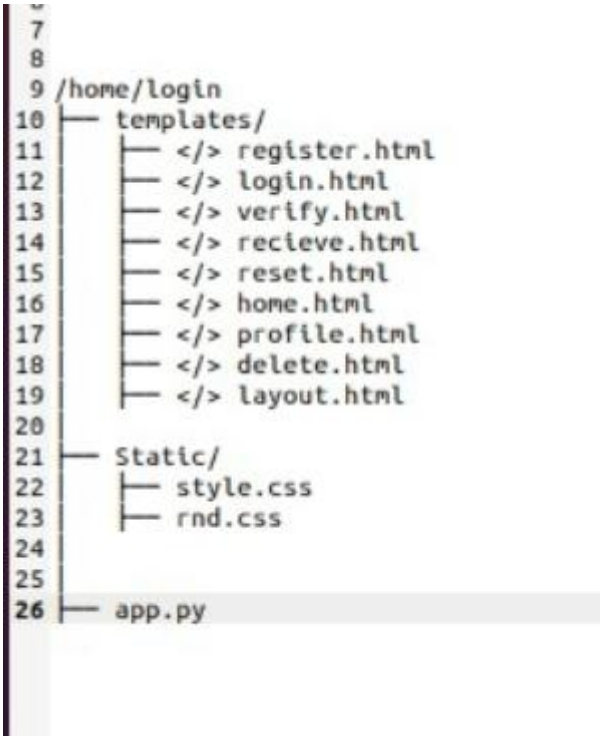
CSS :

```
.button {
        background-color: #236B8E;
        border: none;
        color: white;
        padding: 15px 32px;
        text-align: center;
        text-decoration: none;
        display: inline-block;
        font-size: 16px;
        margin: 4px 2px;
        cursor: pointer;
}
.textbox{
        padding: 10px 40px;
        background-color: #236B8E;
        text-color: #FFFFFF;
        border-radius: 10px;
}

::placeholder {
        color: #FFFFFF;
        opacity: 1;
        font-style: oblique;
        font-weight: bold;
}
```

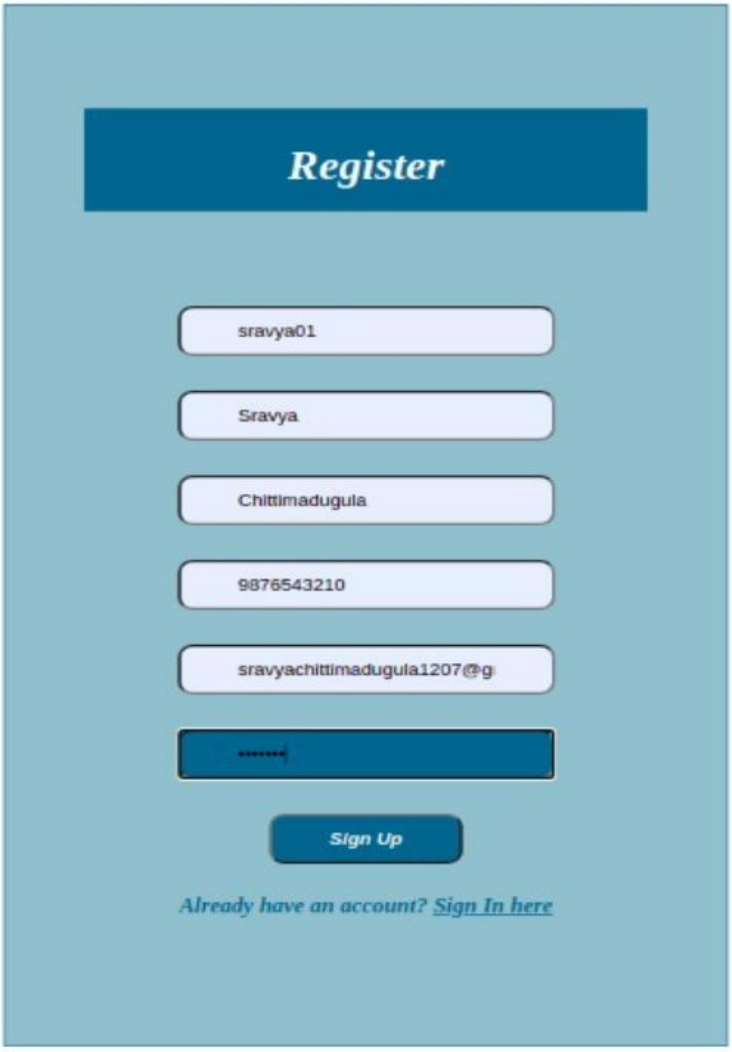
CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML and can be used with any XML-based markup language

FlowChart/File Structure

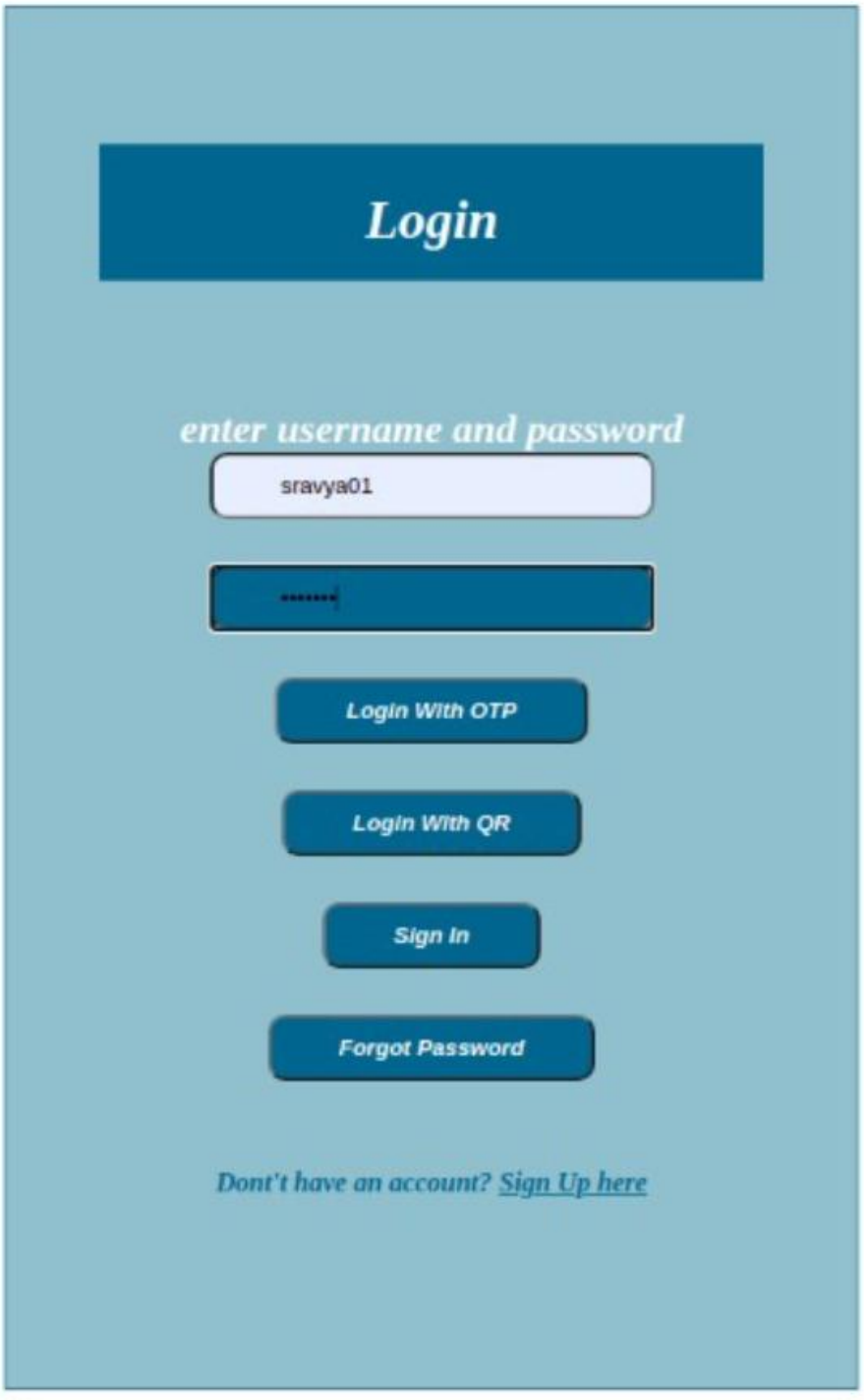


3.3 SCREENSHOTS

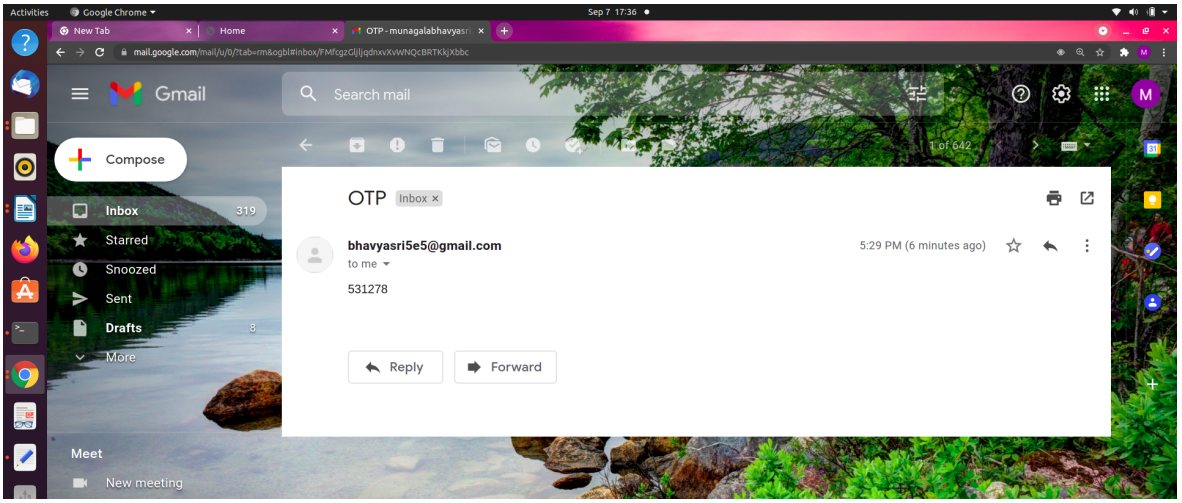
User registration

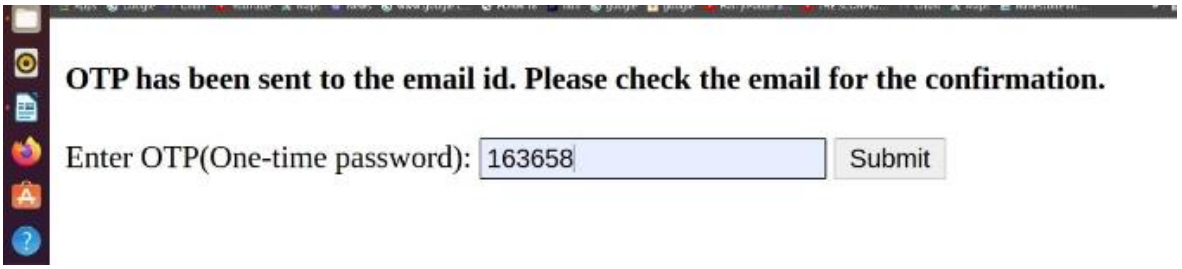


User login

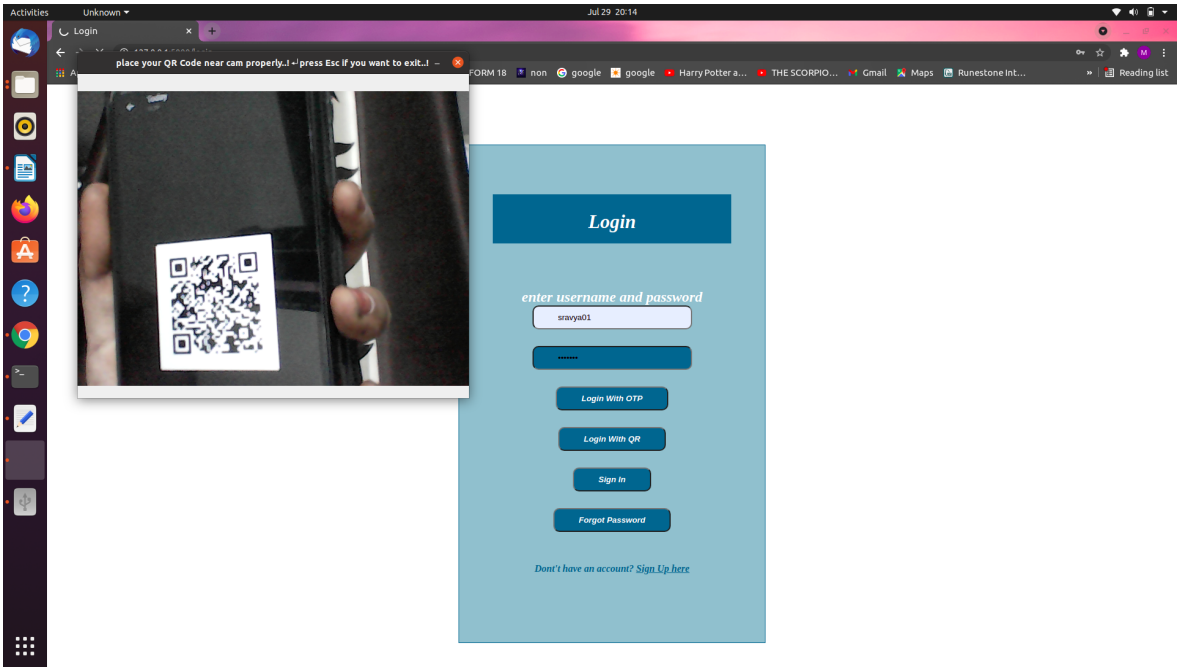

Login
enter username and password
sravya01
Login With OTP
Login With QR
Sign In
Forgot Password
Dont't have an account? Sign Up here

If user chooses to login through OTP :

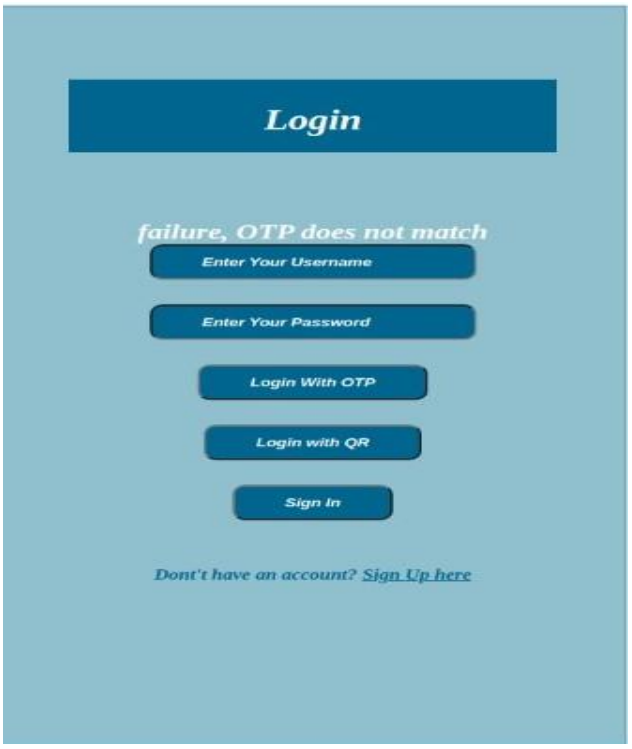




If user chooses to login through QR code

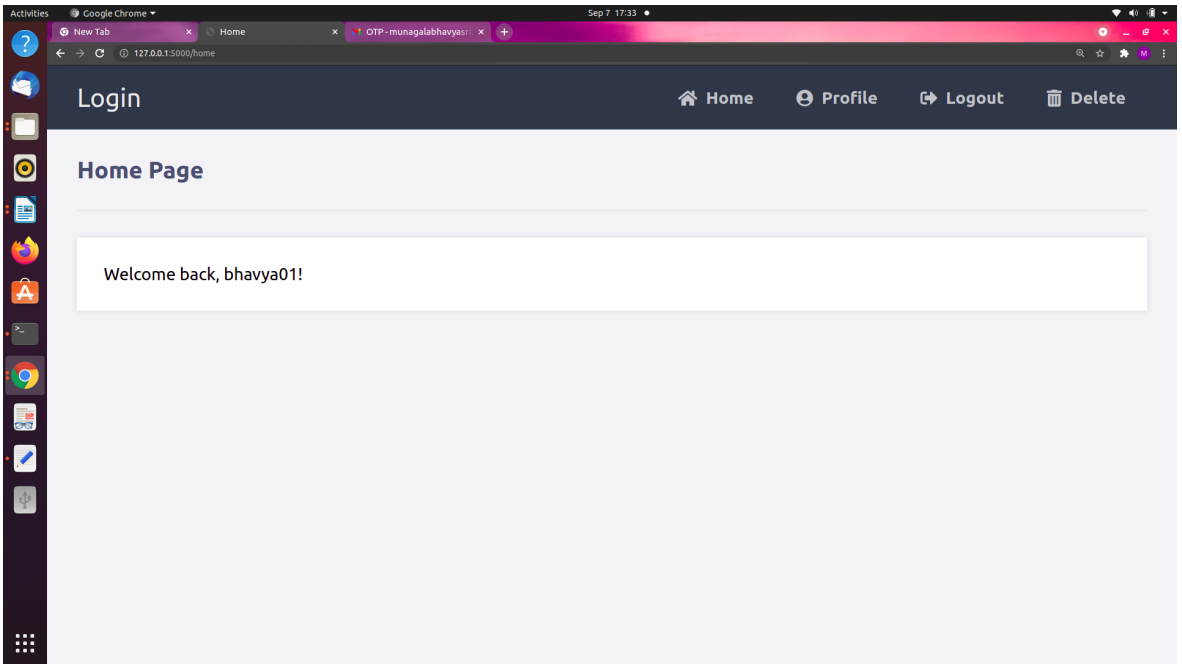


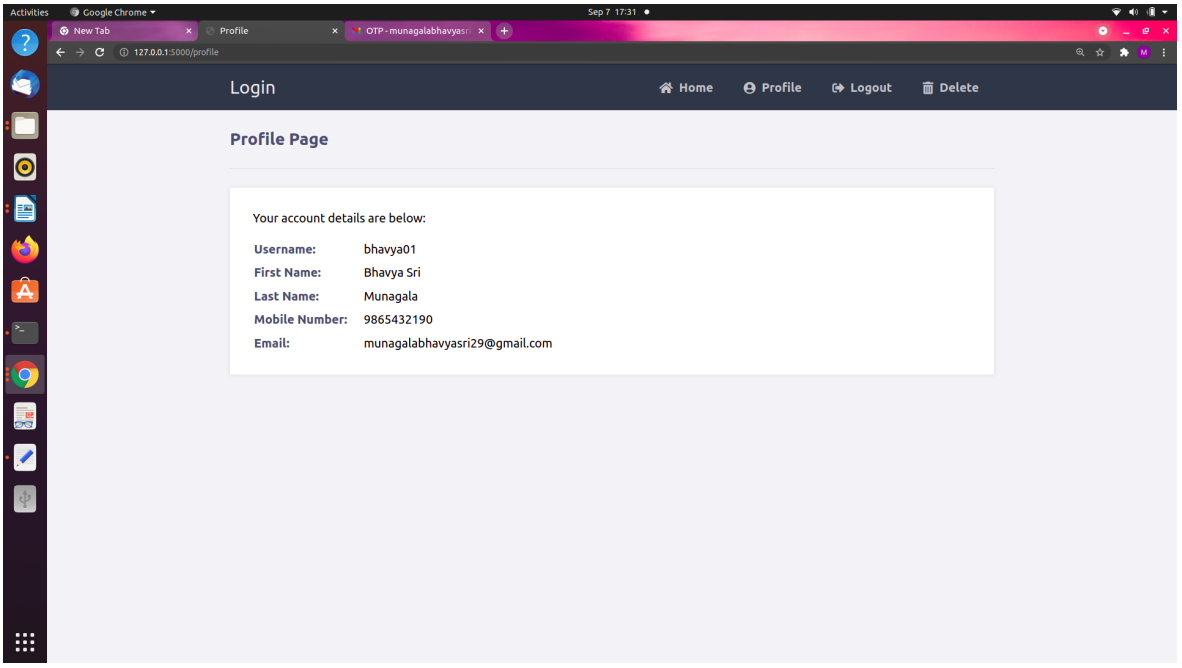
If the OTP is not correct then



If QR code scanned is Incorrect

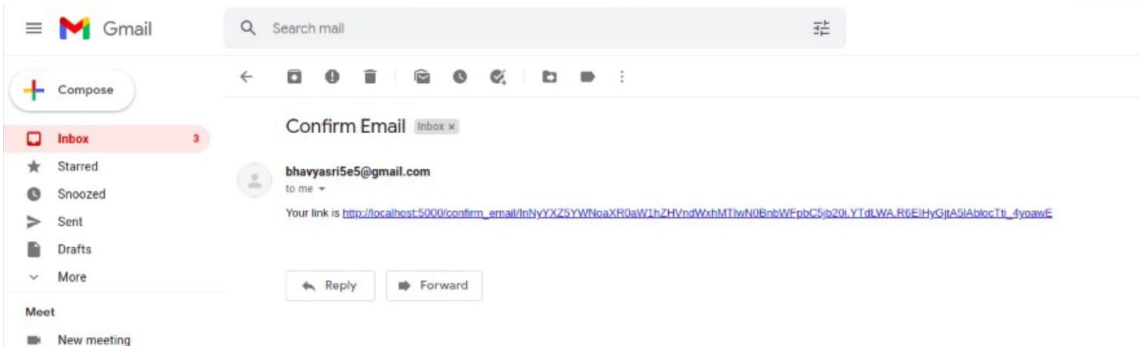
If in either of OTP or QR code ,login is successful then (home page and profile page)

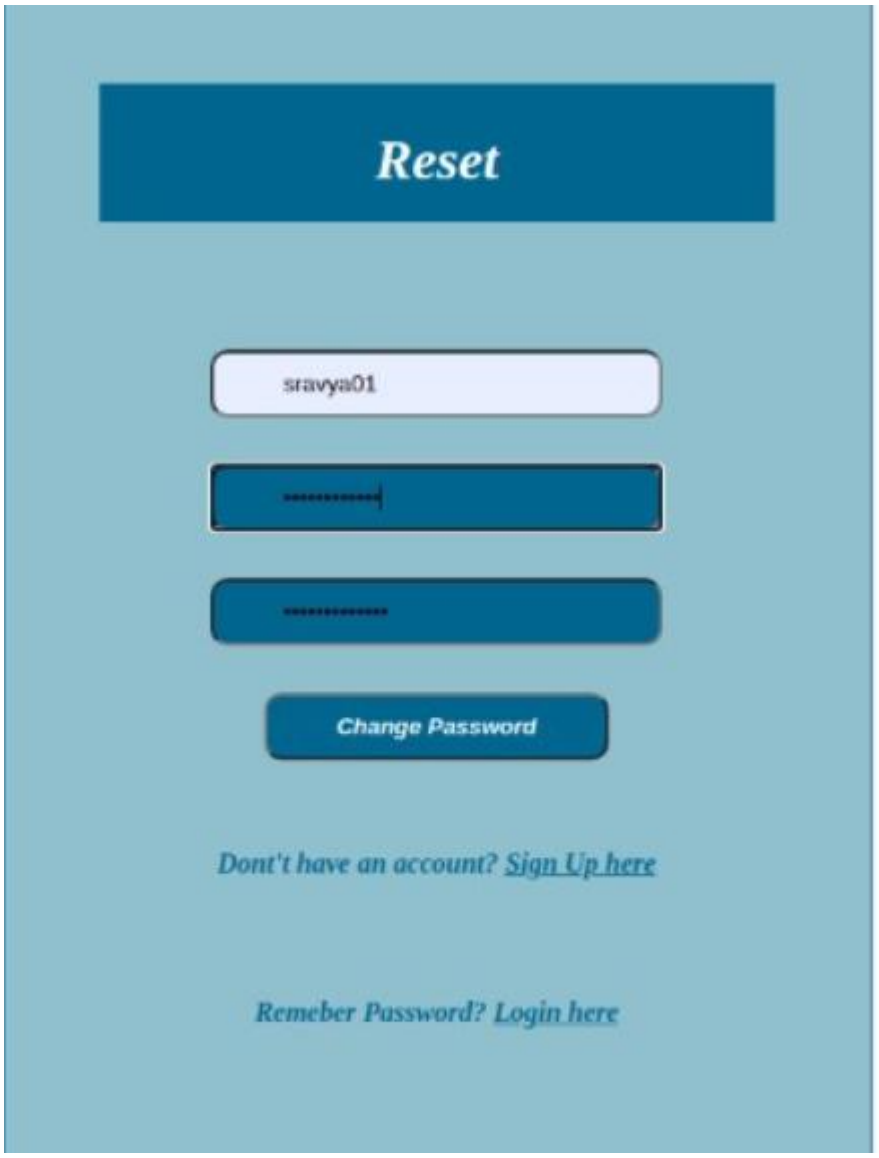




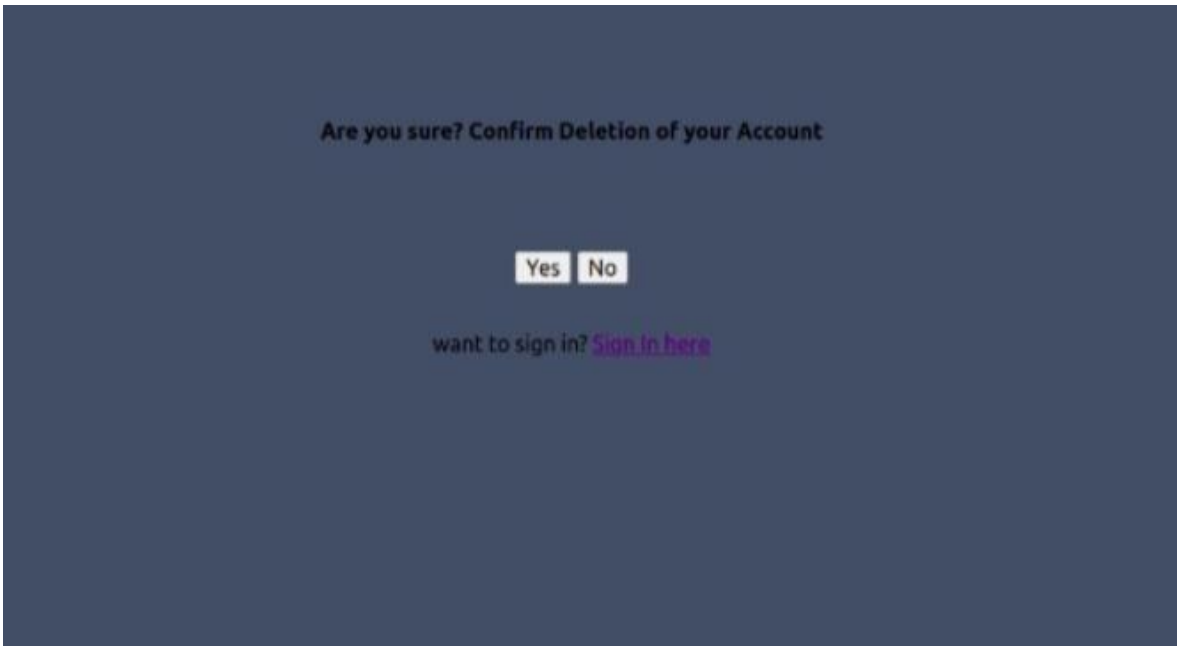
Incase user forgets password


localhost:5000/passwordreset
Check your mail for changing your password!

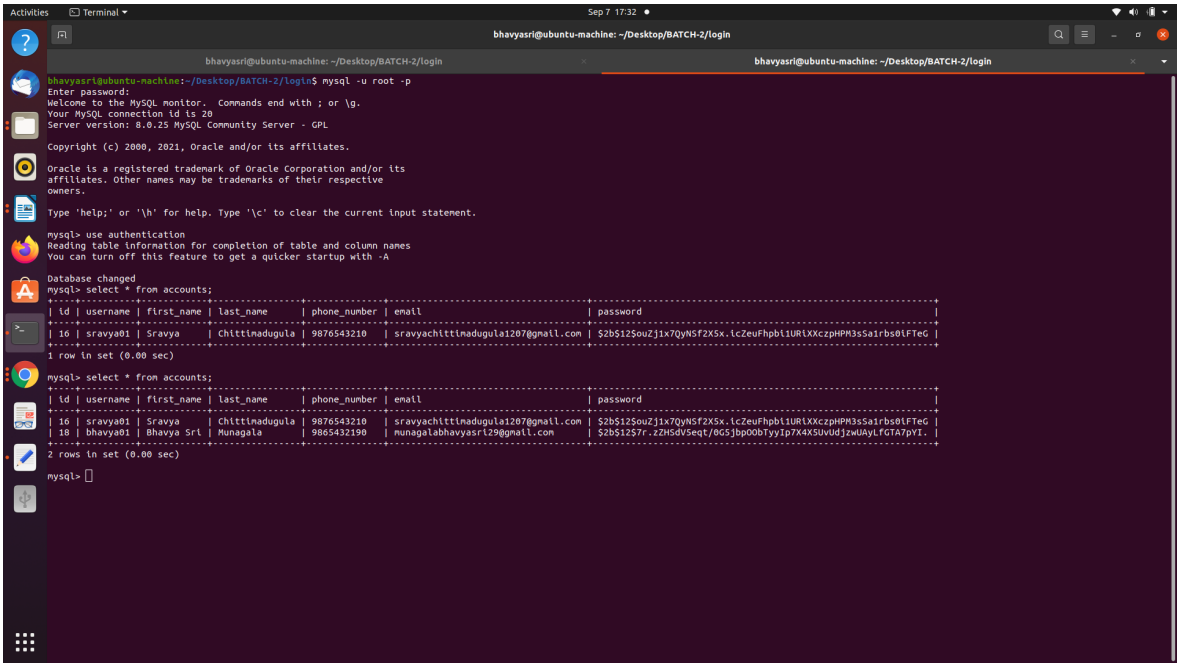

Gmail
Search mail
Compose
Inbox 3
Starred
Snoozed
Sent
Drafts
More
Meet
New meeting
Confirm Email
Inbox ×
bhavyasri5e5@gmail.com
to me
Your link is http://localhost:5000/confirm_email/InNyYXZ5YWNoaXR0aW1hZHVndWxhMThwN0BnbWFpbC5jb20i.YTdLWA.R6EIHyGjtA5IAblocTti_4yoawE
Reply
Forward


Reset
sravya01
Change Password
Dont't have an account? Sign Up here
Remeber Password? Login here

If a user wishes to delete their account



database:



CHAPTER 4

CONCLUSION :

We have successfully implemented an E-authentication system using OTP and QR code . We faced some problems while sending mails and configuring the flask environment and python virtual environment.

This Authentication system is more secure than a general password login system where user passwords can be hacked and user details will be in danger.

OTP generated will only be available till some time then it becomes invalid and also each time a new OTP will be generated,so the login system using OTP is more secure.

QR code, it will be helpful for secure data transactions also.QR code is also generated for each login and is unique and is also available only for some time .

QR codes are a little difficult to hack and decrypt ,so login using QR code is also more safe.

With the increase in usage of internet facilities and websites,Secure login systems are more essential for user safety.Through this login system ,we can maintain security to some more extent compared to general password login systems.

BIBLIOGRAPHY:

- flask tutorial : https://www.tutorialspoint.com/flask/index.html
- https://pythonprogramming.net/flask-user-registration-form-tutorial/ (forms)
- using and learning css - https://www.w3schools.com/css/
- using learning html https://www.w3schools.com/html/
- using cv2 https://pypi.org/project/opencv-python/

https://docs.opencv.org/master/d6/d00/tutorial_py_root.html