

Name	Jagadish R
Roll No	SSNCE195001039
Date	10 September 2022
Team ID	PNT2022TMID53061
Project Name	Project - Personal Expense Tracker

Assignment - 2

CRUD Operations Using Flask

Question

1. Create a User table with users with email,username,roll number, password.
2. Perform UPDATE,DELETE Queries with user table
3. Connect python code to db2.
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 the database and navigate to the login page to authenticate user username and password. If the user is valid, show the welcome page.

IBM db2 is connected with our flask App using ibm-db pip module

Source Code

app.py

```
from flask import Flask, render_template, url_for, redirect, flash
from flask_login import login_user, LoginManager, login_required,
logout_user
from wtforms.validators import InputRequired, Length, ValidationError
from wtforms import StringField, PasswordField, SubmitField
from flask_wtf import FlaskForm
import os
from database import fetchUserByUsername, initialise, fetchUserById,
insert_application, insert_user, update_password
```

```
app = Flask(__name__)
app.config['SECRET_KEY'] = 'B7-1A3E'

login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'

@login_manager.user_loader
def load_user(user_id):
    user = fetchUserById(user_id)[0]
    usr_obj = User(user["ID"], user["EMAIL"], user["USERNAME"])
    return usr_obj

class User:

    def __init__(self, id, email, username):
        self.id = id
        self.username = username
        self.email = email

    def to_json(self):
        return {"username": self.username,
                "email": self.email}

    def is_authenticated(self):
        return True

    def is_active(self):
        return True

    def is_anonymous(self):
        return False

    def get_id(self):
        return str(self.id)
```

```
class RegisterForm(FlaskForm):
    email = StringField(validators=[
        InputRequired(), Length(min=4, max=50)], render_kw={"placeholder":
"Email"})
    username = StringField(validators=[
        InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})
    rollnumber = StringField(validators=[
        InputRequired(), Length(min=5, max=10)], render_kw={"placeholder":
"RollNumber"})
    password = PasswordField(validators=[
        InputRequired(), Length(min=8, max=20)],
render_kw={"placeholder": "Password"})

    submit = SubmitField('Register')

    def validate_username(self, username):
        existing_user_username = fetchUserByUsername(username.data)
        if existing_user_username != []:
            raise ValidationError(
                'That username already exists. Try another one.')

class LoginForm(FlaskForm):
    username = StringField(validators=[
        InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})

    password = PasswordField(validators=[
        InputRequired(), Length(min=8, max=20)],
render_kw={"placeholder": "Password"})

    submit = SubmitField('Login')

class UpdateForm(FlaskForm):
    username = StringField(validators=[
        InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})
```

```

oldpassword = PasswordField(validators=[
    InputRequired(), Length(min=8, max=20)], render_kw={"placeholder":
"Previous Password"})

password = PasswordField(validators=[
    InputRequired(), Length(min=8, max=20)],
render_kw={"placeholder": "Password"})

submit = SubmitField('Update')

class ApplicationForm(FlaskForm):
    applicant_name = StringField(validators=[
        InputRequired(), Length(min=4, max=20)], render_kw={"placeholder":
"Name"})

    email = StringField(validators=[
        InputRequired(), Length(min=4, max=20)], render_kw={"placeholder":
"Email"})

    mobile = StringField(validators=[
        InputRequired()], render_kw={"placeholder": "Mobile"})
    role = StringField(validators=[
        InputRequired(), Length(min=4, max=20)], render_kw={"placeholder":
"Role"})
    experience = StringField(validators=[
        InputRequired(), Length(min=4, max=20)], render_kw={"placeholder":
"Experience"})
    education = StringField(validators=[
        InputRequired(), Length(min=4, max=20)], render_kw={"placeholder":
"Education"})

    submit = SubmitField('Submit Application')

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

```

```

@app.route('/login', methods=['GET', 'POST'])
def login():
    form = LoginForm()
    if form.validate_on_submit():
        user = fetchUserByUsername(form.username.data)
        if user != []:
            user= user[0] #get the first user
            if user["PASSWORD"] == form.password.data:
                usr_obj = User(user["ID"], user["EMAIL"], user["USERNAME"])
                login_user(usr_obj)
                return redirect(url_for('application'))

            else:
                flash(f'Invalid credentials, check and try logging in
again.', 'danger')
                return redirect(url_for('login'))

        return render_template('login.html', form=form)

@app.route('/application', methods=['GET', 'POST'])
@login_required
def application():
    form = ApplicationForm()
    if form.validate_on_submit():
        insert_application(form.applicant_name.data, form.email.data,
form.mobile.data, form.role.data, form.experience.data,
form.education.data)
        return redirect(url_for('welcome'))
    else:
        flash(f'Some fields are missing, please try again.', 'danger')
        return render_template('application.html', form=form)

@app.route('/welcome', methods=['GET', 'POST'])
@login_required
def welcome():
    flash(f'Your job application has been submitted.', 'success')
    return render_template('welcome.html')

```

```
@app.route('/logout', methods=['GET', 'POST'])
@login_required
def logout():
    logout_user()
    return redirect(url_for('login'))

@ app.route('/register', methods=['GET', 'POST'])
def register():
    form = RegisterForm()
    if form.validate_on_submit():
        insert_user(form.email.data, form.username.data,
form.rollnumber.data, form.password.data)
        return redirect(url_for('login'))
    return render_template('register.html', form=form)

@ app.route('/update', methods=['GET', 'POST'])
def update():
    form = UpdateForm()
    if form.validate_on_submit():
        user = fetchUserByUsername(form.username.data)
        if user != []:
            user = user[0]
            if user["PASSWORD"] == form.oldpassword.data:
                new_password = form.password.data
                username = form.username.data
                update_password(username, new_password)
                flash(f'Password changed successfully.', 'success')
                return redirect(url_for('home'))
            else:
                flash(f'Invalid password, Enter valid password.', 'danger')
                return redirect(url_for('update'))
        else:
            flash(f'Invalid user, Enter valid User.', 'danger')
            return redirect(url_for('update'))
    return render_template('update.html', form=form)
```

```
if __name__ == "__main__":
    initialise()
    port = int(os.environ.get('PORT', 5000))
    app.run(debug=True, host='0.0.0.0', port=port)
```

database.py

```
import ibm_db

## get the IBM db2 credentials from your IBM cloud
# driver and protocol remains

dsn_hostname = "XXX"
dsn_uid = "XXX"
dsn_pwd = "XXX"
dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "XXX"
dsn_port = 0000
dsn_protocol = "TCPIP"

dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY=SSL"
).format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol,
dsn_uid, dsn_pwd)

def connect_db():
    conn = ibm_db.connect(dsn, "", "")
    return conn
```

```

def initialise():
    conn = connect_db()
    dropUser = "drop table if exists user;"
    ibm_db.exec_immediate(conn, dropUser);

    createUser = "CREATE TABLE user (id INT GENERATED BY DEFAULT AS
IDENTITY NOT NULL,email VARCHAR(20) NOT NULL,username VARCHAR(20) NOT
NULL,roll_number INTEGER NOT NULL,password VARCHAR(20) NOT NULL);"

    ibm_db.exec_immediate(conn, createUser);

    dropJob = "drop table if exists job;"
    ibm_db.exec_immediate(conn, dropJob);

    createJob = "CREATE TABLE job (\
applicant_name VARCHAR(20) NOT NULL,\
email VARCHAR(20) NOT NULL,\
mobile INTEGER UNIQUE NOT NULL,\
apply_role VARCHAR(20) NOT NULL,\
experience VARCHAR(20) NOT NULL,\
education VARCHAR(20) NOT NULL\
);"
    ibm_db.exec_immediate(conn,createJob);

    ibm_db.close(conn)

def fetchResults(res):
    result_set = []
    d = ibm_db.fetch_assoc(res)
    while d != False:
        result_set.append(d)
        d = ibm_db.fetch_assoc(res)
    return result_set

def fetchAllUsers():
    conn = connect_db()
    query = f'SELECT * FROM user'
    res = ibm_db.exec_immediate(conn, query)
    result_set = fetchResults(res)

```



```

    ibm_db.close(conn)
    return result_set

def fetchUserById(id):
    conn = connect_db()
    query = 'SELECT * FROM user WHERE id = ?'
    stmt = ibm_db.prepare(conn, query)
    param = (id,)
    ibm_db.execute(stmt,param)
    result_set = fetchResults(stmt)
    ibm_db.close(conn)
    return result_set

def fetchUserByUsername(username):
    conn = connect_db()
    query = 'SELECT * FROM user WHERE username = ?'
    stmt = ibm_db.prepare(conn, query)
    param = (username,)
    ibm_db.execute(stmt,param)
    result_set = fetchResults(stmt)
    ibm_db.close(conn)
    return result_set

def insert_user(email, username, rollno, password):
    conn = connect_db()
    query = 'INSERT INTO user (email, username, roll_number, password)
VALUES (?, ?, ?, ?)'
    stmt = ibm_db.prepare(conn, query)
    param = (email, username, rollno, password)
    res = ibm_db.execute(stmt,param)
    return res

def insert_application(applicant_name, email, mobile, apply_role,
experience, education):
    conn = connect_db()
    query = 'INSERT INTO job (applicant_name, email, mobile, apply_role,
experience, education) VALUES (?, ?, ?, ?, ?, ?)'
    stmt = ibm_db.prepare(conn, query)
    param = (applicant_name, email, mobile, apply_role, experience,
education)

```

```

    res = ibm_db.execute(stmt,param)
    return res

def update_password(username, password):
    conn = connect_db()
    query = 'UPDATE user set password = ? where username = ?'
    stmt = ibm_db.prepare(conn, query)
    param = (password, username)
    res = ibm_db.execute(stmt,param)
    return res

```

style.css

```

.header {
    padding: 5px 120px;
    width: 150px;
    height: 70px;
    background-color: #236b8e;
}

.border {
    padding: 80px 50px;
    width: 400px;
    height: 450px;
    border: 1px solid #236b8e;
    border-radius: 0px;
    background-color: #9ac0cd;
}

.btn {
    padding: 10px 40px;
    background-color: #236b8e;
    color: #ffffff;
    font-style: oblique;
    font-weight: bold;
    border-radius: 10px;
}

.textbox {

```

```
padding: 10px 40px;
background-color: #236b8e;
text-shadow: #ffffff;
border-radius: 10px;
}

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

.word {
  color: #ffffff;
  font-style: oblique;
  font-weight: bold;
}

.bottom {
  color: #236b8e;
  font-style: oblique;
  font-weight: bold;
}

body {
  background: #fafafa;
  color: #333333;
  margin-top: 5rem;
}

h1, h2, h3, h4, h5, h6 {
  color: #444444;
}

.bg-steel {
  background-color: #5f788a;
}

.site-header .navbar-nav .nav-link {
```

```
    color: #cbd5db;
}

.site-header .navbar-nav .nav-link:hover {
    color: #ffffff;
}

.site-header .navbar-nav .nav-link.active {
    font-weight: 500;
}

.content-section {
    background: #ffffff;
    padding: 10px 20px;
    border: 1px solid #dddddd;
    border-radius: 3px;
    margin-bottom: 20px;
}

.article-title {
    color: #444444;
}

a.article-title:hover {
    color: #428bca;
    text-decoration: none;
}

.article-content {
    white-space: pre-line;
}

.article-img {
    height: 65px;
    width: 65px;
    margin-right: 16px;
}

.article-metadata {
    padding-bottom: 1px;
}
```

```

margin-bottom: 4px;
border-bottom: 1px solid #e3e3e3
}

.article-metadata a:hover {
  color: #333;
  text-decoration: none;
}

.article-svg {
  width: 25px;
  height: 25px;
  vertical-align: middle;
}

.account-img {
  height: 125px;
  width: 125px;
  margin-right: 20px;
  margin-bottom: 16px;
}

.account-heading {
  font-size: 2.5rem;
}

```

welcome.html

```

<!DOCTYPE html>
<html lang="en">

<head>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.

```

```

css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
  <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Welcome Page</title>
</head>

<body>
  <h1>Welcome!</h1> <br>
  <a href="{{url_for('logout')}}">Press here to logout</a>

  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
1Pi6jizo" crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDs4x0xI
M+B07jRM" crossorigin="anonymous"></script>
</body>
</html>

```

login.html

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>

```

```

    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.
css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />
</head>

<body>
    <div class="col-md-8">
        {% with messages = get_flashed_messages(with_categories=true) %}
            {% if messages %}
                {% for category, message in messages %}
                    <div class="alert alert-{{category}}">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endif %}
        {% endwith %}
        {% block content %} {% endblock %}
    </div>

    <h1>Login Page</h1> <br>

    <form method="POST" action="">
        {{ form.hidden_tag() }}
        <fieldset class="form-group">
            <div>
                {{ form.username.label(class="form-control-label") }}

                {% if form.username.errors %}
                    {{ form.username(class="form-control form-control-lg
is-invalid") }}

                    <div class="invalid-feedback">
                        {% for error in form.username.errors %}
                            <span>{{ error }}</span>
                        {% endfor %}
                    </div>
                {% else %}

```

```

        {{ form.username(class="form-control form-control-lg")
    }}

    {% endif %}
</div>

<div>
    {{ form.password.label(class="form-control-label") }}

    {% if form.password.errors %}
        {{ form.password(class="form-control form-control-lg
is-invalid") }}

        <div class="invalid-feedback">
            {% for error in form.password.errors %}
                <span>{{ error }}</span>
            {% endfor %}
        </div>
    {% else %}
        {{ form.password(class="form-control form-control-lg")
    }}

    {% endif %}
</div>

</fieldset>
<div class="form-group">
    {{ form.submit(class="btn btn-outline-info") }}
</div>

<small class="text-muted ml-2">
    <a href='{{ url_for('register') }}'>Do not have an account? Sign
Up?</a>
</small>

</form>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
1Pi6jizo" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"

```



```

integrity="sha384-U02eT0CpHqdSjQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js
"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/njGzIxFDsf4x0xI
M+B07jRM" crossorigin="anonymous"></script>
</body>

</html>

```

home.html

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.
css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
</head>

<body>
    <div class="col-md-8">
        {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
                <div class="alert alert-{{category}}">
                    {{ message }}
                </div>
            {% endfor %}
        {% endwith %}
    </div>

```

```

        {% endfor %}
    {% endif %}
{% endwith %}
{% block content %} {% endblock %}
</div>
<h1>Flask Application</h1> <br>
<a href="{{ url_for('login') }}">Login Page</a><br> <br>
<a href="{{ url_for('register') }}">Register Page</a><br> <br>
<a href="{{ url_for('update') }}">Update Password</a><br> <br>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
1Pi6jizo" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaEaFf/nJGzIxFDs4x0xI
M+B07jRM" crossorigin="anonymous"></script>
</body>
</html>

```

register.html

```

<!DOCTYPE html>
<html lang="en">

<head>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.
css"

```

```

integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>

<body>
    <h1>Register Page</h1> <br>

    <!-- <form method="POST" action="">
        {{ form.hidden_tag() }}
        {{ form.email }} <br> <br>
        {{ form.username }} <br> <br>
        {{ form.rollnumber }} <br> <br>
        {{ form.password }} <br> <br>
        {{ form.submit }} <br> <br>
    </form> -->

    <form method="POST" action="">
        {{ form.hidden_tag() }}
        <fieldset class="form-group">
            <!-- <legend class="border-bottom mb-4">Registration
Page</legend> -->

            <div>
                {{ form.email.label(class="form-control-label") }}

                {% if form.email.errors %}
                    {{ form.email(class="form-control form-control-lg
is-invalid") }}

                    <div class="invalid-feedback">
                        {% for error in form.email.errors %}
                            <span>{{ error }}</span>
                        {% endfor %}
                    </div>
                {% else %}

```

```

        {{ form.email(class="form-control form-control-lg") }}
    {% endif %}
</div>

<div>
    {{ form.username.label(class="form-control-label") }}

    {% if form.username.errors %}
        {{ form.username(class="form-control form-control-lg
is-invalid") }}

        <div class="invalid-feedback">
            {% for error in form.username.errors %}
                <span>{{ error }}</span>
            {% endfor %}
        </div>
    {% else %}
        {{ form.username(class="form-control form-control-lg")
}}

    {% endif %}
</div>

<div>
    {{ form.rollnumber.label(class="form-control-label") }}

    {% if form.rollnumber.errors %}
        {{ form.rollnumber(class="form-control form-control-lg
is-invalid") }}

        <div class="invalid-feedback">
            {% for error in form.rollnumber.errors %}
                <span>{{ error }}</span>
            {% endfor %}
        </div>
    {% else %}
        {{ form.rollnumber(class="form-control
form-control-lg") }}

    {% endif %}
</div>

```

```

        <div>
            {{ form.password.label(class="form-control-label") }}

            {% if form.password.errors %}
                {{ form.password(class="form-control form-control-lg is-invalid") }}

                <div class="invalid-feedback">
                    {% for error in form.password.errors %}
                        <span>{{ error }}</span>
                    {% endfor %}
                </div>
            {% else %}
                {{ form.password(class="form-control form-control-lg") }}
            {% endif %}
        </div>

    </fieldset>
    <div class="form-group">
        {{ form.submit(class="btn btn-outline-info") }}
    </div>

    <small class="text-muted ml-2">
        <a href="{{ url_for('login') }}">Already have an account? Log
In</a>
    </small>

</form>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
1Pi6jizo" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
"

```

```
integrity="sha384-JjSmVgyd0p3pXBlrRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
</body>

</html>
```

update.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.
css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoR
xT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />
</head>

<body>
  <div class="col-md-8">
    {% with messages = get_flashed_messages(with_categories=true) %}
      {% if messages %}
        {% for category, message in messages %}
          <div class="alert alert-{{category}}">
            {{ message }}
          </div>
        {% endfor %}
      {% endif %}
    {% endwith %}
    {% block content %} {% endblock %}
  </div>
```

```

<h1>Update Password</h1> <br>

<form method="POST" action="">
  {{ form.hidden_tag() }}
  <fieldset class="form-group">
    <div>
      {{ form.username.label(class="form-control-label") }}

      {% if form.username.errors %}
        {{ form.username(class="form-control form-control-lg
is-invalid") }}

        <div class="invalid-feedback">
          {% for error in form.username.errors %}
            <span>{{ error }}</span>
          {% endfor %}
        </div>
      {% else %}
        {{ form.username(class="form-control form-control-lg")
}}

      {% endif %}
    </div>

    <div>
      {{ form.oldpassword.label(class="form-control-label") }}

      {% if form.oldpassword.errors %}
        {{ form.oldpassword(class="form-control
form-control-lg is-invalid") }}

        <div class="invalid-feedback">
          {% for error in form.oldpassword.errors %}
            <span>{{ error }}</span>
          {% endfor %}
        </div>
      {% else %}
        {{ form.oldpassword(class="form-control
form-control-lg") }}

      {% endif %}
    </div>
  </fieldset>
</form>

```

```

        <div>
            {{ form.password.label(class="form-control-label") }}

            {% if form.password.errors %}
                {{ form.password(class="form-control form-control-lg is-invalid") }}

                <div class="invalid-feedback">
                    {% for error in form.password.errors %}
                        <span>{{ error }}</span>
                    {% endfor %}
                </div>
            {% else %}
                {{ form.password(class="form-control form-control-lg") }}
            {% endif %}
        </div>

    </fieldset>
    <div class="form-group">
        {{ form.submit(class="btn btn-outline-info") }}
    </div>
</form>

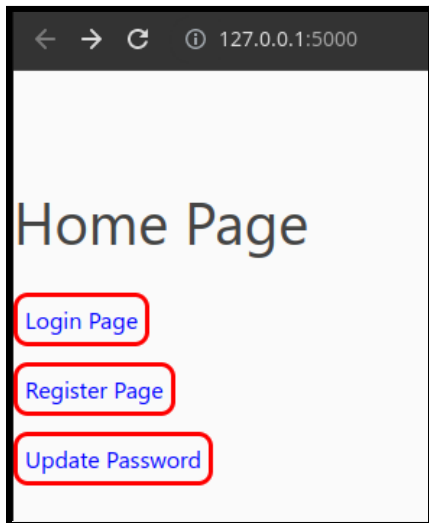
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
1Pi6jizo" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1c1HTMga3JDZwrnQq4sF86d
IHNDz0W1" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/njGzIxFDsf4x0xI
M+B07jRM" crossorigin="anonymous"></script>
</body>

</html>

```


Output

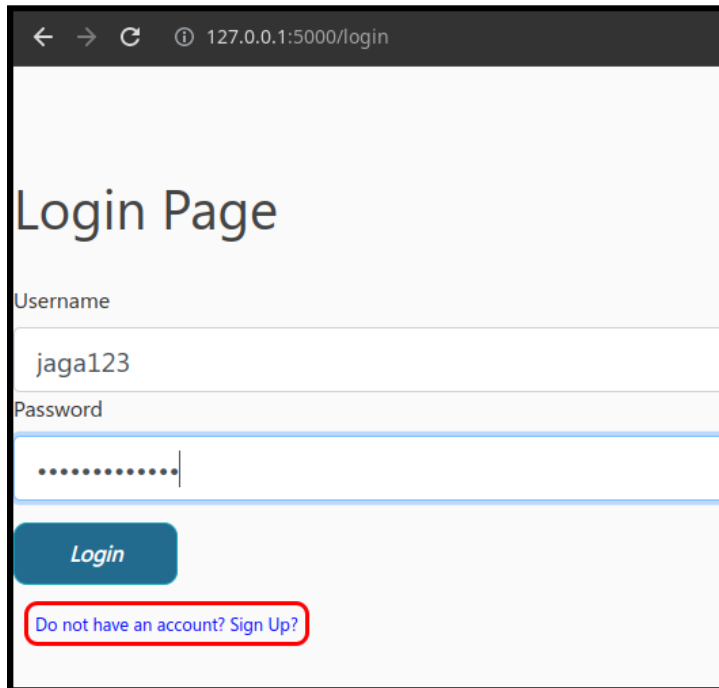
Home Page



Registration

A screenshot of a web browser displaying the Register Page. The address bar shows the URL 127.0.0.1:5000/register. The page has a light gray background with the title "Register Page" in a large, dark font. Below the title, there are five input fields for registration details: "Email" (containing jagadish@ssn.edu.in), "Username" (containing jaga123), "Rollnumber" (containing 195001039), and "Password" (containing a masked password with dots). Below the input fields is a blue "Register" button. At the bottom, there is a red rounded rectangular border containing the text "Already have an account? Log In".

Login



A screenshot of a web browser window with the address bar showing "127.0.0.1:5000/login". The page title is "Login Page". It features a form with a "Username" field containing "jaga123" and a "Password" field with masked characters. A blue "Login" button is visible. Below the button is a link "Do not have an account? Sign Up?" which is highlighted with a red rectangular border.

← → ↻ ⓘ 127.0.0.1:5000/login

Login Page

Username

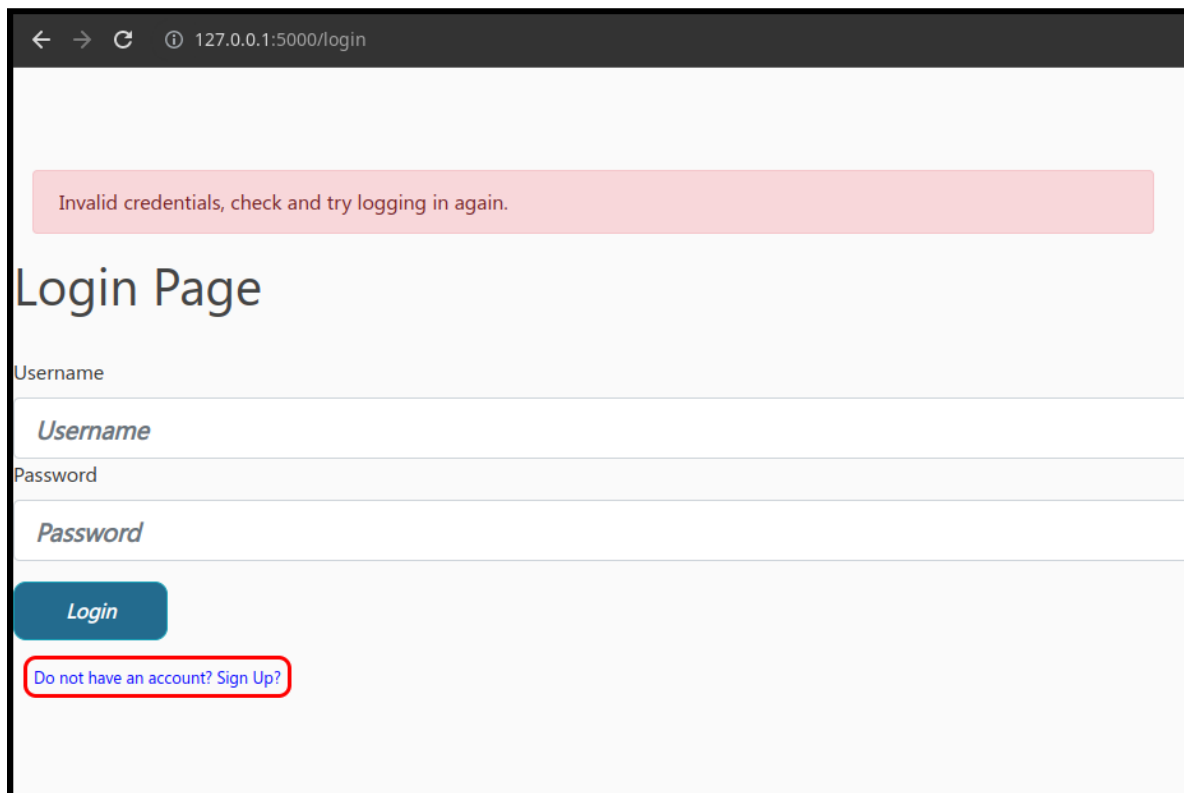
jaga123

Password

.....

Login

Do not have an account? Sign Up?



A screenshot of a web browser window with the address bar showing "127.0.0.1:5000/login". The page title is "Login Page". A red error message "Invalid credentials, check and try logging in again." is displayed at the top. The form fields "Username" and "Password" are empty and have placeholder text "Username" and "Password" respectively. A blue "Login" button is visible. Below the button is a link "Do not have an account? Sign Up?" which is highlighted with a red rectangular border.

← → ↻ ⓘ 127.0.0.1:5000/login

Invalid credentials, check and try logging in again.

Login Page

Username

Username

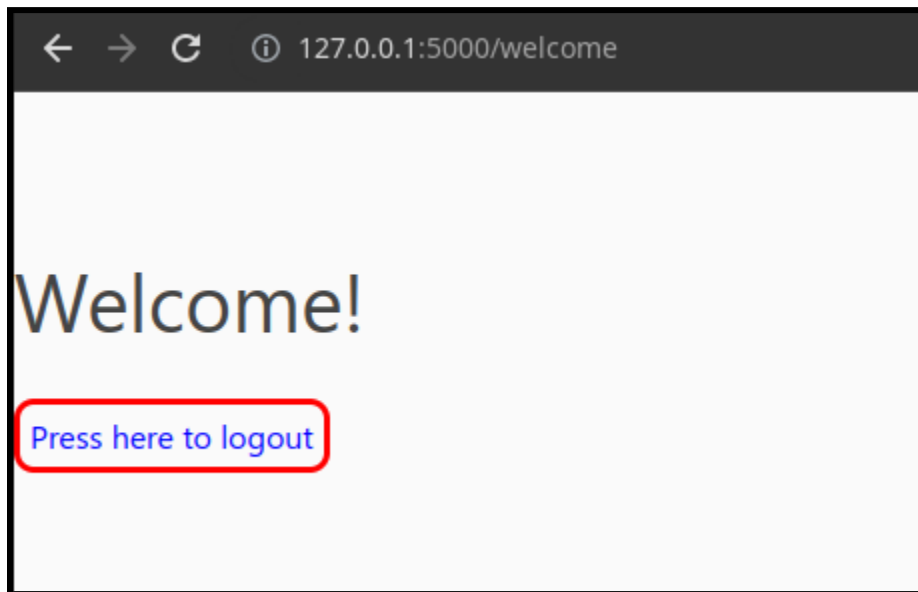
Password

Password

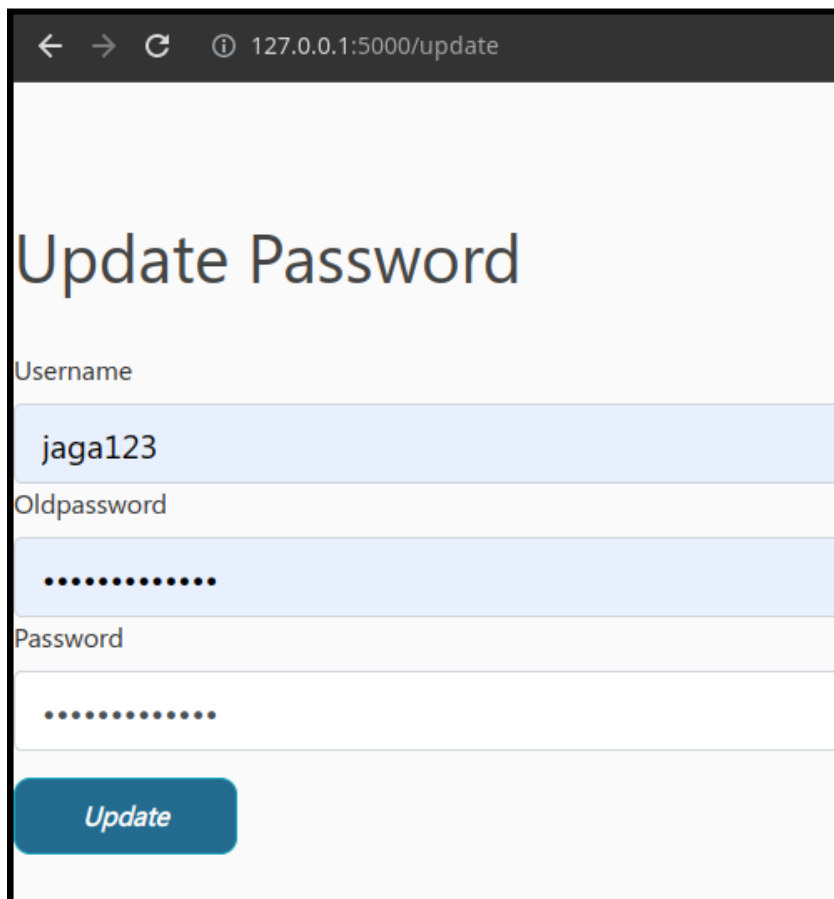
Login

Do not have an account? Sign Up?

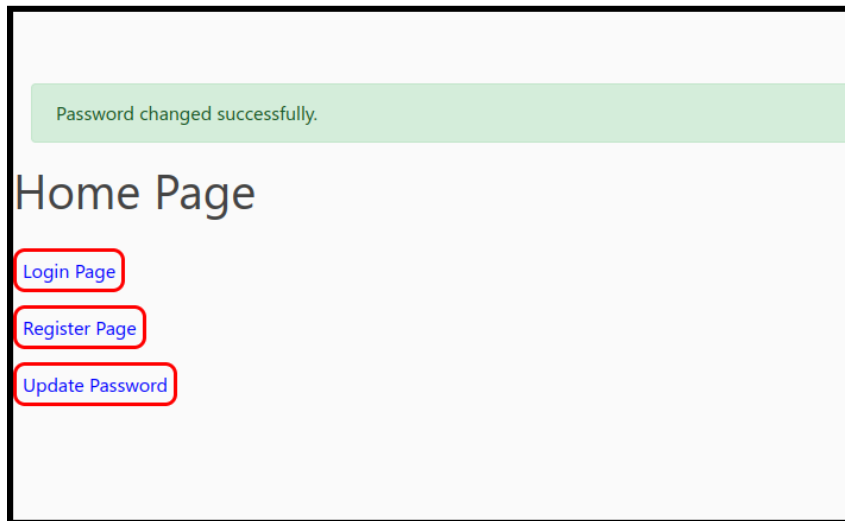
Welcome



Password Updation



A screenshot of a web browser window. The address bar shows the URL '127.0.0.1:5000/update'. The main content area displays the text 'Update Password' in a large, dark font. Below this, there is a form with three input fields: 'Username' (containing 'jaga123'), 'Oldpassword' (containing a series of dots), and 'Password' (containing a series of dots). At the bottom of the form, there is a blue button labeled 'Update'.



IBM db2 Database

Table definition

USER

No statistics available.

Name	Data type	Nullable	Length	Scale	
ID	INTEGER	N		0	👁
EMAIL	VARCHAR	N	20	0	👁
USERNAME	VARCHAR	N	20	0	👁
ROLL_NUMBER	INTEGER	N		0	👁
PASSWORD	VARCHAR	N	20	0	👁

View data

Service Details - IBM Clou x IBM Db2 on Cloud x IBM Db2 on Cloud x Search results - Google x Assignment 2 - Google x

bpe61bfd0365e9u4psdglite.db2.cloud.ibm.com/crn%3Av1%3Abluemix%3Apublic%3Adashdb-for-transactions%3Aus-south%3Aa%2Fd28a079e757e47...

IBM Db2 on Cloud

Load DataLoad HistoryTablesViewsIndexesAliasesMQTsSequencesApplication objects

QST27374.USER

Back

Export to CSV

ID	EMAIL	USERNAME	ROLL_NUMBER	PASSWORD
1	jagadish@ssn.edu.in	jaga123	195001039	hello@1234