**CIS 530 – Bonus Assignment**

**Name: Saritha Reddy Loka**

**CSU ID: 2862662**

**Object: Build a search function in a company website**

**Problem Description: You are building a site called a Company Website for users who can search for an employee working in the company. This is a bonus lab for the students who want to improve their grades. This is not a mandatory lab.**

**Functionality explained:**

As per the problem description, I created a simple website titled “**Employee Search”** for a company database which is useful for searching for an employee working in the company.

The primary functionality of this website is user will have a text box to enter the employee’s last name and a search button to click. On clicking the search button, the system will hit the database and search for the employee based on the last name that user has given.

If an employee found, his/her first name, last name, SSN, and birth date details will be fetched and displayed to the user.

If employee not found, system displays the message saying, “No employee data found with the entered name!” to the user.

If user clicks Search button without entering any name, system displays an error message saying, “Last name not provided!”.

Users will also have a Reset button to clear the details or error message displayed to the user.

**Languages and Tools used:**

To build this website I have used Python language for the front end and MySQL for the backend/database. I have used Visual Studio Code (Version: 1.88.1) and Microsoft SQL Server Management Studio (Version: 19.3.4.0) and operating system windows 11.

**Github repository link to access code, document, screenshots:**

<https://github.com/sarithareddy367/sarithareddy367/tree/main/CIS530_BonusProject_EmployeeSearch>

**Code explained:**

In visual studio, I have created a folder or project name called EMPLOYEESEARCH. It has a main python file named **app.py** where the program execution starts from this file where it connects to database and fetches the data. And a folder called **templates** which contains html files / templates that is **EmpSearch.html** to enter and display the user data fetched from database.

**Below is the code from app.py:**

I have imported python libraries that are used in application Flask, render\_template, request, jsonify, pyodbc.

from flask import Flask, render\_template, request, jsonify

import pyodbc

To connect to database, I built connection string like below.

connection\_string = (

f'DRIVER={{SQL Server}};'

f'SERVER=DESKTOP-52QCFAR;'

f'DATABASE=Company\_SarLoka;'

f'Trusted\_Connection=yes;'

)

To execute python script, we need to start Flask web application. Below script runs directly on python interpreter in debug mode on port 8000 using app.run() method.

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, port=8000)

Below code is to set up a Flask route for handling form submissions, retrieving employee details based on the last name provided in the form, and rendering a template with appropriate messages or data based on the result of the form submission.

@app.route('/', methods=['GET', 'POST'])

def index():

if request.method == 'POST':

message=''

empLastName = request.form['LastName']

if not empLastName:

return render\_template('EmpSearch.html', message='Last name not provided!')

employeeData = fetch\_employee\_details(empLastName)

if not employeeData:

return render\_template('EmpSearch.html', message='No employee data found with the name entered!')

else:

return render\_template('EmpSearch.html', employeeData=employeeData)

else:

return render\_template('EmpSearch.html')

Here, I retrieved the last name entered by user using request.form[] method. To post this data to database I set up request.method to POST. I passed this last name value to fetch\_employee\_details() function to send to database. If the data is not found or last name not entered, I display appropriate messages to the user. If the data is found, I send this data to EmpSearch.html template to render the data using render\_template() method.

Below is the code of fetch\_employee\_details() function.

def fetch\_employee\_details(empLastName):

try:

connection = pyodbc.connect(connection\_string)

with connection.cursor() as cursor:

query = "SELECT FNAME, LNAME, SSN, BDATE FROM EMPLOYEE WHERE LNAME=?"

cursor.execute(query, empLastName)

result = cursor.fetchall()

return result

finally:

if connection:

connection.close()

Here, I connected to the database using the library pyodbc.connect() method. And I passed last name as parameter and executed SQL query to retrieve the result using execute() method. And fetched the data using fetchall() method and returned the result.

**Below is the code from Emp.Search.html:**

I created a form with a post method to display a field/text box to enter last name and submit using Search button. To reset the data displayed there is a Reset button too. I used CSS to style the page and all the html tags.

<h1>Employee Search</h1>

<form method="post">

<label for="LastName" id="lblLastName">Last Name:</label>

<input type="text" id="LastName" name="LastName">

<br><br>

<input type="submit" id="btnSearch" value="Search">

<input type="reset" id="btnReset" onclick="clearResult()" value="Reset">

</form>

I displayed the result, or any messages retrieved from app.py to the user using the code below. I created an unordered list and displayed the data.

{% if message %}

<h4>{{message}}</h4>

{% endif %}

{% if employeeData %}

<h3>Employee Details</h3>

<ul>

<li>First Name: {{employeeData[0][0]}}</li>

<li>Last Name: {{employeeData[0][1]}}</li>

<li>SSN: {{employeeData[0][2]}}</li>

<li>Birth Date: {{employeeData[0][3]}}</li>

</ul>

{% endif %}

To clear the result, I wrote a function called clearResult(). When the user clicks on the Reset button, this function will hit and does accordingly. Below is the code of this function.

function clearResult() {

var employeeDetails = document.getElementById('empDetails');

if (employeeDetails) {

employeeDetails.innerHTML = '';

}

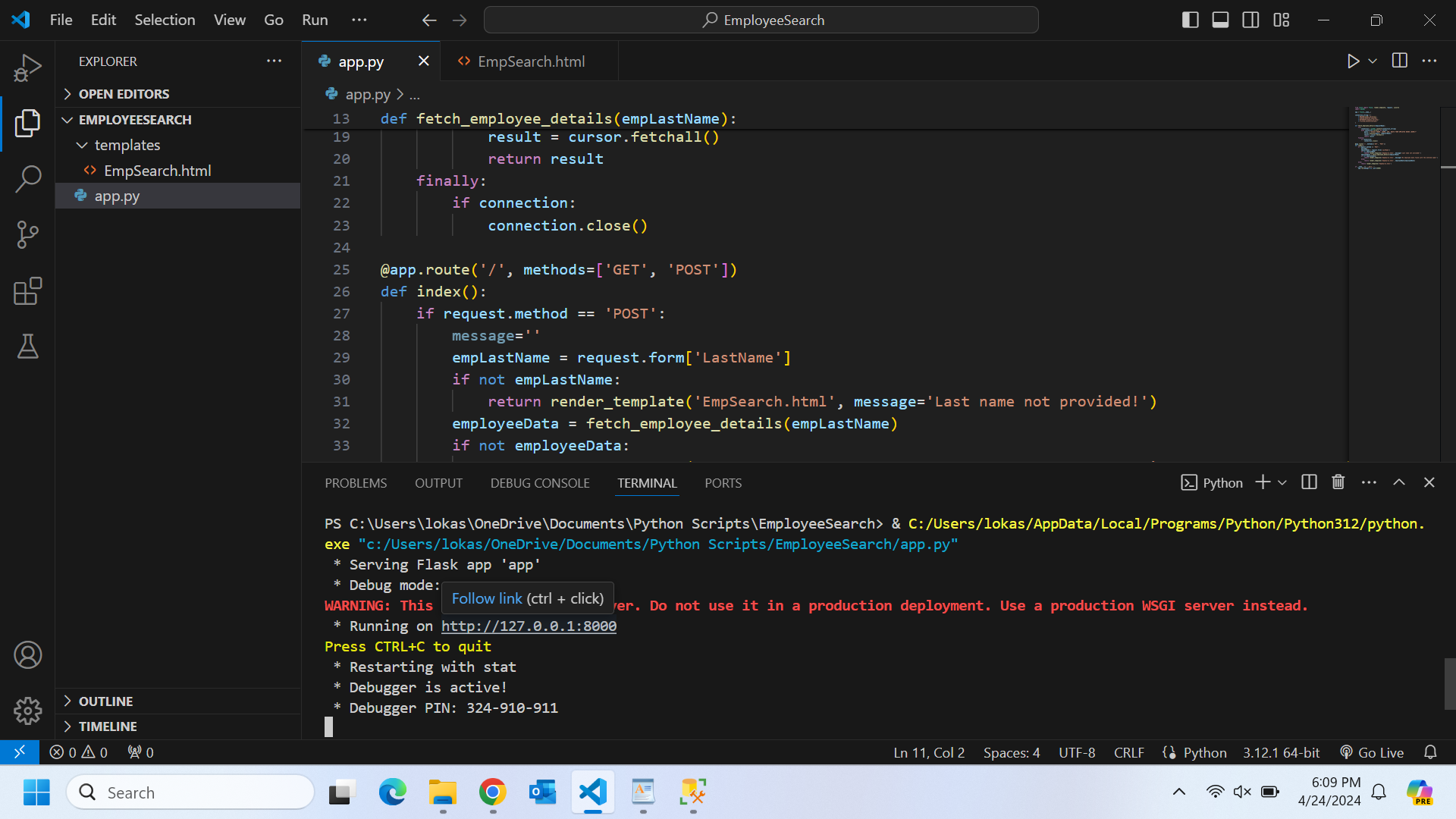
employeeData = {};

document.getElementById('LastName').val('');

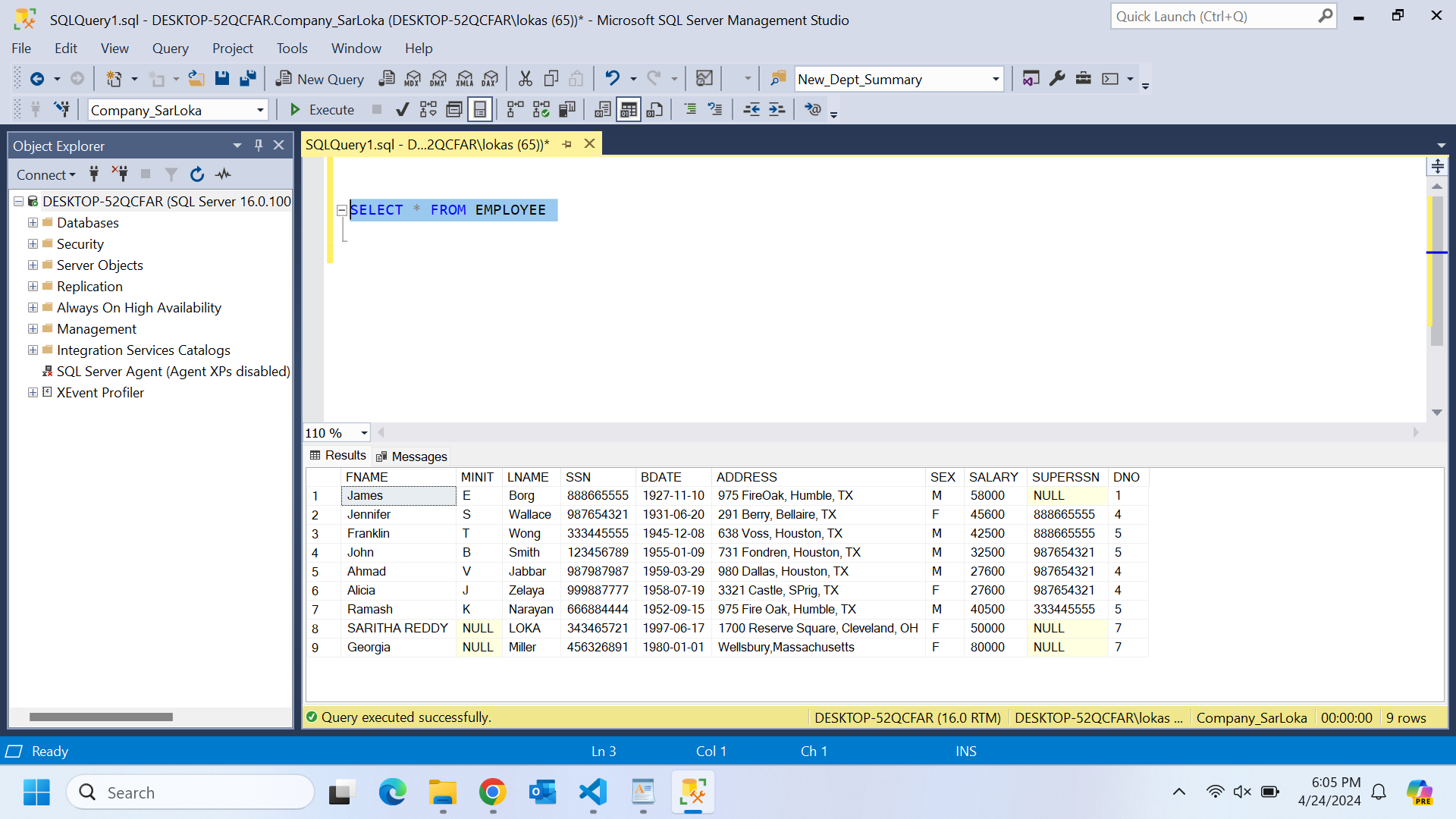
}

**Screenshots:**

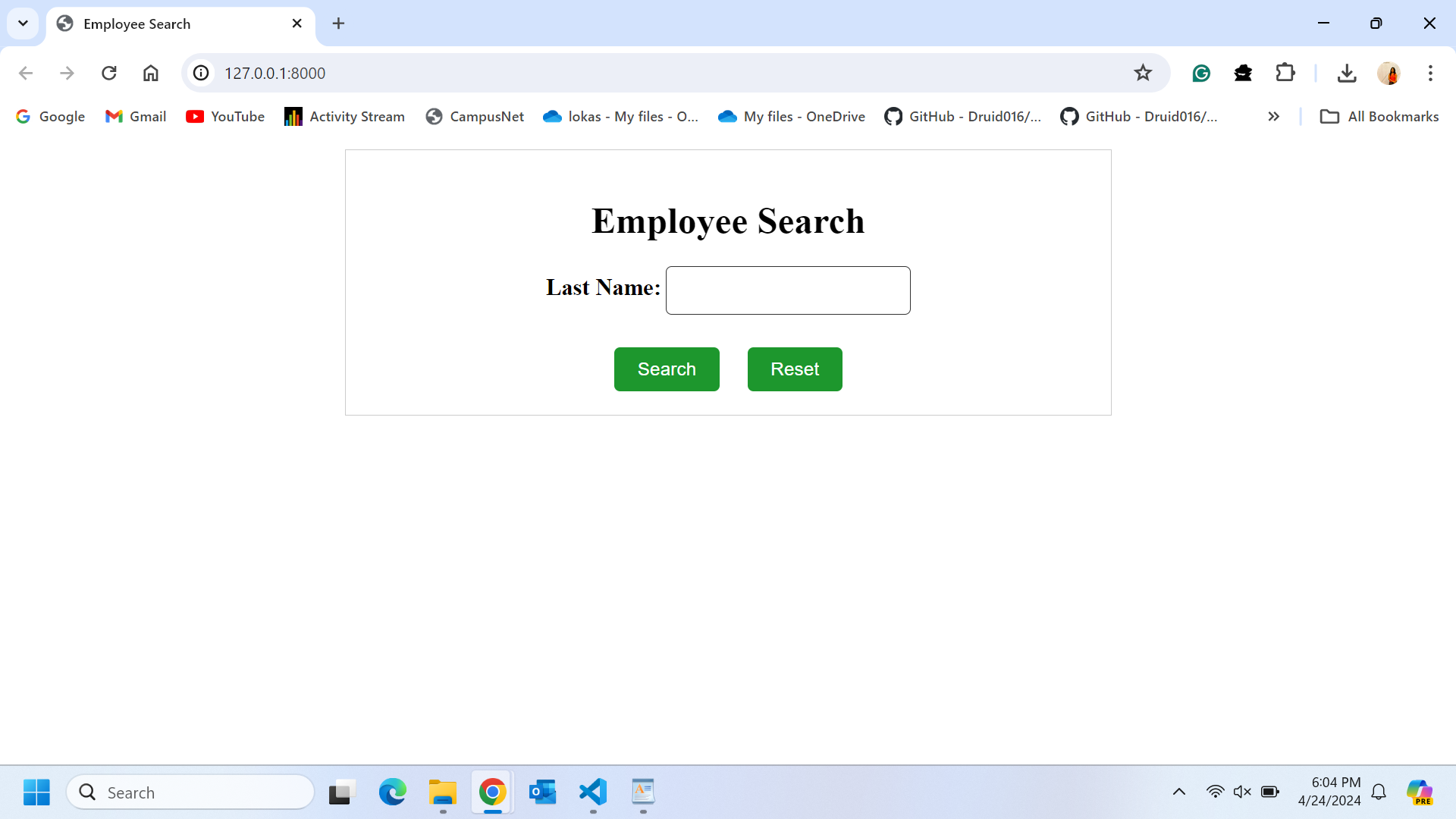
When we execute the application or app.py, it starts Flask app and runs locally using the port 8000 that we created. Below is the screenshot of when we execute the application, and it displays the link <http://127.0.0.1:8000/>.



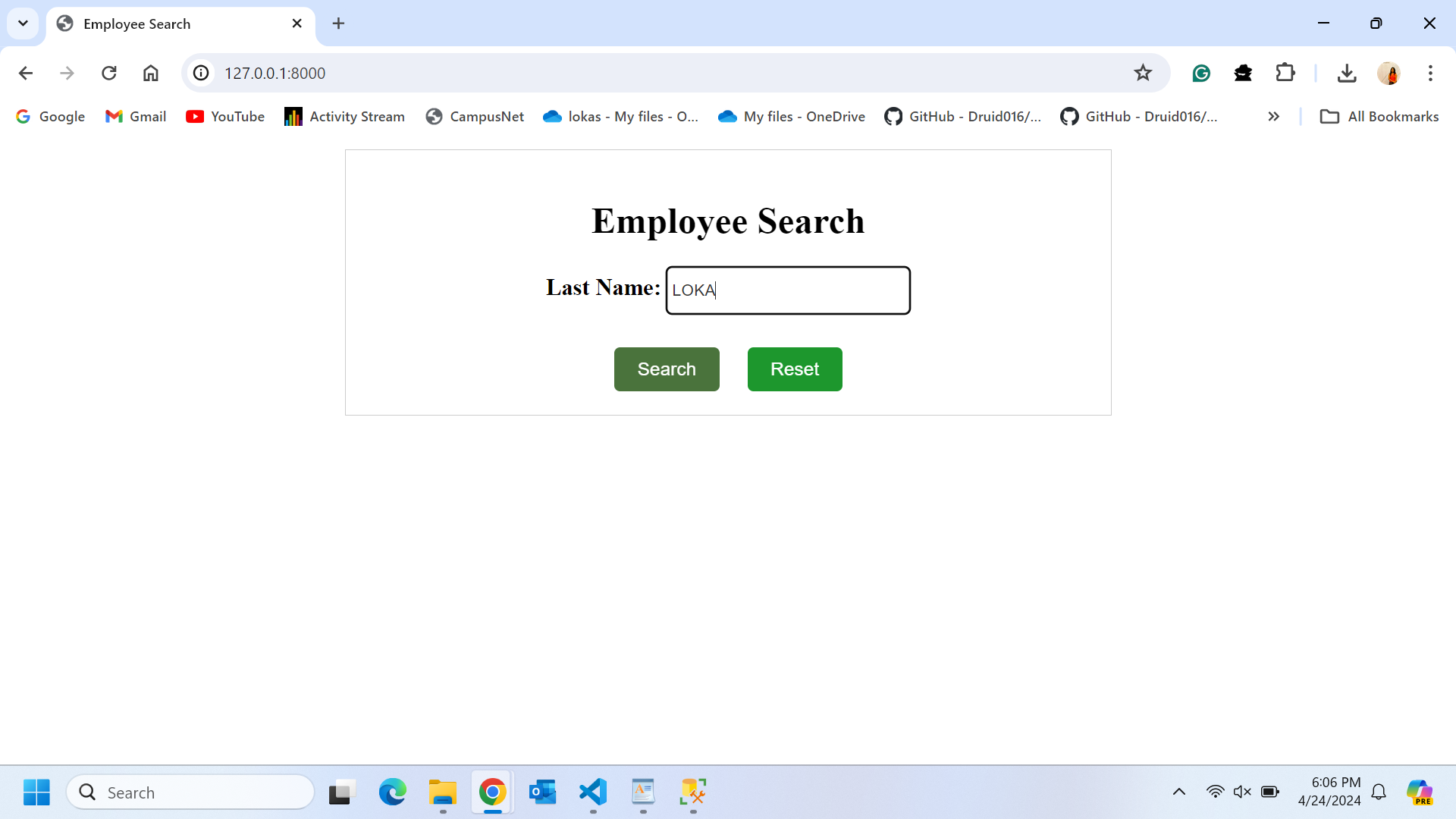
Below is the screenshot of the EMPLOYEE table in the database. The screenshot shows below is the employees list in our database.



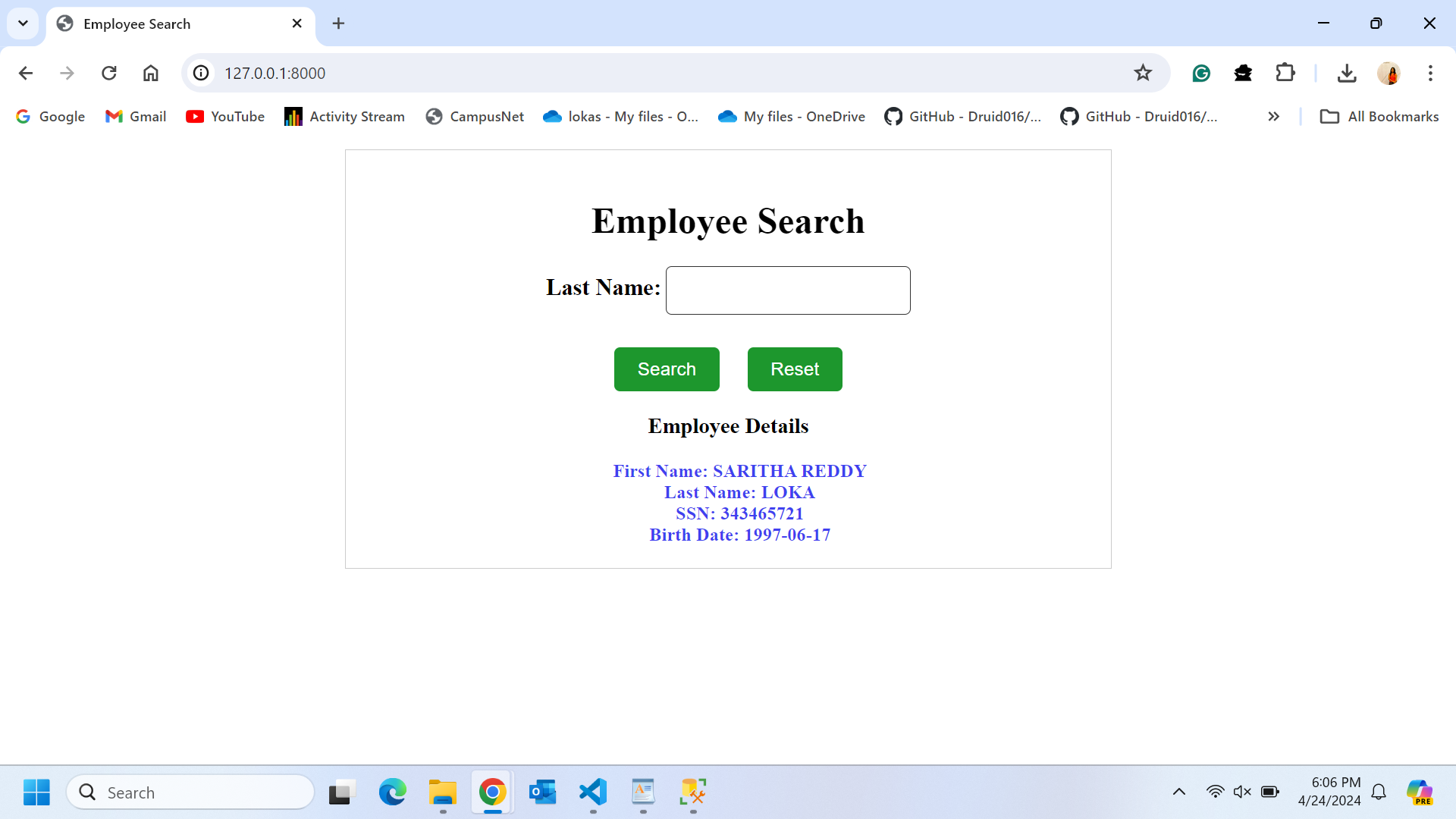
Below screenshot is the main page of our application.



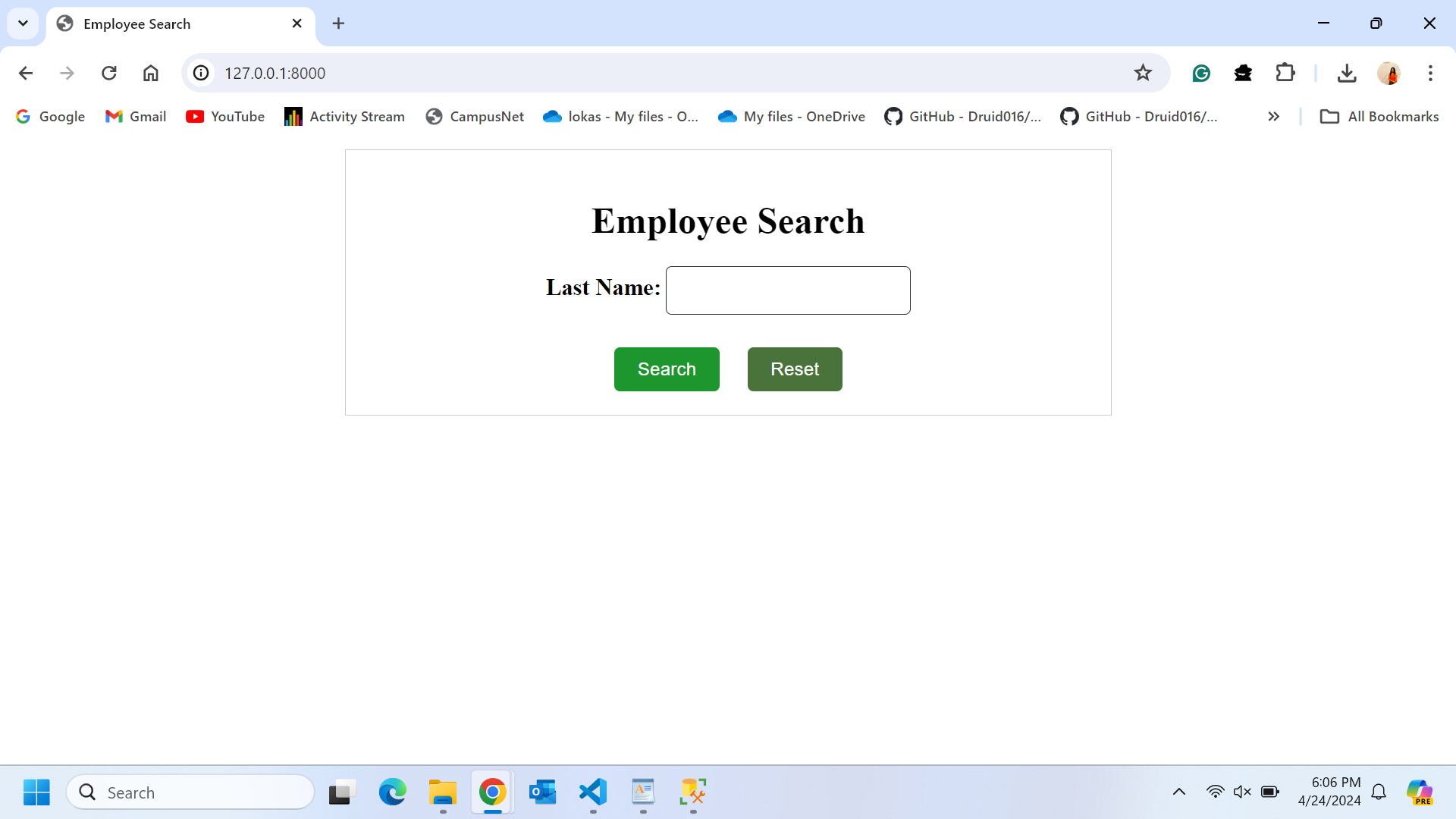
Below is the screenshot when user is entering the data in last name field and clicking on Search button.



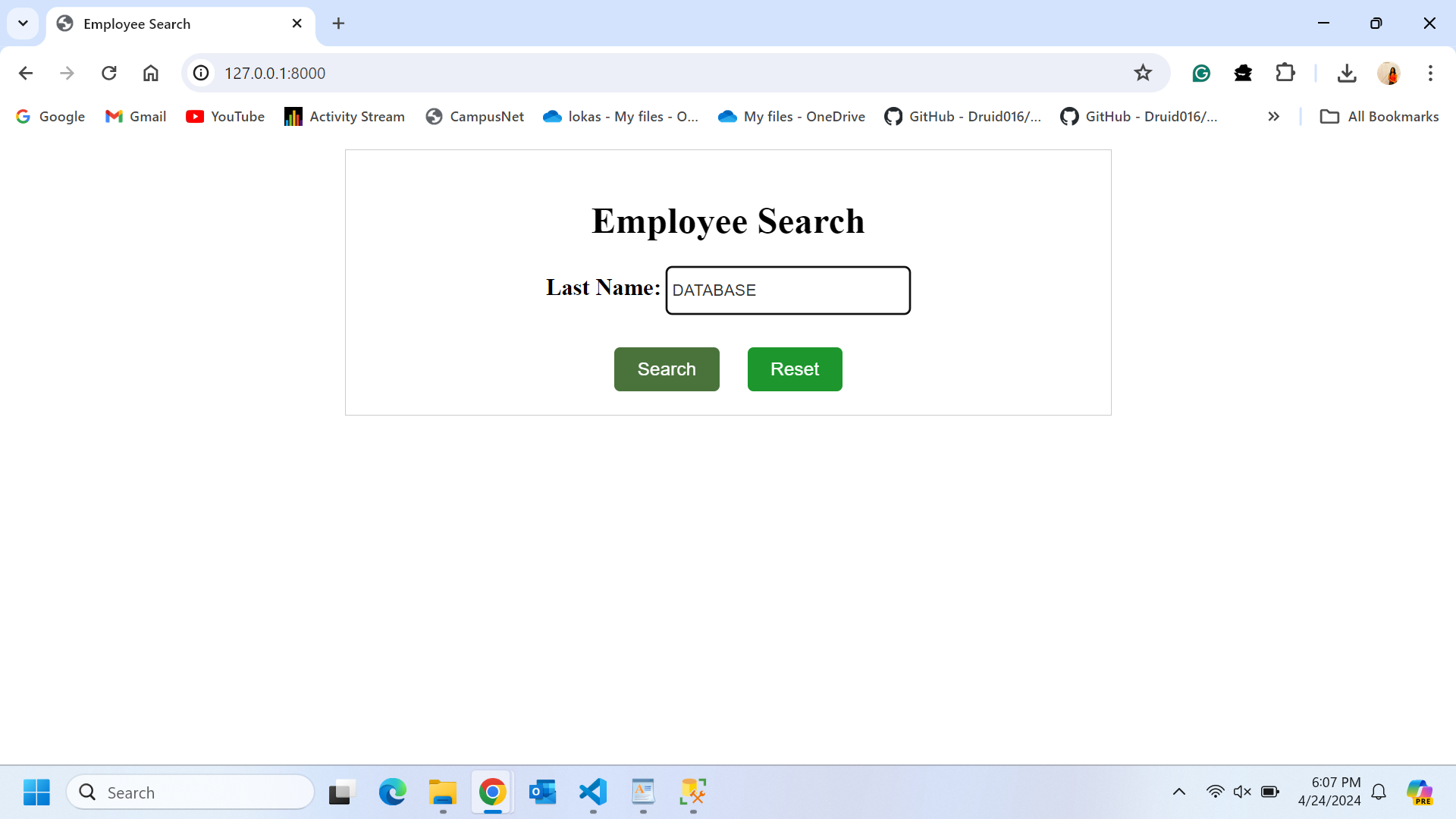
After clicking the Search button, the system will connect to database and fetches the result and displays it to user. Below is the screenshot for that. We can see that employee details are gathered from the database.

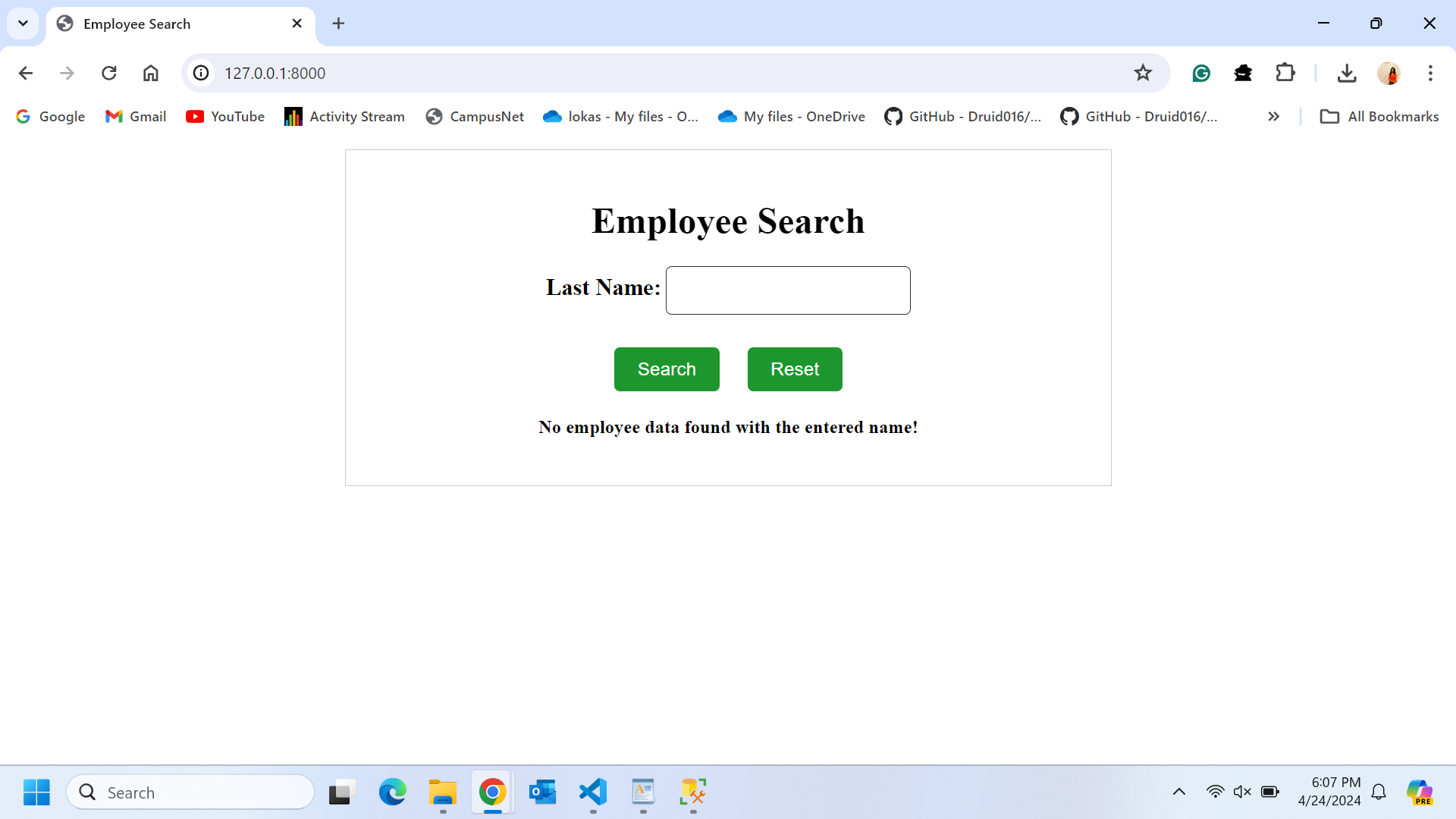


When the user clicks on the Reset button, the data will be erased. Below is the screenshot for that.

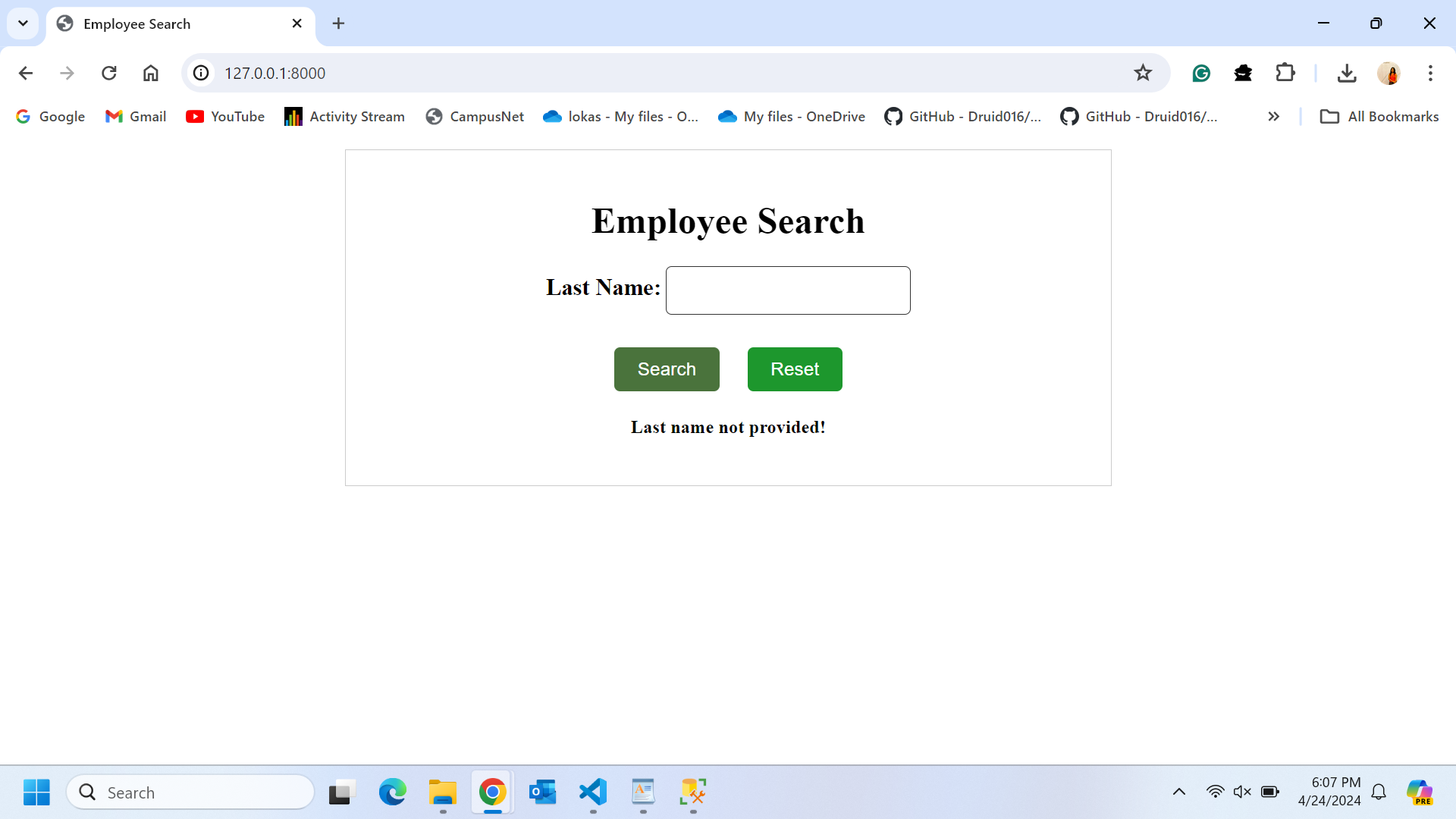


If the user enters an incorrect employee's last name which is not available in the database, the system will display an appropriate message. Below are the screenshots.





If the user clicks on the Search button without providing any last name, the system will display an appropriate error message to the user. Below is the screenshot for that.



The last name value entered by user is not case-sensitive, allowing the user to input data in either uppercase or lowercase. The data will be fetched regardless of the case. Below is the screenshot of this scenario.

