

ASSIGNMENT-2

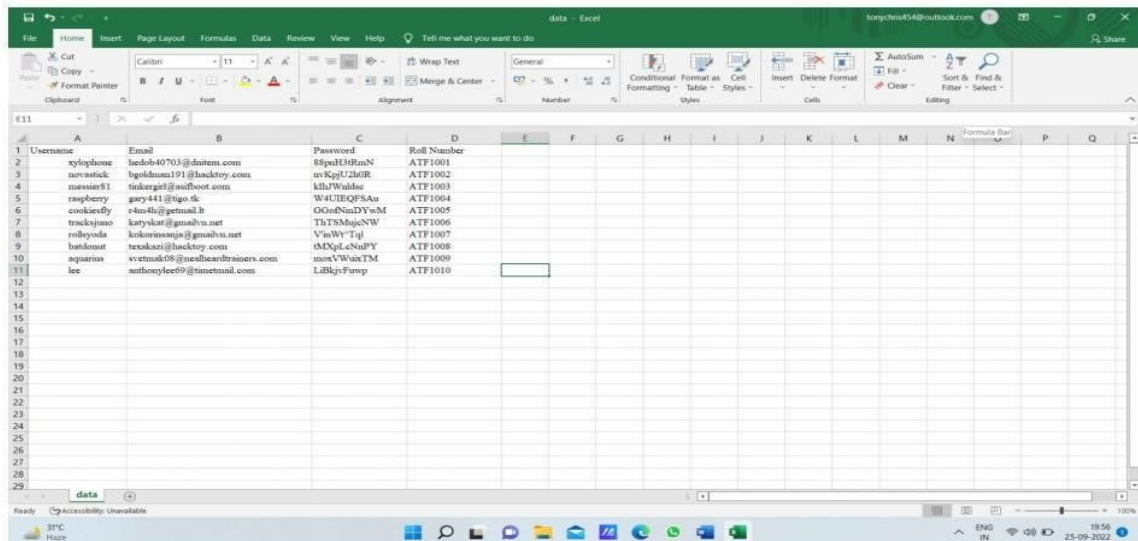
Assignment Date	19 September 2022
Student Name	M.Saraswathi
Student Roll Number	820419104062
Team ID	PNT2022TMID32929
Maximum Marks	2 Marks

Question-1:

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

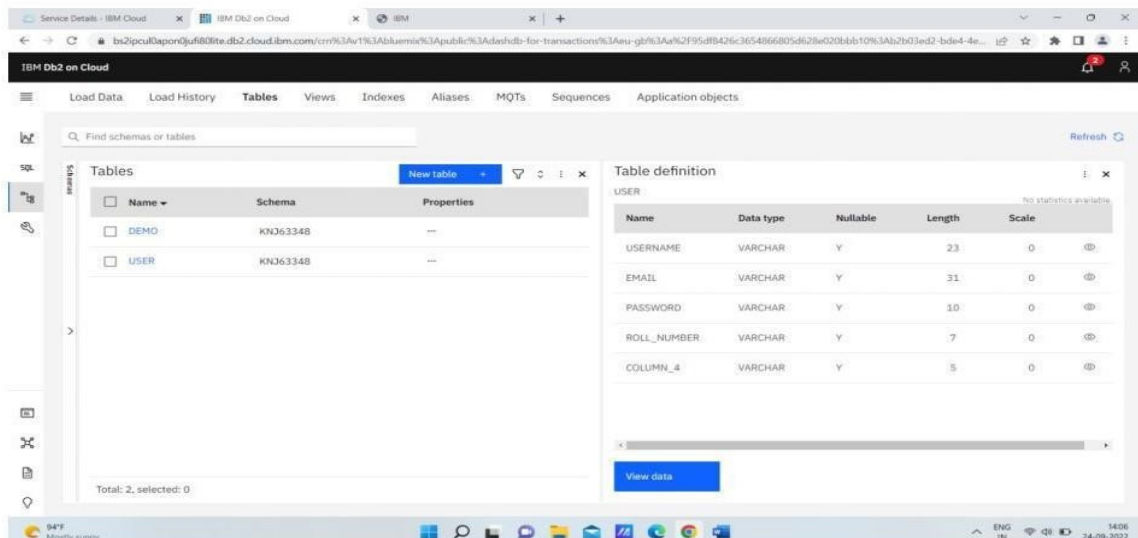
Solution:

Excel comma file



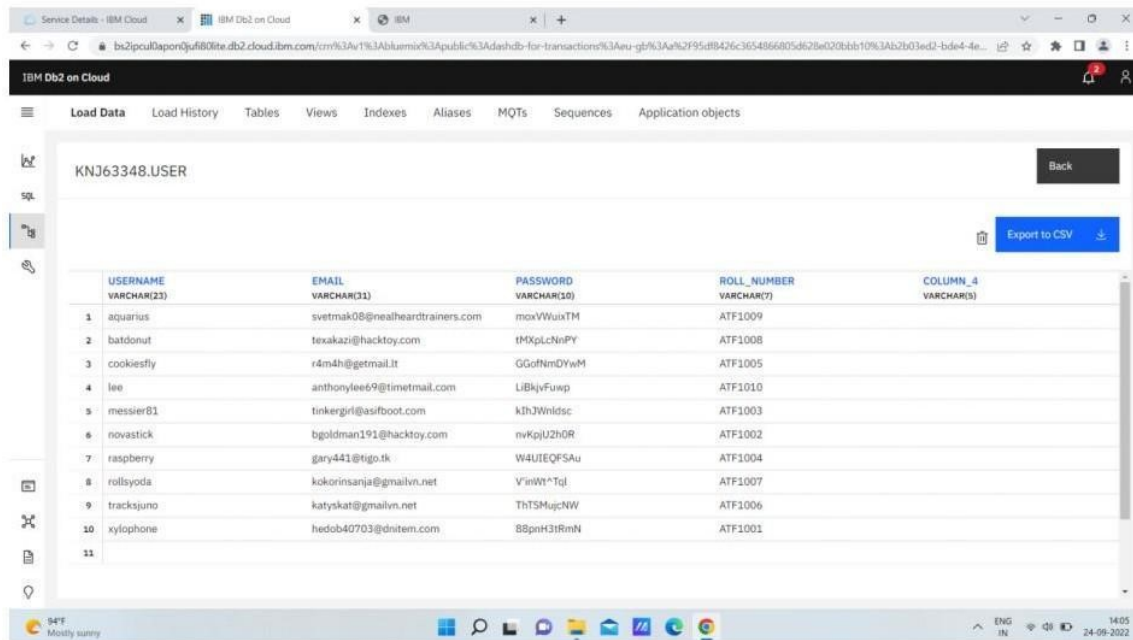
Username	Email	Password	Roll Number
xylophone	lacedo40703@duitem.com	8SpaH30RmN	ATF1001
servetick	bgolhuasi191@hacktoy.com	enKpL7Zb0R	ATF1002
mesier11	tukergd@asuloot.com	LiLWadles	ATF1003
raspberry	gary441@tigo.de	W4UEQFSAn	ATF1004
cookiesfly	r4m4lu@gmail.li	OGedNmDYwM	ATF1005
truckyjmo	kutyklat@gmail.ru.net	TbTSMajcNW	ATF1006
robbyoda	kokotomusju@gmail.ru.net	V5uWY7qj	ATF1007
baldomur	texakazi@hacktoy.com	tMXqLeNulPY	ATF1008
aquarius	svetmak08@senheardrainers.com	monVWuixTM	ATF1009
lee	anthonylee09@tinetmail.com	LBkyvFump	ATF1010

User table in Db2



Name	Data type	Nullable	Length	Scale
USERNAME	VARCHAR	Y	23	0
EMAIL	VARCHAR	Y	31	0
PASSWORD	VARCHAR	Y	10	0
ROLL_NUMBER	VARCHAR	Y	7	0
COLUMN_4	VARCHAR	Y	5	0

Data in User table



The screenshot shows the IBM Db2 on Cloud console interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The main content area displays the 'KNJ63348.USER' table. A 'Back' button is in the top right, and an 'Export to CSV' button is on the right side of the table. The table has five columns: USERNAME (VARCHAR(23)), EMAIL (VARCHAR(31)), PASSWORD (VARCHAR(30)), ROLL_NUMBER (VARCHAR(7)), and COLUMN_4 (VARCHAR(5)). The data is as follows:

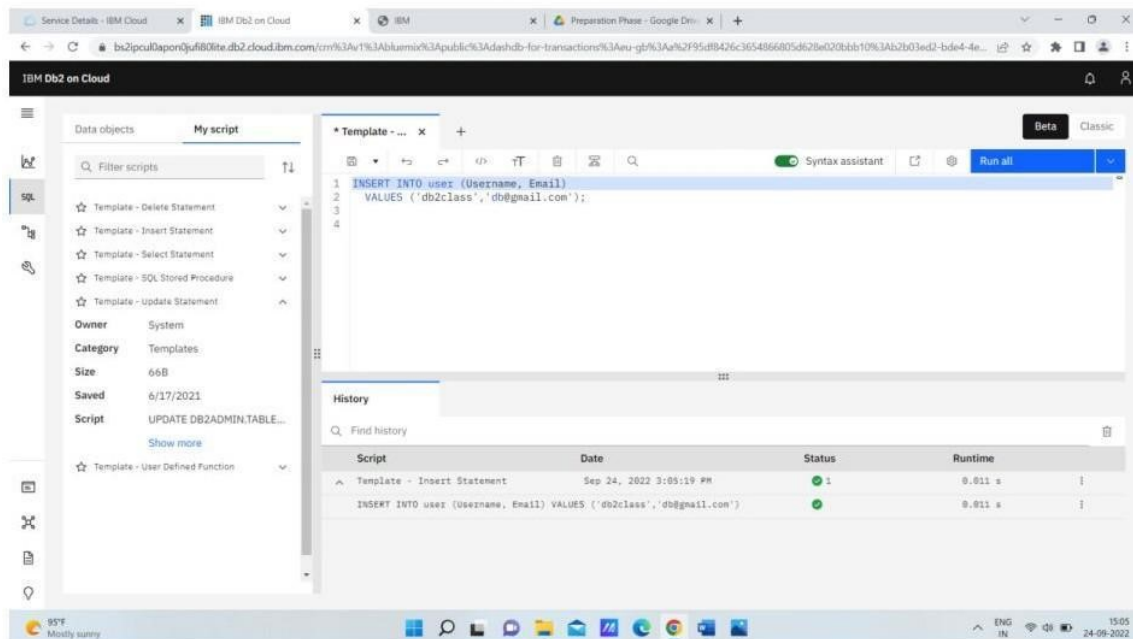
	USERNAME VARCHAR(23)	EMAIL VARCHAR(31)	PASSWORD VARCHAR(30)	ROLL_NUMBER VARCHAR(7)	COLUMN_4 VARCHAR(5)
1	aquarius	svetmak08@nealheardtrainers.com	moxVWuixTM	ATF1009	
2	batdonut	texakazi@hacktoy.com	IMXpLnPY	ATF1008	
3	cookiesfly	r4m4h@getmail.it	GGofNmDYwM	ATF1005	
4	lee	anthonylee69@timetmail.com	LiBkvFuwp	ATF1010	
5	messier81	tinkergirl@asifboot.com	kthJWldsc	ATF1003	
6	novastick	bgoldman191@hacktoy.com	nvKpjU2hOR	ATF1002	
7	raspberry	gary441@tigo.tk	W4UIEQFSau	ATF1004	
8	rollsodya	kokorinsanja@gmailvn.net	V'wWt*Tql	ATF1007	
9	tracksjuno	katyskat@gmailvn.net	ThT5MujcNW	ATF1006	
10	xylophone	hedob40703@dnitem.com	B8pnH3IRmN	ATF1001	
11					

Question-2:

Perform INSERT, SELECT, DELETE Queries in user table

Solution:

Insert Query



The screenshot shows the IBM Db2 on Cloud console interface. The top navigation bar includes 'Data objects', 'My script', and 'Preparation Phase - Google Drive'. The main content area displays the 'My script' editor. The script contains the following SQL query:

```
1 INSERT INTO user (Username, Email)
2 VALUES ('db2class', 'db@gmail.com');
3
4
```

The 'History' tab is selected, showing the execution details of the script:

Script	Date	Status	Runtime
Template - Insert Statement	Sep 24, 2022 3:05:19 PM	1	0.011 s
INSERT INTO user (Username, Email) VALUES ('db2class', 'db@gmail.com')			0.011 s

Insert Query Output

Service Details - IBM Cloud x IBM Db2 on Cloud x IBM x Preparation Phase - Google Drive x +

bs2ipculdaponjuf680ite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transaction%3Aeu-gb%3Aa%2F95d8426c3654866805d628e020bbb10%3Ab2b3ed2-bde4-4e...

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

KNJ63348,USER

Back

Export to CSV

USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
aquarius	svetmak08@nealheardtrainers.com	moxWvixTM	ATF1009	
batdonut	texakazi@hacktoy.com	IMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGoNmDyWm	ATF1005	
db2class	db@gmail.com			
lee	anthonylee69@timetmail.com	LIBkxFuwp	ATF1010	
messier81	tinkergirl@asifboot.com	kHjWnldsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpiU2hOR	ATF1002	
raspberry	gary441@tigo.tk	W4UITEQFSau	ATF1004	

95°F Partly sunny

ENG IN 15:36 24-09-2022

Update Query

Service Details - IBM Cloud x IBM Db2 on Cloud x IBM x Preparation Phase - Google Drive x +

bs2ipculdaponjuf680ite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transaction%3Aeu-gb%3Aa%2F95d8426c3654866805d628e020bbb10%3Ab2b3ed2-bde4-4e...

IBM Db2 on Cloud

Data objects My script

Filter scripts

- Template - Delete Statement
- Template - Insert Statement
- Template - Insert Statement
- Template - Select Statement
- Template - SQL Stored Procedure
- Template - Update Statement

Owner System

Category Templates

Size 66B

Saved 6/17/2021

Script UPDATE DB2ADMIN.TABLE...

Show more

Template - User Defined Function

Template - Insert ...

* Template - ...

Beta Classic

Syntax assistant Run all

```
1 UPDATE user;
2 SET Username = 'anto'
3 WHERE Username='lee';
4
```

Line: 4

History

Find history

Script	Date	Status	Runtime
Template - Update Statement	Sep 24, 2022 3:14:06 PM	1	0.005 s
UPDATE user SET Username = 'anto' WHERE Username='lee'			0.005 s

95°F Mostly sunny

ENG IN 15:34 24-09-2022

Update Query Output

Service Details - IBM Cloud x IBM Db2 on Cloud x IBM x Preparation Phase - Google Drive x +

bs2ipculdaponjuf80ite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transaction%3Aeu-gb%3Aa%2F95d8426c3654866805d628e020bb10%3Ab2b3e42-bde4-4e...

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

KNJ63348.USER Back

Export to CSV

USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
anto	anthonylee69@tinetmail.com	LiBkjvFuwp	ATF1010	
aquarius	svetmak08@nealheardtrainers.com	moxVWuxTM	ATF1009	
batdonut	texakazi@hacktoy.com	tMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGofNmDYwM	ATF1005	
db2class	db@gmail.com			
messier81	tinkergirl@asifboot.com	kthJWnldsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpU2h0R	ATF1002	
raspberry	gary441@tigo.tk	W4UIEQFSAu	ATF1004	

95°F Partly sunny 15:38 24-09-2022

Delete Query

bs2ipculdaponjuf80ite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transaction%3Aeu-gb%3Aa%2F95d8426c3654866805d628e020bb10%3Ab2b3e42-bde4-4e...

IBM Db2 on Cloud

Data objects My script

Filter scripts

- Template - Delete Statement
- Template - Insert Statement
- Template - Insert Statement
- Template - Select Statement
- Template - SQL Stored Procedure
- Template - Update Statement

Owner System

Category Templates

Size 66B

Saved 6/17/2021

Script UPDATE DB2ADMIN.TABLE...

Show more

Template - User Defined Function

Template - Insert ... * Template - Upda... * Template - ... x +

Beta Classic

Syntax assistant Run all

```
1 DELETE FROM user
2 WHERE Username = 'db2class';
3
```

History Results

Find history

Script	Date	Status	Runtime
Template - Delete Statement	Sep 24, 2022 3:38:56 PM	✓ 1	0.009 s
DELETE FROM user WHERE Username = 'db2class'		✓	0.009 s

Delete Query Output

bs2ipculdaponjuf80ite.db2.cloud.ibm.com/crm%3Av1%3Abluemix%3Apublic%3Adashdb-for-transaction%3Aeu-gb%3Aa%2F95d8426c3654866805d628e020bb10%3Ab2b3e42-bde4-4e...

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

KNJ63348.USER Back

Export to CSV

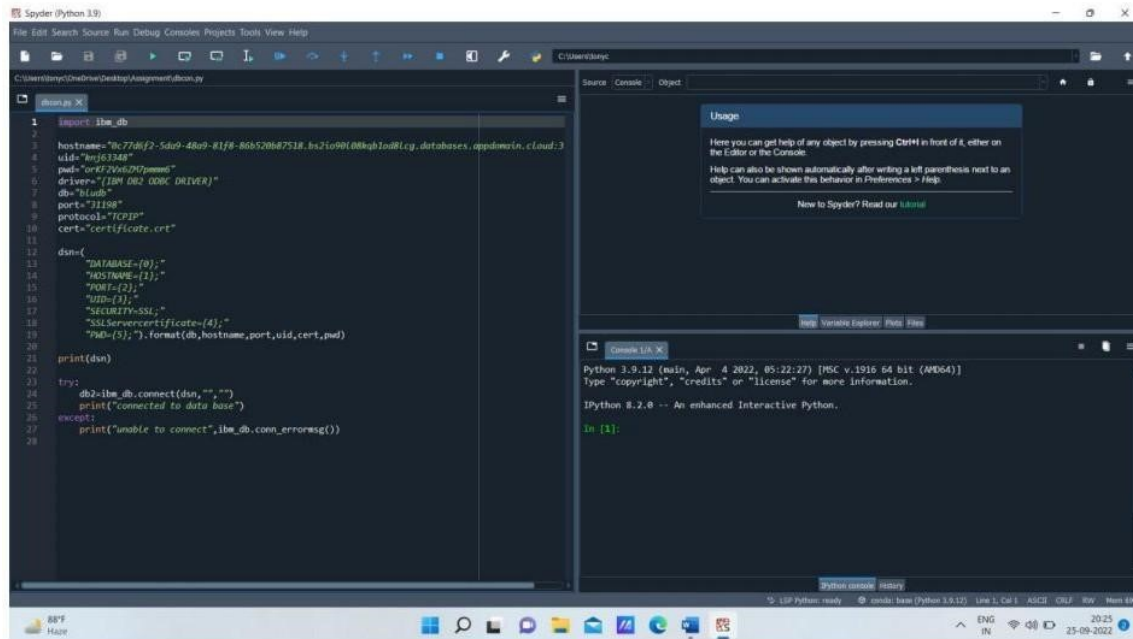
USERNAME	EMAIL	PASSWORD	ROLL_NUMBER	COLUMN_4
anto	anthonylee69@tinetmail.com	LiBkjvFuwp	ATF1010	
aquarius	svetmak08@nealheardtrainers.com	moxVWuxTM	ATF1009	
batdonut	texakazi@hacktoy.com	tMXpLcNnPY	ATF1008	
cookiesfly	r4m4h@gmail.it	GGofNmDYwM	ATF1005	
messier81	tinkergirl@asifboot.com	kthJWnldsc	ATF1003	
novastick	bgoldman191@hacktoy.com	nvKpU2h0R	ATF1002	
raspberry	gary441@tigo.tk	W4UIEQFSAu	ATF1004	
rollyoda	kokorinsanja@gmailvn.net	V'inWn^Tql	ATF1007	

Question-3:

Connect Python to DB2

Solution:

Source Code for Python to DB2 Connection



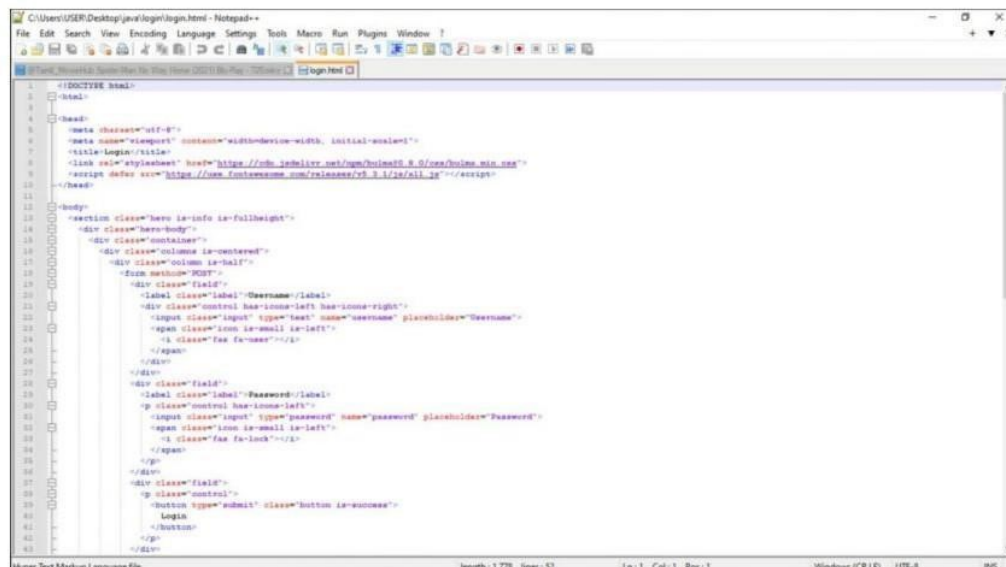
```
1 import ibm_db
2
3 hostname="bc77d6f2-5da9-48a9-81f8-86b520b87518.hs2io90l00qph1ndllcy.databases.appdomain.cloud:3
4 uid="my@3245"
5 pwd="mK7Zv6Zm7pmms"
6 driver="IBM DB2 ODBC DRIVER"
7 db="bluadb"
8 port="21198"
9 protocol="TCP/IP"
10 cert="certificate.crt"
11
12 dsn={
13     "DATABASE":(0),"
14     "HOSTNAME":(1),"
15     "PORT":(2),"
16     "UID":(3),"
17     "SECURITY-SSL:"
18     "SSLServerCertificate=(4),"
19     "PWD=(5);".format(db,hostname,port,uid,cert,pwd)
20 }
21 print(dsn)
22
23 try:
24     db2=ibm_db.connect(dsn,"","")
25     print("connected to data base")
26 except:
27     print("unable to connect",ibm_db.conn_errorsg())
28
```

Question-4:

Create a flask app with 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

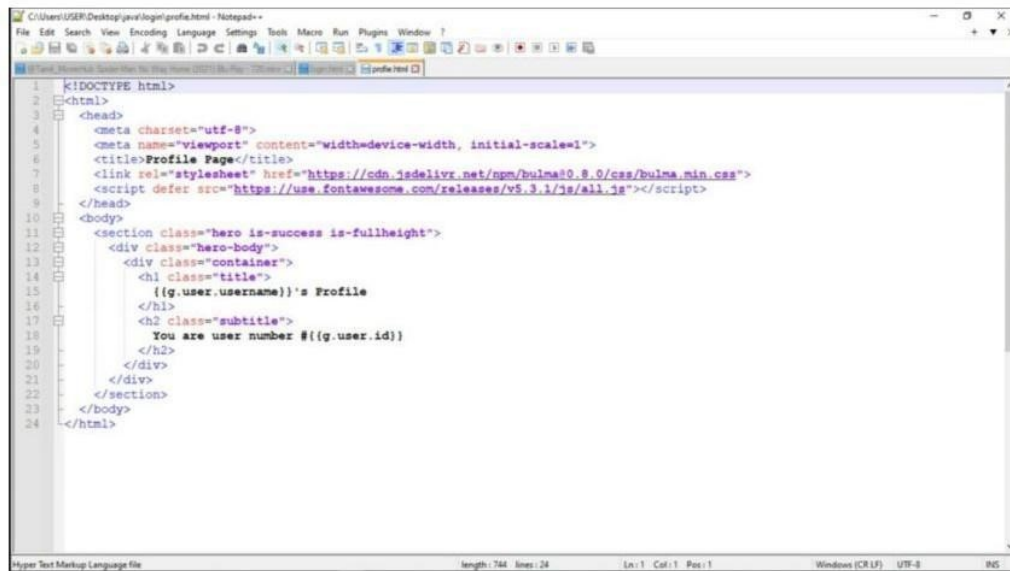
Solution:

Source code for login page



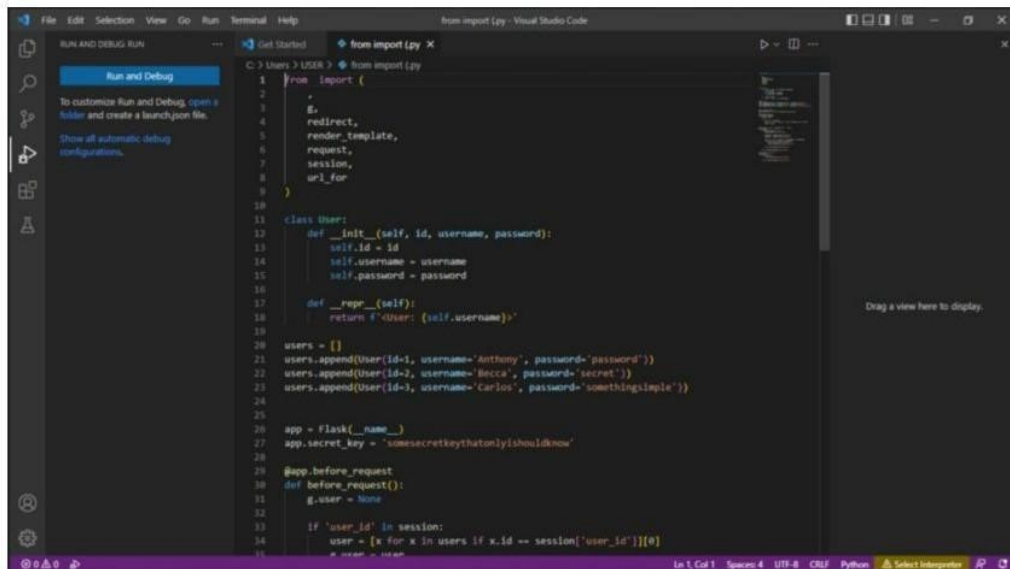
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>Login</title>
7 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/css/bootstrap.min.css">
8 <script defer src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
9 </head>
10 <body>
11 <section class="hero" id="info" id="fullheight">
12 <div class="hero-body">
13 <div class="container">
14 <div class="column is-centered">
15 <div class="column is-half">
16 <div class="field">
17 <label class="label">Username</label>
18 <input class="input" type="text" name="username" placeholder="Username">
19 <input class="input is-small is-left">
20 <input class="fa fa-lock"></input>
21 </div>
22 <div class="field">
23 <label class="label">Password</label>
24 <input class="input" type="password" name="password" placeholder="Password">
25 <input class="input is-small is-left">
26 <input class="fa fa-lock"></input>
27 </div>
28 <div class="field">
29 <button type="submit" class="button is-success">
30 Login
31 </button>
32 </div>
33 </div>
34 </div>
35 </div>
36 </body>
37 </html>
```


Source code for welcome page



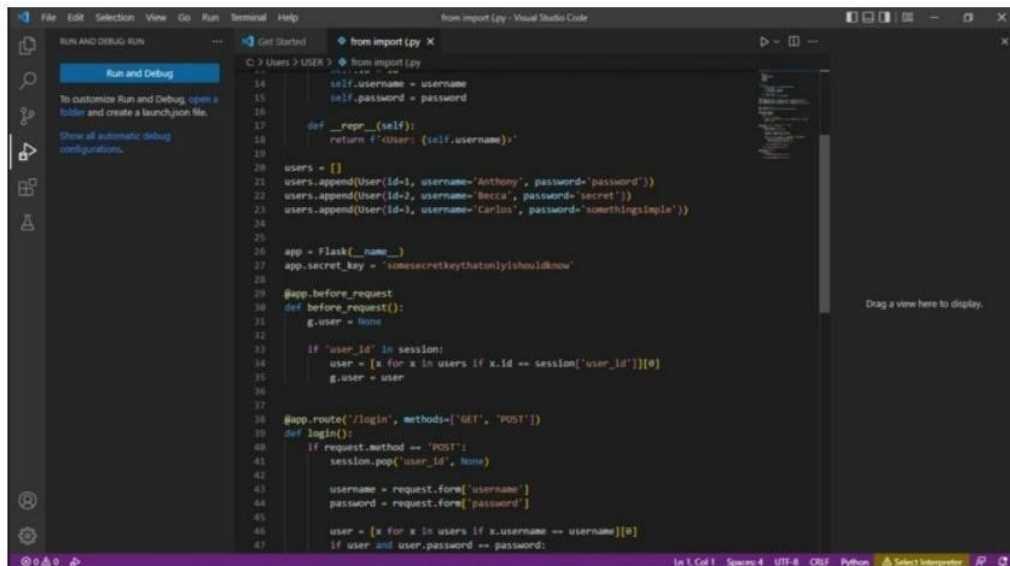
```
1<!DOCTYPE html>
2<html>
3<head>
4<meta charset="utf-8">
5<meta name="viewport" content="width=device-width, initial-scale=1">
6<title>Profile Page</title>
7<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bulma@0.9.0/css/bulma.min.css">
8<script defer src="https://use.fontawesome.com/releases/v5.3.1/js/all.js"></script>
9</head>
10<body>
11<section class="hero is-success is-fullheight">
12<div class="hero-body">
13<div class="container">
14<h1 class="title">
15({{user.username}})'s Profile
16</h1>
17<h2 class="subtitle">
18You are user number {{user.id}}
19</h2>
20</div>
21</div>
22</section>
23</body>
24</html>
```

Flask code



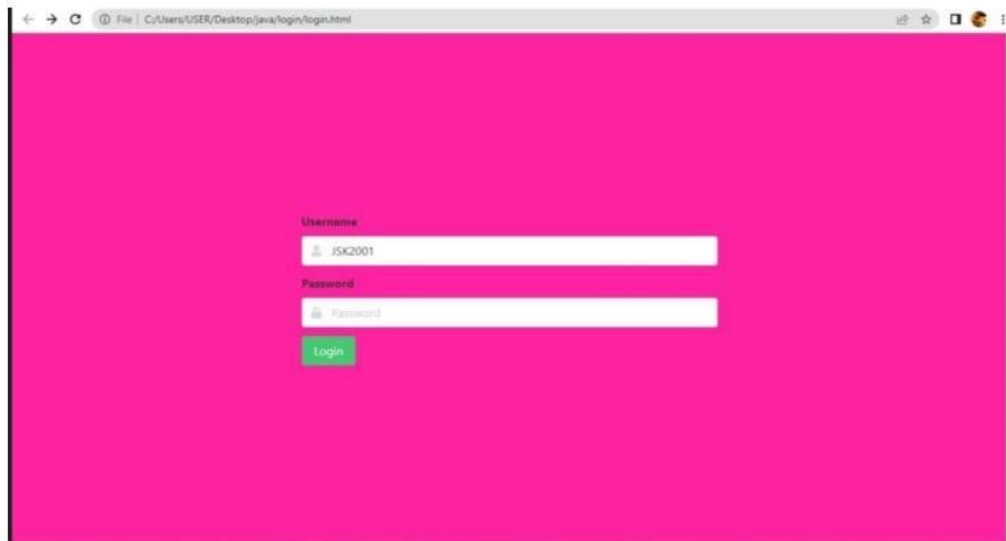
```
1from flask import Flask, redirect, render_template, request, session, url_for
2
3
4
5
6
7
8
9
10
11
12class User:
13    def __init__(self, id, username, password):
14        self.id = id
15        self.username = username
16        self.password = password
17
18    def __repr__(self):
19        return f'User: {self.username}'
20
21
22users = []
23users.append(User(id=1, username='Anthony', password='password'))
24users.append(User(id=2, username='Becca', password='secret'))
25users.append(User(id=3, username='Carlos', password='somethingimple'))
26
27
28app = Flask(__name__)
29app.secret_key = 'somekeythatonlyishouldknow'
30
31@app.before_request
32def before_request():
33    g.user = None
34
35    if 'user_id' in session:
36        user = [x for x in users if x.id == session['user_id']][0]
37        g.user = user
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Flask code



```
1from flask import Flask, redirect, render_template, request, session, url_for
2
3
4
5
6
7
8
9
10
11
12class User:
13    def __init__(self, id, username, password):
14        self.id = id
15        self.username = username
16        self.password = password
17
18    def __repr__(self):
19        return f'User: {self.username}'
20
21
22users = []
23users.append(User(id=1, username='Anthony', password='password'))
24users.append(User(id=2, username='Becca', password='secret'))
25users.append(User(id=3, username='Carlos', password='somethingimple'))
26
27
28app = Flask(__name__)
29app.secret_key = 'somekeythatonlyishouldknow'
30
31@app.before_request
32def before_request():
33    g.user = None
34
35    if 'user_id' in session:
36        user = [x for x in users if x.id == session['user_id']][0]
37        g.user = user
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Login Page Output



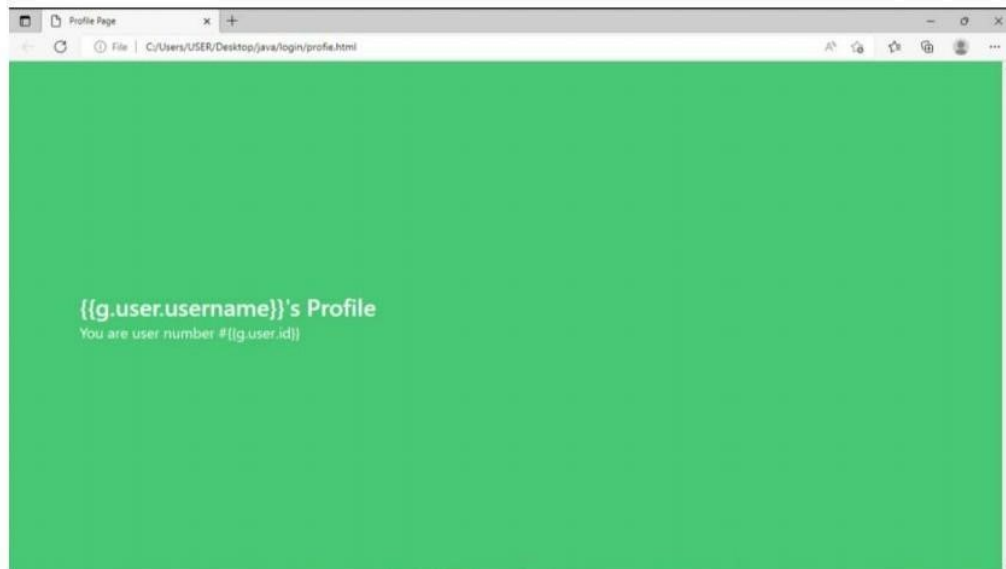
A screenshot of a web browser displaying a login page. The page has a solid pink background. In the center, there is a white rectangular area containing the login form. The form consists of two input fields: the top one is labeled 'Username' and contains the text 'JSK2001'; the bottom one is labeled 'Password' and contains the text 'Password'. Below these fields is a green button with the text 'Login' in white. The browser's address bar shows the file path 'C:/Users/USER/Desktop/java/login/login.html'.

Username
JSK2001

Password
Password

Login

Welcome Page Output



A screenshot of a web browser displaying a welcome page. The page has a solid green background. In the center, there is white text that reads: `{{g.user.username}}`'s Profile. Below this, it says: You are user number `#{{g.user.id}}`. The browser's address bar shows the file path 'C:/Users/USER/Desktop/java/login/prof.html'.

`{{g.user.username}}`'s Profile
You are user number `#{{g.user.id}}`