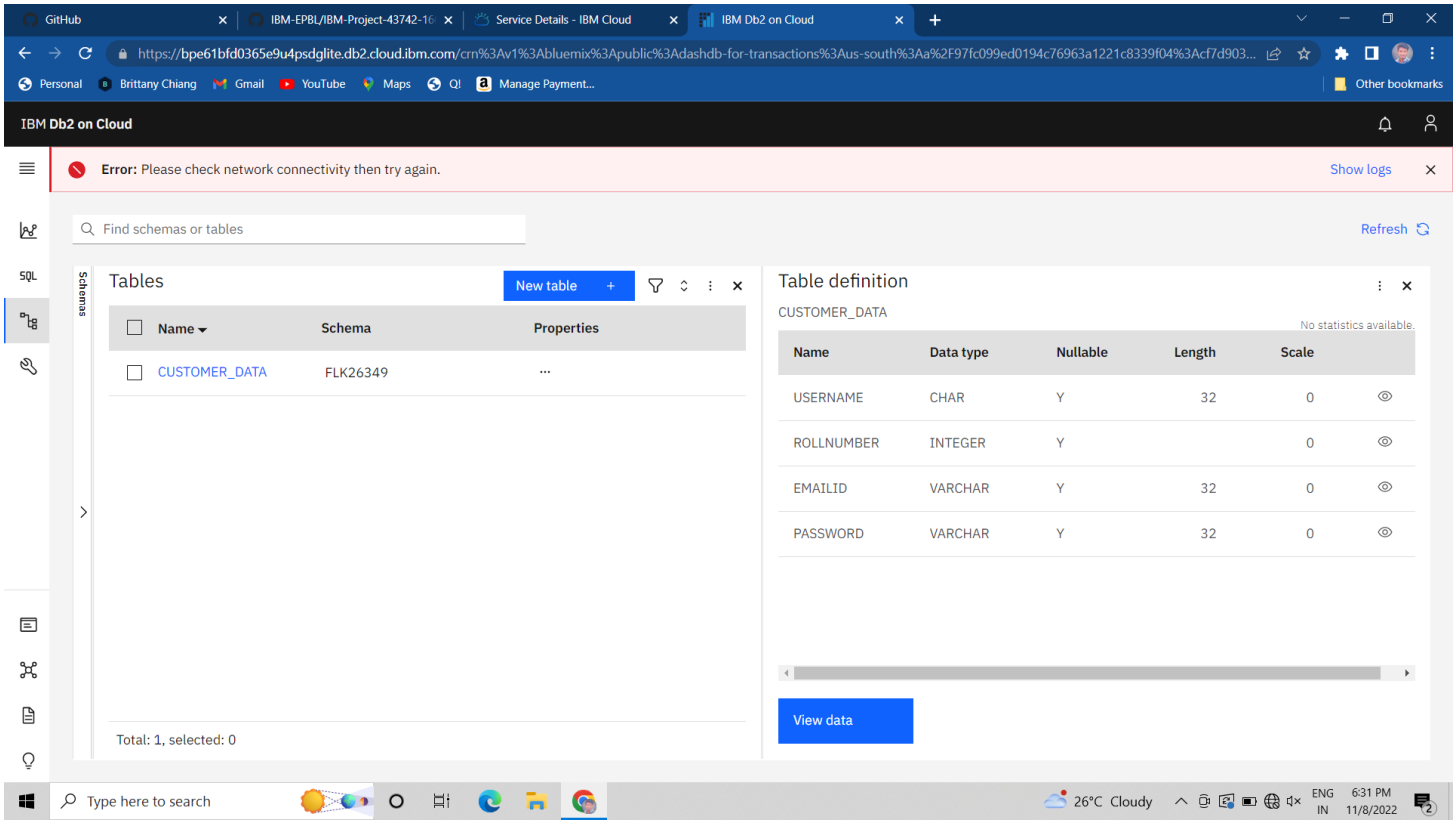
**Assignment -2**

IBM CLOUD DB2

| | |
|---|---|
| Assignment Date | 1 OCTOBER 2022 |
| Student Name | RAGUL R |
| Student Roll Number | 812019205030 |
| Maximum Marks | 2 Marks |

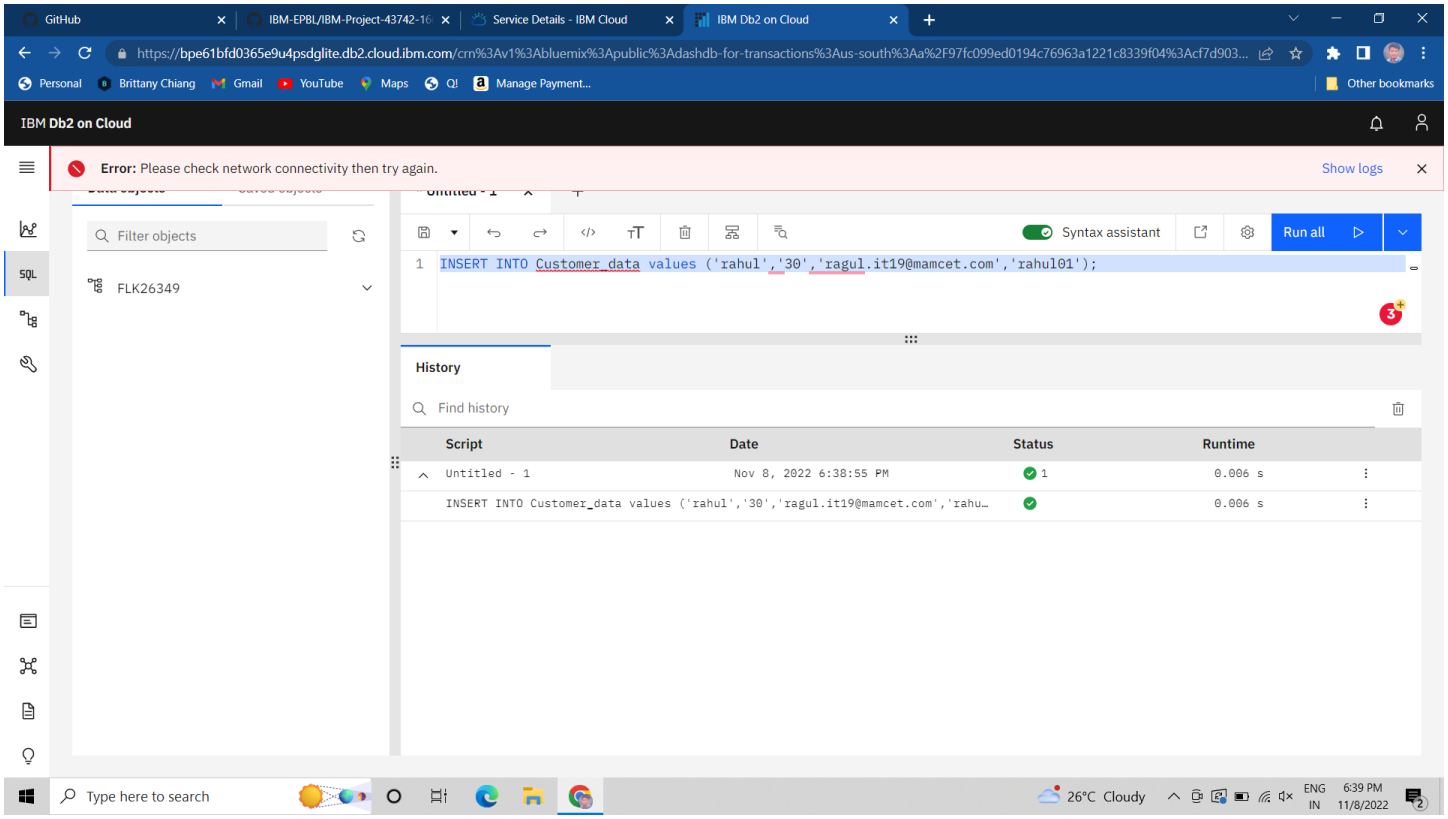
**Questions:**

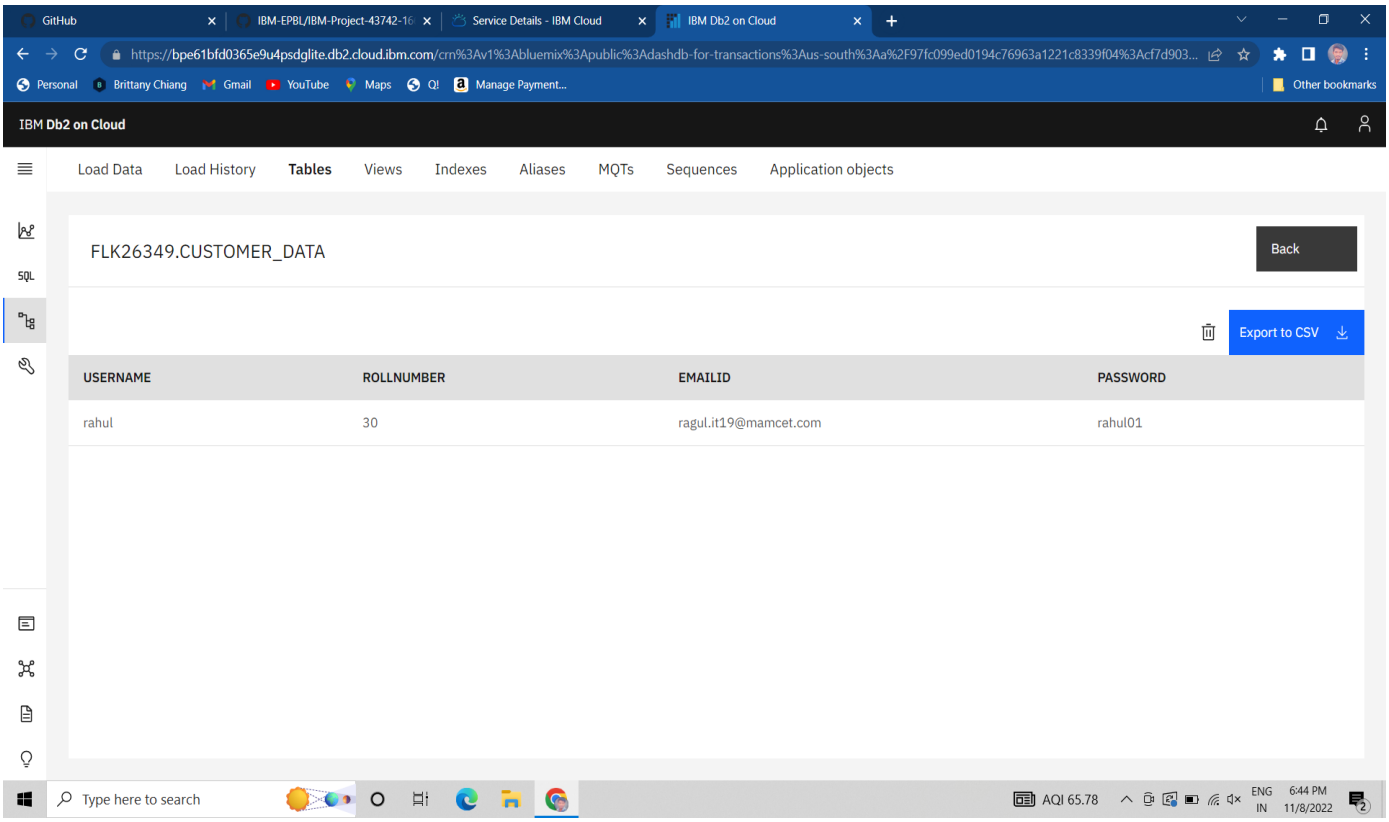
- Create User table with user with email, username, roll number, password.



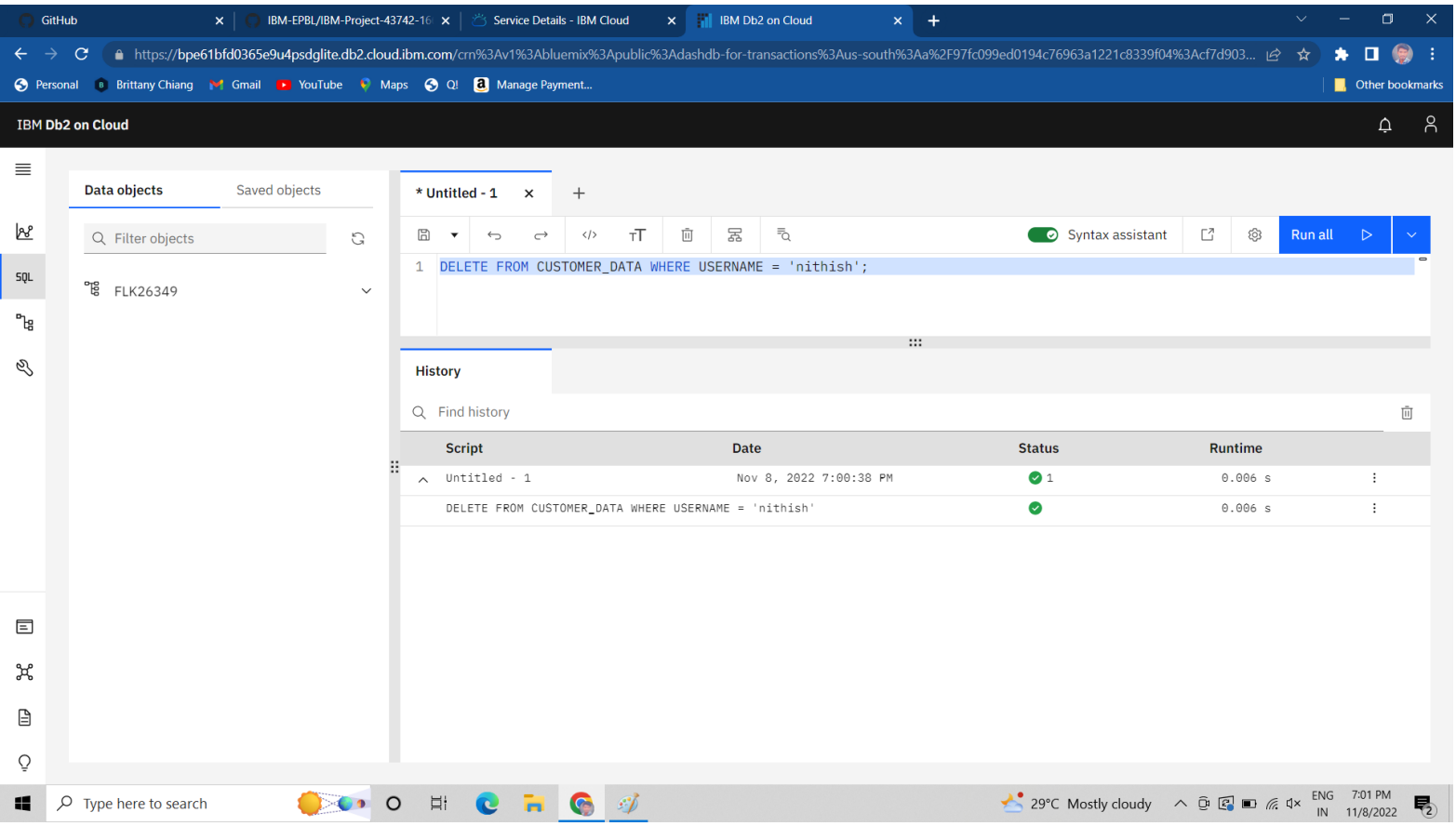
- Perform UPDATE,DELETE Queries with user table.

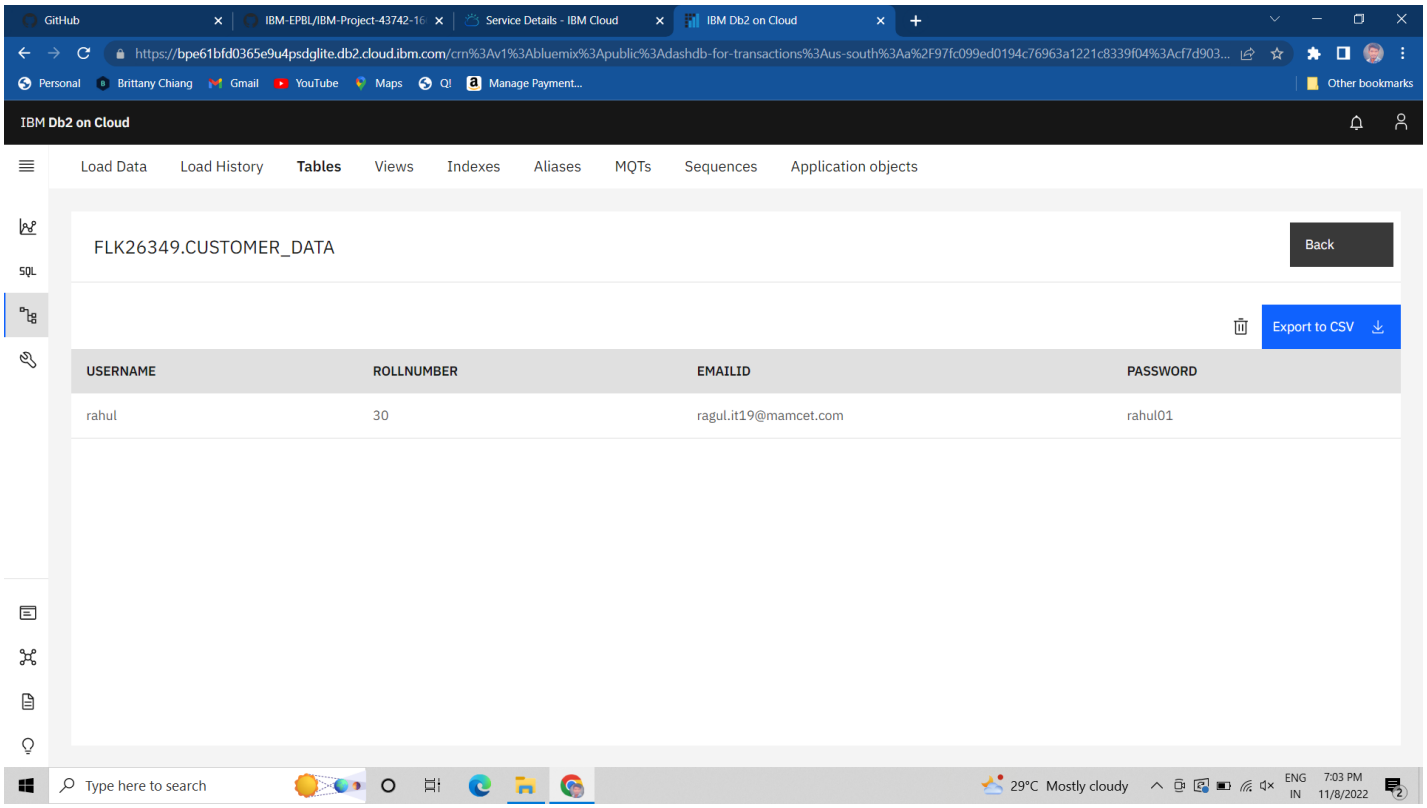
**UPDATE TABLE:-**




GitHub
IBM-EPBL/IBM-Project-43742-16
Service Details - IBM Cloud
IBM Db2 on Cloud
https://bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2F97fc099ed0194c76963a1221c8339f04%3Acf7d903...
Personal
Brittany Chiang
Gmail
YouTube
Maps
Manage Payment...
Other bookmarks
IBM Db2 on Cloud
Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
FLK26349.CUSTOMER_DATA
Back
Export to CSV
USERNAME
ROLLNUMBER
EMAILID
PASSWORD
rahul
30
ragul.it19@mamcet.com
rahul01
Type here to search
AQI 65.78
ENG
IN
6:44 PM
11/8/2022

**DELETE TABLE:-**






**3.CONNECT PYTHON CODE TO DB2.**

**NOTE:- Question 4 contains Question 3 answer**

4. Create a flask app with registration page, login page and welcome page. By default load the registration page once the user enters all the fields store the data in database and navigate to login page authenticate user username and password. If the user is valid show the welcome page

**Answers**

**App.py**

```
from flask import Flask, render_template, request,
redirect, url_for, session import ibm_db import re   app =
Flask(__name__)   app.secret_key =
'a'   conn
=
ibm_db.connect(
    "DATABASE=Db2New;HOSTNAME=19af6446-6171-4641-8aba-
9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;USERNAM
E=kgb92696;PASSWORD=3IlWyGYU3bMIvwnX;SECURITY=SSL;SSLSERVERCERTIFICATE=DigiCer
tGlobalRootCA.crt;", "", "")

@app.route("/", methods=['GET', 'POST'])
def register():     msg = ''     if
request.method == 'POST':
        username = request.form['username']
email = request.form['email']         password =
request.form['password']         sql = "SELECT * FROM
users WHERE username =?"         stmt =
ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt, 1, username)
ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
print(account)         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 = 'name must contain only characters and numbers !'
```

```
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prep_stmt, 1, username)
ibm_db.bind_param(prep_stmt, 2, email)
ibm_db.bind_param(prep_stmt, 3, password)
ibm_db.execute(prep_stmt)             msg = 'You have successfully registered
!'      elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg=msg)

@app.route('/login', methods=['GET', 'POST']) def login():      global userid
msg = ''       if request.method == 'POST':
username = request.form['username']          password =
request.form['password']          sql = "SELECT * FROM users WHERE username =?
AND password=?"          stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt, 1, username)          ibm_db.bind_param(stmt,
2, password)          ibm_db.execute(stmt)          account =
ibm_db.fetch_assoc(stmt)          print(account)          if account:
            session['loggedin'] = True              session['id']
= account['USERNAME']              userid
= account['USERNAME']              session['username'] =
account['USERNAME']             msg = 'Logged in successfully
!'


            msg = 'Logged in successfully !'              return
render_template('dashboard.html', msg=msg)          else:
            msg = 'Incorrect username / password !'      return
render_template('login.html', msg=msg)


if __name__ == '__main__':
app.run(host='0.0.0.0')
    # app.run(debug=True)
```

**Signup.html**

```
<!DOCTYPE html>
<head>
<title>signup page</title>
<link href={{url_for('static',filename='signup.css')}} rel="stylesheet">
</head>
<body >
```

```
<div id="form">
<form action="/login" method="post"  style="border:1px solid #ccc">
    <div class="container">
      <h1>Sign Up</h1>
      <hr>
      <label for="uname"><b>Username</b></label>
      <input type="text" placeholder="Enter username" name="uname" required>
      <label for="roll"><b>Roll Number</b></label>
      <input type="text" placeholder="Enter roll number" name="roll" required>
      <label for="email"><b>Email</b></label>
      <input type="text" placeholder="Enter Email" name="email" required>
      <label for="psw"><b>Password</b></label>
      <input type="password" placeholder="Enter Password" name="psw" required>

      <label>
        <input type="checkbox" checked="checked" name="remember" style="margin-bottom:15px">
      </label>

      <div class="clearfix">
        <button type="button" class="cancelbtn">Cancel</button>
        <button type="submit" class="signupbtn">Sign Up</button><br><br
        <p>Already registered user <a href="{{ url_for('signin') }}">signin</a></p>
      </div>
    </div>
  </form></div></body>
```

```
<!DOCTYPE html>
<head>
    <title>Signin page</title>
    <link href={{url_for('static',filename='signup.css')}} rel="stylesheet">
</head><div id="for">
<form action="action_page.php" method="post" id="forme" >
    <div class="imgcontainer">
    </div>

    <div class="container">
        <h1>Sign in</h1>
      <label for="email"><b>Email</b></label>
      <input type="text" placeholder="Enter email" name="email" required>
      <label for="psw"><b>Password</b></label>
```

**Login.html**

```
    <input type="password" placeholder="Enter Password" name="psw" required>
    <button type="submit">sign in</button>
    <label>
      <input type="checkbox" checked="checked" name="remember"> Remember me
    </label>
  </div>
```

```
    <div class="container" style="background-color:#f1f1f1">
      <button type="button" class="cancelbtn">Cancel</button>      </div>
  </form></div>
```

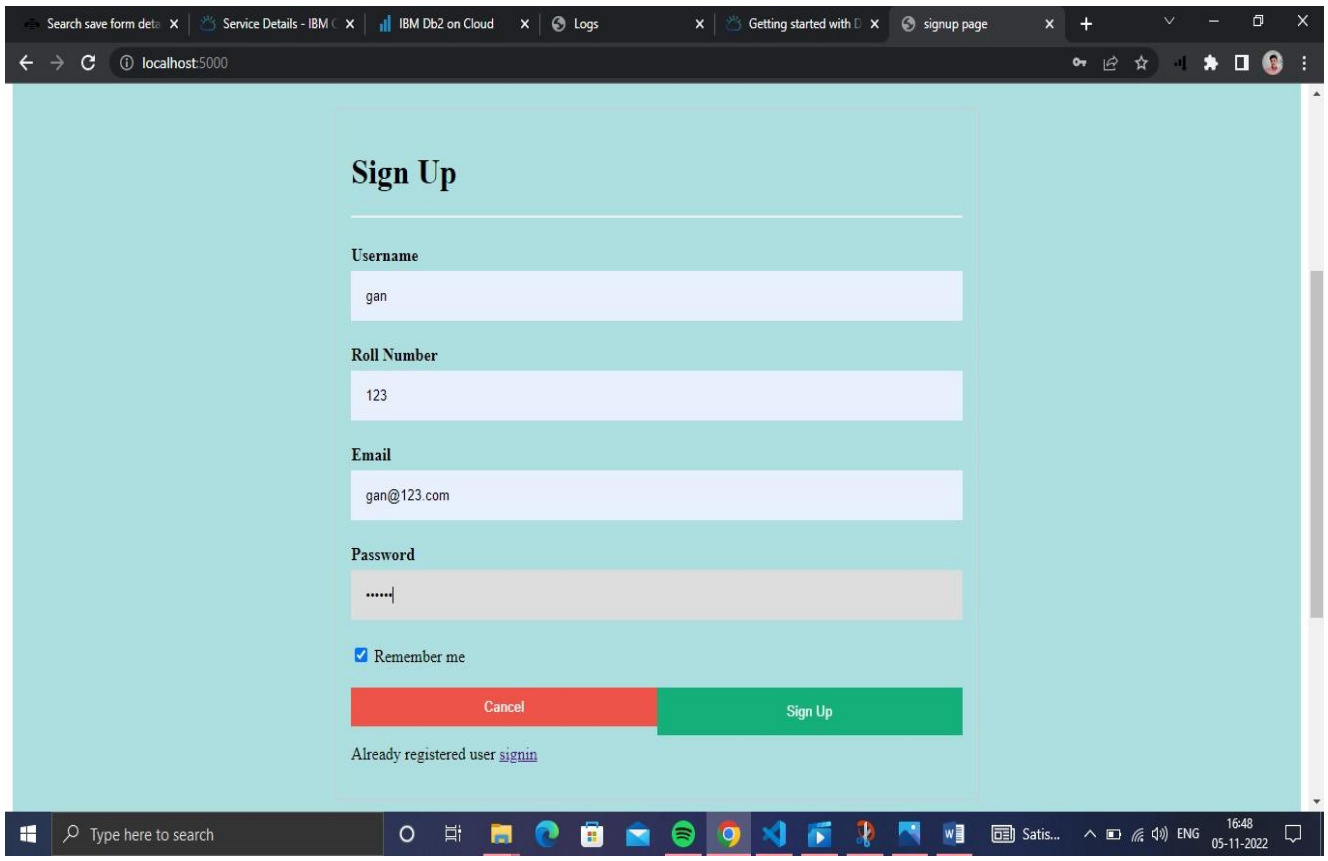
**Style.css**

```
* {box-sizing: border-box}

/* Full-width input fields */
input[type=text], input[type=password] {
width: 100%;   padding: 15px;   margin:
5px 0 22px 0;   display: inline-block;
border: none;   background: #f1f1f1;
}  input[type=text]:focus,
input[type=password]:focus {   background-color:
#ddd;   outline: none;
}  hr {   border: 1px solid
#f1f1f1;   margin-bottom:
25px;
}

/* Set a style for all buttons
*/ button {   background-color:
#04AA6D;   color: white;
padding: 14px 20px;   margin:
8px 0;   border: none;   cursor:
pointer;   width: 50%;
opacity: 0.9;
```

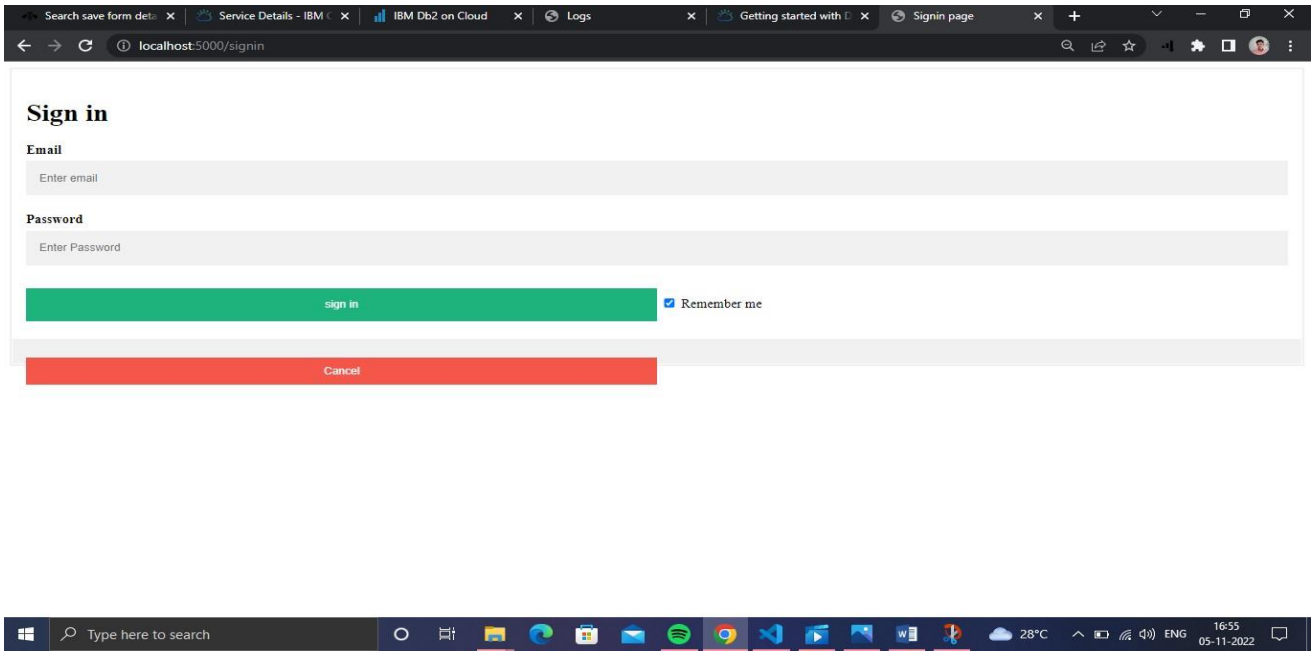
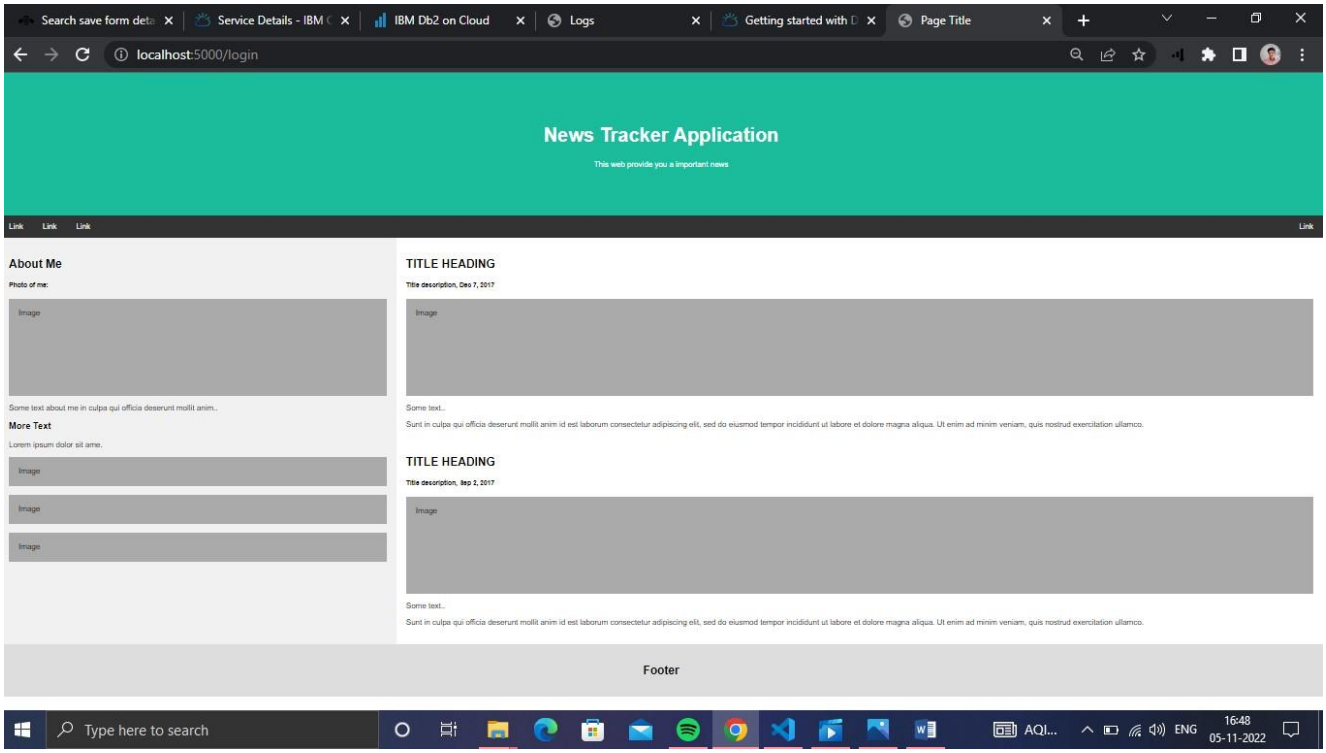
```
}
button:hover {
opacity:1;
}

/* Extra styles for the cancel button */
.cancelbtn {   padding:
14px 20px;   background-
color: #f44336;
}

/* Float cancel and signup buttons and add an equal width */
.cancelbtn, .signupbtn
{   float: left;
width: 50%;
}

/* Add padding to container elements */
.container {
padding: 16px;
}

/* Clear floats */
.clearfix::after {
content: "";   clear:
both;   display:
table;
}

/* Change styles for cancel button and signup button on extra small screens */
@media screen and (max-width: 300px) {
  .cancelbtn, .signupbtn {
width: 50%;
  }
}
/* Bordered form */ form {
border: 3px solid #f1f1f1;
  }




  /* Extra style for the cancel button (red) */   .cancelbtn
{
```

```
    width: 50%;     padding:
10px 18px;     background-
color: #f44336;
  }



  /* Add padding to containers */
  .container {
padding: 16px;
  }



  /* Change styles for span and cancel button on extra small screens */
  @media screen and (max-width: 300px)
{     span.psw {       display: block;
float: none;
    }
    .cancelbtn {
width: 50%;
    }
  }   #form{     padding: 25%;
background-color: rgb(172, 223, 223);
  }   #forme {
padding: 25%;
  }
```

**OUTPUT:- SIGNUP PAGE:-**




**SIGNIN PAGE:-**



**WELCOME PAGE:-**