LABORATORY REPORT

# Application Development Lab (CS33002)

## B.Tech Program in ECSc

Submitted By

**Name:-**Pruthibiraj Nayak

**Roll No:** 2230183

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

Spring 2024-2025

# Table of Content

|  |  |
|---|---|
| **Experiment Number** | 7 |
| **Experiment Title** | Natural Language Database Interaction with LLMs |
| **Date of Experiment** | 18.03.2025 |
| **Date of Submission** | 24.03.2025 |

## 1. Objective:-

To interact with databases using natural language queries powered by LLMs.

## 2. Procedure:- (Steps Followed)

1. Set up a MySQL database and populate it with sample data.
2. Integrate an LLM to convert natural language queries into SQL commands.
3. Develop a Flask backend to interact with the database.
4. Create a frontend for users to enter queries and view results.

## Code:-

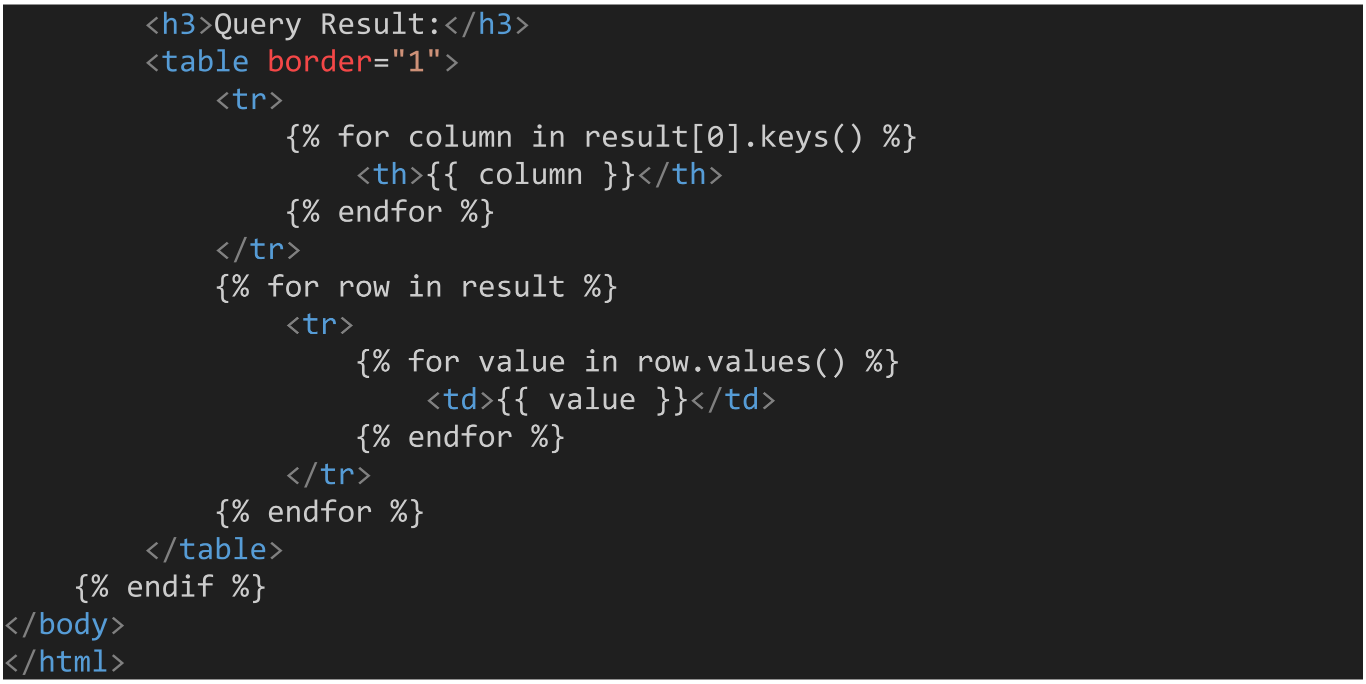
## FLASK CODE

```
from flask import Flask, request, jsonify, render_template
import mysql.connector
import google.generativeai as genai

app = Flask(__name__)

# Set your API key here
genai.configure(api_key="AIzaSyBBXyfZMhgHH6T0fHoj1XjJVZMrzZAyryo")

# Configure MySQL Connection
db_config = {
    "host": "localhost",
    "user": "root",  # Change if needed
    "password": "passofmysql",  # Change to your MySQL password
    "database": "nl_query_db"  # Change to your schema name
}

# Function to Convert Natural Language to SQL
def convert_nl_to_sql(nl_query):
    model = genai.GenerativeModel("gemini-1.5-pro-latest")  # Ensure the
correct model
    response = model.generate_content(f"Convert this to SQL: {nl_query}")
    # Clean the SQL query (Remove markdown formatting)
    sql_query = response.text.strip().replace("```sql", "").replace("```",
"").strip()

    return sql_query
```

```
# Function to Execute SQL Query
def execute_sql_query(query):
    conn = mysql.connector.connect(**db_config)
    cursor = conn.cursor(dictionary=True)  # Return results as dictionaries
    cursor.execute(query)
    result = cursor.fetchall()
    cursor.close()
    conn.close()
    return result

@app.route("/", methods=["GET", "POST"])
def process_query():
    if request.method == "POST":
        nl_query = request.form["nl_query"]
        sql_query = convert_nl_to_sql(nl_query)  # Convert NL to SQL

        # Debugging: Print generated SQL query
        print("Generated SQL Query:", sql_query)
        result = execute_sql_query(sql_query)  # Execute SQL query
        return render_template("index.html", nl_query=nl_query,
sql_query=sql_query, result=result)

    return render_template("index.html", nl_query="", sql_query="",
result=None)
if __name__ == "__main__":
    app.run(debug=True)
```

## HTML CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Natural Language to SQL</title>
</head>
<body>
    <h2>Natural Language to SQL Query</h2>
    <form method="post">
        <label for="nl_query">Enter Query:</label>
        <input type="text" id="nl_query" name="nl_query" required>
        <button type="submit">Submit</button>
    </form>

    {% if sql_query %}
        <h3>Generated SQL:</h3>
        <p><strong>SQL Query:</strong> {{ sql_query }}</p>
```



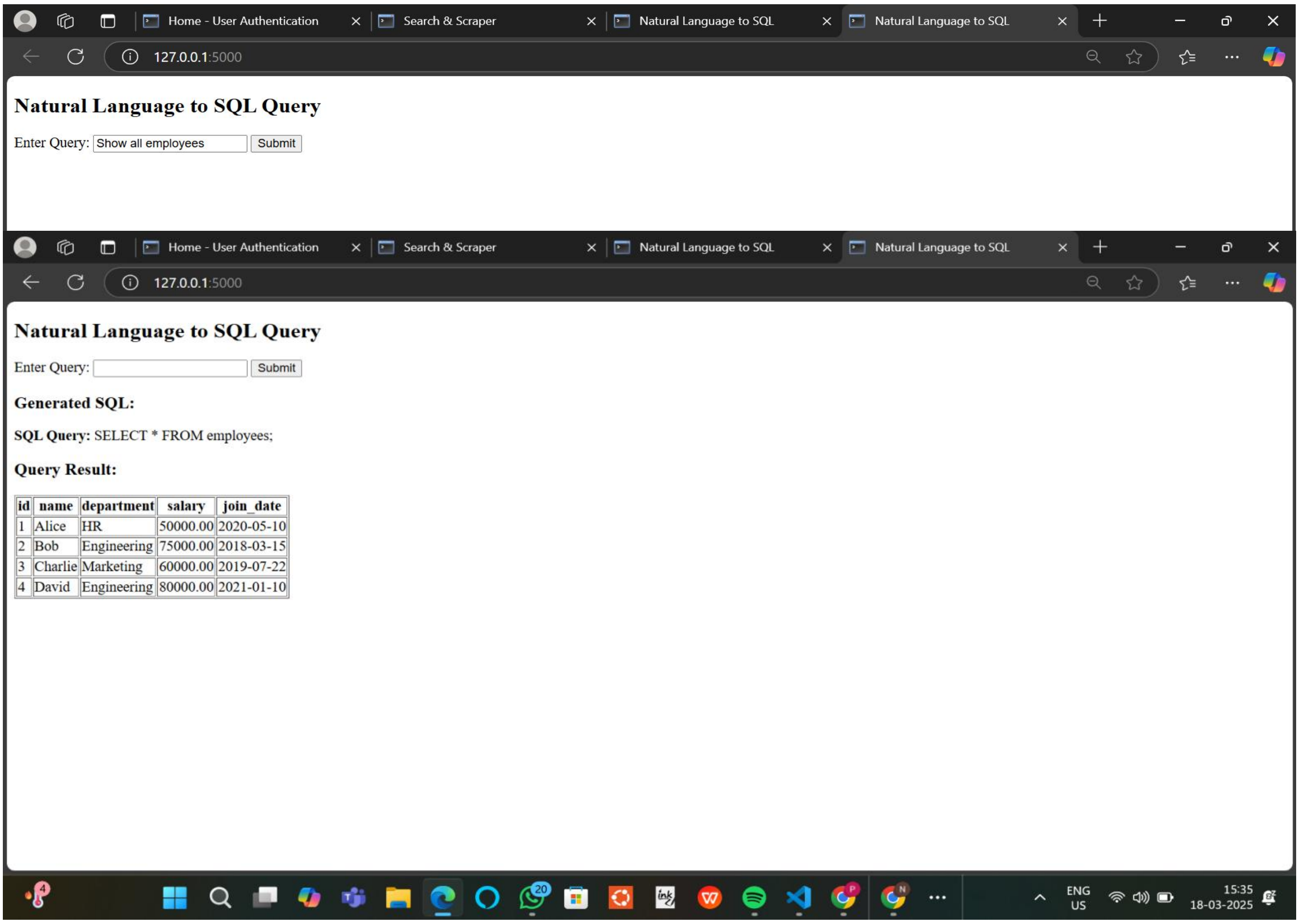
```
        <h3>Query Result:</h3>
        <table border="1">
            <tr>
                {% for column in result[0].keys() %}
                    <th>{{ column }}</th>
                {% endfor %}
            </tr>
            {% for row in result %}
                <tr>
                    {% for value in row.values() %}
                        <td>{{ value }}</td>
                    {% endfor %}
                </tr>
            {% endfor %}
        </table>
    {% endif %}
</body>
</html>
```

## SQL FILE

```
CREATE DATABASE nl_query_db;
USE nl_query_db;

CREATE TABLE employees (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    department VARCHAR(50),
    salary DECIMAL(10,2),
    join_date DATE
);

INSERT INTO employees (name, department, salary, join_date) VALUES
('Alice', 'HR', 50000, '2020-05-10'),
('Bob', 'Engineering', 75000, '2018-03-15'),
('Charlie', 'Marketing', 60000, '2019-07-22'),
('David', 'Engineering', 80000, '2021-01-10');
```

```
SHOW TABLES;

DESC employees;

SELECT * FROM employees;

SHOW ERRORS;
```

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



**Natural Language to SQL Query**

Enter Query: Show all employees Submit

**Natural Language to SQL Query**

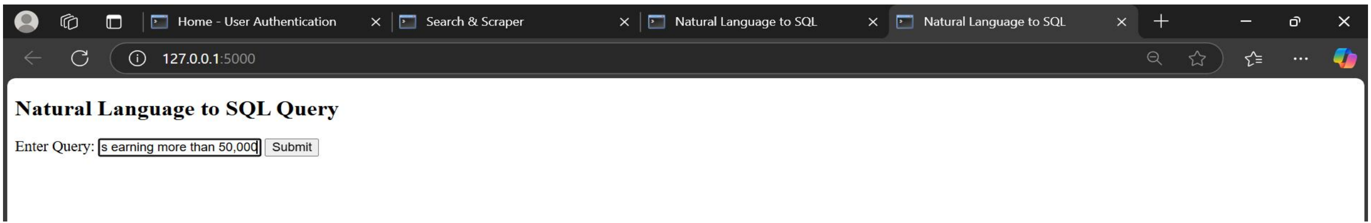
Enter Query: Submit

**Generated SQL:**

**SQL Query:** SELECT * FROM employees;

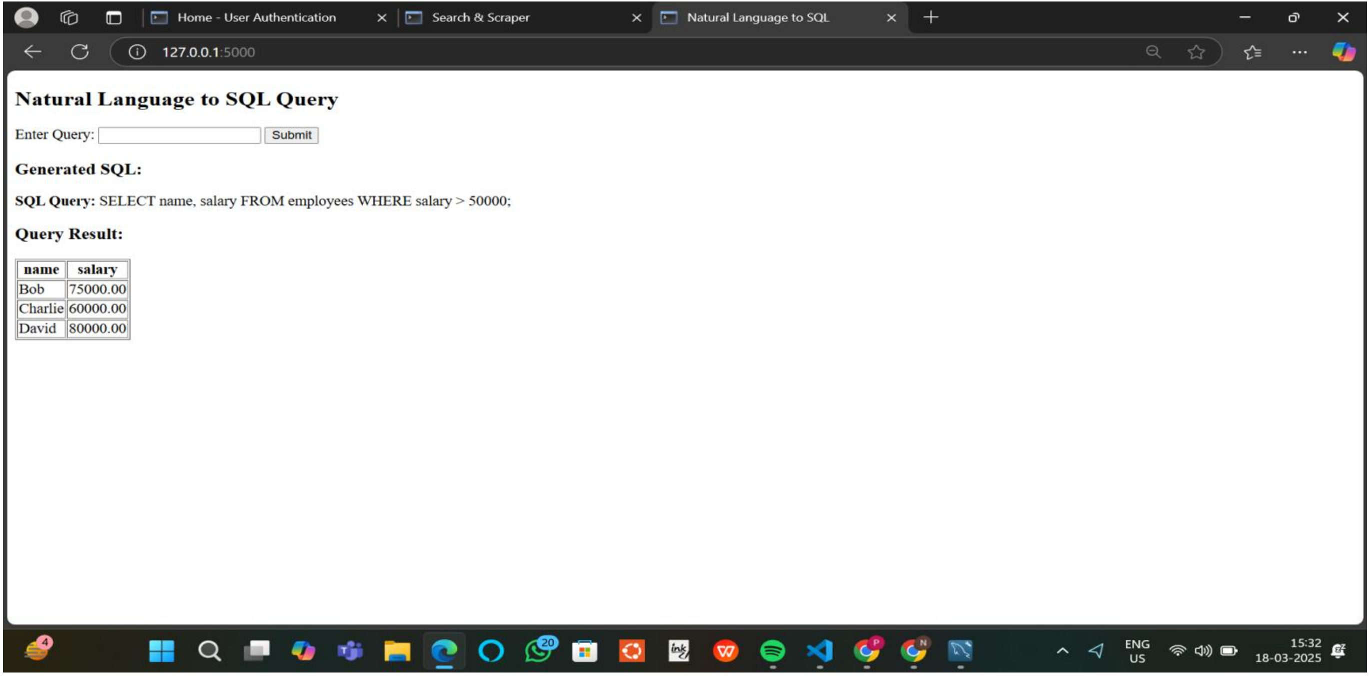
**Query Result:**

| id | name | department | salary | join_date |
|---|---|---|---|---|
| 1 | Alice | HR | 50000.00 | 2020-05-10 |
| 2 | Bob | Engineering | 75000.00 | 2018-03-15 |
| 3 | Charlie | Marketing | 60000.00 | 2019-07-22 |
| 4 | David | Engineering | 80000.00 | 2021-01-10 |



### Natural Language to SQL Query

Enter Query: s earning more than 50,000 Submit



### Natural Language to SQL Query

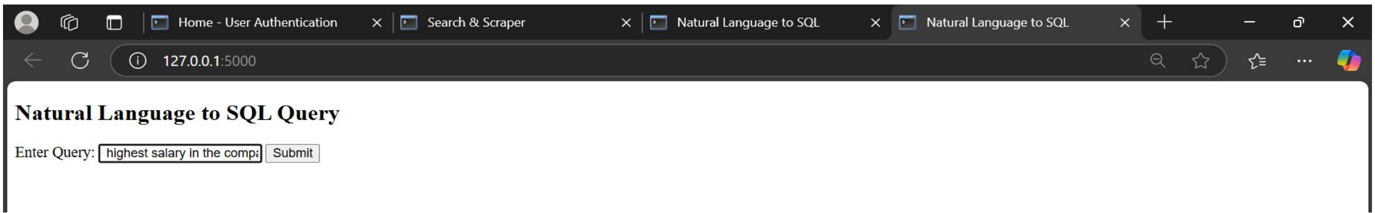
Enter Query: Submit

#### Generated SQL:

**SQL Query:** SELECT name, salary FROM employees WHERE salary > 50000;

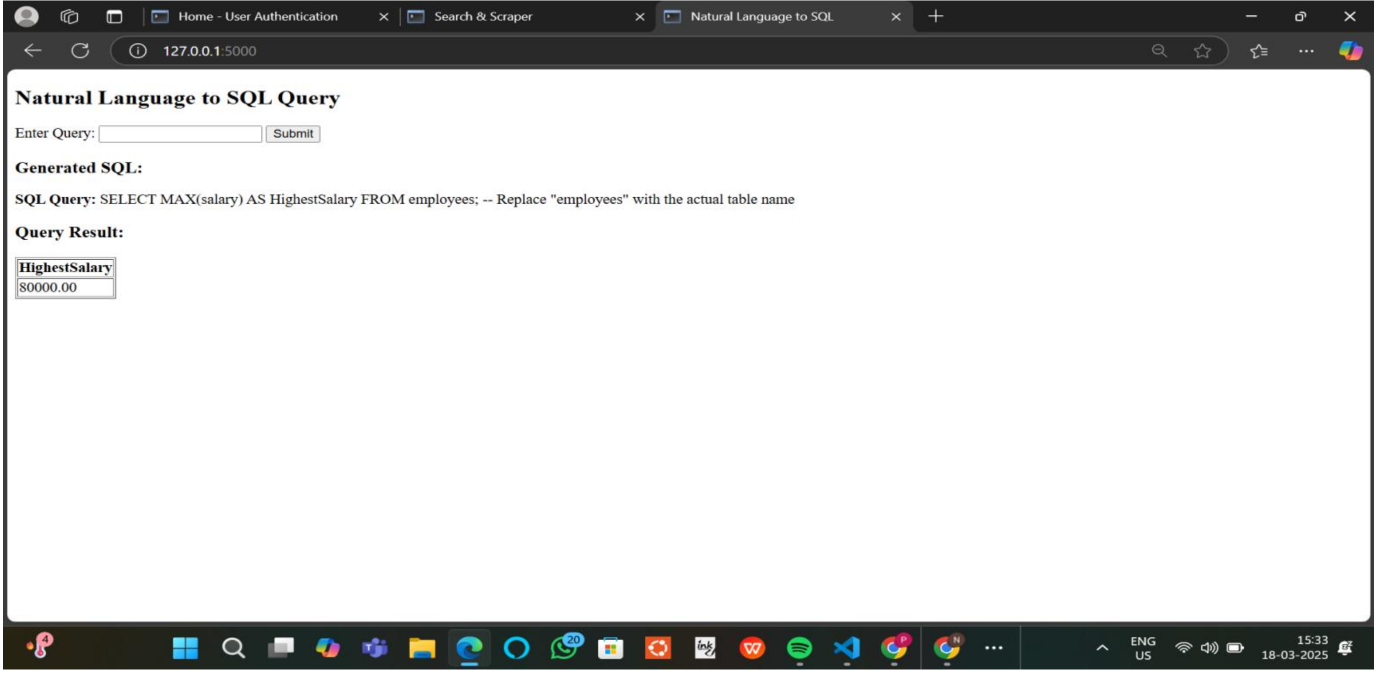
#### Query Result:

| name | salary |
|---|---|
| Bob | 75000.00 |
| Charlie | 60000.00 |
| David | 80000.00 |



### Natural Language to SQL Query

Enter Query: highest salary in the compa Submit



### Natural Language to SQL Query

Enter Query: Submit

#### Generated SQL:

**SQL Query:** SELECT MAX(salary) AS HighestSalary FROM employees; -- Replace "employees" with the actual table name

#### Query Result:

| HighestSalary |
|---|
| 80000.00 |

## 4. Remarks:-

This experiment successfully demonstrated how to convert natural language queries into SQL statements and execute them using Flask and MySQL. By integrating a user-friendly frontend, we enabled seamless interaction with the database, allowing users to retrieve information without needing SQL expertise. The project highlights the potential of natural language processing in simplifying database management and can be further improved by refining query translation accuracy and expanding database support.

Pruthibiraj Nayak (2230183)

_______________________ ______________________________________

(Name of the Student) (Name of the Coordinator)