DBS Mini Project

Global Avian Invasions Atlas
A database of alien bird distributions worldwide

Team Members:

- Vaishnavi D – 225890242
- Dhruv Thejas KJ – 225890040

Aim:
The project aims to establish a novel high-level language query system integrating the Global Avian Invasions Atlas with an AI model. This system will enable users to enter natural language queries and obtain SQL outputs that meet their information requirements regarding invasive bird species and their ranges globally.

Principle:
The project employs machine learning and natural language processing (NLP) approaches to enable the AI model to comprehend and analyze high-level language input. It utilizes preprocessing of the dataset to improve compatibility with the AI model and enhance its performance in understanding user inquiries. Training the AI model involves using sophisticated language questions about bird ranges, habitats, and invasion trends. Iterative testing and optimization are conducted to enhance the model's ability to produce precise and contextually relevant SQL outputs.

Problem Statement:

The Avian Species Database aims to address the challenge of efficiently managing and accessing comprehensive information on avian biodiversity. This encompasses the taxonomy, distribution, and introduction status of avian species, crucial for various stakeholders including researchers, conservationists, and policymakers. The problem entails:

1. Data Fragmentation: Existing information on avian species is scattered across disparate sources, leading to fragmentation and difficulty in accessing consolidated data for analysis and decision-making.

2. Taxonomic Complexity: The intricate taxonomy of avian species requires a structured approach to capture and classify taxonomic details accurately, facilitating the organization and retrieval of species data.

3. Geographic Variation: Avian species exhibit diverse distribution patterns across countries, regions, and habitats, necessitating a systematic framework to document their geographic distribution and associated attributes.

4. Introduced Species Monitoring: With the increasing introduction of avian species into new habitats, there is a pressing need to track and manage introduced species' impact on ecosystems, requiring robust systems to record introduction details and monitor their spread.

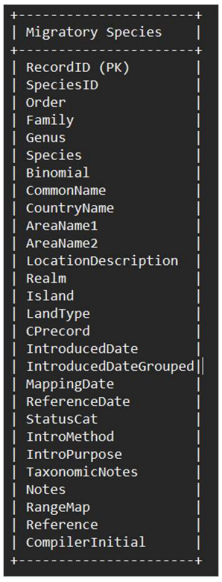
5. Data Accessibility: Accessing avian species data should be user-friendly and intuitive, enabling stakeholders to retrieve relevant information efficiently for research, conservation planning, and policy formulation.

Addressing these challenges requires the development of a centralized database solution capable of consolidating avian species data, providing comprehensive taxonomic classification, capturing geographic distribution patterns, monitoring introduced species, and ensuring ease of data access and management for diverse user needs.

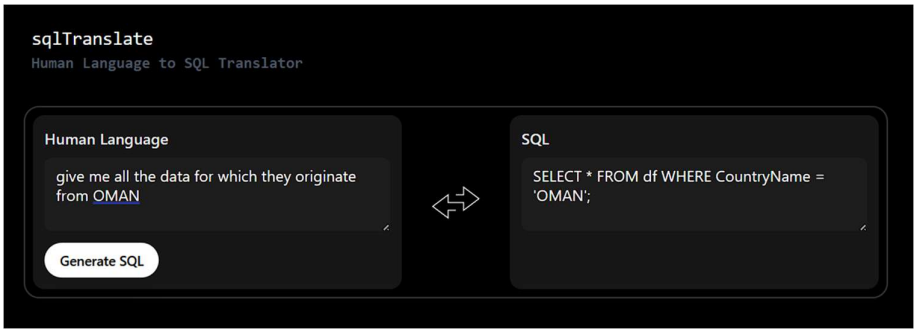
Technologies Used –
Flask: Flask is a lightweight Python web framework known for its simplicity and flexibility, allowing developers to quickly create web applications with a clean and concise design philosophy.
React: React is a popular JavaScript library for building user interfaces, known for its component-based architecture and efficient rendering capabilities. It enables developers to create interactive and dynamic web applications with ease.
Python: Python is a widely used programming language known for its simplicity and readability, making it ideal for web development.
TypeScript: TypeScript is a superset of JavaScript that adds static typing to the language, providing improved code maintainability and scalability. It enhances developer productivity by catching errors early in the development process and providing better tooling support.
CSS: CSS is used for styling the HTML elements and making the user interface visually appealing.

Database:
GAVIA.xlsx

E-R Diagram:



Frontend: (CSS, Typescript including React was used)
We used Flask to connect the backend to the front end.
The website is hosted locally.



Output:

output.xlsx (Since we have made use of a large database, the website couldn't be embedded with the output. So, the output is re-directed to a CSV file in the file directory)

src > pages > index.tsx > page

```tsx
import React, { useEffect, useState } from "react";
import dynamic from "next/dynamic";
import Head from "next/head";
import ThemeButton from "../components/ThemeButton";
import LoadingDots from "../components/LoadingDots";
import { useTheme } from "next-themes";
import Toggle from "../components/Toggle";
import { Header } from "../components/Header/Header";
import { GreenOrb, OrangeOrb, WhiteOrb } from "../components/atoms/Orbs";
import { Table, TableHead, TableHeader, TableRow } from "@/components/ui/table";



const page = () => {
  const [inputText, setInputText] = useState("");
  const [data, setData] = useState([]);
  const [outputText, setOutputText] = useState("");
  const submit = (event: React.MouseEvent<HTMLButtonElement, MouseEvent>) => {
    event.preventDefault();
    const input = inputText.trim();
    if (input === "") {
      return;
    }
    fetch(
      "http://127.0.0.1:5000/translate",
      {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify({
          inputText: input,
        }),
      }
    ).then((response) => response.json())
      .then((data) => {
        setOutputText(data.outputText);
        setData(data.result);
        console.log(data);
```

Backend: (Python with API integration)

```python
import pandas as pd
import pandasql as ps
import requests
from flask import Flask,request
from flask_cors import CORS, cross_origin

app = Flask(__name__)

CORS(app, supports_credentials=True)

@app.route('/')
def hello():
    return 'Hello, World!'

@app.route('/translate', methods=['POST'])
def translate():
    print (request.get_json())
    user_input = request.get_json()['inputText']
    print(user_input)
    payload = {
      "inputText": user_input,
      "tableSchema": "CREATE TABLE df (RecordID INT PRIMARY KEY, SpeciesID INT, OrderName VARCHAR(255), FamilyName VARCHAR(255), GenusNam
    }
    url = "https://www.sqltranslate.app/api/translate"
    response = requests.post(url=url, json=payload)
    print(response)
    if response.status_code == 200:
        df = pd.read_csv('GAVIA.csv', encoding='cp1252')
        q1 = response.json()['outputText']
        result = ps.sqldf(q1, locals())
        result.to_csv('output.csv', index=False)
        json1 ={
            "outputText": q1,
            "result": result.to_json(orient='records')
        }
        return json1
    else:
        return f"Request failed with status code {response.status_code}"
```

Environment:

```
home = C:\Python312
include-system-site-packages = false
version = 3.12.3
executable = C:\Python312\python.exe
command = C:\Python312\python.exe -m venv C:\Users\vaish\Downloads\Vaish-DBMS-main\Vaish-DBMS-main\myenv
```

Sources:

The global avian invasions atlas, a database of alien bird distributions worldwide | Scientific Data (nature.com)