Group: - 1 from T4 Batch
0077 – Aakash Joshi
2039 – Akanksha Lokhande

Title-: Write a program to implement MySQL/Oracle database connectivity with any front end language to implement Database navigation operations (add, delete, edit etc.)

Date of Completion-:

Objectives-:

- To develop Database programming skills.
- To develop basic Database administration skills.
- To develop skills to handle SQL database.
- To learn, understand and execute process of software application development.

Outcomes-:

- Design schema in appropriate normal form considering actual requirements.
- Implement SQL queries for given requirements , using different SQL concepts.
- Implement PL/SQL Code block for given requirements.
- Design and develop application considering actual requirements and using database concepts.

Problem Statement-:

Write a program to implement MySQL/Oracle database connectivity with any front end language to implement Database navigation operations (add, delete, edit etc.)

Software and Hardware requirement-:

- 64-bit Open source Linux or its derivative.
- MYSQL/Oracle.
- Xampp Software

Theory-:

MySQL Connectivity:

a. Overview of MySQL: Discuss the origins, features, and advantages of MySQL as a relational database management system.

b. Understanding Connectivity: Highlight the significance of establishing a stable and secure connection between the application and the MySQL database. Discuss the role of APIs, drivers, and protocols in facilitating this connection.

c. Authentication and Security: Explain the importance of authentication mechanisms in MySQL connectivity and emphasize the implementation of security measures to safeguard sensitive data.

Database Operations:

a. Data Definition Language (DDL): Explain the role of DDL in MySQL for creating and modifying database structures. Discuss the use of statements like CREATE, ALTER, and DROP for defining tables, indexes, and constraints.

b. Data Manipulation Language (DML): Explore the functionalities of DML in MySQL for managing data within the database. Elaborate on the use of statements such as INSERT, UPDATE, DELETE, and SELECT for adding, modifying, and retrieving data.

c. Transaction Management: Discuss the significance of transactions in maintaining data consistency and integrity. Highlight the role of COMMIT, ROLLBACK, and SAVEPOINT statements in managing transactions effectively.

Connectivity Implementation:

a. Establishing Connection: Provide a step-by-step guide on how to establish a connection between MySQL and a programming language (e.g., Python, Java) using appropriate connectors.

b. Executing Queries: Demonstrate the execution of basic SQL queries through the established connection. Showcase examples of SELECT, INSERT, UPDATE, and DELETE operations for data retrieval and manipulation.

c. Error Handling: Discuss common errors encountered during MySQL connectivity and suggest strategies for effective error handling and debugging.

Practical Applications:

Data Storage and Retrieval: Discuss how MySQL facilitates efficient data storage and retrieval, enabling users to manage large datasets effectively.

Web Application Development: Illustrate the use of MySQL in the development of web applications, emphasizing its role in storing user data, session management, and content management.
Data Analysis and Reporting: Elaborate on how MySQL can be integrated with data analysis tools for generating meaningful insights and comprehensive reports based on stored data.

Implementation :

```
const express = require('express');
const mysql = require('mysql');
const ejs = require('ejs');
const bcrypt = require('bcrypt');
const dotenv = require('dotenv');
const path = require('path');
const session = require('express-session');
const { error } = require('console');

const app = express();
const port = 3000;

const db = mysql.createConnection({
    host : 'localhost',
    user : 'root',
    password : '',
    database : 'mydatabase1'
});

db.connect( err => {
    if(err){
        console.log('database connection failed' , err);
    }
    else{
        console.log('Database connected');
    }
});

app.set('view engine' , 'ejs');
app.use(express.urlencoded({ extended: true }));
```

```
app.use(express.static(path.join('public')));


app.use(session({
    secret : 'secret-key',
    resave: false,
    saveUninitialized: true
}));

app.get('/' , (req , res) => {
    if(req.session.user){
        res.render('index', { loggedin : true});
    }
    else{
        res.render('index' , {loggedin : false});
    }
});

app.get('/signup', (req , res) => {
    res.render('signup' , {errors : {} , email : '' , password : '' , username : ''});
});

app.post('/signup' , async (req , res) => {
    const {username , email , password ,confirmPassword} = req.body;
    const errors = {};

    if(password.length < 8){
        errors.passwordlength = true;
    }
```

```
    else{

        if(confirmPassword == password)
        {

            db.query('SELECT * FROM users WHERE email = ?' , [email] , async (err, results) => {
                if(err){
                    console.log('error in checking the existing database' , err);
                    return
                }
                if(results.length > 0){
                    return res.send('account already exists');
                }

                const hashPassword = await bcrypt.hash(password , 10);

                db.query('INSERT INTO users (username , email , password) VALUES (? , ? , ?)' , [username , email ,
                    if(err){
                        console.log('Insertion of data in the database failed');
                        res.send('Sign up failed');
                    }
                    else{
                        console.log('Data inserted in the databse');
                        res.redirect('/login');
                    }
                });
            });

        }
        else{
```

```
        else{
            errors.passwordmatch = true;
        }
    }

    if(Object.keys(errors).length > 0){
        return res.render('signup' , {errors , email , password , username});
    }


});

app.get('/login' , (req ,res) => {
    res.render('login' , {errors : {} , email : '' , password : ''});
})

app.post('/login' , (req , res) => {
    const {email , password} = req.body;
    const errors = {};
    const status = {};

    db.query('SELECT * FROM users WHERE email = ?' , [email] , async (err , results) => {
        if(err){
            console.log('error in searching email', err);
            return;
        }

        if(results.length > 0){

            if(password.length < 8){
                errors.passwordlength = true;
            }
            else{
```

```
            else{
                const users = results[0];
                const match = await bcrypt.compare(password , users.password);

                if(match){
                    res.redirect('/');
                }
                else{
                    errors.invalidpassword = true;
                }
            }
        }
        else{
            errors.emailnotfound = true;
        }

        if (Object.keys(errors).length > 0) {
            return res.render('login', { errors, email , password});
        }
    });
});




app.listen(port , (req , res) => {
    console.log(`Server started on port ${port}`);
});

```

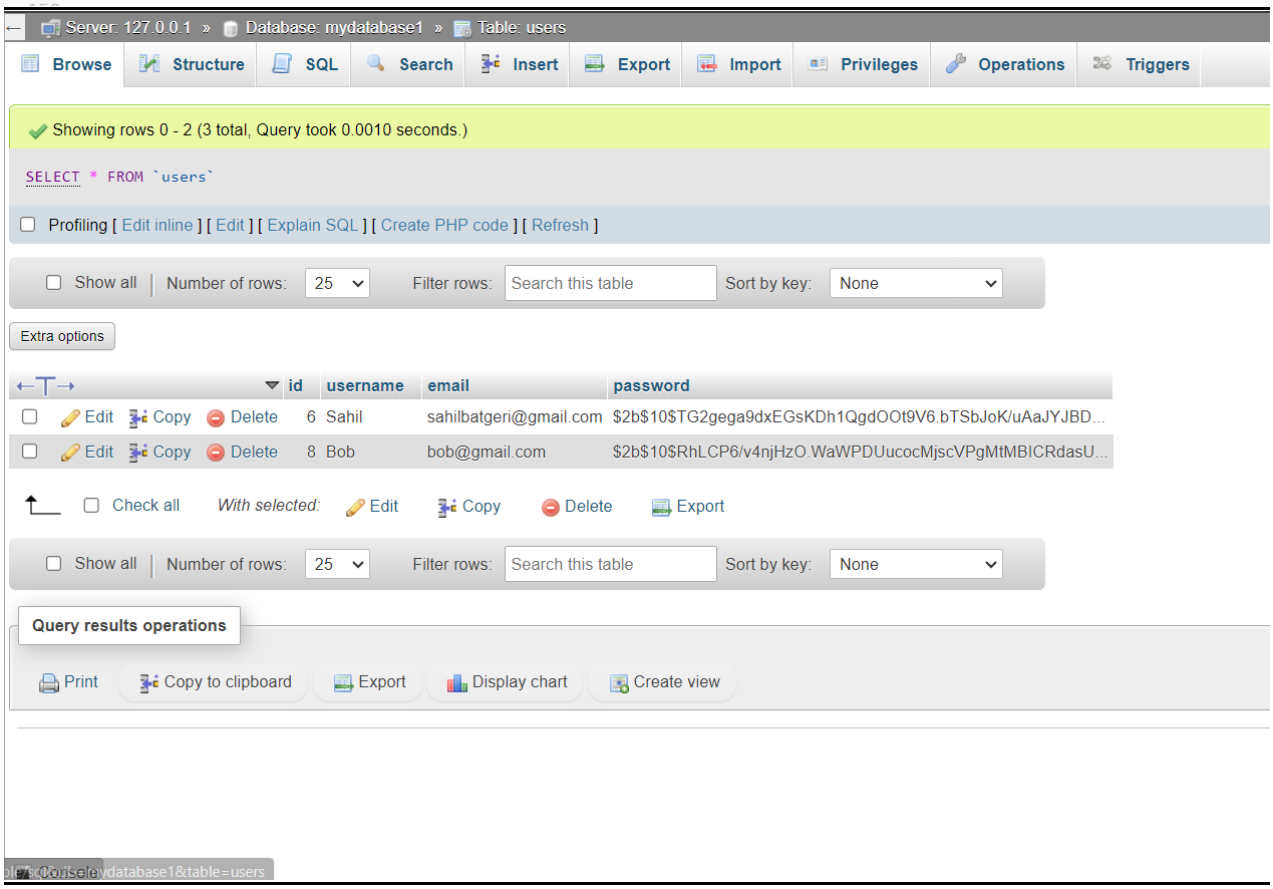
```
            else{
                const users = results[0];
                const match = await bcrypt.compare(password , users.password);

                if(match){
                    res.redirect('/');
                }
                else{
                    errors.invalidpassword = true;
                }
            }
        }
        else{
            errors.emailnotfound = true;
        }

        if (Object.keys(errors).length > 0) {
            return res.render('login', { errors, email , password});
        }
    });
});



app.listen(port , (req , res) => {
    console.log(`Server started on port ${port}`);
});
```



Conclusion-:

In conclusion, MySQL connectivity plays a pivotal role in ensuring seamless data management and manipulation. Understanding the theoretical aspects of MySQL

connectivity and its practical implementation is crucial for harnessing the full potential of this robust database management system. By mastering the art of establishing connections, executing queries, and implementing security measures, users can effectively leverage MySQL for diverse data management tasks in various domains..