Experiment No 10 Database Management System Lab 2021-22

Faculty: Sana Shaikh

Class: SE Comp

Experiment No: 10

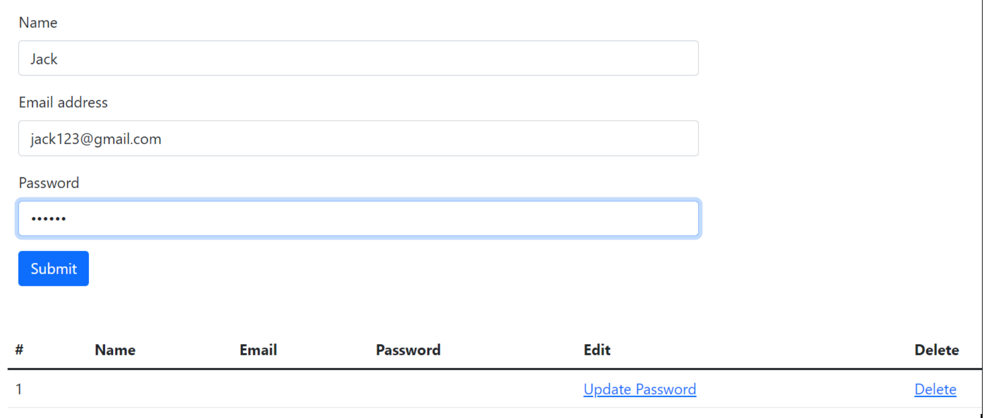
Name : Ashish Jha Roll no. 27 Batch :B

Topic:	Develop one database application (for the assigned application) using Flask with MySQL and apply CURD(Create, Update, Read, Delete) operations.
Prerequisite :	Knowledge of concepts Flask, Python, MySQL, CURD operations
Mapping With COs:	CSL402.6
Objective:]	To develop database application using Flask and MySQL. Also apply basic CURD operations.
Outcome:	After completion of this lab, the students will understand and be able to do the following: - Able to install Flask, MySQL - Able to apply CURD operations - Able to connect a Flask Application to a MySQL Database
Instructions :	1. This experiment is a compulsory experiment. All the students are required to perform this experiment individually. 2. Students need to make basic database application (based on assigned topic) using Flask and MySQL. 3. Studnets needs to apply CURD operations of the given application.
Deliverables :	Code : from sqlalchemy import create_engine import flask import json from flask import request, render_template, redirect

```
import uuid
app = flask.Flask(__name__)
app.config["DEBUG"] = True
engine = create_engine('sqlite:///database.db', echo=True)

try:
    conn = engine.connect()
    conn.execute("CREATE TABLE users (name VARCHAR(45) NOT
NULL,email VARCHAR(45) NOT NULL,password VARCHAR(45)
NOT NULL,PRIMARY KEY (`email`));")
except:
    pass

def execute_query(query):
    conn = engine.connect()
    return conn.execute(query)

def get_users():
    users = execute_query("SELECT * FROM users;").fetchall()
    userdata = []
    temp = {}
    for i in users:
        temp["name"] = i[0]
        temp["email"] = i[1]
        temp["password"] = i[2]
        userdata.append(temp)
    return userdata

def insert_user(name, email, password):
    print(f"INSERT INTO users(name, email, password)
VALUES('{name}', '{email}', '{password}');")
    execute_query(f"INSERT INTO users(name, email, password)
VALUES('{name}', '{email}', '{password}');")

def update_password(email, password):
    print(f"UPDATE users SET password = '{password}' WHERE
email='{email}';")
    execute_query(f"UPDATE users SET password = '{password}'
WHERE email='{email}';")

def delete_user(email):
   execute_query(f"DELETE FROM users WHERE email='{email}';")

def find_user(email):
   user = execute_query(f"SELECT * FROM users WHERE
email='{email}';").fetchone()
   userdata = []
   temp = {}
   for i in users:
       temp["name"] = i[0]
       temp["email"] = i[1]
       temp["password"] = i[2]
       userdata.append(temp)
   return user


@app.route('/', methods=['GET', 'POST'])
def UsersController():
   if request.method == 'GET':
      users = get_users()
      print(users)
      return render_template('index.html',users=users)
      # return json.dumps(get_users())
   else:
      print(request.form["name"])
      print(request.form["email"])
      print(request.form["password"])
      try:
         insert_user(request.form["name"], request.form["email"],
request.form["password"])
         # return {"message" : "done"}
         return redirect('/')
      except Exception as e:
         return {"message" : "error"}

@app.route('/updatepassword/<int:id>', methods=['GET', 'POST'])
```

```
def UpdatePassword(id):
    if request.method=='POST':
        # email=request.form['email']
        # password=request.form['password']
        update_password(request.form['email'],request.form['password'])
        return redirect('/')
    else:
        users = get_users()
        return render_template('updatepassword.html',user=users[id])

@app.route('/delete/<email>')
def DeleteUser(email):
    # email = request.args['email']
    delete_user(email)
    return redirect('/')
app.run()
```

Output :



Name

Jack

Email address

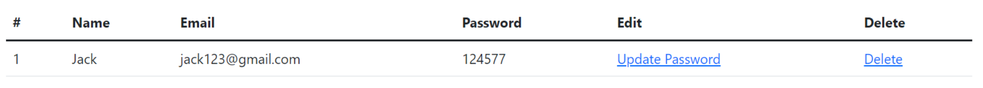
jack123@gmail.com

Password

••••••

Submit

#	Name	Email	Password	Edit	Delete
1				Update Password	Delete



#	Name	Email	Password	Edit	Delete
1	Jack	jack123@gmail.com	124577	Update Password	Delete

Conclusion:	understand and be able to install Flask and MySQL. Also learned how to connect Flask application with MySQL. Further, students were able to apply CURD operations.
References:	ACM Workshop.

Experiment No 10 Database Management System Lab 2021-22

Faculty: Sana Shaikh

Class: SE Comp

Don Bosco Institute of Technology

Department of Computer Engineering

<u>Assessment Rubric for Experiment No. 10</u>

Title of Experiment : Develop Flask Application with MySQL and apply CURD operations **Performance Date :08/4/2022** **Year and Semester** : 2nd Year and IVth Semester **Submission Date :08/04/2022**
Name: Ashish Jha **Batch : B**
<u>Roll No. : 27</u>

Sr. No.	Criteria	1 Marks	2 Marks	3 Marks	4 Marks	5 Marks
1	Execution	Executed 10- 30% based on following: - Good User interface design - Apply All CURD	Executed 31- 50% based on following: - Good User interface design - Apply All CURD	Executed 51- 70% based on following: - Good User interface design - Apply All	Executed 71- 89% based on following: - Good User interface design - Apply	Executed 90- 100% based on following: - Good User interface design - Apply All

		operations	operations	CURD operations	All CURD operations	CURD operations
2	Documentation	20-39% of solutions are documented properly.	40-59% of solutions are documented properly.	60-79% of solutions are documented properly.	80-100% of the solution is documented properly.	
3	Viva	Students hardly answered.	Students have problems while answering.	Questions are answered fairly well.	Questions are answered completely and correctly.	
4	Submission on Time	Submitted after the given deadline	Submitted before the given deadline			