LABORATORY REPORT

Application Development Lab
(CS33002)

B.Tech Program in ECSc

Submitted By

Name:- Harsh Kumar

Roll No: 2230084

Kalinga Institute of Industrial Technology
(Deemed to be University)
Bhubaneswar, India

Spring 2024-2025

Experiment Number	6
Experiment Title	Database Management Using Flask
Date of Experiment	11-03-2025
Date of Submission	16 -03-2025

1. Objective:-

To develop an application for user authentication and document sharing.

2. Procedure:- (Steps Followed)

1. Install MySQL workbench in your system and install flask-mysqldb package.
2. Create a database where you wish to store your user name and the password
3. Implement user authentication/registration form using Flask and the database. For a new user the account is created using the 'signup' button. Existing users can directly login with their crendtials.
4. Inside the users can update their personal details, reset their passwords.
5. Inside the users can see the grades for their marks, which they cannot edit personally
6. Build a responsive frontend for user interactions.

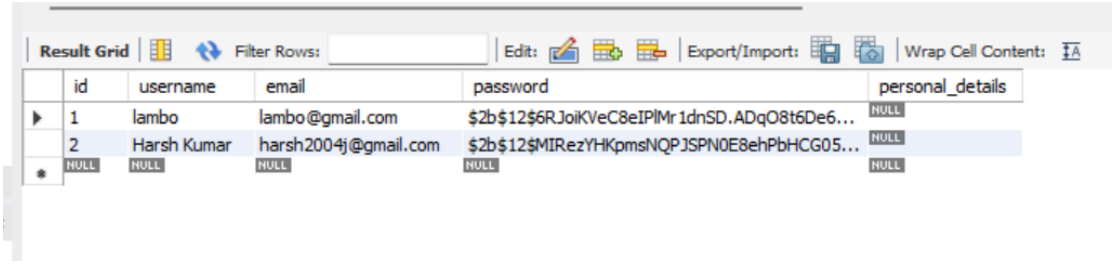
3. Code:-

- *Create a database where you wish to store your user name and the password*

```
CREATE DATABASE user_db;
use user_db;
drop database user_db;

CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    personal_details TEXT
);

CREATE TABLE documents (
    id INT AUTO_INCREMENT PRIMARY KEY,
```

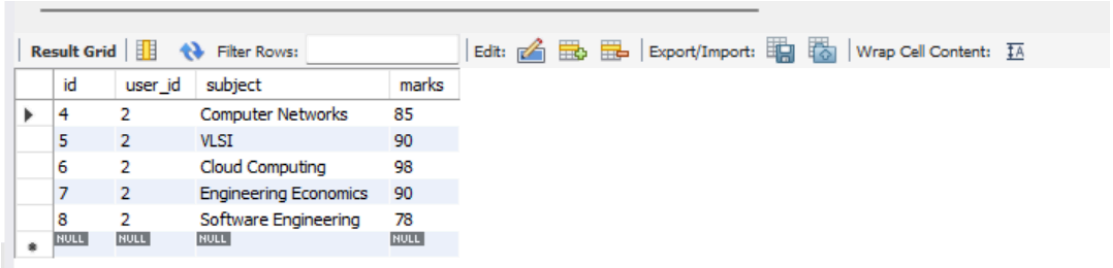
```
    user_id INT,
    file_path VARCHAR(255),
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);

drop table grades;
CREATE TABLE grades (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    subject VARCHAR(100),
    marks INT,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);


select* from users;
select* from grades;
SELECT * FROM grades WHERE user_id = 1;
INSERT INTO grades (user_id, subject, marks) VALUES
(1, 'Math', 85),
(1, 'Science', 90),
(1, 'History', 78);
drop table grades;
INSERT INTO grades (user_id, subject, marks) VALUES
(2, 'Computer Networks', 85),
(2, 'VLSI', 90),
(2, 'Cloud Computing', 98),
(2, 'Engineering Economics', 90),
(2, 'Software Engineering', 78);
```

select from users;*



id	username	email	password	personal_details
1	lambo	lambo@gmail.com	$2b$12$6RJoiKVeC8eIPlMr1dnSD.ADqO8t6De6...	NULL
2	Harsh Kumar	harsh2004j@gmail.com	$2b$12$MIRezYHKpmsNQPJSPN0E8ehPbHCG05...	NULL
NULL	NULL	NULL	NULL	NULL

*select * from grades WHERE user_id = 2;*



id	user_id	subject	marks
4	2	Computer Networks	85
5	2	VLSI	90
6	2	Cloud Computing	98
7	2	Engineering Economics	90
8	2	Software Engineering	78
NULL	NULL	NULL	NULL

- *Implement user authentication/registration form using Flask and the database. For a new user the account is created using the 'signup' button. Existing users can directly login with their crendtials.*

- *Inside the users can update their personal details, reset their passwords.*

- *Inside the users can see the grades for their marks, which they cannot edit personally*

```
from flask import Flask
from flask_mysqldb import MySQL
from flask_bcrypt import Bcrypt
from flask_login import LoginManager
from routes import init_routes

app = Flask(__name__)
app.config.from_object("config")

mysql = MySQL(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)
login_manager.login_view = "login"

init_routes(app, mysql, bcrypt, login_manager)

if __name__ == "__main__":
   app.run(debug=True)
```

```
PS C:\Users\KIIT\Desktop\AD Lab> python -u "c:\Users\KIIT\Desktop\AD Lab\AD_Lab6\app.py"
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WS
GI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 960-475-524
127.0.0.1 - - [16/Mar/2025 22:51:29] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [16/Mar/2025 22:51:29] "GET /static/styles.css HTTP/1.1" 304 -
```

```
 import os
SECRET_KEY = os.urandom(24)
MYSQL_HOST = "localhost"
MYSQL_USER = "root"
MYSQL_PASSWORD = ""
MYSQL_DB = "user_db"
MYSQL_CURSORCLASS = "DictCursor"

from flask import render_template, request, redirect, url_for, flash
from flask_bcrypt import Bcrypt
from flask_login import LoginManager, UserMixin, login_user, login_required,
logout_user, current_user
class User(UserMixin):
```

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

def init_routes(app, mysql, bcrypt, login_manager):
    @login_manager.user_loader
    def load_user(user_id):
        cur = mysql.connection.cursor()
        cur.execute("SELECT id, username, email FROM users WHERE id = %s",
        (user_id,))
        user_data = cur.fetchone()
        cur.close()
        return User(user_data["id"], user_data["username"], user_data["email"]) if user_data
         else None

    @app.route("/")
    def index():
        return render_template("index.html")
    @app.route("/signup", methods=["GET", "POST"])
    def signup():
        if request.method == "POST":
            username = request.form["username"]
            email = request.form["email"]
            password =
             bcrypt.generate_password_hash(request.form["password"]).decode("utf-8")
            cur = mysql.connection.cursor()
            cur.execute("INSERT INTO users (username, email, password) VALUES (%s,
              %s, %s)", (username, email, password))
            mysql.connection.commit()
            cur.close()
            flash("Signup successful! Please log in.", "success")
            return redirect(url_for("login"))
        return render_template("signup.html")
    @app.route("/login", methods=["GET", "POST"])
    def login():
        if request.method == "POST":
            email = request.form["email"]
            password = request.form["password"]

            cur = mysql.connection.cursor()
            cur.execute("SELECT * FROM users WHERE email = %s", (email,))
```

```
        user_data = cur.fetchone()
        cur.close()

        if user_data and bcrypt.check_password_hash(user_data["password"], password):
            user = User(user_data["id"], user_data["username"], user_data["email"])
            login_user(user)
            flash("Login successful!", "success")
            return redirect(url_for("dashboard"))
        else:
            flash("Invalid email or password", "danger")

    return render_template("login.html")

        @app.route("/dashboard")
        @login_required
        def dashboard():
    cur = mysql.connection.cursor()
    cur.execute("SELECT subject, marks FROM grades WHERE user_id = %s",
        (current_user.id,))
    grades_data = cur.fetchall()

    grades_list = [(row["subject"], row["marks"]) for row in grades_data]

    cur.close()

    return render_template("dashboard.html", username=current_user.username,
    grades=grades_list)


@app.route("/update_profile", methods=["GET", "POST"])
@login_required
def update_profile():
    if request.method == "POST":
        new_username = request.form["username"]
        new_email = request.form["email"]

        cur = mysql.connection.cursor()
        cur.execute("UPDATE users SET username = %s, email = %s WHERE id = %s",
                (new_username, new_email, current_user.id))
        mysql.connection.commit()
        cur.close()
```

```
            flash("Profile updated successfully!", "success")
            return redirect(url_for("dashboard"))

        return render_template("update_profile.html")

    @app.route("/reset_password", methods=["GET", "POST"])
    @login_required
    def reset_password():
        if request.method == "POST":
            current_password = request.form["current_password"]
            new_password = request.form["new_password"]

            cur = mysql.connection.cursor()
            cur.execute("SELECT password FROM users WHERE id = %s",
             (current_user.id,))
            user_data = cur.fetchone()

            if user_data and bcrypt.check_password_hash(user_data["password"],
             current_password):
                hashed_password =
                 bcrypt.generate_password_hash(new_password).decode("utf-8")
                cur.execute("UPDATE users SET password = %s WHERE id = %s",
                        (hashed_password, current_user.id))
                mysql.connection.commit()
                cur.close()

                flash("Password updated successfully!", "success")
                return redirect(url_for("dashboard"))
            else:
                flash("Current password is incorrect.", "danger")

        return render_template("reset_password.html")

    @app.route("/grades")
    @login_required
    def grades():
        cur = mysql.connection.cursor()
        cur.execute("SELECT subject, marks FROM grades WHERE user_id = %s",
(current_user.id,))
        grades_data = cur.fetchall()
        cur.close()
        return render_template("grades.html", grades=grades_data)
```

```
@app.route("/logout")
@login_required
def logout():
    logout_user()
    flash("Logged out successfully.", "info")
    return redirect(url_for("login"))
```

- *Build a responsive frontend for user interactions.*

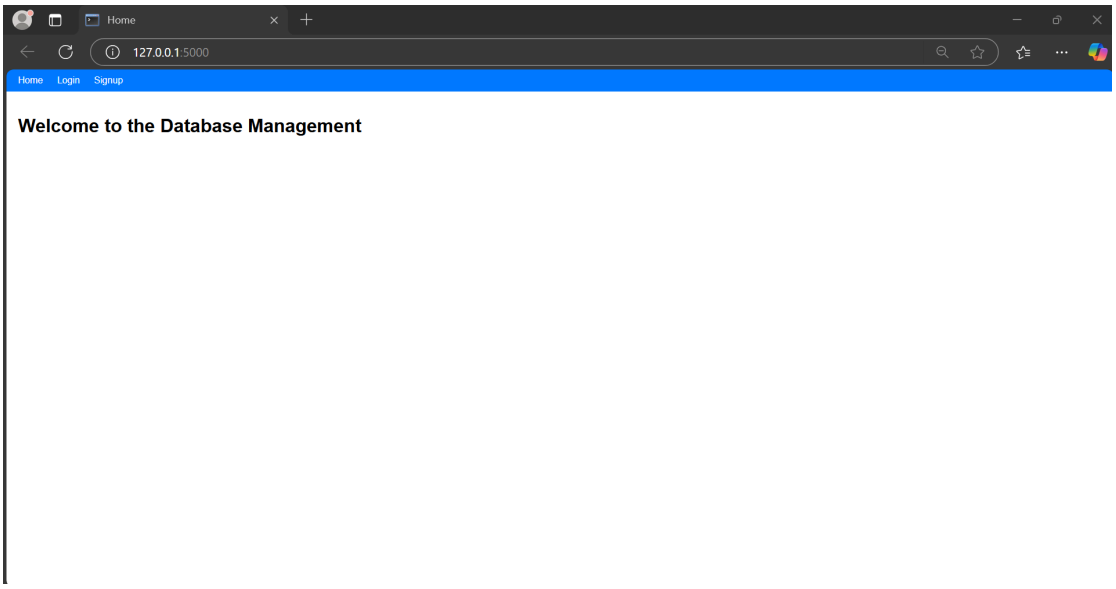
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
        }
        .container {
            width: 50%;
            margin: 50px auto;
            background: white;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        table {
            width: 100%;
            margin-top: 20px;
            border-collapse: collapse;
        }
        th, td {
            border: 1px solid black;
            padding: 10px;
```

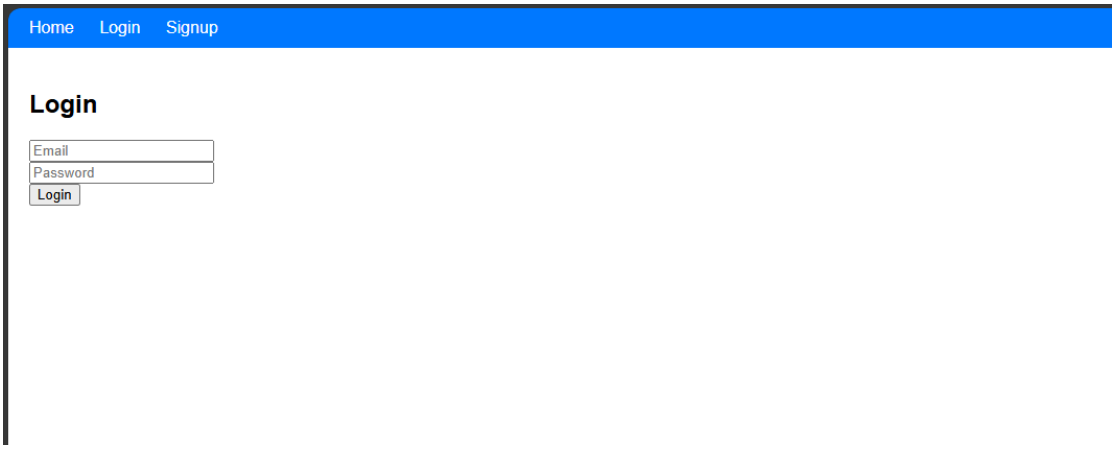
```
            text-align: center;
        }
        th {
            background-color: #007BFF;
            color: white;
        }
        h1, h2, h3 {
            color: #333;
        }
    </style>
</head>
<body>

    <h1>Home | <a href="{{ url_for('dashboard') }}">Dashboard</a> | <a href="{{ url_for('logout') }}">Logout</a></h1>

    <div class="container">
        <h2>Welcome, {{ username }}!</h2>
        <p>This is your dashboard.</p>

        <h3>Your Grades:</h3>

        {% if grades %}
        <table>
            <tr>
                <th>Subject</th>
                <th>Marks</th>
            </tr>
            {% for subject, marks in grades %}
            <tr>
                <td>{{ subject }}</td>
                <td>{{ marks }}</td>
            </tr>
            {% endfor %}
        </table>
        {% else %}
        <p>No grades available.</p>
        {% endif %}
    </div>

</body>
</html>
```

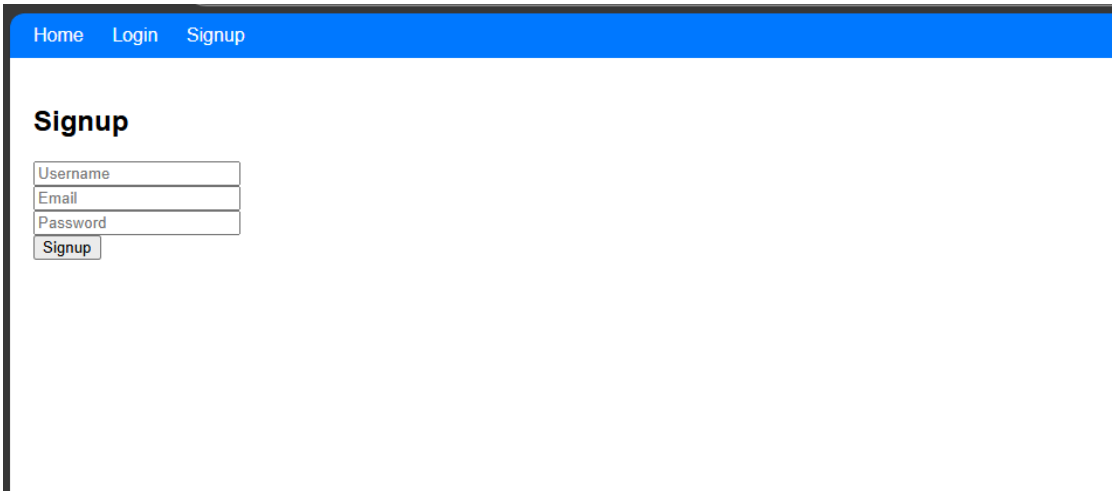
```
{% extends "base.html" %}

{% block content %}
<div class="container mt-4">
  <h2 class="text-center">Your Grades</h2>
  {% if grades %}
  <table class="table table-bordered">
    <thead class="table-dark">
      <tr>
        <th>Subject</th>
        <th>Marks</th>
      </tr>
    </thead>
    <tbody>
      {% for grade in grades %}
      <tr>
        <td>{{ grade.subject }}</td>
        <td>{{ grade.marks }}</td>
      </tr>
      {% endfor %}
    </tbody>
  </table>
  {% else %}
  <p class="text-center text-danger">No grades available.</p>
  {% endif %}
</div>
{% endblock %}

{% extends "base.html" %}
{% block title %}Home{% endblock %}
{% block content %}
<h1>Welcome to the Flask App</h1>
{% endblock %}

{% extends "base.html" %}
{% block title %}Login{% endblock %}
{% block content %}
<h2>Login</h2>
<form method="POST">
  <input type="email" name="email" placeholder="Email" required><br>
  <input type="password" name="password" placeholder="Password" required><br>
```

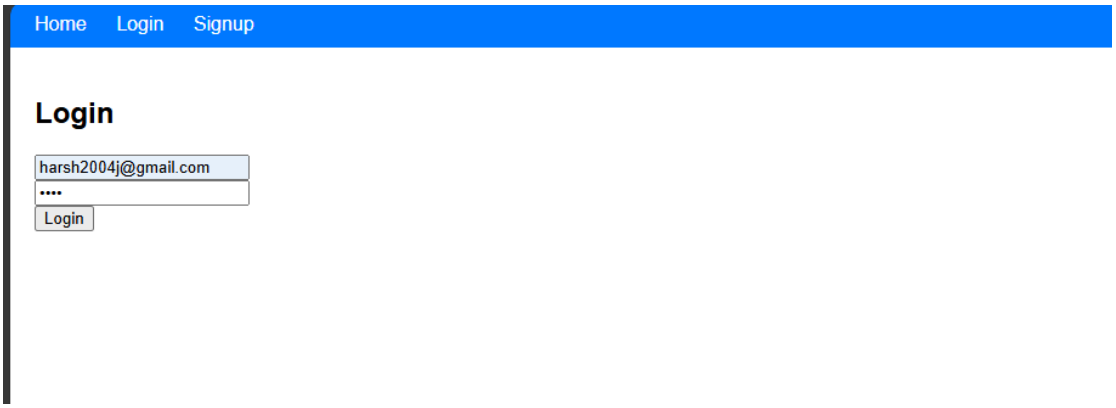
```
    <button type="submit">Login</button>
</form>
{% endblock %}
```

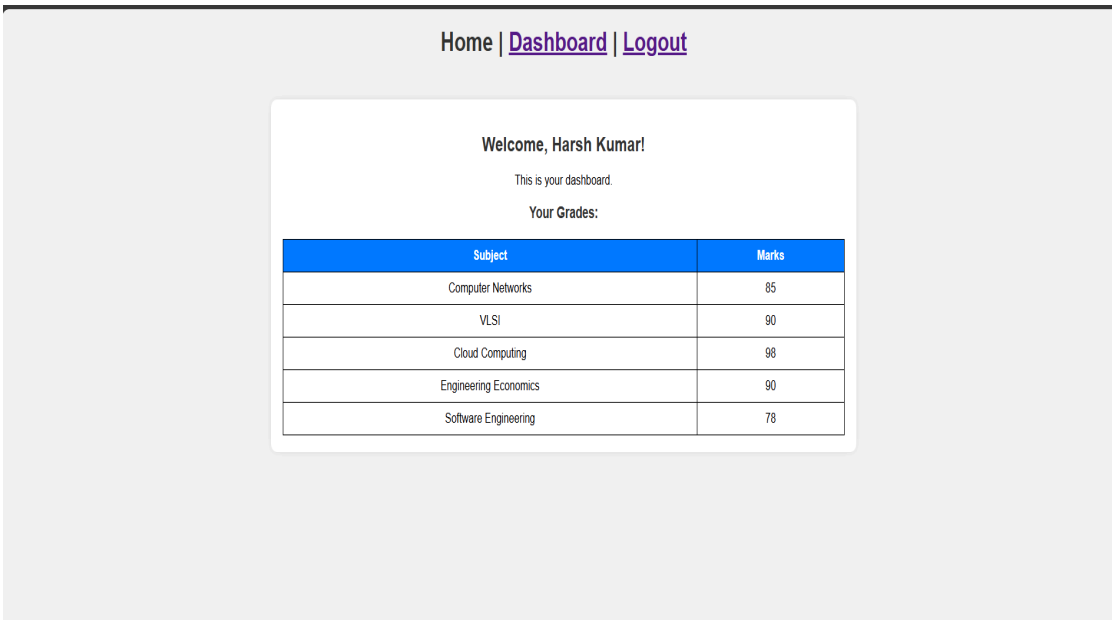
4. Results/Output:- Entire Screen Shot including Date & Time











5. Remarks:-

In this experiment, we successfully developed a Flask-based database management system with user authentication and document sharing functionalities. The implementation included MySQL for data storage, allowing users to register, log in, update personal details, and reset passwords securely. Additionally, users could view their grades but not modify them. A responsive frontend was built to enhance user experience, ensuring accessibility across different devices. This project provided hands-on experience in integrating Flask with MySQL, handling authentication, and designing interactive web applications.

Signature of the Student

_____

(Harsh Kumar)

Signature of the Lab Coordinator

_____

(Name of the Coordinator)