

CUSTOMER CARE REGISTRY

TEAM ID:PNT2022TMID37726

AGENT LOGIN. HTML:

```
<!DOCTYPE
html>

<html>
<head>
<link rel="stylesheet" href="https://static86.s3.jp-tok.cloud-object-
storage.appdomain.cloud/mystyle1.css">
</head>
<body>

<form action="action_page.php" method="post">
  <div class="imgcontainer">
    
  </div>

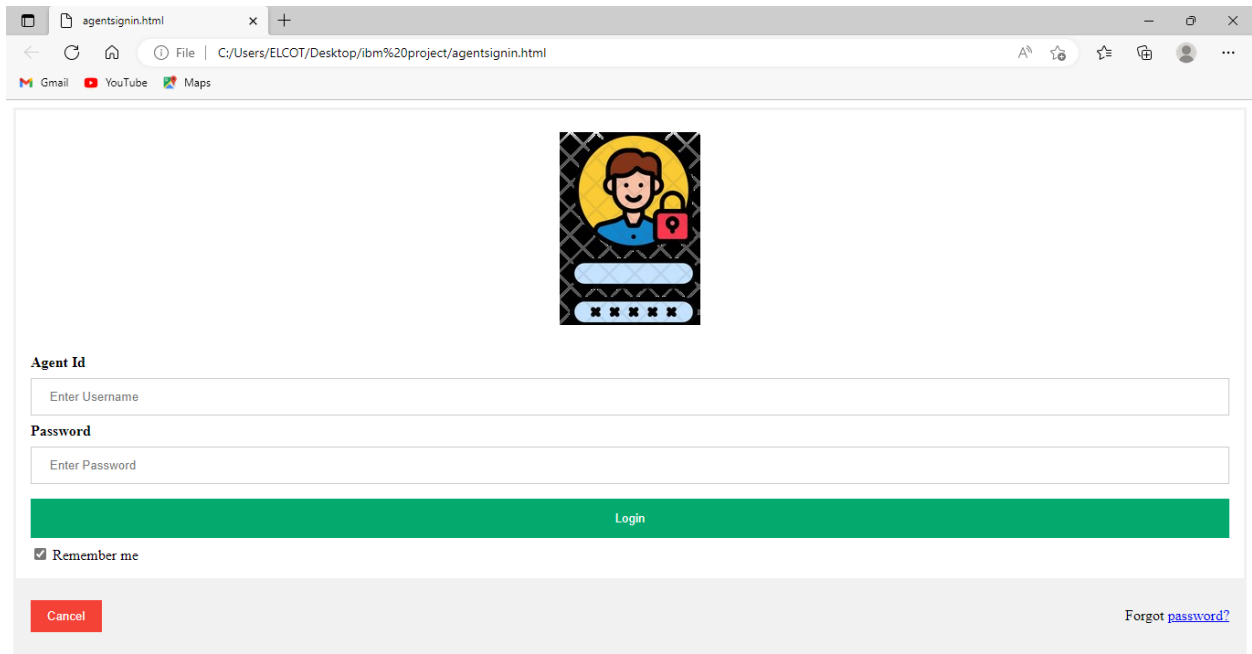
  <div class="container">
    <label for="uname"><b>Agent Id</b></label>
    <input type="text" placeholder="Enter Username" name="uname" required>

    <label for="psw"><b>Password</b></label>
    <input type="password" placeholder="Enter Password" name="psw" required>

    <button type="submit">Login</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>
    <span class="psw">Forgot <a href="#">password?</a></span>
  </div>
</form>
</body>
</html>
```

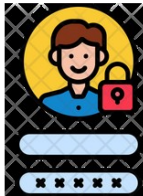
AGENT LOGIN PNG:



agentsignin.html

File | C:/Users/ELCOT/Desktop/ibm%20project/agentsignin.html

Gmail YouTube Maps



Agent Id

Enter Username

Password

Enter Password

Login

☒ Remember me

Cancel

Forgot [password?](#)

APP.PY:

```
1 from __future__ import print_function
2 from audioop import add
3 import datetime
4 from unicodedata import name
5 from sib_api_v3_sdk.rest import ApiException
6 from pprint import pprint
7 from flask import Flask, render_template, request, redirect, url_for, session, flash
8 from markupsafe import escape
9 from flask import *
10 import ibm_db
11 import sib_api_v3_sdk
12 from init import randomnumber
13 from init import id
14 from init import hello
15
16 import datetime
17
18
19
20 conn = ibm_db.connect("DATABASE=b1udb;HOSTNAME="";PORT="";SECURITY=SSL;SSLServerCertificate="";UID="";PID="";', '')
21 print(conn)
22 print("connection successful...")
23
24 app = Flask(__name__)
25 app.secret_key = 'your secret key'
26
27
28 @app.route('/')
```

```

29 def home():
30     message = "TEAM ID : PNT2022THIO37544" + " " + "BATCH ID : B1-1H3E "
31     return render_template('index.html',mes=message)
32
33
34 @app.route('/signinpage', methods=['POST', 'GET'])
35 def signinpage():
36     return render_template('signinpage.html')
37
38
39 @app.route('/agentsigin', methods=['POST', 'GET'])
40 def agentsigin():
41
42     return render_template('signinpageagent.html')
43
44
45 @app.route('/signuppage', methods=['POST', 'GET'])
46 def signuppage():
47     return render_template('signuppage.html')
48
49
50 @app.route('/agentRegister', methods=['POST', 'GET'])
51 def agentRegister():
52     return render_template('agentregister.html')
53
54
55
56 @app.route('/forgotpass', methods=['POST', 'GET'])
57 def forgotpass():
58     return render_template('forgot.html')
59
60
61 @app.route('/newissue/<name>', methods=['POST', 'GET'])

```

```

62 def newissue(name):
63     name = name
64     return render_template('complaint.html',msg=name)
65
66
67 @app.route('/forgot', methods=['POST', 'GET'])
68 def forgot():
69
70     try:
71         global randomnumber
72         ida = request.form['custid']
73         print(ida)
74         global id
75         id = ida
76         sql = "SELECT EMAIL,NAME FROM Customer WHERE id=?"
77         stmt = ibm_db.prepare(conn, sql)
78         ibm_db.bind_param(stmt, 1, ida)
79         ibm_db.execute(stmt)
80         emailif = ibm_db.fetch_both(stmt)
81         while emailif != False:
82             e = emailif[0]
83             n = emailif[1]
84             break
85
86         configuration = sib_api_v3_sdk.Configuration()
87         configuration.api_key['api-key'] = ""
88
89         api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
90             sib_api_v3_sdk.ApiClient(configuration))

```

```

91     subject = "Verification for Password"
92     html_content = "<html><body><h1>Your verification Code is : <h2>" + \
93         str(randomnumber)+"</h2> </h1> </body></html>"
94     sender = {"name": "IBM CUSTOMER CARE REGISTRY",
95              "email": "ibmdemo6@yahoo.com"}
96     to = [{"email": e, "name": n}]
97     reply_to = {"email": "ibmdemo6@yahoo.com", "name": "IBM"}
98     headers = {"Some-Custom-Name": "unique-id-1234"}
99     params = {"parameter": "My param value",
100             "subject": "Email Verification"}
101     send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
102         to=to, reply_to=reply_to, headers=headers, html_content=html_content, params=params, sender=sender, subject=subject)
103
104     api_response = api_instance.send_transac_email(send_smtp_email)
105
106     pprint(api_response)
107     message = "Email send to:"+e+" for password"
108     flash(message, "success")
109
110     except ApiException as e:
111         print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
112         flash("Error in sending mail")
113     except:
114         flash("Your didn't Signin with this account")
115     finally:
116         return render_template('forgot.html')
117
118
119 @app.route('/verifyemail', methods=['POST', 'GET'])
120 def verifyemail():

```

```

121     try:
122         email = request.form['verifyemail']
123         sql = "SELECT ID,NAME FROM Customer WHERE email=?"
124         stmt = ibm_db.prepare(conn, sql)
125         ibm_db.bind_param(stmt, 1, email)
126         ibm_db.execute(stmt)
127         emailf = ibm_db.fetch_both(stmt)
128         while emailf != False:
129             id = emailf[0]
130             name = emailf[1]
131             break
132         configuration = sib_api_v3_sdk.Configuration()
133         configuration.api_key['api-key'] = ""
134
135         api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
136             sib_api_v3_sdk.ApiClient(configuration))
137         subject = "Regarding of your Customer Id"
138         html_content = "<html><body><h1>Your Customer Id is : <h2>" + \
139             str(id)+"</h2> </h1> </body></html>"
140         sender = {"name": "IBM CUSTOMER CARE REGISTRY",
141                  "email": "ibmdemo6@yahoo.com"}
142         to = [{"email": email, "name": name}]
143         reply_to = {"email": "ibmdemo6@yahoo.com", "name": "IBM"}
144         headers = {"Some-Custom-Name": "unique-id-1234"}
145         params = {"parameter": "My param value",
146                 "subject": "Email Verification"}
147         send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
148             to=to, reply_to=reply_to, headers=headers, html_content=html_content, params=params, sender=sender, subject=subject)
149
150         api_response = api_instance.send_transac_email(send_smtp_email)
151

```

```

152         pprint(api_response)
153         message = "Email send to:"+email+" for password"
154         flash(message, "success")
155
156     except ApiException as e:
157         print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
158         flash("Error in sending mail.")
159     except:
160         flash("Database not found in mail! Please Register Your account.", "danger")
161     finally:
162         return render_template('signinpage.html')
163
164
165 @app.route('/otp', methods=['POST', 'GET'])
166 def otp():
167     try:
168         otp = request.form['otp']
169         cusid = id
170         print(id)
171         sql = "SELECT PASSWORD FROM Customer WHERE id=?"
172         stmt = ibm_db.prepare(conn, sql)
173         ibm_db.bind_param(stmt, 1, cusid)
174         ibm_db.execute(stmt)
175         otpf = ibm_db.fetch_both(stmt)
176         while otpf != False:
177             verify = otpf[0]
178             break
179         if otp == str(randomnumber):
180             msg = "Your Password is "+verify+"
181             flash(msg, "success")

```

```

182         return render_template('forgot.html')
183     else:
184         flash("Wrong Otp", "danger")
185     finally:
186         return render_template('forgot.html')
187
188
189 @app.route('/admin', methods=['POST', 'GET'])
190 def admin():
191     userdatabase = []
192     sql = "SELECT * FROM customer"
193     stmt = ibm_db.exec_immediate(conn, sql)
194     dictionary = ibm_db.fetch_both(stmt)
195     while dictionary != False:
196         userdatabase.append(dictionary)
197         dictionary = ibm_db.fetch_both(stmt)
198     if userdatabase:
199         sql = "SELECT COUNT(*) FROM customer;"
200         stmt = ibm_db.exec_immediate(conn, sql)
201         user = ibm_db.fetch_both(stmt)
202
203     users = []
204     sql = "select * from ISSUE"
205     stmt = ibm_db.exec_immediate(conn, sql)
206     dict = ibm_db.fetch_both(stmt)
207     while dict != False:
208         users.append(dict)
209         dict = ibm_db.fetch_both(stmt)
210     if users:
211         sql = "SELECT COUNT(*) FROM ISSUE;"
212         stmt = ibm_db.exec_immediate(conn, sql)
213         count = ibm_db.fetch_both(stmt)
214
215     agent = []
216     sql = "SELECT * FROM AGENT"
217     stmt = ibm_db.exec_immediate(conn, sql)
218     dictionary = ibm_db.fetch_both(stmt)
219     while dictionary != False:
220         agent.append(dictionary)
221         dictionary = ibm_db.fetch_both(stmt)
222
223     if agent:
224         sql = "SELECT COUNT(*) FROM AGENT;"
225         stmt = ibm_db.exec_immediate(conn, sql)
226         cot = ibm_db.fetch_both(stmt)
227
228     return render_template("admin.html", complaint=users, users=userdatabase, agents=agent, message=user[0], issue=count[0], msgagent = cot[0])
229
230
231 @app.route('/remove', methods=['POST', 'GET'])
232 def remove():
233
234     otp = request.form['otp']
235     if otp == 'C':
236         try:
237             insert_sql = f"delete from customer"
238             prep_stmt = ibm_db.prepare(conn, insert_sql)
239             ibm_db.execute(prepare_stmt)
240             flash("deleted successfully the Customer", "success")
241         except:
242             flash("No data found in Customer", "danger")
243     finally:
244         return redirect(url_for('signuppage'))
245     if otp == 'A':
246         try:
247             insert_sql = f"delete from AGENT"
248             prep_stmt = ibm_db.prepare(conn, insert_sql)
249             ibm_db.execute(prepare_stmt)
250             flash("deleted successfully the Agents", "success")
251         except:
252             flash("No data found in Agents", "danger")
253     finally:
254         return redirect(url_for('signuppage'))
255
256     if otp == 'C':
257         try:
258             insert_sql = f"delete from AGENT"
259             prep_stmt = ibm_db.prepare(conn, insert_sql)
260             ibm_db.execute(prepare_stmt)
261             flash("deleted successfully the Complaints", "success")
262         except:
263             flash("No data found in Complaints", "danger")
264     finally:
265         return redirect(url_for('signuppage'))
266
267 @app.route('/login', methods=['GET', 'POST'])
268 def login():
269     if request.method == 'POST':
270         try:

```

```

271
272     id = request.form['idn']
273     global hello
274     hello = id
275     password = request.form['password']
276     print(id, password)
277     if id == '1111' and password == '1111':
278         return redirect(url_for('admin'))
279
280     sql = f"select * from customer where id='{escape(id)}' and password='{escape(password)}'"
281     stmt = ibm_db.exec_immediate(conn, sql)
282     data = ibm_db.fetch_both(stmt)
283
284     if data:
285         session["name"] = escape(id)
286         session["password"] = escape(password)
287         return redirect(url_for("welcome"))
288
289     else:
290         flash("Mismatch in credentials", "danger")
291     except:
292         flash("Error in Insertion operation", "danger")
293
294     return render_template('signinpage.html')
295
296 @app.route('/welcome', methods=['POST', 'GET'])
297 def welcome():
298     try:
299         id = hello
300         sql = "SELECT ID,DATE,TOPIC,SERVICE_TYPE,SERVICE_AGENT,DESCRIPTION,STATUS FROM ISSUE WHERE CUSTOMER_ID =?"
301         agent = []

```

```

302         stmt = ibm_db.prepare(conn, sql)
303         ibm_db.bind_param(stmt, 1, id)
304         ibm_db.execute(stmt)
305         otpf = ibm_db.fetch_both(stmt)
306         while otpf != False:
307             agent.append(otpf)
308             otpf = ibm_db.fetch_both(stmt)
309         sql = "SELECT COUNT(*) FROM ISSUE WHERE CUSTOMER_ID = ?"
310         stmt = ibm_db.prepare(conn, sql)
311         ibm_db.bind_param(stmt, 1, id)
312         ibm_db.execute(stmt)
313         t = ibm_db.fetch_both(stmt)
314         return render_template("welcome.html",agent=agent,message=t[0])
315     except:
316
317         return render_template("welcome.html")
318
319 @app.route('/loginagent', methods=['GET', 'POST'])
320 def loginagent():
321     if request.method == 'POST':
322         try:
323             global loginagent
324             id = request.form['idn']
325             loginagent = id
326             password = request.form['password']
327
328             sql = f"select * from AGENT where id='{escape(id)}' and password='{escape(password)}'"
329             stmt = ibm_db.exec_immediate(conn, sql)
330             data = ibm_db.fetch_both(stmt)
331
332             if data:

```

```

333                 session["name"] = escape(id)
334                 session["password"] = escape(password)
335                 return redirect(url_for("agentwelcome"))
336
337             else:
338                 flash("Mismatch in credentials", "danger")
339             except:
340                 flash("Error in Insertion operation", "danger")
341
342             return render_template("signinpageagent.html")
343
344 @app.route('/delete/<ID>')
345 def delete(ID):
346     sql = f"select * from customer where Id='{escape(ID)}'"
347     print(sql)
348     stmt = ibm_db.exec_immediate(conn, sql)
349     student = ibm_db.fetch_row(stmt)
350     if student:
351         sql = f"delete from customer where id='{escape(ID)}'"
352         stmt = ibm_db.exec_immediate(conn, sql)
353
354         flash("Deleted Successfully", "success")
355         return redirect(url_for("admin"))

```

CSS:

```
@import url('https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;500;600;700&display=swap');
```

```
*{  
  box-sizing: border-box;  
  padding: 0;  
  margin: 0;  
}  
  
body{  
  font-family: 'Playfair Display', serif;  
  display: grid;  
  background-color: #4158D0;  
  background-image: linear-gradient (43deg, #4158D0 0%, #C850C0 46%, #FFCC70 100%);  
  
  align-content: center;  
  min-height: 100vh;  
}  
  
section{  
  display: grid;  
  grid-template-columns: 1fr 1fr;  
  min-height: 70vh;  
  width: 75vw;  
  margin: 0 auto;  
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);  
  border-radius: 12px  
}  
  
.image{  
  background-color: #12192c;  
  display: flex;  
  border-radius: 12px 0 0 12px;  
}  
  
.image img{  
  height: 50vh;  
  margin: 50px auto  
}
```

```
.content{  
    background-color: #12192c;  
    display: flex;  
    justify-content: center;  
    flex-direction: column;  
    align-items: center;  
    border-radius: 0 12px 12px 0;  
    color: #fff;  
}
```

```
.content h2{  
    text-transform: uppercase;  
    font-size: 36px;  
    letter-spacing: 6px;  
    opacity: 0.9;  
}
```

```
.content span{  
    height: 0.5px;  
    width: 80px;  
    background: #777;  
    margin: 30px 0;  
}
```

```
.content p{  
    padding-bottom: 15px;  
    font-weight: 300;  
    opacity: 0.7;  
    width: 60%;  
    text-align: center;  
    margin: 0 auto;  
    line-height: 1.7;  
    color: #ffffff  
}
```

```
.links{  
    margin: 15px 0;  
}
```

```
.links li{  
    border: 2px solid #4158D0;
```



```
list-style: none;

border-radius: 5px;

padding: 10px 15px;

width: 160px;

text-align: center;
}

.links li a{

text-transform: uppercase;

color: #fff;

text-decoration: none;
}

.links li:hover{

border-color: #C850C0;
}
```

```
.vertical-line{

height: 30px;

width: 3px;

background: #C850C0;

margin: 0 auto;
}
```

```
.icons{

display: flex;

padding: 15px 0;
}
```

```
.icons li{

display: block;

padding: 5px;

margin: 5px;
}
```

```
.icons li i{

font-size: 26px;

opacity: 0.8;
}
```

```
.icons li i:hover{

color: #C850C0;
}
```

```
    cursor: pointer;
}
```

```
/******
```

```
@media(max-width: 900px){
  section{
    grid-template-columns: 1fr;
    width: 100%;
    border-radius: none;
  }
  .image{
    height: 100vh;
    border-radius: none;
  }
  .content{
    height: 100vh;
    border-radius: none;
  }
  .content h2{
    font-size: 20px;
    margin-top: 50px;
  }
  .content span{
    margin: 20px 0;
  }
  .content p{
    font-size: 14px;
  }
  .links li a{
    font-size: 14px;
  }
  .links{
    margin: 5px 0;
  }
}
```

```
.links li{
    padding: 6px 10px;
}
.icons li i{
    font-size: 15px;
}
}

.credit{
    text-align: center;
    color: #000;
    font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif;
}

.credit a{
    text-decoration: none;
    color: #000;
    font-weight: bold;
}
```

CUSTOMER LOGIN.HTML:

```
<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://static86.s3.jp-tok.cloud-object-storage.appdomain.cloud/mystyle1.css">

</head>

<body>

<form action="action_page.php" method="post">

<div class="imgcontainer">


```

</div>

<div class="container">

<label for="uname">User Id</label>

<input type="text" placeholder="Enter Username" name="uname" required>

<label for="psw">Password</label>

<input type="password" placeholder="Enter Password" name="psw" required>

<button type="submit">Login</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>

Forgot password?

</div>

</form>

</body>


</html>

CUSTOMER LOGIN.PNG:

usersignin.html x +

File | C:/Users/ELCOT/Desktop/lbm%20project/usersignin.html

Gmail YouTube Maps



User Id

Enter Username

Password

Enter Password

Login

☒ Remember me

Cancel

Forgot [password?](#)

