

Assignment -2

Assignment Date	19 September 2022
Student Name	Sivanandhini R
Student Roll Number	820419104068
Maximum Marks	2 Marks

Question-1:

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

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 'Tables' section is active, displaying a search bar and a 'Refresh' button. The 'Schemas' panel on the left shows a table with columns 'Name', 'Type', and 'Tables'. It contains one entry: 'VSC20823' of type 'User' with 1 table. The 'Tables' panel on the right shows a table with columns 'Name', 'Schema', and 'Properties'. It contains one entry: 'PLASMADONOR' in schema 'VSC20823'. Both panels show 'Total: 1, selected: 1' at the bottom.

The screenshot shows the IBM Db2 on Cloud console interface with the 'Table definition' panel open for the 'PLASMADONOR' table. The 'Tables' panel on the left shows the table 'PLASMADONOR' in schema 'VSC20823'. The 'Table definition' panel on the right displays the table's structure with columns: 'NAME', 'MAIL', 'PHONE', 'CITY', 'COVIDINFECTIONSTATUS', and 'BLOODGROUP'. All columns are of type 'VARCHAR' with a length of 32 and are nullable. The 'View data' button is visible at the bottom of the table definition panel.

Question-2:

Perform UPDATE,DELETE Queries with user table

Solution:

insert into plasmadonor

values('caitlin','caitlin33@gmail.com','7778889997','Washington','uninfected','O negative','12345');

insert into plasmadonor values('joe','joe33@gmail.com','7778889998','Washington','uninfected','B negative','12346');

insert into plasmadonor values('barry','barry33@gmail.com','7778889999','Washington','infected','A negative','12349');

update plasmadonor set phone = '7776666555' where name = 'barry';

delete from plasmadonor where name = 'joe';

The screenshot shows the IBM Db2 on Cloud interface. The left sidebar contains navigation icons for Data objects, My script, and SQL. The main area displays a script editor with the following SQL code:

```
1 insert into plasmadonor values('caitlin','caitlin33@gmail.com','7778889997','Washington','uninfected','O negative','12345');
2 insert into plasmadonor values('joe','joe33@gmail.com','7778889998','Washington','uninfected','B negative','12346');
3 insert into plasmadonor values('barry','barry33@gmail.com','7778889999','Washington','infected','A negative','12349');
4 update plasmadonor set phone = 7776666555 where name = 'barry';
5 delete from plasmadonor where name = 'joe';
6
```

Below the script editor, the 'History' tab is active, showing a table of executed scripts:

Script	Date	Status	Runtime
Untitled - 1	Oct 27, 2022 9:53:13 PM	✓ S	0.026 s
insert into plasmadonor values('caitlin','caitlin33@gmail.com','7778889997','W...		✓	0.007 s
insert into plasmadonor values('joe','joe33@gmail.com','7778889998','Washingto...		✓	0.005 s
insert into plasmadonor values('barry','barry33@gmail.com','7778889999','Washi...		✓	0.005 s
update plasmadonor set phone = 7776666555 where name = 'barry'		✓	0.004 s
delete from plasmadonor where name = 'joe'		✓	0.005 s

IBM Db2 on Cloud

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

VSC20823.PLASMADONOR

Export to CSV

NAME	MAIL	PHONE	CITY	COVIDINFECTIONSTATUS	BLOODGROUP	PASSWORD
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
barry	barry33@gmail.com	7776666555	Washington	infected	A negative	12349
caitlin	caitlin33@gmail.com	7778889997	Washington	uninfected	O negative	12345
caitlin	caitlin33@gmail.com	7778889997	Washington	uninfected	O negative	12345
caitlin	caitlin33@gmail.com	7778889997	Washington	uninfected	O negative	12345

WPS Office - data (2).csv

Menu Home Insert Page Layout Formulas Data Review View Tools Smart Toolbox

Click to find commands

Field List

PivotTable Areas

Filters

Columns

Rows

Values

Question-3:

Connect python code to db2.

Solution:

```
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lbg.databases.appdomain.cloud;PORT=31498;SECURITY=SSL;SSL ServerCertificate=DigiCertGlobalRootCA.crt;UID=sch83401;PWD=j7QZUHGaTUGbPhns",",")
```

Plasma Donor App

• [Home](#)

caitlin

caitlin33@gmail.com

7778889997

Washington

Uninfected

☐ Negative

Register

{{ pred }}

WPS Office - data (1).csv

Menu File Edit View Insert Page Layout Formulas Data Review View Tools Smart Toolbox

Click to find commands

Paste Copy Cut Format Painter Bold Italic Underline Text Color Fill Color Font Size Orientation Merge and Center Wrap Text Conditional Formatting Cell Style AutoSum Autoriser Sort Filter Rows and Columns Worksheet Freeze Panes Find Reps

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	NAME	MAIL	PHONE	CITY	COVIDINFE	BLOODGRO	PASSWORD								
2	caitlin	caitlin33@gmail.com	7778889997	Washington	uninfected	O negative	12345								
3															
4															
5															
6															
7															
8															
9															
10															
11															
12															
13															
14															
15															
16															
17															
18															
19															
20															
21															
22															
23															
24															
25															
26															
27															
28															
29															
30															

Field List

Drag fields onto PivotTable area

PivotTable Areas

Drag fields between areas below

FILTERS

COLS

ROWS

VALUES

meet.google.com is sharing your screen. Stop sharing Hide

VSC20823.PLASMADONOR

NAME	MAIL	PHONE	CITY	COVIDINFECTIONSTATUS	BLOODGROUP	PASSWORD
caitlin	caitlin33@gmail.com	7778889997	Washington	uninfected	O negative	12345

Question-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

Solution:

register.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Plasma Donor App</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
  <link rel="stylesheet" href="{{ url_for('static', filename='style1.css') }}">
  <link rel="stylesheet" href="style.css">

<style>
.login{
top: 20%;
}
```

```

</style>
</head>

<body>
<div class="header">
<div>Plasma Donor App</div>
    <ul>
    <li><a class="active" href="/login">Home</a></li>
    </ul>
</div>
<div class="login">
    <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('register')}}" method="post">
        <input type="text" name="name" placeholder="Enter Your Name" required="required"
style="color:black"/>
        <input type="email" name="email" placeholder="Enter Email" required="required"
style="color:black"/>
        <input type="text" name="phone" placeholder="Enter 10-digit mobile number"
required="required" style="color:black"/>
        <input type="city" name="city" placeholder="Enter Your City Name" required="required"
style="color:black"/>
        <select name="infect">

            <option value="select" selected>Select COVID infection status</option><option
value="infected">Infected</option>
            <option value="uninfected">Uninfected</option>
        </select>
        <select name="blood">
<option value="select" selected>Choose your blood group</option>
            <option value="O Positive">O Positive</option>
            <option value="A Positive">A Positive</option>
            <option value="B Positive">B Positive</option>
            <option value="AB Positive">AB Positive</option>
            <option value="O Negative">O Negative</option>
            <option value="A Negative">A Negative</option>
            <option value="B Negative">B Negative</option>
            <option value="AB Negative">AB Negative</option>
        </select>
        <input type="password" name="passw" placeholder="Enter Password" required="required"
style="color:black"/>
        <button type="submit" class="btn btn-primary btn-block btn-large">Register</button>
    </form>
    <br><br>
<div style="color:black">
    {{ pred }}</div>
</div>

```

```
</body>
</html>
```

login.html

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Plasma Donor App</title>
    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet'
type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300'
rel='stylesheet' type='text/css'>
    <link rel="stylesheet" href="{ { url_for('static', filename='style1.css') } }">
    <link rel="stylesheet" href="style.css">

<style>
.login{
top: 20%;
}
</style>
</head>

<body>
<div class="header">
<div>Plasma Donor App</div>
  <ul>
    <li><a href="/registration">Register</a></li>
    <li><a class="active" href="/login">Home</a></li>
  </ul>
</div>
<div class="login" >
  <div>
    <div>

<!-- Main Input For Receiving Query to our ML -->
```

```

<form action="{{ url_for('loginpage')}}"method="post">
    <input type="text" name="user" placeholder="Enter UserName" required="required"
style="color:black" />
    <input type="password" name="passw" placeholder="Enter Password" required="required"
style="color:black" />
    <button type="submit" class="btn btn-primary btn-block btn-large">Login</button>

</form>
<br><br>
<div style="color :black">
{{ pred }}</div>
</div>
</body>
</html>

```

style.css

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn {
    display: inline-block;
    *display: inline;
    *zoom: 1; padding:
    4px 10px 4px;
    margin-bottom: 0;
    font-size: 13px;
    line-height: 18px;
    color: #333333;
    text-align: center;
    text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75);
    vertical-align: middle;
    background-color: #f5f5f5;
    background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6);
    background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6);
    background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6));
    background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6);
    background-image: -o-linear-gradient(top, #ffffff, #e6e6e6);
    background-image: linear-gradient(top, #ffffff, #e6e6e6);
    background-repeat: repeat-x;
    filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff,
endColorstr=#e6e6e6, GradientType=0);
    border-color: #e6e6e6 #e6e6e6 #e6e6e6;
    border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25);
    border: 1px solid #e6e6e6;
    -webkit-border-radius: 4px;
    -moz-border-radius: 4px;

```



```

border-radius: 4px;
-webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
-moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
cursor: pointer; *margin-left: .3em;
}

.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
.btn-large {
padding: 9px 14px;
font-size: 15px;
line-height: normal;
-webkit-border-radius: 5px;
-moz-border-radius: 5px;
border-radius: 5px;
}
.btn:hover {
color: #333333;
text-decoration: none;
background-color: #e6e6e6;
background-position: 0 -15px;
-webkit-transition: background-position 0.1s linear;
-moz-transition: background-position 0.1s linear;
-ms-transition: background-position 0.1s linear;
-o-transition: background-position 0.1s linear;
transition: background-position 0.1s linear;
}
.btn-primary, .btn-primary:hover {
text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25);
color: #ffffff;
}
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
.btn-primary {
background-color: #4a77d4;
background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4);
background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4);
background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4));
background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4);
background-image: -o-linear-gradient(top, #6eb6de, #4a77d4);
background-image: linear-gradient(top, #6eb6de, #4a77d4);
background-repeat: repeat-x;
filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de,
endColorstr=#4a77d4, GradientType=0);
border: 1px solid #3762bc;
text-shadow: 1px 1px 1px rgba(0,0,0,0.4);
box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5);
}

```

```

        .btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-
primary[disabled] {
            filter: none;
            background-color: #4a77d4;
        }
        .btn-block { width: 100%; display:block; }

* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box; -o-box-
sizing:border-box; box-sizing:border-box; }
html { width: 100%; height:100%; overflow:hidden; }
body {
    width: 100%;
    height:100%;
    font-family: 'Open Sans', sans-serif;
    background: #ffffff;
    color: #000000;

    font-size: 18px;
    text-align:center;
    letter-spacing:1.2px;
}
.header {
    top:0;
    margin:0px;
    left: 0px;
    right: 0px;
    position: fixed;
    background: #4a77d4;
    color: white;
    box-shadow: 0px 8px 4px grey;
    overflow: hidden;
    padding: 15px;
    font-size: 1.5vw;
    width: 100%;
    text-align: center;
}

.login {
    position: absolute;
    top: 70%;
    left: 50%;
    margin: -25px 0 0 -150px;
    width:400px;
    height:400px;
}

.header div { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-
align:center; float:left; padding-left:150px;}

```

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  padding-right: 150px;
  overflow: hidden;
}
li {
  float: right;
}
li a {
  display: block;
  color: white;
  text-align: center;
  padding: 0px 15px;
  text-decoration: none;
}
input {
  width: 100%;
  margin-bottom: 10px;
  background: rgba(255,255,255,255);
  border: none;
  outline: none;
  padding: 10px;
  font-size: 13px;
  color: black;
  text-shadow: black;
  border: 1px solid rgba(0,0,0,0.3);
  border-radius: 4px;
  box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
  -webkit-transition: box-shadow .5s ease;
  -moz-transition: box-shadow .5s ease;
  -o-transition: box-shadow .5s ease;
  -ms-transition: box-shadow .5s ease;
  transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px
rgba(255,255,255,0.2); }
select {
  width: 100%;
  margin-bottom: 10px;
  background: rgba(255,255,255,255);
  border: none;
  outline: none;
  padding: 10px;
  font-size: 13px;
```

```

        color: #000000;
        text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
        border: 1px solid rgba(0,0,0,0.3);
        border-radius: 4px;
        box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
        -webkit-transition: box-shadow .5s ease;
        -moz-transition: box-shadow .5s ease;
        -o-transition: box-shadow .5s ease;
        -ms-transition: box-shadow .5s ease;
        transition: box-shadow .5s ease;
    }

```

app.py

```

from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
import json
import requests
app = Flask(__name__)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=3883e7e4-18f5-4afe-be8c-
fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31498;SECURITY=SSL;SSL
ServerCertificate=DigiCertGlobalRootCA.crt;UID=sch83401;PWD=j7QZUHGAUGbPhns","")

@app.route('/registration')
def home():
    return render_template('register.html')

@app.route('/register',methods=['POST'])
def register():
    x = [x for x in request.form.values()]
    print(x)
    name=x[0]
    email=x[1]
    phone=x[2]
    city=x[3]
    infect=x[4]
    blood=x[5]
    password=x[6]
    sql = "SELECT * FROM plasmadonor WHERE email =?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print(account)
    if account:

```

```

    return render_template('register.html', pred="You are already a member, please login using
your details")
else:
    insert_sql = "INSERT INTO plasmadonor VALUES (?, ?, ?, ?, ?, ?, ?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prepare_stmt, 1, name)
    ibm_db.bind_param(prepare_stmt, 2, email)
    ibm_db.bind_param(prepare_stmt, 3, phone)
    ibm_db.bind_param(prepare_stmt, 4, city)
    ibm_db.bind_param(prepare_stmt, 5, infect)
    ibm_db.bind_param(prepare_stmt, 6, blood)
    ibm_db.bind_param(prepare_stmt, 7, password)
    ibm_db.execute(prepare_stmt)
    return render_template('register.html', pred="Registration Successful, please login using your
details")
@app.route('/')
@app.route('/login')
def login():
    return render_template('login.html')
@app.route('/loginpage', methods=['POST'])
def loginpage():
    user = request.form['user']
    passw = request.form['passw']
    sql = "SELECT * FROM plasmadonor WHERE email =? AND password=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt, 1, user)
    ibm_db.bind_param(stmt, 2, passw)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    print (account)
    print(user, passw)
    if account:
        return redirect(url_for('stats'))
    else:
        return render_template('login.html', pred="Login unsuccessful. Incorrect username /
password !")
@app.route('/stats')
def stats():
    '''sql = "SELECT blood FROM user group by blood"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.execute(stmt)
    count = ibm_db.fetch_assoc(stmt)
    print(count)'''
    return render_template('stats.html', b=5, b1=2, b2=3, b3=4, b4=2, b5=1, b6=2, b7=1, b8=1)
@app.route('/requester')
def requester():

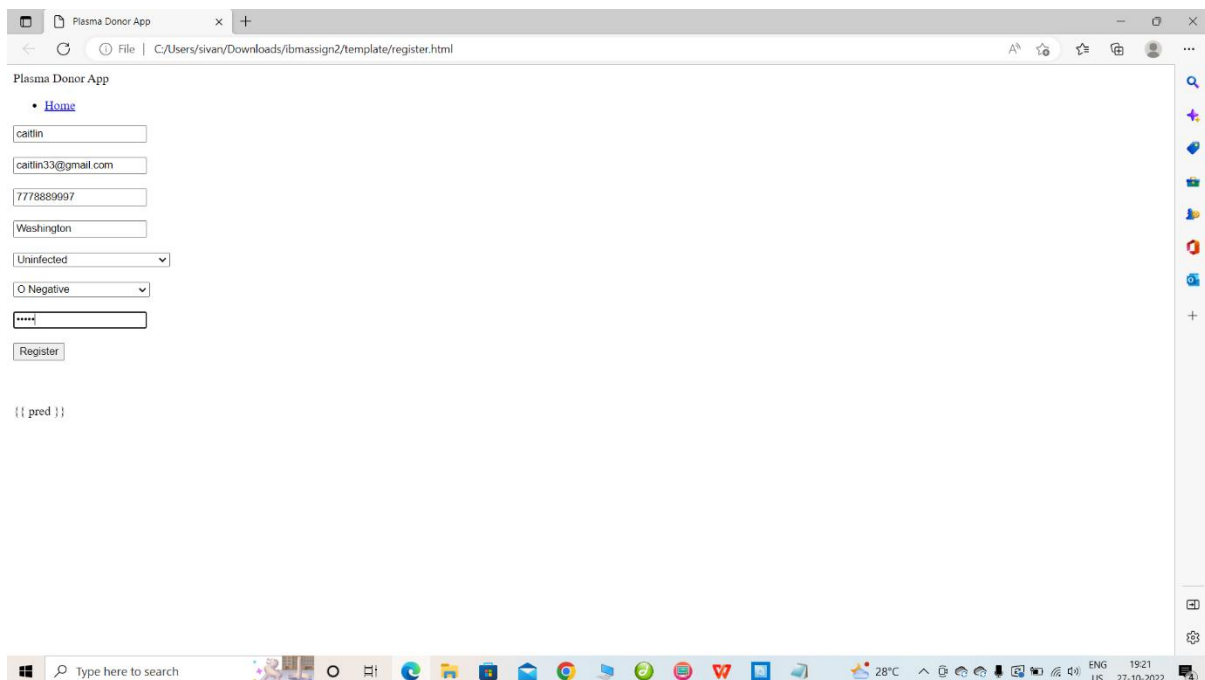
```

```

    return render_template('request.html')
@app.route('/requested',methods=['POST'])
def requested():
    bloodgrp = request.form['bloodgrp']
    address = request.form['address']
    print(address)
    sql = "SELECT * FROM plasmadonor WHERE blood=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,bloodgrp)
    ibm_db.execute(stmt)
    data = ibm_db.fetch_assoc(stmt)
    msg = "Need Plasma of your blood group for: "+address
    while data != False:
        print ("The Phone is : ", data["PHONE"])
url="https://www.fast2sms.com/dev/bulk?authorization=xCXuWTzyjOD2ARd1EngbH3a7tKlq5P
klJ8YSf0Lh4FQZecs9iNI1dSvuqprxFwCKYJXA5amQkBE36RI&sender_id=FSTSMS&message="+msg+"
&language=english&route=p&numbers="+str(data["PHONE"])
        result=requests.request("GET",url)
        print(result)
        data = ibm_db.fetch_assoc(stmt)
    return render_template('request.html', pred="Your request is sent to the concerned people.")


if __name__ == "__main__":
    app.run(host='0.0.0.0', port=8080)

```



Plasma Donor App

- [Register](#)
- [Home](#)



{{ pred }}

