



NUS
National University
of Singapore



GRADUATE CERTIFICATE

INTELLIGENT REASONING SYSTEM (IRS)

PROJECT REPORT



- Enhancing Birdwatching with AI

GROUP MEMBERS

Ashiwin Rajendran (A0291191E)

Jithin Krishnan (A0249481W)

Prem Varijakzhan (A0291913B)

Shankar Sai Ganesh (A0168149U)

Table of Content

1. Executive Summary	5
Preface.....	5
Problem.....	5
Opportunity.....	5
Solution	5
Design TechniquES	5
Closing Thoughts	6
2. Project Background	7
Market Context.....	7
Problem Statement.....	7
Objective.....	7
3. System Design	9
Application Features	9
Bird Informational Queries	9
Objective:	9
Technology/Algorithms Used:	9
Working Mechanism:	9
Text Based Bird Identification	9
Objective:	10
Technology/Algorithms Used:	10
Working Mechanism:	10
Image Based Bird Identification	10
Objective:	10
Technology/Algorithms Used:	10
Working Mechanism:	10
Bird Sighting Prediction.....	11
Objective:	11
Technology/Algorithms Used:	11
Working Mechanism:	11
Park Recommendation	12
Heat Map	12
Operational Flow	13
System Architecture.....	14
Architectural Diagram	14
System Components.....	15

User Interfaces	15
Backend Server	15
LLM Models	16
Machine Learning Models	16
Data Storage	16
4. Implementation	17
Application Frameworks and User Interfaces.....	17
Backend API	17
Web User Interface	17
Key features.....	17
Mobile Application (Android).....	18
Key Components	18
Large Language Models Integration	19
Integration Details	19
User Intent detection to use respective ML models.	19
Intent Detection Workflow:	19
Model creation for each Features.....	20
Bird Informational Queries	20
Data Collection.....	20
Data Conversion and Storage.....	21
Model Creation	21
Model Performance	24
Text Based Bird Identification	24
DATA COLLECTION	24
DATA CONVERSION AND STORAGE	24
MODEL CREATION	24
Model Performance	26
Image Based Bird Identification	27
Data Collection.....	27
DaTA CONVERSION and StORAGE	28
Model Creation	28
Model Performance	29
Bird Sighting Prediction.....	30
Data Preparation.....	30
Model Creation	31
Feature Importance and Model Tuning	32
Validation and Performance Metrics.....	33
Conclusion	34
Park Recommendation	34
Overview	34

Data Collection.....	35
Data preparation.....	35
Model Creation	36
Predictions and Rating Estimation:.....	37
Model Performance	37
Evaluation of other recommendation models for comparison	38
Heatmap	38
Data Preparation.....	38
Model Creation	39
5. Version Control and Deployment	41
Version Control with Git and GitHub.....	41
DEPLOYMENT and Hosting in Heroku CLOUD	41
Continuous Integration/Continuous Deployment (CI/CD)	41
6. Testing and Validation	42
7. Conclusion	46
8. Recommendations	47
9. References.....	48
APPENDIX A: PROJECT PROPOSAL	48
APPENDIX B: INDIVIDUAL REPORT	49
APPENDIX C: MAPPED SYSTEM FUNCTIONALITIES AGAINST KNOWLEDGE OF COURSES KNOWLEDGE OF COURSE.....	64
APPENDIX D: INSTALLATION AND USER GUIDE	65

1. EXECUTIVE SUMMARY

PREFACE

Singapore's diverse avian population often goes unnoticed during urbanization. To address this, we introduce an innovative AI chatbot application called WingSpan designed to enhance birdwatching experiences. This report outlines the problem, the opportunity it presents, our proposed solution, design techniques employed, and closing thoughts on the project.

PROBLEM

Despite Singapore's rich bird species, there is a lack of accessible tools for bird identification and locating birdwatching spots. Traditional methods are time-consuming and require expertise, while information on bird hotspots is often scattered and outdated and is not easily accessible.

OPPORTUNITY

Advancements in AI, including GPT-4 and ML models, offer an opportunity to revolutionize birdwatching. By leveraging these technologies, we can create a user-friendly platform that simplifies bird identification and enables easy discovery of birdwatching spots, fostering greater appreciation for Singapore's natural environment.

SOLUTION

Our solution is an AI chatbot called WingSpan, which offers comprehensive functionalities:

- **Bird Information:** This feature provides users with extensive information about various bird species, including their habitat and conservation status, enabling users to expand their ornithological knowledge.
- **Bird Identification by Description:** Users can input descriptions of birds they encounter, and the system will assist in identifying the species, aiding in the exploration and discovery of avian life.
- **Bird Identification from Photos:** This feature allows users to upload photos of birds for identification, leveraging visual data for species recognition.
- **Bird Spotting Predictions:** Utilizing historical data, the application predicts the likelihood of spotting specific birds, aiding enthusiasts in planning their birdwatching activities.
- **Park Recommendation:** By analysing data on bird populations and activity, the system recommends parks for birdwatching.
- **Bird Abundance Heatmap in Singapore:** An interactive heatmap feature displays areas of high bird abundance within Singapore, serving as a strategic tool for birdwatching planning.

DESIGN TECHNIQUES

WingSpan incorporates several design techniques to ensure a seamless and enjoyable birdwatching experience:

- **User-Friendly Interface:** The chatbot interface is designed to be intuitive and easy to navigate, allowing users to interact effortlessly with the application.
- **Natural Language Processing:** Leveraging GPT-4's language understanding capabilities, WingSpan interprets user queries and descriptions accurately, enhancing the interaction experience.
- **Image Recognition:** WingSpan utilizes GPT-4 Vision Preview's image recognition capabilities to accurately identify bird species from user-uploaded images, providing instant results.
- **Personalized Recommendations:** WingSpan offers personalized recommendations for bird sightings and park locations based on user preferences, ensuring tailored suggestions for each user.
- **Data-Driven Bird Heat Map:** WingSpan provides a bird heat map generated by ML models that predict where birds will be sighted based on historical data for each week. While not real-time, this feature offers valuable insights into bird activity patterns.

CLOSING THOUGHTS

WingSpan represents a significant step forward in leveraging AI and machine learning to enhance birdwatching experiences. By addressing the existing challenges of bird identification and locating birdwatching spots, WingSpan empowers both seasoned enthusiasts and newcomers to explore and appreciate Singapore's avian diversity.

Through intuitive design and advanced technology, WingSpan not only simplifies the process of bird identification but also encourages users to engage with nature in a meaningful way. By fostering greater accessibility and interest in birdwatching, WingSpan contributes to the conservation and appreciation of Singapore's natural environment.

As we continue to refine and expand WingSpan's capabilities, we are excited about the potential to inspire more people to connect with the natural world and contribute to the preservation of our avian habitats. With ongoing updates and enhancements, WingSpan will remain a valuable tool for birdwatchers, promoting conservation efforts and enriching the experience of exploring Singapore's vibrant birdlife.

2. PROJECT BACKGROUND

WingSpan is envisaged as an innovative solution at the intersection of artificial intelligence and user experience design, specifically tailored for the bird-watching community in Singapore. This project is borne out of the observation that despite the rich biodiversity and the popularity of bird watching as a recreational activity, enthusiasts face considerable barriers that diminish their experience and engagement. These barriers include difficulties in species identification, lack of predictive insights for bird sightings, and absence of an integrated, user-friendly platform for accessing relevant information. WingSpan proposes a comprehensive AI-driven application that leverages machine learning algorithms, data analytics, and advanced user interface designs to overcome these challenges, thereby enhancing the bird-watching experience.

MARKET CONTEXT

The bird-watching community is diverse, ranging from casual observers to dedicated ornithologists. However, despite the widespread interest, enthusiasts face significant challenges in identifying species, predicting sightings, and locating birds. The current methods are time-consuming and often rely on disparate sources, lacking real-time data and predictive capabilities. Furthermore, the growing interest in eco-tourism and environmental conservation underscores the need for innovative solutions that enhance outdoor experiences while fostering a deeper connection with nature.

PROBLEM STATEMENT

Despite the availability of online databases and mobile applications for bird identification and logging, bird watchers in Singapore and globally face significant hurdles. These include the difficulty of identifying bird species from physical characteristics or photographs, the unpredictability of bird sightings, and the lack of personalized, location-specific advice for optimal bird-watching experiences. Furthermore, existing tools often do not account for temporal and environmental variables that significantly influence bird presence and behavior. These gaps in the current bird-watching landscape underscore the need for a holistic, data-driven solution.

OBJECTIVE

The overarching goal of WingSpan is to develop a multifaceted, AI-enhanced application that streamlines the bird-watching process, from species identification to sighting prediction, while providing a rich, educational user experience. Specifically, the project aims to:

1. Implement bird information enquiries.
2. Utilize image recognition technology for accurate species identification from user-uploaded photographs.
3. Implement the prediction of bird presence based on location, date, and environmental factors, utilizing comprehensive historical datasets.
4. Generate dynamic heatmaps reflecting bird sighting probabilities across Singapore, leveraging linear regression models and historical sighting data.
5. Offer personalized park recommendations using a content-based recommendation system, enhancing the user's bird-watching itinerary planning.
6. Create an engaging and user-friendly interface with a sophisticated LLM-powered chatbot for user interaction, capable of handling a variety of queries and guiding users through the application's features.

Through these objectives, WingSpan aspires to not only enrich the bird-watching experience but also contribute valuable data to bird conservation efforts, promote environmental education, and encourage public engagement with nature.

3. SYSTEM DESIGN

APPLICATION FEATURES

Following are the key application features in detail:

BIRD INFORMATIONAL QUERIES

OBJECTIVE:

To provide following information about various birds based on trained data:

- Scientific Name, name in different Languages (Chinese and Malay), Locations found across the globe, Taxonomy, Subspecies, Local Subspecies, Size, Identification details, Behavior and Ecology, Local Status and Conservation Status

TECHNOLOGY/ALGORITHMS USED:

- Python script using the Selenium library for web scraping and data preparation.
- 'GPT-4-turbo-preview' model for contextual understanding and searching data, providing bird information to the user.
- Open AI's Assistant API feature, fine-tuned with bird information data.

WORKING MECHANISM:

- **User Interaction:** Users make requests through a chat interface to provide the information they need.
- **Backend Reception:** The request is received at the backend API endpoint.
- **Intent Recognition:** The user's intent is identified by the LLM model, taking the conversation history into consideration.
- **Data Integration and Prompt Crafting:** Respective methods to communicate with the Assistant API are called with a crafted prompt to search the attached data and provide the bird information requested.
- **Querying the data:** The Assistant API that uses 'GPT-4-turbo-preview' model, already having the information of common birds as an attachment, receives a carefully crafted prompt to fetch the information. It searched the attached data to fetch the bird information.
- **Response Formulation:** The model formulates a response based on the retrieved information, which includes the details about the bird.
- **User Response Delivery:** This response is delivered back to the user through the interface, completing the identification process.

TEXT BASED BIRD IDENTIFICATION

OBJECTIVE:

To identify bird species based on textual descriptions or queries.

TECHNOLOGY/ALGORITHMS USED:

- GPT-4 LLM model for contextual understanding and identification based on textual input and additional prompts and data.
-

WORKING MECHANISM:

- **User Interaction:** Users make requests through a chat interface to provide the information they need.
- **Backend Reception:** The request is received at the backend API endpoint.
- **Intent Recognition:** The user's intent is identified by the LLM model, taking the conversation history into consideration.
- **Data Integration and Prompt Crafting:** Alongside the contextual analysis, the model integrates additional data provided for bird identification. This includes a specialized dataset of bird species characteristics. A specialized prompt is crafted using input data and contextual analysis. This prompt is designed to extract the most relevant information from the database for bird identification. The GTP4-model used in the assistant API uses the crafted prompt to retrieve information from its data sources that can help identify the bird species mentioned in the user's query.
- **Response Formulation:** The model formulates a response based on the retrieved information, which includes the bird's species name, characteristics, and potentially other relevant details.
- **User Response Delivery:** This response is delivered back to the user through the interface, completing the identification process.

IMAGE BASED BIRD IDENTIFICATION

OBJECTIVE:

To identify bird species from user-uploaded images.

TECHNOLOGY/ALGORITHMS USED:

- GPT-4-vision LLM model for contextual understanding and identification based on image analysis and additional prompts and data.
-

WORKING MECHANISM:

- **User Interaction:** A user uploads an image of a bird through the chat interface to request identification.
- **Backend Reception:** The uploaded image is received at the backend API endpoint, where it is prepared for analysis. The GPT-4-vision LLM model assesses the image's content, considering any conversation history that might provide context for the identification request.
- **Intent Recognition:** The user's intent is identified by the LLM model, taking the conversation history into consideration.
- **Bird Identification:** Using advanced image recognition capabilities of GTP-4 Vision model, the model conducts a contextual analysis of the image to detect features specific to bird species, such as color patterns, size, and shape. Concurrently, the model integrates additional data pertinent to bird identification. This data includes a curated list of common birds in Singapore. Utilizing the generated

prompt, the GPT-4-vision LLM model queries its data sources to find a match for the bird species in the image.

- **Response Formulation:** Once a match is found, the model formulates a response. This response will include the identified bird's species name, visual characteristics that led to the identification, and other pertinent details.
- **User Response Delivery:** The identification details are communicated back to the user via the chat interface, thus completing the bird species identification process based on the image provided.

BIRD SIGHTING PREDICTION

OBJECTIVE:

To leverage advanced machine learning techniques to predict the presence of bird species at a specific location and date, enhancing the bird-watching experience by providing accurate, data-driven insights.

TECHNOLOGY/ALGORITHMS USED:

- **Random Forest Classifier:** A machine learning algorithm known for its accuracy and robustness in classification tasks.
- **Data Collection & Preprocessing Techniques:** Selenium to extract data from sources and pre-processing. Includes normalization, outlier detection, and encoding to prepare the dataset for analysis.
- **Hyperparameter Tuning:** Techniques such as grid search and cross-validation to find the optimal settings for the model.
- **Environmental Data:** Incorporating current and forecasted weather data like temperature, humidity, and precipitation levels.
- **User Interaction:** GPT-4 LLM Assistant API model for contextual understanding from the user prompt and query handling to extract the relevant input features for the prediction model.

WORKING MECHANISM:

- **Advanced Preprocessing and Feature Engineering:** Sophisticated preprocessing techniques to standardize and normalize the data and removal of inconsistencies to ensure that ML algorithm can interpret it effectively.
- **Incorporation of Temporal and Spatial Analysis:** Utilizes temporal data analysis to understand seasonal patterns. Spatial analysis helps in understanding geographical distribution and location preferences, also integration of Environmental Variables is done for real-time and historical weather data as predictive variables. All these add a layer of prediction accuracy.
- **Machine Learning Model Optimization:** Detail the use of ensemble learning techniques, such as Random Forest, emphasizing their advantage in reducing overfitting and improving prediction robustness. Discuss the implementation of grid search and cross-validation in hyperparameter tuning, elaborating on how these techniques methodically explore various parameter combinations to identify the most effective model settings.
- **Data Handling and Prediction:** The chat handling mechanism for bird prediction integrates GPT-4 LLM to understand user queries and answer accordingly. It utilizes NLP techniques to interpret and extract data from queries effectively. It involves:
 - **Data Extraction:** Extracting relevant information (location, date) from user inputs using entity recognition and extraction of weather information from weather forecast API.

- **Response Generation:** Leveraging the prediction model to provide specific bird species predictions and presenting the information in a user-friendly manner through the chat interface.
- This approach ensures a seamless, interactive experience, guiding users with precision and efficiency.

PARK RECOMMENDATION

Objective:

To recommend parks that are most likely to feature specific bird species based on historical sighting data, enhancing the bird-watching experience for enthusiasts.

Technology/Algorithms Used:

- **Data Collection & Preprocessing Techniques:** Selenium to extract data from sources and pre-processing
- **Item-based Collaborative Filtering with KNNBasic:** Employs collaborative filtering techniques to predict the presence of bird species in various parks. This approach personalizes park recommendations based on historical data.
- **Hyperparameter Tuning:** Utilizes cross-validation to optimize the collaborative filtering model for better prediction accuracy.
- **GPT-4 LLM model** for contextual understanding of user intent and identifying the specific bird's name for prediction to feed to the model, providing bird information to the user.

Working Mechanism:

- **Data Handling and Preprocessing:** The system retrieves and preprocesses bird sighting data, normalizing counts and detecting outliers to prepare for collaborative filtering.
- **Model and Prediction:** Utilizes the KNNBasic algorithm to generate predictions based on similarities between parks and historical sightings data. The model is tuned through cross-validation to ensure robustness and accuracy.
- **User Interaction and Response Generation:** Users can query the system via a chat interface to request park recommendations for bird watching. The system uses natural language processing to understand and extract relevant information from user queries.
- **Recommendation Delivery:** The system provides a list of recommended parks, sorted by the likelihood of spotting the queried bird species. The recommendations are presented in a user-friendly format, making it easy for bird watchers to plan their visits.

HEAT MAP

Objective:

A heat map of the bird abundance is generated for the current week based on the predicted count of birds across Singapore.

Technology/Algorithms Used:

- Python Selenium to extract data
- Python with pandas for data preprocessing and manipulation

- scikit-learn for model training and evaluation (Linear Regression, Random Forest Regressor, Gradient Boosting Regressor, Support Vector Regressor, K-Nearest Neighbours Regressor).
- Joblib for model persistence.
- Django REST framework for API development.
- OpenAI ChatGPT Assistant API for conversational interface.
- Plotly for heatmap visualization

Working Mechanism:

- **Chat Interface with Assistant API:**
 - Users send queries (e.g. "I want a heat map") through the chat interface.
 - Django backend interacts with the OpenAI ChatGPT Assistant API to process the query.
 - The backend retrieves the trained model for predicting bird counts.
 - Current week and location data are prepared based on user input or current time.
 - Predictions are generated using the trained model.
 - Plotly creates a heatmap based on predicted counts and location data.
 - The heatmap is returned to the user interface for display.
- **Heatmap Visualization:**
 - Latitude, longitude, and current week data are collected.
 - Distance is calculated from latitude and longitude.
 - Trained model predicts bird populations based on features.
 - Plotly generates a density heatmap using latitude, longitude, and predicted counts.
 - The map is centered around the mean latitude and longitude of the locations.
 - Heatmap is returned to the user interface for display.
- **Model Loading:**
 - Trained model is loaded from the file path.
 - Current week and location data are prepared for prediction.
 - Predictions are generated using the loaded model.

OPERATIONAL FLOW

The integrated workflow of our application encapsulates the intricate collaboration between various technological components to deliver a seamless user experience and provide accurate responses to bird-related queries. User interactions are initiated via an Android application or a Web UI, both of which connect to our Django backend hosted on Heroku Cloud.

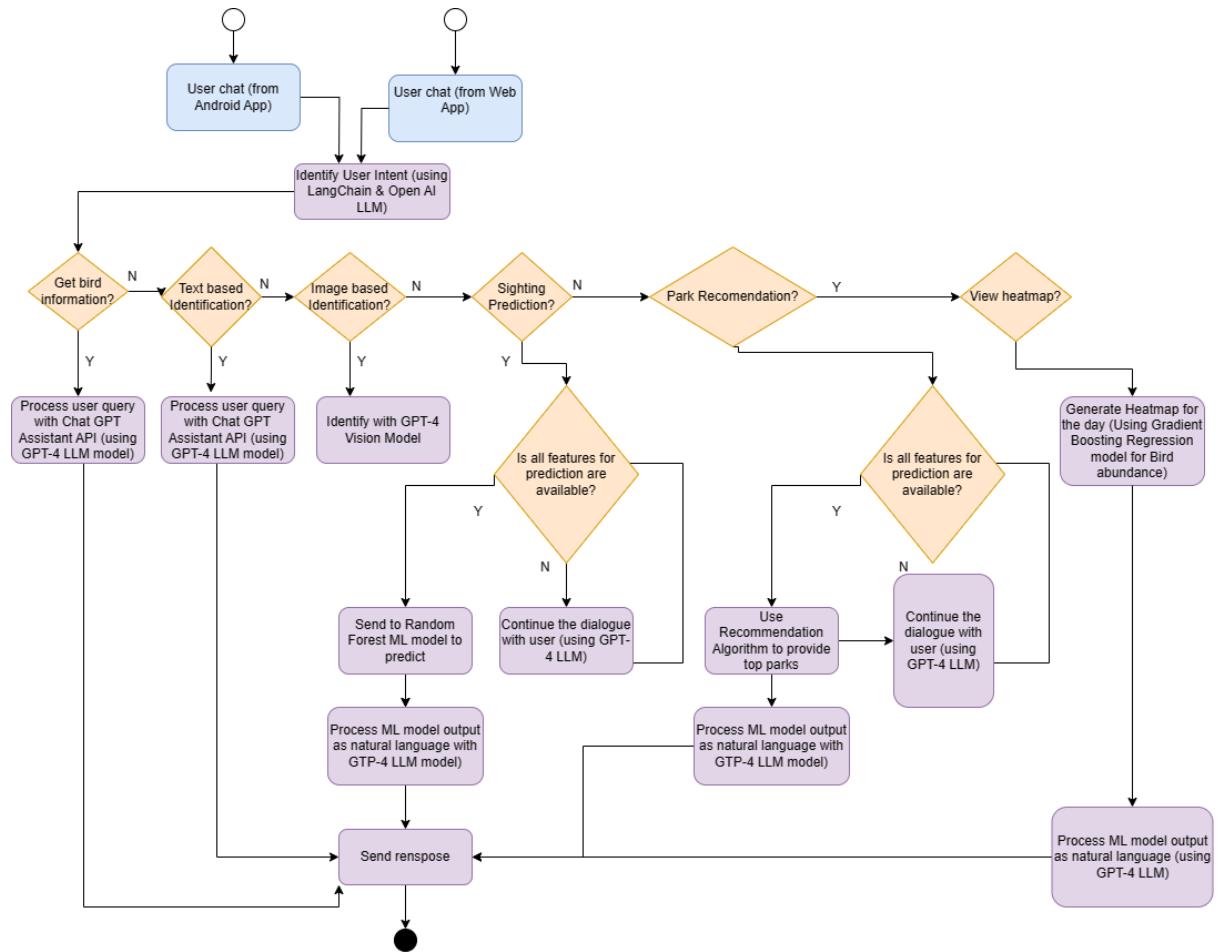


Figure 1 Operational Flow Chart

1. Users send a chat either through an Android or Web App.
 2. User intent is identified using GPT-4 Large Language Model.
 3. If it is a bird information request, the query is processed with a chat API using an OpenAI GPT-4 Language Model.
 4. For text-based identification, the query is processed in a similar fashion.
 5. If image-based identification is needed, the system utilizes GPT-4 with a Vision Model.
 6. For sighting prediction, the workflow checks if all features are available. a. If yes, it proceeds to prediction using a Random Forest ML model. b. If no, the dialogue continues with the user.
 7. For park recommendations, an algorithm provides top park suggestions if all features for prediction are available.
 8. Viewing a heatmap involves generating it using Gradient Boosting Regression Model if all data is available.
 9. The output from the ML models is processed as natural language with GPT-4.
 10. The final response is sent to the user.

SYSTEM ARCHITECTURE

ARCHITECTURAL DIAGRAM

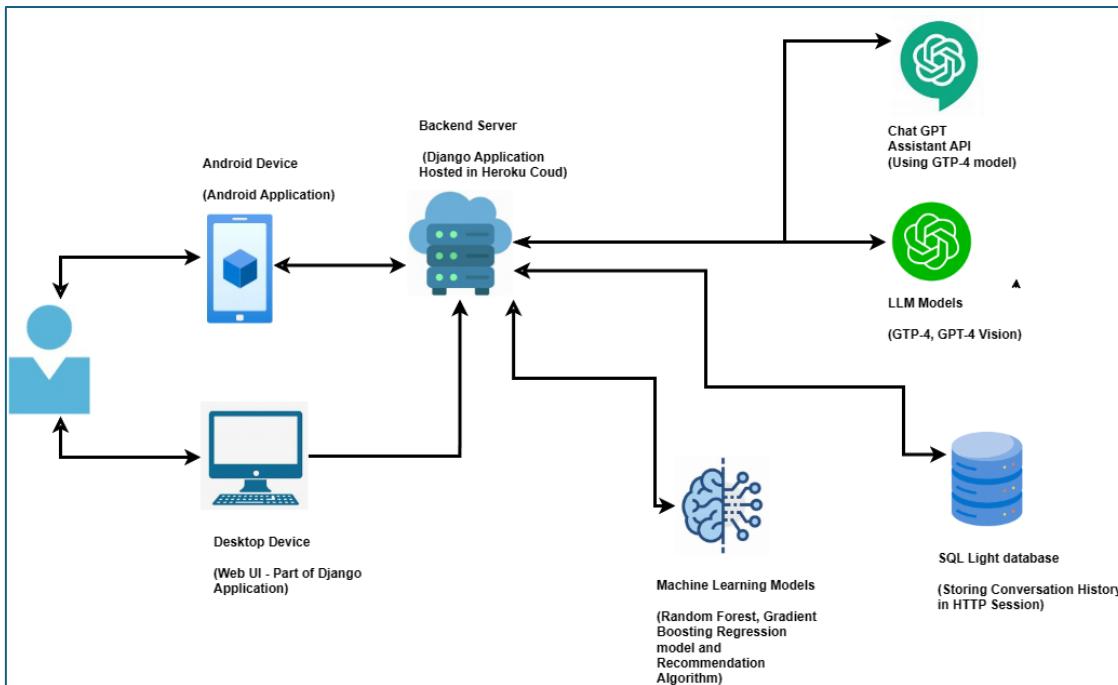


Figure 2 System Architectural Diagram

SYSTEM COMPONENTS

USER INTERFACES

There are both Web and Android mobile User interfaces for the application. Both interfaces are designed to offer a seamless user experience, allowing users to input text or upload images for bird identification and access various bird-related information and services.

- **Android Device**: Users interact with the system through an Android application, which serves as one of the primary interfaces for the service.
- **Desktop Device**: Alternatively, users can engage with the application via a web-based user interface (UI) powered by the Django application, facilitating access from desktop devices.

BACKEND SERVER

- **Django Application Hosted in Heroku Cloud**: The backend logic of the application is handled by a Django application. This Python-based framework is known for its simplicity and flexibility. The entire application is hosted on Heroku Cloud, providing scalable server infrastructure with easy deployment processes.
- **Chat GPT Assistant API**

LLM MODELS

Leveraging the GPT-4 model, the Chat GPT Assistant API processes natural language inputs from users. This component is responsible for understanding user queries and generating appropriate responses.

Two specialized Large Language Models (LLMs) are employed:

- **GPT-4/GPT-4 Turbo:** Used for processing and responding to text-based user queries.
- **GPT-4 Vision:** Specifically utilized for analyzing images uploaded by users to identify bird species.

These models are crucial for enabling intelligent and context-aware interactions, capable of understanding and providing information or performing bird identification based on user inputs.

MACHINE LEARNING MODELS

- **Machine Learning Models:** A suite of machine learning models is incorporated into the system for various predictive and analytical tasks:
- **Random Forest:** This is used for prediction bird sighting based on user inputs.
- **Gradient Boosting Regression Model:** Utilized for generating heatmaps for bird abundance.
- **Recommendation Algorithm:** To provide park recommendations for the birds using interest to watch.
- These models are central to providing advanced services beyond simple information retrieval, such as predictive analytics and personalized recommendations.

DATA STORAGE

- **SQLite Database:** Serving as the application's data repository, the SQLite database stores conversation history within HTTP sessions. This storage solution enables the system to maintain context during interactions, enhancing the user experience by allowing the application to reference previous inquiries and user data.

4. IMPLEMENTATION

APPLICATION FRAMEWORKS AND USER INTERFACES

This section details the software architecture of the application, highlighting the Backend API, Web User Interface, and the Mobile Application for Android.

BACKEND API

The backend of the application is built using Django, a high-level Python web framework that encourages rapid development and clean, pragmatic design. The application is using Django version 5.0.3, which follows the Model-View-Controller (MVC) architectural pattern.

The backend manages several API endpoints, in its Controller class (Views.py file), providing various functionalities:

- **Querying the data with LLM:** Utilizes classes such as BirdInformationBot, BirdIdentificationBot, and related services for each feature to provide information and identify birds based on user queries.
- **LLM based Image Processing:** The ImagelidentificationBot handles image-based bird identification. Images are managed to ensure the server maintains only the latest files, with a limit of five images at a time.
- **Predictive Analytics:** Using BirdPredictionBot and BirdHeatMapBot, the application predicts bird spotting opportunities and generates heat maps for bird activity.
- **Location Services:** LocationRecommendationBot suggests optimal bird-watching locations.

WEB USER INTERFACE

The web interface is designed using HTML templates integrated within the Django framework, enhanced by Bootstrap 5 for a rich user interface and custom CSS for styling. AJAX requests are employed to send user queries to the backend, facilitating a dynamic and responsive user experience without reloading the page.

KEY FEATURES

- **AJAX Integration:** Enhances user experience by making asynchronous requests to the server, allowing for live updates of content without reloading the webpage.
- **Responsive Design:** Bootstrap 5 ensures that the application is accessible and visually appealing across different devices and screen sizes.



Figure 3 Web User Interface

MOBILE APPLICATION (ANDROID)

The mobile application is developed using Kotlin in Android Studio, focusing on providing a seamless mobile user experience. It primarily utilizes WebView to embed the mobile-responsive web interface within the app. This approach ensures that all functionalities available on the web are also accessible on mobile devices.

KEY COMPONENTS

- **WebView:** Allows embedding of web content within the mobile app, ensuring feature parity with the web interface.
- **File Upload Handling:** Integrated support for file uploads, particularly for image-based bird identification, using Android's WebChromeClient and handling file chooser intents.

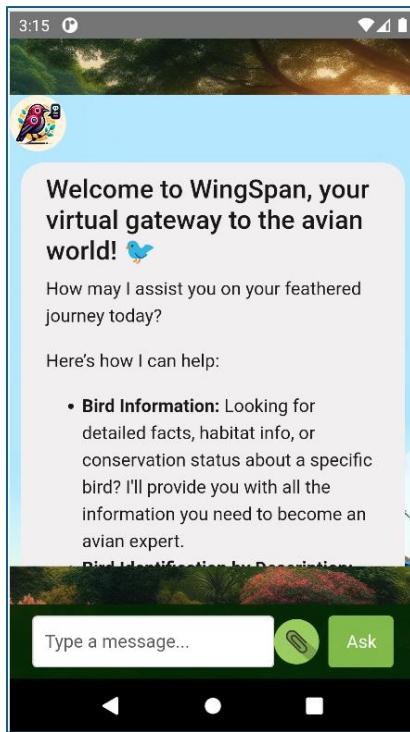


Figure 4 Mobile User Interface (Screenshot from Android Studio Emulator)

Mobile User Interface (Screenshot from Android Studio Emulator)

LARGE LANGUAGE MODELS INTEGRATION

The application leverages OpenAI's large language models (LLMs) to enhance its functionality by providing advanced natural language processing capabilities. This integration allows the application to understand and respond to user queries with high accuracy and contextual relevance.

INTEGRATION DETAILS

- **API Key Configuration:** The integration starts with setting up the OpenAI API key. The key is securely stored in the environment variables of the server where the Django application is hosted. This approach not only protects the key from unauthorized access but also makes it easy to manage across different deployment stages (development, testing, production).
- **Client Initialization:** In the Django application, the OpenAI client is initialized using the API key retrieved from the environment variables. This client acts as the gateway for sending requests to OpenAI's servers and receiving responses. The code snippet below demonstrates the initialization process:

USER INTENT DETECTION TO USE RESPECTIVE ML MODELS.

The User Intent Detection system is a sophisticated component of the application that uses OpenAI's large language model (LLM), specifically GPT-4, to accurately understand and respond to user inquiries. It is a crucial intermediary that connects the user's natural language requests with the appropriate machine learning models for handling specific tasks.

INTENT DETECTION WORKFLOW:

- **User Interaction:** The user interacts with the application via a web or mobile interface. They may pose a question, describe a bird, upload a photo, or ask for recommendations or predictions, as displayed in the UI screenshot.
- **API Endpoint Invocation:** When the user submits their query, an AJAX call is made to the handleRequest API endpoint on the backend, which triggers the intent detection process.
- **Session Management:** A user's conversation history is maintained in the session storage, which helps in understanding the context of the user's current intent.
- **Intent detection and processing logic:** First the system verifies if there is an image attached to the user's request. If an image is present, image-based identification logic is processed. The 'handle_message' method in ImgelidentificationBot processes it for identification.
- If no image is present, the user's input is then sent to the GTP-4 model with any relevant conversation history to identify the intent. The intent is identified based on the message content and the role labels 'system' and 'user', which provide context to the LLM. The identified intent is categorized as one of the following the following:
 - Greeting
 - Bird Information
 - Bird Identification
 - Spotting Prediction
 - Hotspot Map
 - Park Recommendation
 - Unclear Intent

Once the intent is identified, the system decides which specialized ML model to use:

- **Bird Information:** Inquiries about bird facts, habitat, and conservation status are managed by the BirdInformationBot class. Its handle_message method is called upon identifying the intent as 'Bird Information'.
- **Bird Identification by Description:** Requests for bird identification by description are handled by BirdIdentificationBot. The class's handle_message method is activated when the intent is recognized as 'Bird Identification'.
- **Bird Spotting Predictions:** The BirdPredictionBot is responsible for predicting bird spotting opportunities once the intent 'Spotting Prediction' is identified.
- **Bird Abundance Heatmap:** The HotspotMap generates a heatmap showing bird abundance and activity. The intent associated with this function is 'Hotspot Map', and the HotspotMap class's handle_message method is called to produce the heatmap.
- **Park Recommendation:** Users seeking the best parks for birdwatching are supported by LocationRecommendationBot. For requests with the intent 'Park Recommendation', the LocationRecommendationBot class's handle_message method is invoked to provide recommendations.
- **Response Generation:** After processing the user's request through the respective ML model, a response is generated and sent back to the user. The conversion history is added to the HTTP session and saved in the database to keep track of the subsequent request from same user.

MODEL CREATION FOR EACH FEATURES

BIRD INFORMATIONAL QUERIES

DATA COLLECTION

Using Python with the Selenium library, we automated the scraping of bird species information from the "Singapore Birds Project" website. Our script systematically navigated the site, extracting data from individual species pages. To maintain ethical web scraping

practices, we incorporated delays between page requests to minimize server load. Data on available 415 common birds in Singapore were collected.

Northern Shoveler
Scientific Name: *Spatula clypeata*
Malay Name: Itik-Sudu Utara
Chinese Name: 虎嘴鸭
Alternative Name(s): Shoveller, Common Shoveller, Red Shoveller
Range: Found from Europe to North Africa, temperate Asia and North America, wintering to Equatorial Africa, Indian subcontinent, southern China, Korea, Japan, mainland Southeast Asia, Philippines, Central and South America.
Taxonomy: Monotypic.
Size: 43-52 cm
Identification: Male has long, wide, spatula-shaped bill, dark green head and upper neck, white breast, chestnut flanks and black tail-coverts. Female and juvenile can be easily distinguished from other duck species from the distinctive bill shape.
Habitat: Lakes, rivers, marshes and various wetland habitats.
Behaviour/Ecology: Feeds in very shallow water, using its bill to sift through the mud for minute aquatic animals, plants and seeds.
Local Status: Vagrant
Conservation Status: Least Concern (BirdLife International 2019)

Figure 5 Sample of collected Data.

DATA CONVERSION AND STORAGE

The data was then stored systematically for easy retrieval, as an offline resource to be used later by the chat GPT assistant API with 'GPT-4-turbo-preview.'

This processed dataset was then structured in a format that could be efficiently queried by the GPT-4 LLM model during live user interactions, facilitating prompt and accurate data retrieval.

MODEL CREATION

The core of the bird information system is built on the 'GPT-4-turbo-preview' LLM model, enhanced with a fine-tuning process using the prepared bird data. This model creation step involved training the model to understand and generate contextually relevant responses based on user queries. The fine-tuning process utilized the Open AI's Assistant API feature, where the bird data collected was converted to text file and integrated as a searchable attachment.

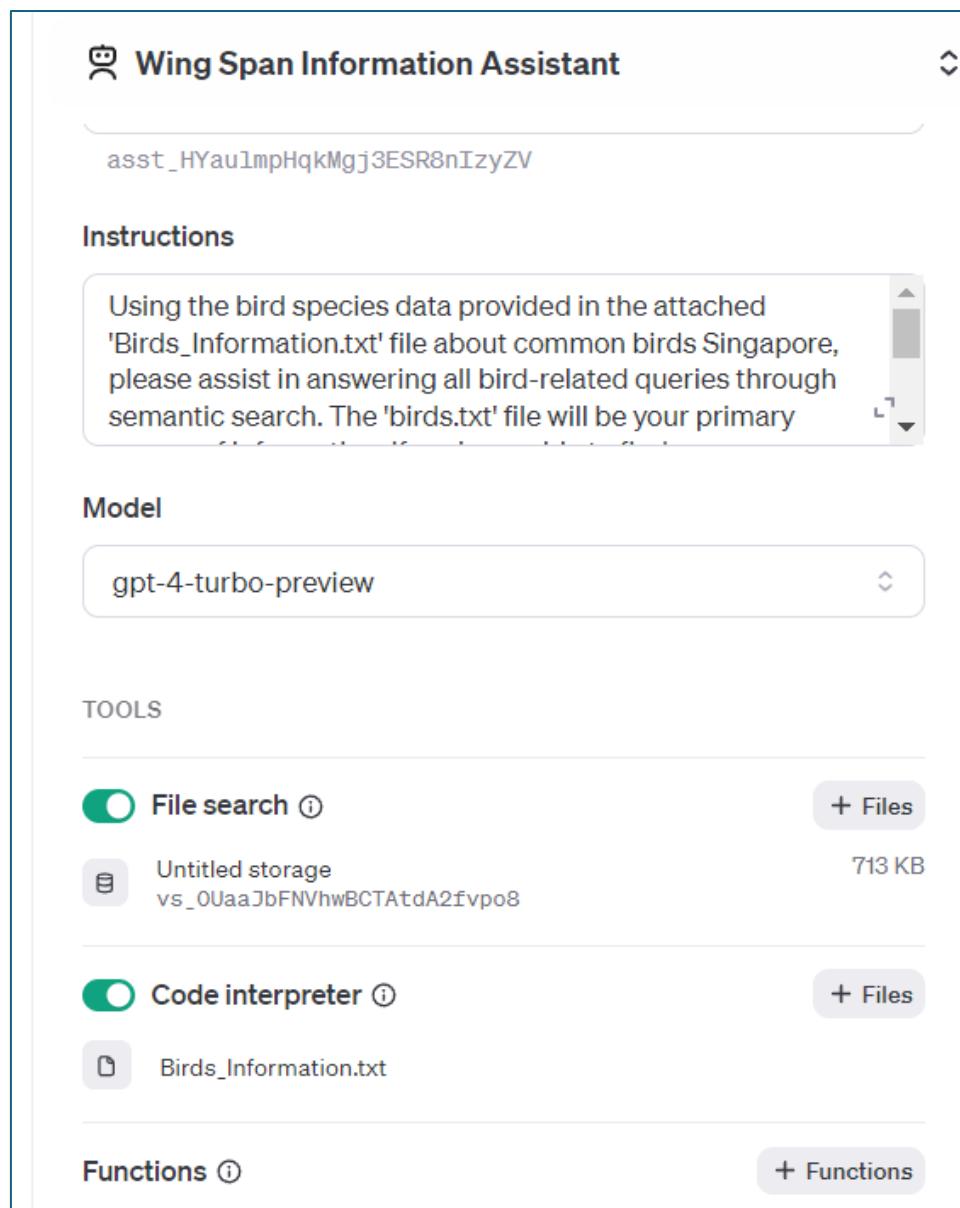


Figure 6 ChatGPT Assistant API for Bird information with attached data

During model creation, along with the Instructional prompt for Assistant API, custom prompts were crafted to do a semantic search in the attached document and fetch the information asked.

Playground Assistants ▾

← Edit instructions

Using the bird species data provided in the attached 'Birds_Information.txt' file about common birds Singapore, please assist in answering all bird-related queries through semantic search. The 'birds.txt' file will be your primary source of information. If you're unable to find an answer within the document, feel free to ask me for more details.

If the user asks for the list of all common birds in Singapore, provide the answer in single response

Figure 7 Figure 6 Instructional prompt for Assistant API

```
Codumate: Options | Test this method
def handle_message(self, request):
    conversation_history = request.session.get("conversation_history", [])
    # Get the last 10 items of the conversation history
    conversation_history_recent = conversation_history[-10:]
    try:
        user_query = request.data.get("query", "")
        extended_prompt = f"message: '{user_query}'. Your goal is to get \
            accurate bird information from the attached 'Birds_Information.txt' \
            data with a semantic search. If my intention isn't clear, \
            please ask for more details on the kind of support I need from \
            your chat service regarding birds. Make the conversation so \
            realistic and give clarity so that I can provide correct \
            information. Do not tell what document use, i.e., the file name \
            'Birds_Information.txt' to the user in your reply. \
            Format your response to sound natural. Take Conversation \
            history:{conversation_history_recent} into consideration to \
            understand what I am saying \
            Don't include source and References from the attachment \
            in your response. Don't mention any technical detail or \
            attachment used as well"
        response = self.talk_to_assistant(extended_prompt)
        return {"response": response}

    except Exception as e:
        print(f"An error occurred: {e}")
        return "An error occurred while processing your request."
```

Figure 8 Extended prompt to query the attached data along with conversational history

MODEL PERFORMANCE

- The model's performance was assessed using Python test scripts designed to submit a series of bird-related informational queries to the model through an API call. (Refer to Application root folder>Test>'bird_information_test.py' for the test script)
- A pre-prepared list of correct answers was used for comparison.
- The model's responses were evaluated against these answers by a manual comparison, achieving an impressive accuracy rate of 95%.

Questions	Answers	Model Answers	Pass/Fail
What is the conservation status of the Long-toed Stint?	Least Concern (BirdLife International 2016).	{"response": "The Long-toed Stint is currently listed as being of Least Concern according to the BirdLife International 2016 assessments."}	Pass
Where can the Red-necked Stint be spotted in Singapore?	Locations include Marina East Drive, Sungei Buloh Wetland Reserve, and Seletar Dam .	{"response": "The Red-necked Stint can be spotted in various locations throughout Singapore. Some of the notable places where this uncommon migratory bird can be seen are Marina East Drive, Sungei Buloh Wetland Reserve, Seletar Dam, Pulau Tekong, Chek Jawa at Pulau Ubin, and Tuas South."}	Pass
What is the identifying feature of the Spoon-billed Sandpiper?	It has a spatulate bill .	{"response": "The identifying feature of the Spoon-billed Sandpiper is its distinctive spatulate bill. This unique trait sets it apart from other small waders such as the Red-necked Stint and Little Stint with which it shares similarities in size and appearance."}	Pass
What is the local status of the Arctic Warbler in Singapore?	It is a very common migrant.	{"response": "The Arctic Warbler is considered a very common migrant in Singapore. It is known for its yellowish-white supercilium, olive-green crown, and upperparts with subtle wing bars. You might find them in a variety of habitats, including forests, wooded areas, and urban parks and gardens. The conservation status of the Arctic Warbler, based on BirdLife International 2016, is listed as 'Least Concern,' highlighting its stable population at a global scale."}	Pass

Figure 9 Samples from the output CSV file.

TEXT BASED BIRD IDENTIFICATION

DATA COLLECTION

We utilized the same bird informational data that was discussed in the Bird Informational Queries section. This dataset includes size, identification information, behavior, and other details for 415 common birds found in Singapore.

DATA CONVERSION AND STORAGE

The data was stored in Attachment E to be later used by the ChatGPT Assistant API with the 'GPT-4-turbo-preview' model.

MODEL CREATION

The text-based bird identification model employs the 'GPT-4-turbo-preview' LLM, enhanced through a fine-tuning process with the prepared bird data. This step involved training the model to understand and generate contextually relevant responses based on user queries. We utilized OpenAI's Assistant API feature, where the collected bird data was converted into a text file and integrated as a searchable attachment. Additionally, a carefully crafted instructional prompt was provided to the assistant API.

Drawing from the attached text data on bird species of Singapore, which encompasses scientific and common names in multiple languages, geographical ranges, taxonomy (including subspecies), physical dimensions, distinguishing characteristics, comparison with similar species, habitats, behaviors, ecological roles, local stats and conservation Status please address the inquiry below:

A user is in pursuit of identifying a bird or acquiring exhaustive details about a particular species. They may offer insights regarding the bird's physical traits, size, preferred environments, or behavioral patterns. Employ the extensive knowledge you've been endowed with to achieve the following:

Identify the Bird: Ascertain the bird species from the user's description, paying attention to unique identifiers or resemblances to other species.

Detail the Range: Provide comprehensive info on the bird's habitat range, encompassing breeding and nonbreeding locales, alongside notable subspecies distinctions.

Describe Physical Features: Elaborate on the bird's visual attributes, emphasizing key markers that aid in differentiation from closely related species.

Discuss Ecology and Behavior: Relay information on preferred habitats, typical activities, and the bird's role within its ecosystem.

Status and Conservation: Note the species' local prevalence (e.g., common resident, rare visitor). Aim for responses that are enlightening, accessible, and precise, designed to either educate the user about the bird in focus or assist in pinpointing the observed bird. Should the provided details be insufficient or applicable to several species, initiate a comparative analysis to refine the identification.

Moreover, if the initial description falls short for a definitive identification, continue the dialogue by asking more questions. After four questions if a definite answer is not found, propose potential matches based on the available data, encouraging an interactive discovery process. Your response should not be more than 200 characters while trying to identify birds.

Figure 10 Figure 9 Instructional prompt for Assistant API

For user queries about bird identification, an extended prompt with conversation history was also employed to fine-tune the identification process.

```

def handle_message(self, request):
    conversation_history = request.session.get("conversation_history", [])
    # Get the last 10 items of the conversation history
    conversation_history_recent = conversation_history[-10:]

    try:
        user_query = request.data.get("query", "")
        # From General predictionn chat to Identifying the prediction mode type

        extended_prompt = f"Based on the message: '{user_query}', your goal is to\
        accurately identify a bird using a semantic search of the \
        'Birds_Information.txt' data provided. If it seems like I'm\
        looking for general bird information instead, please clarify\
        what specific details or support I need regarding bird \
        identification. Keep our conversation realistic and clear, \
        so I can provide the correct information. Remember, avoid \
        mentioning the document name directly in your response. \
        Let's make our exchange as natural as possible\
        take Conversation history:{conversation_history_recent} into\
        consideration to understand what i am saying\
        Don't include source and References from the attachment \
        in your response. Don't mention any technical detail or\
        attachment used as well"
        response = self.talk_to_assistant(extended_prompt)
        return {"response": response}

    except Exception as e:
        print(f"An error occurred: {e}")
        return "An error occurred while processing your request."

```

Figure 11 Extended prompt for the attached data along with conversational history.

MODEL PERFORMANCE

- The model's performance was assessed using Python test scripts designed to submit a series of bird-related identification queries through an API call. (Refer to Application root folder>Test>'bird_identification_Text_Test.py' for the test script).
- A pre-prepared list of correct answers was used for comparison.
- The model's responses were evaluated against these answers through manual comparison, achieving an good accuracy rate of 80%.

Questions	Answer	Model Answers	Pass/Fail
I spotted a bird that is smaller than similar species, with broad yellowish-white supercilium and two broad yellowish-white wing bars	Arctic Warbler	{"response": "The bird you described sounds like the Yellow-browed Warbler. It is notably smaller with broad yellowish-white supercilium and two broad yellowish-white wing bars, distinguishing it from similar species like the Arctic Warbler and Eastern Crowned Warbler. This bird inhabits forests, forest edges, wooded areas, parks, and gardens and is an uncommon migrant."}	Pass
I saw a bird with dark to greyish-brown upperparts without any pale wing bars and buffish wash on breast-sides	Dusky Warbler	{"response": "The bird you described matches the Dusky Warbler. Found from eastern Eurasia to Siberia, it prefers open, often wetter areas. It has dark to greyish-brown upperparts, buffish wash on breast-sides, and lacks pale wing bars."}	Pass
I noticed a bird that resembles Arctic Warbler but has yellow-green upperparts and a pale median coronal stripe	Eastern Crowned Warbler	{"response": "Based on your description, the bird you've observed is likely the Eastern Crowned Warbler. It's distinguished by the yellow-green upperparts and a pale median coronal stripe, which closely matches the details you've provided. This warbler frequents forests, mangroves, gardens, parks, and wooded areas. It's a common migrant noted for foraging in the canopy. In Singapore, it can be seen in locations such as Central Catchment Forest and Sungai Buloh."}	Pass
I observed a bird resembling Arctic Warbler but with dark greyish crown and warmish olive-brown rump	Sakhalin Leaf Warbler	{"response": "The bird you observed, resembling the Arctic Warbler but with a dark greyish crown and warmish olive-brown rump, is likely the Pale-legged Leaf Warbler. This species, similar in appearance to the Arctic Warbler, is differentiated by its unique crown and rump colors, and its unique song or call frequency."}	Pass

Figure 12 Sample from the output CSV file

IMAGE BASED BIRD IDENTIFICATION

DATA COLLECTION

We downloaded common bird names from '<https://records.singaporebirds.com/checklist/>' and compiled them into text files. These files are used alongside user-submitted images to assist in bird identification communications.

Ducks, Geese, Swans (Anatidae)
1 Lesser Whistling Duck Dendrocygna javanica
2 Cotton Pygmy Goose Nettapus coromandelianus
3 Garganey Spatula querquedula
4 Northern Shoveler Spatula clypeata
5 Gadwall Mareca strepera
6 Eurasian Wigeon Mareca penelope
7 Northern Pintail Anas acuta
8 Tufted Duck Aythya fuligula
Pheasants & Allies (Phasianidae)
9 Red Junglefowl Gallus gallus
10 King Quail Syoenicus chinensis

Figure 13 Sample of data in the text file.

DATA CONVERSION AND STORAGE

The bird's name list was stored in the Application root folder under "Data" > "Singapore_Bird_Name_List.txt".

MODEL CREATION

For the image-based bird identification, we implemented an interactive bot using Python and the Django framework. The bot employs OpenAI's **gpt-4-vision-preview** model to interpret and respond to user-submitted bird images. Here's a step-by-step breakdown of the model's operation:

Image and Data Handling: When a user submits an image and a query for bird identification:

- The image path is combined with the server settings to access the actual file.
- A list of bird names is read from a text file to assist in generating contextually relevant responses.
- The image is resized to 200x200 pixels, converted to JPEG format, and encoded in base64 for processing.

Query Processing: The bot constructs a detailed prompt that includes the user's text query, the encoded image, the list of common bird names, and conversation history to the **gpt-4-vision** model.

```
def handle_message(self, request, image_path):
    # Retrieve the full conversation history from the session
    conversation_history = request.session.get("conversation_history", [])
    # Get the last 10 items of the conversation history
    conversation_history_recent = conversation_history[-10:]
    try:
        user_query = request.data.get("query", "")

        extended_prompt = f"Based on the message: '{user_query}', and the attached\\
            image determine if I'm trying to identify a specific bird.\\
            My goal is to accurately identify a bird using image provided.\\
            Please look into the attached 'Singapore_Bird_Name_List.txt' \\
            to look for the possible birds Remember, avoid mentioning the\\
            document name directly in your response. Let's make the response\\
            as natural as possible\\
            take Conversation history:{conversation_history_recent} into \\
            consideration to understand what i am saying "
        response = self.identify_bird(extended_prompt, image_path)
        return {"response": response}

    except Exception as e:
        print(f"An error occurred: {e}")
        return "An error occurred while processing your request."
```

Figure 14 Prompt for image-based bird identification.

```
messages=[  
    {  
        "role": "user",  
        "content": [  
            {  
                "type": "text",  
                "text": f"{user_query}. Give me the possible answers if you do not have a definite answer.\nHere are possible birds: {birds_list}. Do not mention the documents/record used or the locations in it in the response",  
            },  
            {  
                "type": "image_url",  
                "image_url": {  
                    "url": f"data:image/jpeg;base64,{base64_image}"  
                },  
            },  
        ],  
    }]
```

Figure 15 API call with prompt, bird list, and image.

Response Generation: The model processes the inputs and returns a response, which is either a direct identification of the bird or a list of possible birds based on the query and image content.

Conversation Context Management: The bot manages and utilizes conversation history to better understand and respond to ongoing user interactions, ensuring responses are as accurate and relevant as possible.

MODEL PERFORMANCE

We assessed the model's performance using Python test scripts that submit various bird images to the model through an API call. (Refer to Application root folder>Test> 'bird_identification_image_test.py' for the test script).

- A pre-prepared list of correct answers was used for comparison.
- The model's responses were evaluated against these answers through manual comparison, achieving a good accuracy rate of 82%.
- This approach ensures that the image-based bird identification system is robust and efficient, leveraging advanced AI capabilities for accurate bird identification.

Actual Bird	Identified Bird	Result
1 Indian cuckoo 1.jpg	{"response":"The bird in the image is a cuckoo, specifically the Asian Koel (Eudynamys scolopaceus). This is evident from the distinctive plumage and overall shape typical of this species. The Asian Koel is known for its dark plumage and prominent white spots and streaks."}	Pass
4 ruddy breasted crake 2.jpeg	{"response":"The bird in the image you've provided appears to be a Ruddy-breasted Crake (Zapornia fusca), which is characterized by its warm brown plumage on the upperparts, paler ruddy on the breast, and bright red legs and eyes. This species is among the list you referenced, under the family of Rails, Crakes & Coots (Rallidae). It's a bird commonly found in wetland areas across parts of Asia."}	Pass
5 ruddy breasted crake.jpg	{"response":"The bird in the image is a Ruddy-breasted Crake (Zapornia fusca). Its identifying features are the reddish-brown plumage on the breast and upperparts, the grey face and throat, and the bright red legs. This species is consistent with one of the birds from the provided list of birds."}	Pass
Asian koel.webp	{"response":"The bird in the image is an Asian Koel (Eudynamys scolopaceus), which is characterized by its glossy black plumage and distinctive red eye. It is a member of the cuckoo family and is known for its loud, distinctive calls. This bird can be found in many parts of Asia and is often heard during the breeding season."}	Pass

Figure 16 Sample from the output CSV file

BIRD SIGHTING PREDICTION

DATA PREPARATION

Overview

Efficient data preparation is pivotal in constructing a reliable predictive model for bird sighting predictions. This section outlines the systematic process of preparing data to be fit for Random Forest modeling.

Data Collection

Initial data collection involved gathering extensive records of bird sightings, which included date, time, environmental factors, and species observed. This data is sourced from Singapore bird databases, and from the Singapore Weather data records and from public submission platforms like eBird.

Data Cleaning and Preprocessing

Given the nature of observational data, it typically requires rigorous cleaning^[1]. This includes:

- Removing duplicates: Ensuring unique entries for analysis.
- Handling missing values: Imputing or discarding missing data based on the extent and nature of the gaps.
- Data transformation: Converting timestamps into usable date-time formats, categorizing continuous variables where necessary, and encoding categorical variables, such as bird species and locations.

Feature Engineering

Feature engineering^[1] involves creating new variables from existing data to enhance the model's predictive power:

- Time features: Extracting day of the week, day of the year and month from date records, as these can influence bird visibility.
- And eliminating certain environmental weather data which doesn't contribute much to the model performance.

Data Splitting

The dataset is then divided into training and test sets. This allows the model to be trained on one subset of the data and validated on another to evaluate its performance and generalize ability. 70% of the data has been assigned for training sets and 30% of data has been assigned to test sets.

	A Species	B Location	C Observation_Date	D Year	E Month	F Day	G tempmax	temp
2	Black Hornbill	Pulau Ubin	2018-10-05	2018	10	5	33.9	27.6
3	Little Grebe	Lorong Halus Wetland	2018-10-05	2018	10	5	33.9	27.6
4	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-05	2018	10	5	33.9	27.6
5	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-06	2018	10	6	32.9	26.6
6	Black Hornbill	Pulau Ubin	2018-10-06	2018	10	6	32.9	26.6
7	Little Grebe	Lorong Halus Wetland	2018-10-06	2018	10	6	32.9	26.6
8	Little Grebe	Lorong Halus Wetland	2018-10-07	2018	10	7	30.9	25.9
9	Black Hornbill	Pulau Ubin	2018-10-07	2018	10	7	30.9	25.9
10	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-07	2018	10	7	30.9	25.9
11	Black Hornbill	Pulau Ubin	2018-10-08	2018	10	8	32.2	25.7
12	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-08	2018	10	8	32.2	25.7
13	Little Grebe	Lorong Halus Wetland	2018-10-08	2018	10	8	32.2	25.7
14	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-09	2018	10	9	29.6	24.1
15	Black Hornbill	Pulau Ubin	2018-10-09	2018	10	9	29.6	24.1
16	Little Grebe	Lorong Halus Wetland	2018-10-09	2018	10	9	29.6	24.1
17	Barred Eagle-Owl	Dairy Farm Nature Park	2018-10-10	2018	10	10	31.4	24.1
18	Black Hornbill	Pulau Ubin	2018-10-10	2018	10	10	31.4	24.1
19	Chinese Egret	Pulau Tekong	2018-10-10	2018	10	10	31.4	24.1
20	Little Grebe	Lorong Halus Wetland	2018-10-10	2018	10	10	31.4	24.1
21	Black Hornbill	Pulau Ubin	2018-10-11	2018	10	11	32.3	25

Figure 17 Sample snap from the dataset prepared for prediction

MODEL CREATION

Overview

After evaluating various classification models, the Random Forest classifier was identified as the most effective for bird sighting predictions.

Comparison with Other Models

- **Logistic Regression and SVM:** Struggle with non-linear data and require normalization.
- **Neural Networks:** Need extensive preprocessing and are prone to overfitting without large datasets.
- **k-Nearest Neighbors (k-NN):** Highly sensitive to data scale and irrelevant features.

Advantages of Random Forest

Random Forest excels in handling complex ecological datasets due to its robustness against overfitting and its ability to manage high-dimensional data^[2]. It also offers clear insights into feature importance. More detailed analysis of model training and evaluation has been discussed in the below sections.

Model Theoretical Framework

The Random Forest model integrates multiple decision trees to improve prediction accuracy and control overfitting. It builds on the concept of ensemble learning, where multiple models (trees) are trained, and their predictions aggregated.

Building Decision Trees

Each tree in a Random Forest works on a bootstrap sample drawn from the original data. For each node:

- A subset of features is randomly selected to decide the best split based on criteria like Gini Impurity or Entropy.
- Gini Impurity for a set is calculated as ^[3]

$$G = 1 - \sum_{i=1}^n p_i^2$$

where p_i , represents the probability of an instance being classified to a specific class.

- Entropy, which measures the purity of a split, is computed as ^[4]

$$H = - \sum_{i=1}^n p_i \log_2(p_i)$$

Aggregation and Prediction Making

In classification tasks, the prediction from the forest is the mode of the predictions made by individual trees.

Classification:

$$\hat{y} = \text{mode}(\{h_1(X), h_2(X), \dots, h_N(X)\})$$

FEATURE IMPORTANCE AND MODEL TUNING

Feature importance is assessed to understand the impact of different variables on prediction accuracy. The importance of a feature is indicated by how much the tree nodes that use that feature reduce impurity on average across all trees in the forest. Model hyperparameters like the number of trees, the number of features to consider at each split, and the maximum depth of trees are tuned to optimize the performance.

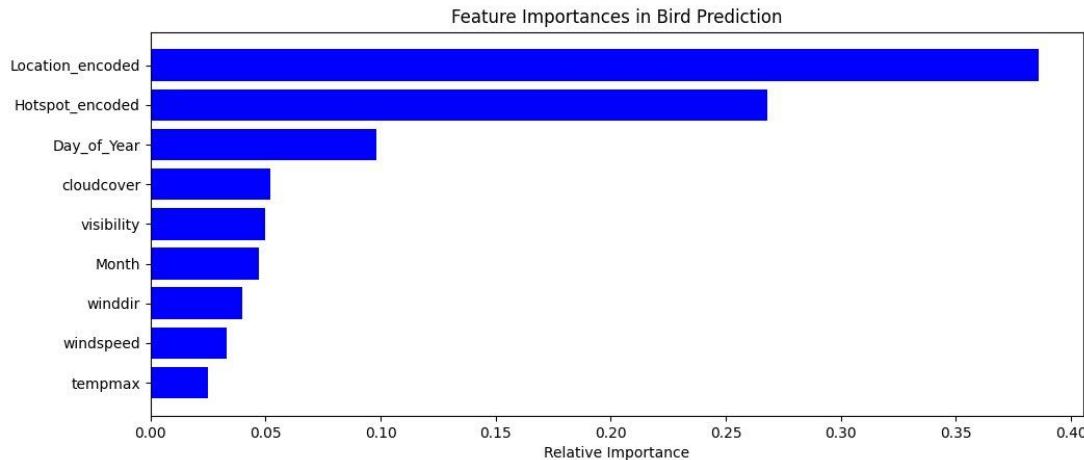


Figure 18 The Feature importance in the Bird prediction model

```
ashwin@ashwin-LOQ-16IRH8: ~/test_rf
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_estimators=300; total time= 4.2s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_estimators=300; total time= 4.1s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_estimators=100; total time= 1.3s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=100; total time= 1.3s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=100; total time= 1.3s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=200; total time= 2.7s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=200; total time= 2.6s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=200; total time= 2.5s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=300; total time= 3.6s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=300; total time= 3.7s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=15, n_estimators=300; total time= 3.9s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=100; total time= 1.3s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=100; total time= 1.3s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=100; total time= 1.1s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=200; total time= 2.1s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=200; total time= 2.1s
CV] END max_depth=30, max_features=sqrt, min_samples_leaf=2, min_samples_split=20, n_estimators=200; total time= 2.1s
```

Figure 19 Hyperparameter tuning to optimize settings for enhanced performance

VALIDATION AND PERFORMANCE METRICS

The model's performance is evaluated using the test set and metrics such as accuracy, precision, recall, and F1-score for classification tasks. The overall **training accuracy** is measured to be around **90%** and the overall **testing accuracy** is measured to be around **77%**.

	precision	recall	f1-score	support
Amur Falcon	0.75	1.00	0.86	3
Amur Stonechat	0.03	0.02	0.02	51
Ashy Drongo	0.91	0.93	0.92	90
Ashy-headed Green Pigeon	1.00	1.00	1.00	1
Asian Emerald Cuckoo	0.83	0.83	0.83	12
Asian Openbill	0.92	0.95	0.93	243
Barn Swallow	0.50	0.33	0.40	3
Barred Eagle-Owl	0.94	0.97	0.96	277
Bat Hawk	0.00	0.00	0.00	1
Besra	0.00	0.00	0.00	3
Besra/Shikra	0.00	0.00	0.00	1
Black Drongo	0.70	0.70	0.70	50

Figure 20 The Precision, Recall, F1 score and Support values for the sample test data

accuracy			0.77	4176
macro avg	0.49	0.47	0.47	4176
weighted avg	0.75	0.77	0.75	4176

Figure 21 Overall accuracy observed in test data

CONCLUSION

In deploying the Random Forest model for bird sighting prediction, the combined strategy of robust data preparation and sophisticated ensemble modeling ensures a balance between complexity and performance, yielding better accuracy and insightful predictions.

PARK RECOMMENDATION

OVERVIEW

The park recommendation is another key feature we have proposed for our project. The goal of this system is to suggest parks where a particular bird, identified by its name, has been observed or is likely to be seen. This enhances the user's birdwatching experience.

In developing the park recommendation system, we employed a novel approach by adapting the user-based collaborative filtering (CF) methodology commonly used in recommendation systems to the context of bird watching. This section of the report outlines the creation of the model, which predicts the likelihood of bird sightings in various parks. The data for this recommendation algorithm was prepared with above mentioned methodology and saved as csv files for training and testing.

Advantages and Disadvantages of Novel CF Methodology vs. Traditional Algorithms

Advantages	Disadvantages
Novel CF vs. Item-Based CF: The novel approach may provide more personalized recommendations because it considers the specific preferences of similar 'users' (bird species in this case), rather than relying on the general popularity of 'items' (parks).	Novel CF vs. Item-Based CF: The novel CF method suffers more significantly from the cold start problem, as it requires sufficient data on new bird species to provide accurate recommendations.
Novel CF vs. Apriori Algorithm: Unlike Apriori, which is designed for uncovering frequent itemsets and association rules, the novel CF method directly focuses on the user's interests, potentially leading to more tailored suggestions.	Novel CF vs. Apriori Algorithm: The Apriori algorithm can be more scalable for transactional datasets but may not be as effective for rating-based recommendation systems.
Novel CF vs. Classification Algorithms: Classification algorithms predict categorical class labels, whereas the novel CF method can recommend a list of parks,	Novel CF vs. Classification Algorithms: Classification algorithms can often make predictions with limited

providing a ranked set of options rather than a single classification.

data, whereas CF needs a larger dataset to find neighbors effectively.

Table 1 Advantages and Disadvantages of Novel CF Methodology vs. Traditional Algorithms

Finally, the novel (user-based) collaborative filtering model was chosen for its ability to deliver personalized and dynamically updated recommendations, ideal for suggesting the top 5 parks based on a user-specified bird. Leveraging extensive historical sighting data, the model effectively mimics individual bird preferences, ensuring recommendations are reflective of the latest and most accurate patterns. This approach is well-suited for our application, providing birdwatchers with tailored park suggestions that enhance their experience and align closely with the specific intent of observing a chosen bird species in its natural habitat.

DATA COLLECTION

For the park recommendation, we collected historical data of bird sighting across different parks in Singapore from ‘Singapore Bird Database’ website. The original data contains records of bird sightings, including the species, date of sighting, and location. This data is stored in a CSV file named Park_Recommendation_Data.csv.

Species	Date	Location
Amur Falcon	26-Nov-17	Changi Coast
Amur Falcon	16-Dec-16	Seletar Dam
Amur Stonechat	11-Apr-21	Raffles Marina
Ashy Drongo	18-Nov-18	Kranji Marshes
Ashy Drongo	18-Mar-23	Mount Faber Park
Ashy Drongo	2-Aug-23	Nanyang Technological University
Ashy Drongo	8-Dec-16	Old Bidadari cemetery
Ashy Drongo	12-May-23	Pulau Ubin
Ashy Drongo	6-Jun-18	Punggol Promenade Nature Walk
Ashy Drongo	6-Mar-23	Singapore Botanic Gardens
Ashy Drongo	24-Mar-19	Telok Blangah Hill Park
Asian Dowitcher	28-Aug-22	Marina East Drive
Asian Dowitcher	16-Sep-23	New reclamation land at P. Tekong
Asian Dowitcher	1-Sep-16	Pulau Tekong
Asian Dowitcher	9-Sep-17	Pulau Tekong
Asian Dowitcher	3-Oct-18	Pulau Tekong
Asian Dowitcher	3-Sep-23	Sungei Buloh Wetland Reserve

Figure 22 Screenshot of collected data for the recommendation system.

DATA PREPARATION

In the data preparation phase, bird sightings are grouped by species and park, with a count calculated to gauge sighting frequencies across locations. These counts are then normalized to a 1 to 10 range using MinMaxScaler from sklearn.preprocessing, ensuring uniform data for the collaborative filtering model and mitigating bias due to count variability.

given an original count x , the scaled count x_scaled is computed as ^[5]

$$X_{\text{std}} = (X - X.\min(\text{axis}=0)) / (X.\max(\text{axis}=0) - X.\min(\text{axis}=0))$$

$$X_{\text{scaled}} = X_{\text{std}} * (\text{max} - \text{min}) + \text{min}$$

where min, max = feature range.

Following normalization, counts are rounded up to integers, creating a simpler rating system for the model's interpretation. The original date of sightings is reintegrated into the transformed data for context, while the Data Frame undergoes cleaning to rename columns for clarity and remove duplicates, thus retaining only unique bird-park combinations which are pivotal for precise recommendations. The final step involves exporting this curated dataset to a new CSV file, final_count_dataframe.csv, ready to be ingested by the collaborative filtering algorithm.

bird	park	count
Common Tern	Singapore Straits	10
White-throated N	Henderson Waves	10
White-throated N	Bukit Timah Nature	9
Cinnamon-heade	Pulau Ubin	7
Black-winged Flyc	Pulau Ubin	6
Common Kestrel	Tuas South	6
Northern Booboo	Tuas South	6
Parasitic Jaeger	Singapore Straits	6
Short-tailed Shear	Singapore Straits	6
Booted Eagle	Henderson Waves	5
Pied Harrier	Henderson Waves	5
Pied Harrier	Mount Faber Park	5
Silver-backed Nee	Bukit Timah Nature	5
Barn Swallow	Turut Track	4
Black-and-red Br	Pulau Ubin	4
Blue Rock Thrush	Pinnacle@Duxton	4

Figure 23 Transformed data.

MODEL CREATION

As mentioned above we adopted innovative adaptation of the User-Based Collaborative Filtering (CF) algorithm, utilizing bird species as 'users' and parks as 'items'. The idea is to predict which parks a given bird is likely to be observed in based on past sightings, leveraging the similarities in sighting patterns across different birds.

- **Training and Test Split:** We partitioned the dataset into a training set, which accounted for 90% of the data, and a test set comprising the remaining 10%. This division allowed us to train the model on a substantial portion of the data while reserving a segment for evaluating the model's predictive performance.
- **Model Training with KNNBasic Algorithm:** Using the KNNBasic algorithm from the Surprise library, we trained the model based on the premise of collaborative filtering. We configured the algorithm to identify the 40 nearest neighbors using cosine similarity, thereby leveraging the parallel between birds with similar sighting patterns.

The mechanism behind the user-based collaborative filtering (CF) model implemented here involves several key concepts from statistics and machine learning:

- **Nearest Neighbors Algorithm:** The model uses the K-Nearest Neighbors (KNN) algorithm. It operates on the principle of finding the 'k' closest users (in this case, bird species) to the target user (target bird species), based on their 'ratings' (sighting counts) across a set of items (parks). Mathematically, for a given bird species b , the algorithm searches for k , other bird species that have the most similar sighting patterns across all parks. The similarity is measured using a distance metric, which in this case is the cosine similarity.
- **Cosine Similarity:** Cosine similarity measures the cosine of the angle between two vectors in a multidimensional space. In this context, each vector represents a bird species' sighting pattern across various parks. Mathematically, for two bird species b_1 and b_2 , with sighting patterns represented by vectors v^1 and v^2 , cosine similarity is calculated as:

$$\text{similarity } (v^1, v^2) = (v^1 \cdot v^2) / (\|v^1\| \|v^2\|)$$

PREDICTIONS AND RATING ESTIMATION:

The algorithm estimates the 'rating' (predicted sighting count) for a target park by aggregating the ratings of the k nearest bird species to that park.

The predicted count for bird b at park p is calculated as a weighted sum of the ratings of the k nearest birds, with weights being the similarity measures.

MODEL PERFORMANCE

The model uses RMSE (Root Mean Square Error) and MAE (Mean Absolute Error) as evaluation metrics.

Evaluating RMSE, MAE of algorithm KNNBasic on 5 split(s).							
	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	1.5062	0.7897	1.2937	0.8994	1.2153	1.1409	0.2623
MAE (testset)	0.7801	0.5684	0.7445	0.5819	0.7132	0.6776	0.0864
Fit time	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Test time	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 24 CF model performance matrix.

In the performance evaluation of our collaborative filtering model, we utilized RMSE and MAE metrics across a 5-fold cross-validation to ensure a robust assessment of the algorithm's accuracy. RMSE values ranged from 0.7897 to 1.5602, with a mean of 1.1409, indicating that while our model predicts with reasonable accuracy, there are variations attributable to the specific characteristics of each data fold. The standard deviation of RMSE, 0.2623, signifies a moderate fluctuation in error magnitudes across the folds, which could be a focal point for model improvement.

MAE, a measure less sensitive to large errors than RMSE, presented lower scores with a mean of 0.6776 and a standard deviation of 0.0864. The consistency in MAE suggests that the model's average error does not drastically vary, supporting the model's reliability in predicting the count of bird sightings. Notably, the zero values reported for 'Fit time' and 'Test time' imply a high computational efficiency, although these figures should be approached with caution as they may not accurately reflect the actual computational time due to measurement resolution limitations.

EVALUATION OF OTHER RECOMMENDATION MODELS FOR COMPARISON

We initially considered a content-based recommendation system for birdwatching experiences for our project case. This model analyzes the features of items, such as the types of birds and their locations, against individual user preferences. Harnessing Python libraries such as pandas for data manipulation, scikit-learn for data preprocessing and similarity calculations, and the MultiLabelBinarizer for converting species data into a binary matrix. Where the user keys in the bird type it shows the top recommended parks for the birds. The model utilizes this binary matrix to calculate the Jaccard similarity, which serves as the crux of our park recommendation logic. The Jaccard similarity for a target bird species A against a park B is computed as follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

```
Current directory: C:\Users\User\Desktop\IS06\WingSpan\ML_Park_Recommendation
Data path: C:\Users\User\Desktop\IS06\WingSpan\ML_Park_Recommendation\Data\final_count_dataframe.csv
C:\Users\User\.conda\envs\wingspan\Lib\site-packages\sklearn\metrics\pairwise.py:2317: DataConversionWarning: Data
was converted to boolean for metric jaccard
    warnings.warn(msg, DataConversionWarning)
Top 3 recommended parks for Black Hornbill: ['Pulau Ubin', 'Bidadari Park', 'Rifle Range Link']
Precision: 0.3333, Recall: 1.0000, F1 Score: 0.5000

(wingspan) C:\Users\User\Desktop\IS06\WingSpan\ML_Park_Recommendation>
```

Figure 25 content-based recommendation system results.

Recommendations against the actual parks where the bird species were sighted:

These equations enabled us to measure the model's precision at 0.3333, indicating that one-third of the recommended parks were indeed relevant for the "Black Hornbill". The recall was 1.0000, showing that all pertinent parks were recommended, while the F1 score balanced the two at 0.5000.

A primary limitation we faced was the model's inability to effectively recommend parks for uniquely sighted bird species. The Jaccard similarity's reliance on common attributes meant that singular sightings received lower similarity scores, making them less likely to be recommended. Thus, we did not proceed with the recommendation system and pivoted it to the Collaborative filtering.

HEATMAP

In this section, we detail the data preparation and model creation process for generating the bird heat map. Our objective is to predict bird sightings based on historical data and geographical coordinates, enabling users to identify areas with a high probability of bird presence.

DATA PREPARATION

The dataset used for training the heat map model comprises historical bird sightings records. Each record includes information such as bird species, latitude, longitude, and the count of bird sightings.

A crucial step in data preparation is feature engineering, where we calculate the Euclidean distance based on latitude and longitude. This distance metric serves as a feature in our model. The intuition behind this feature is that nearby geographical locations may share similar bird habitats and thus have similar bird sighting patterns.

```
# Calculate Euclidean distance
data['Distance'] = ((data['Latitude'] - data['Longitude'])**2)**0.5
```

MODEL CREATION

We experimented with various regression algorithms to predict the count of bird sightings. The models considered are Linear Regression, Random Forest Regressor, Gradient Boosting Regressor, Support Vector Regressor (SVR), and K-Nearest Neighbors Regressor.

We evaluate the models based on the Mean Squared Error (MSE) and R-squared (R^2), which measure the average squared difference between the actual and predicted counts of bird sightings. Lower MSE indicates better model performance.

MSE Formula:

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

R-squared Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

where:

- y_i represents the actual count,
- \hat{y}_i represents the predicted count,
- n is the number of samples, and
- \bar{y} is the mean of the actual counts.

Model	MSE	R-Squared
Linear Regression	2.700	-0.003
Random Forest Regressor	1.183	0.545
Gradient Boosting Regressor	1.164	0.567
Support Vector Regressor (SVR)	4.008	-0.489

K-Nearest Neighbors Regressor	4.571	-0.699
-------------------------------	-------	--------

Table 2 Result of evaluation of different models.

R-squared gives an overall indication of the goodness of fit of the model, while MSE provides a more detailed measure of the average error between predicted and actual values.

R-squared is useful for comparing different models and selecting the best one, while MSE is useful for understanding the magnitude of errors in predictions.

It's important to note that a high R-squared does not necessarily mean the model is good if the MSE is high, and vice versa.

From the scores, the Gradient Boosting Regressor has the lowest MSE (1.164) and the highest R-squared (0.567), indicating better performance compared to other models. Therefore, the Gradient Boosting Regressor is the best model among the ones evaluated.

We have developed a robust model for predicting bird sightings in specific regions in Singapore based on historical data. The Gradient Boosting Regressor model, incorporating Euclidean distance as a feature, provides accurate predictions of bird counts. This model will serve as the foundation for generating the bird heat map, enabling users to explore and enjoy birdwatching in Singapore.

5. VERSION CONTROL AND DEPLOYMENT

VERSION CONTROL WITH GIT AND GITHUB

The application's source code is managed using Git, a distributed version control system that tracks changes and allows for collaborative development among team members. Git facilitates branching, merging, and reverting code, which is essential for maintaining a stable code base during the development lifecycle.

GitHub serves as the remote repository platform where the codebase is hosted. It acts as the central source of truth for the project and offers additional features like pull requests, issue tracking, and code reviews, which enhance collaboration and code quality.

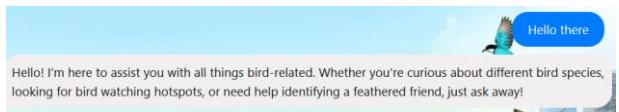
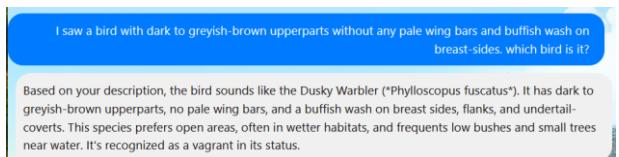
DEPLOYMENT AND HOSTING IN HEROKU CLOUD

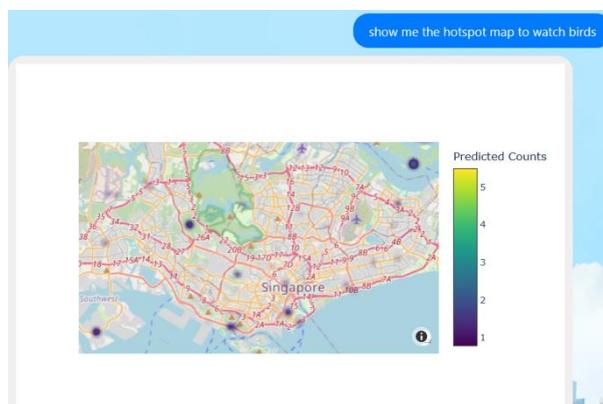
The Django application is hosted on Heroku, a cloud platform service that allows for easy deployment and scaling of web applications. Heroku integrates with GitHub and provides a straightforward, automated deployment process.

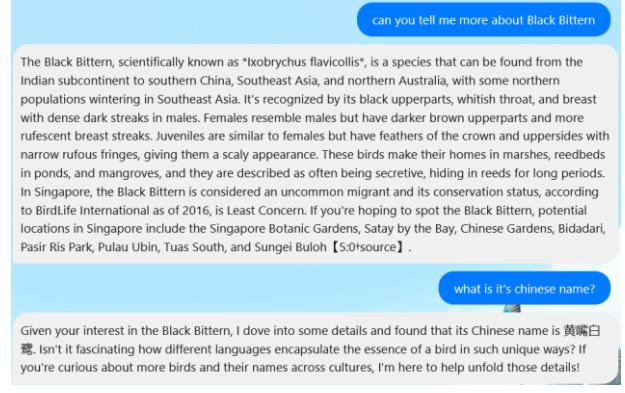
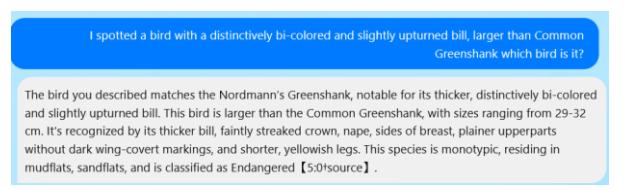
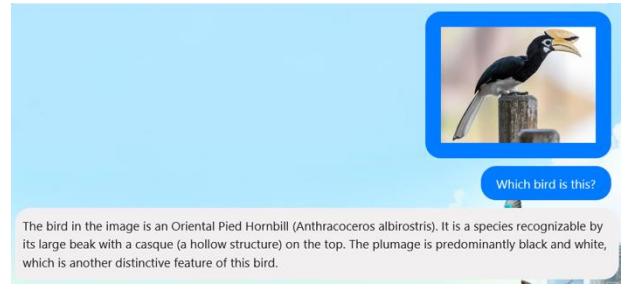
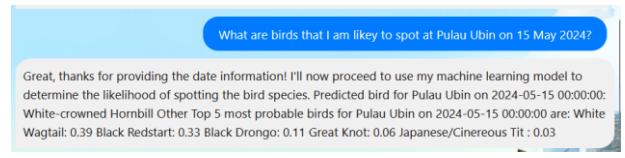
CONTINUOUS INTEGRATION/CONTINUOUS DEPLOYMENT (CI/CD)

The deployment pipeline is set up to trigger automatic deployments to Heroku upon successful merging to the main branch of the GitHub repository. This ensures that the application is always up to date with the latest stable version.

6. TESTING AND VALIDATION

Test Case	Testing results	Pass/Fail
<p>Intent Identification when user make the chat.</p> <p>The following intents should be identified accurately from user chat.</p> <ul style="list-style-type: none"> 1. Greeting 2. Bird Information 3. Bird Identification by from text 4. Bird Identification from Image 5. Bird Spotting Prediction 6. Park Recommendation 7. View Hotspot Map 	<p>1. Greeting</p>  <p>Identified intent: [01/May/2024 11:16:43] "GET /static/icons/favicon.ico HTTP/1.1" 200 123 User: Hello there Greeting [01/May/2024 11:17:14] "POST /handlerequest/ HTTP/1.1" 200 223</p> <p>2. Bird Information</p>  <p>Identified intent: User: what is the size of cotton pigmy goose? Bird Information [01/May/2024 11:20:38] "POST /handlerequest/ HTTP/1.1" 200 101</p> <p>3. Bird Identification by from text</p>  <p>Identified intent: User: I saw a bird with dark to greyish-brown upperparts without any pale wing bars and buffish wash on breast-sides. Which bird is it? Based on your description, the bird sounds like the Dusky Warbler (*Phylloscopus fuscatus*). It has dark to greyish-brown upperparts, no pale wing bars, and a buffish wash on breast sides, flanks, and undertail-coverts. This species prefers open areas, often in wetter habitats, and frequents low bushes and small trees near water. It's recognized as a vagrant in its status.</p> <p>4. Bird identification from image</p>  <p>This bird has glossy black plumage and a distinctive red eye, which are characteristic features of the Asian Koel. The male Asian Koel, in particular, displays this striking coloration. It's a species known for its loud calls and is commonly found across parts of the Asian continent.</p>	✓ Pass

	<p>Identified intent:</p> <pre>image based identification [01/May/2024 11:30:14] "POST /handlerequest/ HTTP/1.1" 200 299</pre> <h3>5. Bird Spotting Prediction</h3> <div style="background-color: #f0f8ff; padding: 10px; border-radius: 10px;"> <p>What are birds that I am likely to spot at Pulau Ubin on 15 May 2024?</p> <p>Great, thanks for providing the date information! I'll now proceed to use my machine learning model to determine the likelihood of spotting the bird species. Predicted bird for Pulau Ubin on 2024-05-15 00:00:00: White-crowned Hornbill Other Top 5 most probable birds for Pulau Ubin on 2024-05-15 00:00:00 are: White Wagtail: 0.39 Black Redstart: 0.33 Black Drongo: 0.11 Great Knot: 0.06 Japanese/Cinereous Tit: 0.03</p> </div>	
	<p>Identified intent:</p> <pre>Spottin Prediction [01/May/2024 11:33:32] "POST /handlerequest/ HTTP/1.1" 200 476</pre> <h3>6. Park Recommendation</h3> <div style="background-color: #f0f8ff; padding: 10px; border-radius: 10px;"> <p>can u recommend the parks to spot cotton pigmy goose</p> <p>Sure, if you're looking to spot the Cotton Pygmy Goose, here are five parks in Singapore that could be your best bet: 1. Henderson Waves: Known for its stunning architectural design, this park might just give you a glimpse of the Cotton Pygmy Goose. 2. Singapore Straits: With a high biodiversity, this place increases your chances of spotting the bird in its natural habitat. 3. Woodlands Waterfront Park: This coastal park with an abundant bird life is another spot where you might possibly find the Cotton Pygmy Goose. 4. Yishun Dam: Among the fresh green and blue, the serene Yishun Dam is a place where you might be able to spot the bird. 5. Sembawang Wagtail Roost: The vast open space and the bird-friendly environment make this park another potential location for spotting the Cotton Pygmy Goose. Remember to take your binoculars with you and happy bird-watching!</p> </div>	
	<p>Identified intent:</p> <pre>User: can u recommend the parks to spot cotton pigmy goose Identified bird: Cotton Pygmy Goose ["Henderson Waves", "Singapore Straits", "Woodlands Waterfront Park", "Yishun Dam", "Sembawang Wagtail Roost"] Park Recommendation [01/May/2024 11:37:32] "POST /handlerequest/ HTTP/1.1" 200 994</pre> <h3>7. View Hotspot Map</h3> <div style="background-color: #f0f8ff; padding: 10px; border-radius: 10px;">  <p>show me the hotspot map to watch birds</p> </div>	

<p>Provide the requested bird information to the user and answer bird related questions</p>	 	✓ Pass
<p>Identify the bird from descriptions provided by the user with good accuracy when sufficient or clear enough information is provided</p>		✓ Pass
<p>Identify the bird based on the image sent via the chat.</p>		✓ Pass
<p>Provide bird sighting prediction when user specify the date and location</p>	<p>The predicted list of bird they might encounter for the specified date and location is displayed in the chat</p> 	✓ Pass

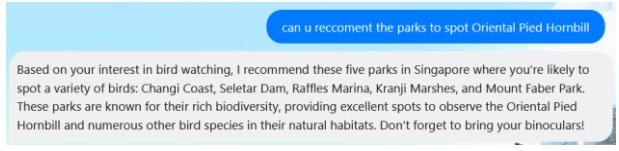
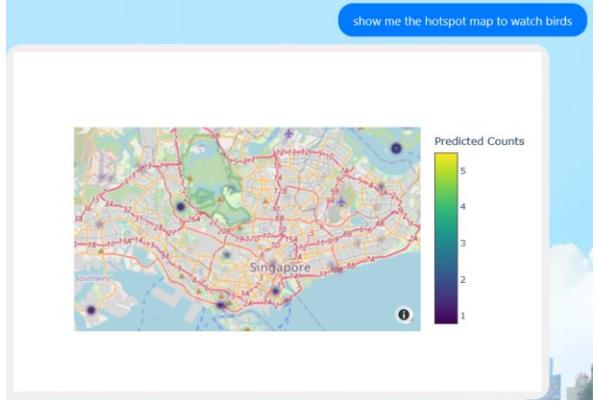
Provide the recommended list of Parks to find a specific bird	The recommended parks are displayed in the chat based when user specify the bird.	 ✓ Pass
User Test Case on querying for heatmap by inputting “heatmap” or “hot spot” into chat	A heat map focused on Singapore with predicted counts across different regions in Singapore for the current week.	 ✓ Pass

Table 3 Application Test Results

7. CONCLUSION

The project successfully implemented a comprehensive software solution for bird identification and birdwatching assistance, leveraging modern software architecture and advanced machine learning technologies. Through the integration of a robust Backend API built on Django, an intuitive Web User Interface enhanced by Bootstrap, and a seamless Mobile Application for Android, the application offers a versatile platform for birdwatchers. The inclusion of OpenAI's large language models has significantly enriched the application's capability to understand and process natural language queries, delivering accurate responses and engaging user experiences. Predictive analytics, location services, and customized user interactions form a solid foundation for an application that not only meets the practical needs of birdwatchers but also enhances their interactive experience.

8. RECOMMENDATIONS

We recommend the following actions to further enhance the application and maintain its relevance:

- **Continuous Integration and Updates:** Given the rapid advancement in machine learning and software development tools, it is recommended that the application undergo continuous updates to both its backend and frontend components to keep pace with technological advancements.
- **User Feedback Incorporation:** To enhance user satisfaction and application relevance, it is advisable to implement a structured feedback system to gather insights from users and make iterative improvements based on this feedback.
- **Expand Database and ML Capabilities:** The application could be improved by expanding its database to include more bird species and updating the machine learning models with new data to improve accuracy and response capabilities.
- **Enhance Security Measures:** As the application uses API keys and handles user data, enhancing security measures to protect against unauthorized access and data breaches is crucial.
- **Cross-Platform Compatibility:** Consider developing versions for other mobile operating systems, such as iOS, to increase the application's user base and accessibility.
- **Localization and Internationalization:** To reach a broader audience, adding support for multiple languages and customizing content based on geographic location could prove beneficial.

By following these recommendations, the project can maintain its relevance and continue to provide a high-quality service to its users, fostering a more engaging and rewarding birdwatching experience.

9. REFERENCES

- [1] Jason Brownlee, Tour of Data Preparation Techniques for Machine Learning, 30 Jun. 2020, <https://machinelearningmastery.com/data-preparation-techniques-for-machine-learning/>
- [2] Ana C. Lorena, Luis F.O. Jacintho, Martinez F. Siqueira, Renato De Giovanni, Lúcia G. Lohmann, André C.P.L.F. de Carvalho, Missae Yamamoto, May. 2011, Comparing machine learning classifiers in potential distribution modelling, <https://www.sciencedirect.com/science/article/pii/S0957417410011759>
- [3] Hanane D, Gini Index (Gini Impurity), <https://machinelearning-basics.com/gini-index-gini-impurity/>
- [4] Vignya Durvasula, Decoding Entropy in Decision Trees: A Beginner's Guide, 28 Dec. 2023, <https://www.askpython.com/python/examples/entropy-decision-trees>
- [5] Jason Brownlee, How to Use StandardScaler and MinMaxScaler Transforms in Python, 28 Aug. 2020, <https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/>

APPENDIX A: PROJECT PROPOSAL

Please refer to:

<https://github.com/jithinkrn/IRS-PM-2024-05-05-IS06PT-WingSpan/blob/main/ProjectReport/Project%20Proposal%20-%20WingSpan.pdf>

APPENDIX B: INDIVIDUAL REPORT

Name: Jithin Krishnan	Matriculation Number: A0249481W
Personal contribution to group project:	
1. Django Application Development:	
Architecture: Developed the overall structure and settings of a Django application, integrating controller, view, and model components for MVC architecture.	
2. Database Integration:	
Implemented SQLite for session management to track conversational history.	
3. Web Interface:	
Designed and implemented a user-friendly Chat UI using HTML, JavaScript, CSS, and Bootstrap. Enhanced user interaction with Ajax for dynamic image handling and display.	
3. Android Application Implementation:	
Developed an Android application with embedded mobile-responsive Web UI using WebView.	
Implemented robust file upload handling for image-based functionalities, leveraging Android's WebChromeClient and file chooser intents.	
4. CI/CD Pipeline and Deployment:	
Set up version control using Git and GitHub, streamlining code management and collaboration.	
Configured and maintained a CI/CD pipeline for automated deployments to Heroku, ensuring seamless updates and stable application performance.	
5. LLM Integration with OpenAI:	
Integrated Language Learning Models (LLM) into the Django application, configuring environmental variables and API communications for OpenAI's LLM models.	
6. User Intent Recognition:	
Developed a top-level user intent identification system using a Language Learning Model (LLM) to delegate requests to appropriate algorithms, enhancing the application's responsiveness and accuracy in user interactions. The intents include main features such as bird informational queries, bird identification by text, bird identification by image, bird sighting predictions, park recommendations, and heatmaps.	

7. Data Collection:

Utilized Python scripts with Selenium to collect and manage data essential for bird information and identification features.

8. Machine Learning Feature implementation:

End-to-end implementation of the following Features:

I. Bird Information Feature:

- User Query Interpretation: Developed algorithms to analyze and interpret user chats to accurately understand queries about birds.
- API Integration and Prompt Crafting: Implemented the Assistant API to generate prompts that facilitate detailed bird information retrieval based on user queries.
- Response Handling and Dialogue Continuation: Designed logic to manage API calls and handle responses, enabling continuous dialogue with users to refine or expand upon the information provided.
- Testing and Optimization: Conducted testing and fine-tuning of the model to improve accuracy and user experience.

II. Text-Based Bird Identification Feature:

- Chat Interpretation: Created code to parse and understand user text inputs for bird identification, utilizing natural language processing techniques.
- Prompt Design and API Interaction: Developed and implemented prompts that direct the Assistant API to retrieve bird identification data, enhancing the relevance and specificity of the information provided.
- Response Generation: Engineered the backend to handle API responses effectively, ensuring that the bird identification information is accurate and comprehensible.
- Model Refinement: Continuously tested and refined the system based on real-world usage and feedback, aiming to enhance the identification accuracy.

III. Image-Based Bird Identification Feature:

- Integration with ChatGPT Vision Model: Integrated OpenAI's ChatGPT Vision model into the Django application to process and analyze bird images submitted by users.
- Image Preprocessing: Implemented image preprocessing steps to ensure that images are in the correct format and quality for optimal analysis by the vision model.
- Prompt Crafting for AI: Crafted specialized prompts to guide the vision model in focusing on relevant features of the bird for identification.
- API Communication: Managed communications with the API to send processed images and receive identification results, handling all data transfers and response parsing.
- Continuous Testing and Tuning: Engaged in continuous testing and tuning of the feature to improve response accuracy and system reliability.

IV. Park Recommendation Feature:

- Data Scripting and Model Preparation: Authored scripts to prepare and clean data for model training, using collaborative filtering techniques.
- Model Development and Tuning: Developed the collaborative filtering model to predict park recommendations.
- Model Deployment and Integration: Implemented functionality to import and apply the trained model within the application framework to generate real-time recommendations.
- Natural Language Interaction: Enhanced the system to interpret user chats for feature extraction using GPT-4, providing recommendations in a conversational, natural language format.

- Iterative Improvement: Performed ongoing testing and fine-tuning to refine the recommendation accuracy and user satisfaction.

8. Reports and Documentation:

Participated in Project report writing, and documentation of work done.

What learnt is most useful for you:

One of the most enriching aspects of this project for me was the seamless integration of web development with machine learning technologies, which offered a comprehensive perspective on building sophisticated, AI-driven applications. This dual focus allowed me to understand and implement an effective system architecture that supports both immediate user interactions through a web interface and complex data processing in the backend.

The architecture of the Django application was particularly instrumental in demonstrating how web technologies and machine learning models can be intertwined. By developing and connecting the MVC components with various machine learning functionalities, I gained a deeper appreciation for designing systems that are both scalable and robust. This experience was crucial in enhancing my ability to integrate complex machine learning models, such as LLM, vision models, recommendation systems, directly into a web-based infrastructure.

The project also provided a valuable hands-on experience with chatbot concepts, particularly through the integration of LLMs to handle user intent recognition. This was not just about parsing text but understanding user needs and providing intelligent, context-aware responses. The LLM integration was pivotal, enabling the chatbot to delegate requests to the most appropriate algorithms or responses, thereby refining the application's interaction flow and overall user experience.

Moreover, my work with specific machine learning models, like the recommendation systems and image-based identification models, deepened my technical expertise. I was involved in every stage of these models' lifecycle—from data collection and preprocessing to training, evaluating, and integrating them in the application. This gave me an in-depth knowledge of how such models are constructed, tuned, and integrated into larger systems to provide dynamic and personalized user experiences.

In conclusion, this project was instrumental in solidifying my understanding of how to architect and implement complex, AI-enhanced web applications. It allowed me to not only apply what I had learned academically about web development and machine learning but also to see the practical implementation of theoretical concepts like chatbots and recommendation algorithms in real-world applications. These experiences have prepared me well for a future where I can contribute to, and lead, technology-driven projects with confidence and expertise.

How you can apply the knowledge and skills in other situations or your workplaces:

As a software professional, the knowledge and skills I've acquired through my project work are immensely transferable and applicable across various stages of the software development lifecycle. By integrating web development with machine learning into cohesive applications, I have learned to not only enhance user interfaces but also make them smarter, more interactive, and responsive to user needs.

Integration of Web Development and Machine Learning:

My experience in combining Django application development with advanced machine learning features is directly applicable to enhancing any web-based software. For example, implementing user intent recognition systems and integrating language learning models can significantly improve the functionality of chatbots on customer service platforms. This integration aids in understanding and effectively responding to customer queries, thereby enhancing user satisfaction and operational efficiency.

Understanding Overall Architecture and Integration of ML Models:

I have developed a deep understanding of system architecture and the integration of machine learning models, which allows me to contribute significantly to the planning and execution of complex software projects. This knowledge is crucial for designing systems that are not only efficient but also scalable, supporting data-driven decision-making, essential for businesses.

Chatbot Concepts and LLM Integration:

Building chat interfaces and integrating Language Learning Models (LLMs) to manage conversational flows has proven especially useful. This knowledge is applicable in developing advanced virtual assistants and support systems that can automate responses and personalize interactions based on user behavior and history—an invaluable asset for enhancing digital services.

In-depth Knowledge of ML Models Used:

The skills I have gained from implementing machine learning algorithms, such as vision models and recommendation systems, are particularly beneficial for e-commerce and media platforms where personalized content is crucial. Developing and fine-tuning algorithms that predict user preferences can significantly boost customer engagement and sales.

Cross-Disciplinary Skills:

Finally, my diverse skill set, ranging from coding to continuous integration and deployment, equips me to oversee the technical aspects of software projects, ensuring they meet the latest standards of technology and security. This comprehensive expertise is essential for maintaining a competitive edge in today's rapidly evolving tech landscape.

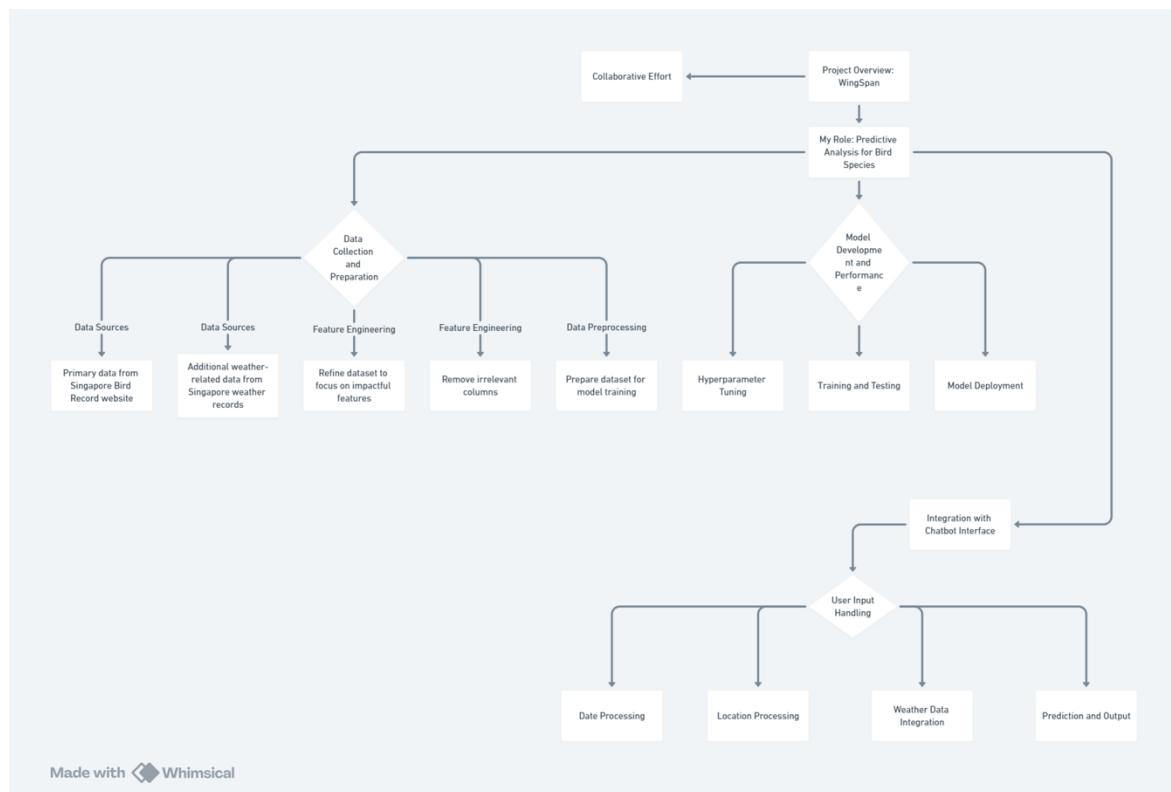
Overall, the competencies I have developed are not just theoretical but have real-world applications that extend beyond traditional boundaries, preparing me for a wide array of challenges and opportunities in the tech industry.

Name: Ashiwin Rajendran

Matriculation Number: A0291191E

Personal contribution to group project:

As part of the multidisciplinary team developing an innovative bird watching platform, my primary responsibility was to work on the development of the bird species prediction feature using Machine Learning algorithms and managing the entire workflow of the respective feature; end-to-end. I utilized the environmental, temporal, and locational data to predict bird sightings through data-driven insights.



Data Preparation:

- Identifying robust techniques and prepared the data through scaling, transformation and normalization for effective ML modelling.
- Data Cleaning:** Performed rigorous data extraction from sources / data cleaning / data preprocessing through several automation scripts to ensure the dataset's integrity.
- Feature Engineering:** Employed feature engineering to extract useful predictors from our knowledge base, while eliminating non-informative variables.

Model Selection:

- Choosing the right model is critical, and it involves testing various robust models to ensure optimal accuracy in both training and testing. I evaluated popular models including time series, XGBoost, and linear regression to determine which best suits our project's needs. After preliminary data

testing, I opted for the Random Forest classifier due to its ability to manage overfitting and its effectiveness in handling high-dimensional data.

Model Tuning, Training and Evaluation:

- Hyperparameter tuning where I utilized grid search and cross-validation to find the optimal model parameters.
- Trained the model with rigorous fine tuning of parameters and dataset variables to achieve the best accuracy score & evaluated the model through precision and recall.

Model Deployment:

- Integrated the trained model into the existing Django application infrastructure, ensuring smooth operation within the platform's backend.

LLM Integration for Bird prediction:

- Integrated a Large Language Model to process user requests through prompt via the chatbot interface, effectively managing and generating responses for users.

Error Handling and Input Validation in Chatbot:

- Developed a sophisticated input validation system to accurately parse and correct user inputs regarding date and location.
- Implemented weather data retrieval logic that adjusts based on the user's specified date, pulling real-time data or using historical weather patterns as needed.

Project documentation:

- Helped in creating promotional video for the project.
- Contributed to project documentation and reporting.

Team Collaboration:

- Assisted together with my team in collecting bird data information along with weather databases from the online sources and managed it accordingly for bird prediction feature.

What learnt is most useful for you:

The most invaluable skill I've acquired through this project is the ability to seamlessly integrate data science and machine learning techniques into real-world applications. Specifically, the experience of working end-to-end has been profoundly impactful for several reasons:

Understanding of Machine Learning Workflow: Managing the complete lifecycle of a machine learning project—from data preparation and model selection to tuning, training, evaluation, and deployment—has provided me with a comprehensive understanding of how each phase contributes to the robust application development.

Practical Application of Theoretical Knowledge: Integrating ML algorithms into a functioning application, bridged the gap between theoretical concepts and real-world execution. This enhances my problem-solving skills and prepares me for complex challenges in the field.

Team Collaboration: Working within a multidisciplinary team has taught me the importance of collaboration and communication across different areas of expertise. This has broadened my perspective on integrating diverse inputs to enrich project outcomes.

Advanced Problem-Solving with LLM Integration: Integrating a Large Language Model for natural language processing within the chatbot interface allowed me to delve deep into AI-driven state of the art models. This has honed my skills in deploying AI technologies for enhancing user engagement and functionality.

Adaptability and Innovation: Error handling, input validation, integrating dynamic weather data, data pre-processing, reinforced my ability to adapt and innovate.

In summary, this project has not only solidified my foundational skills in machine learning and data science but also enhanced my abilities in project management, user experience design, and cross-functional collaboration. These skills are invaluable assets in my ongoing professional development and future career endeavors in technology.

How you can apply the knowledge and skills in other situations or your workplaces:

The skill set developed during this WingSpan project is highly transferable to my current robotics profession and I will be applying it in various situations and workplace scenarios:

The experience of integrating machine learning with web development and natural language processing can be applied to various technology sectors. Since I work in Robotics sector, this ability is so helpful for me to integrate and develop complex software solutions with AI capabilities into the robotics domain. Particularly in enhancing the user experiences of our Autonomous Underwater Vehicle with AI functionalities.

The skills acquired in data collection, cleaning, and analysis empower me to implement evidence-based decision-making and skills towards handling data sources. Particularly, since I work in Robotics division. I will be dealing with several modular sensors and there is a necessity for me to process the data and to make it meaningful & valuable for other robotics perception or navigation tasks.

The practical knowledge of deploying machine learning models can be used in other scenarios where predictive analytics and user behavior modeling are crucial. I gained an overview of both traditional and current technologies used. For instance, I gained the ability to pick and choose the traditional/modern ML technique based on the problem statement especially in perception domain of my robotics career.

Applying advanced problem-solving and adaptability skills will be instrumental in improving robotic product development and driving innovation across various companies in my future career. These skills are essential for roles that demand continuous product optimization and innovative solutions to complex technical challenges.

The ability to quickly adapt and implement new technologies is a valuable skill in the fast-evolving tech landscape. My experience can help organizations stay ahead of technology curves, implementing state of the art models and techniques as they become available.

By utilizing these skills, I can make a substantial impact in any workplace, especially in my current robotics career, by tailoring my approach to effectively address specific challenges.

Name: Shankar Sai Ganesh	Matriculation Number: A0168149U
---------------------------------	--

Personal contribution to group project:

In my role for the Wingspan chatbot project, I contributed to several key areas:

1. Data Extraction and Collection:

- a. I developed Selenium scripts to automate the extraction of data from various websites. These scripts were tailored to extract bird information and historical weather data.
- b. Leveraging Python Selenium with the Chrome browser, I ensured efficient data collection, simulating human interaction with web pages.

2. Data Processing:

- a. I aggregated monthly weather data into a single Excel file using Pandas, streamlining further analysis, and ensuring data consistency.

3. Heatmap Model Preparation:

- a. I transformed raw data such as location and bird species into longitude and latitude coordinates for heatmap generation.
- b. I calculated the count of bird species for each week at each location, providing insights into bird distribution patterns over time.
- c. To facilitate visualization, I converted locations to specific regions and explored various Python packages for displaying the heatmap on the chatbot interface. This involved packages such as Matplotlib, Seaborn and Folium.
- d. I compared different regression models to predict bird species distribution around Singapore, selecting the most accurate one for integration into the chatbot.

4. Integration with Django Application:

- a. I seamlessly integrated the heatmap model into the Django application powering the Wingspan chatbot.
- b. Introduced the usage of the package joblib which helped the team to integrate the trained models with Django.
- c. This involved incorporating the model's functionality into the application's codebase and enhancing the chatbot's user interface to display the heatmap visualization effectively.

5. Presentation and Documentation:

- a. Took responsibility for the System Design Presentation and Video and prepared all the slides and script for the System Design of WingSpan.
- b. Completed the report for both the Project Overview and Heatmap section as well as the Testing of the application.

My contributions ranged from data extraction and processing to model development and integration, showcasing my versatility across different aspects of the project. Through critical thinking and problem-

solving, I ensured that the implemented features met the project's requirements and enhanced the Wingspan chatbot's functionality.

What learnt is most useful for you:

From the contributions I made to the Wingspan chatbot project, I've gained valuable skills and insights that I find particularly useful:

1. **Data Extraction and Processing Skills:** Developing Selenium scripts and using Pandas for data manipulation have been immensely useful. These skills have not only enabled me to automate data collection but also to organize and preprocess the data effectively for analysis.
2. **Geospatial Data Handling:** Transforming data into longitude and latitude coordinates for heatmap generation has deepened my understanding of geospatial data processing. This skill has broad applications beyond this project, particularly in projects involving location-based analytics and visualization.
3. **Model Evaluation and Selection:** Comparing different regression models for predicting bird species distribution has taught me the importance of model selection. Learning how to evaluate and choose the most suitable model for a given problem has been invaluable and can be applied to various machine learning projects.
4. **Integration with Web Applications:** Integrating the heatmap model into the Django application has improved my skills in web development and software integration. This experience has equipped me to integrate machine learning models into web applications seamlessly, a skill that is highly relevant in today's technology landscape.
5. **Problem-solving and Critical Thinking:** Throughout the project, I encountered various challenges that required creative problem-solving and critical thinking. Learning how to approach and overcome these challenges has been a significant takeaway and has enhanced my overall problem-solving abilities.

Overall, the skills and knowledge I've gained from working on the Wingspan project have been instrumental in my professional development. They have equipped me with a diverse skill set that I can apply to future projects in machine learning, data analysis, and software development.

How you can apply the knowledge and skills in other situations or your workplaces:

Transitioning into the AI field in the future, the knowledge and skills I've gained from contributing to the Wingspan project will continue to be valuable. These skills not only benefit my current role as an automation test engineer but also provide a strong foundation for my future endeavours in AI.

In my current role, I can immediately apply these skills to enhance our testing processes. For example, my experience in developing Selenium scripts for data extraction can be leveraged to automate testing tasks more effectively. By automating data extraction and preprocessing, I can streamline test data generation and analysis, improving the efficiency and accuracy of our testing efforts.

Additionally, my understanding of geospatial analysis will be beneficial in testing applications with location-based features. I can accurately simulate geospatial scenarios, ensuring comprehensive test coverage and validating the functionality of these features under different geographic conditions.

Moreover, my ability to evaluate and compare regression models can be applied to performance testing, helping to identify and optimize critical performance metrics. By selecting the most appropriate models for evaluating application performance, I can ensure that our software meets the required standards for speed, scalability, and reliability.

Looking ahead to my transition into the AI field, these skills will be invaluable in various AI projects. For instance, my experience with data extraction, preprocessing, and analysis will be essential for working with large datasets in AI applications. Whether it's cleaning and preparing data for training machine learning models or conducting exploratory data analysis, these skills will enable me to handle data more effectively.

Furthermore, my knowledge of geospatial analysis can be applied to AI projects involving location-based data. From developing geospatial recommendation systems to analysing environmental data, this skill will open up opportunities to work on diverse and impactful AI projects.

Additionally, my experience in integrating machine learning models into web applications will be valuable for deploying AI solutions in production environments. Whether it's building AI-powered chatbots, recommendation systems, or predictive analytics tools, I can seamlessly integrate AI functionality into various applications, ensuring a smooth user experience.

Finally, my problem-solving and critical thinking skills developed during the Wingspan project will be essential for tackling complex challenges in the AI field. From optimizing model performance to addressing data biases and improving AI algorithms, these skills will enable me to innovate and deliver effective solutions in the rapidly evolving field of AI.

Overall, the knowledge and skills gained from the Wingspan project have prepared me well for both my current role as an automation test engineer and my future endeavours in the AI field. By leveraging these skills, I can contribute effectively to testing processes and AI projects, driving innovation and delivering high-quality solutions in both domains.

Name: Prem Varijkzhan	Matriculation Number: A0291913B
-----------------------	---------------------------------

Personal contribution to group project:

1. Data Collection

- Spearheaded the acquisition of essential datasets necessary for the project by actively researching and evaluating various APIs and sources of historical data. Coordinated the gathering of historical and real-time weather data through the most relevant APIs identified, ensuring comprehensive coverage and accuracy. This meticulous approach supported robust analysis and effective modeling by providing high-quality data tailored to the project's needs.

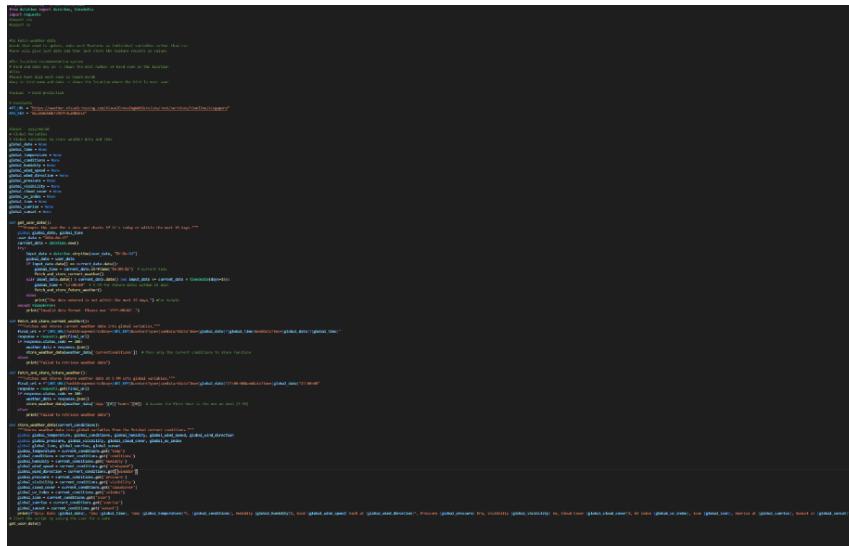
2. Data Processing

- Implemented robust data processing pipelines to cleanse, transform, and prepare data for analysis with the weather data and ecological observations, enhancing the reliability and validity of the data inputs for subsequent modeling tasks. Additionally, standardized critical measurements, such as converting temperature readings from Fahrenheit to Celsius, to ensure consistency and comparability across different data sources.

3. Weather Data Integration

- Authored Python code to efficiently retrieve both historical and predictive weather data via APIs. Developed functions to manage API requests, parse responses in JSON format, and organize the retrieved data into a structured format. This design ensures that the weather data is

accurate and timely but also accessible and usable by other team members for predictive modeling and analysis.



The screenshot shows a terminal window with a large amount of Python code. The code appears to be a script for a park recommendation system, likely using machine learning or a similar algorithm. It includes imports for various libraries like numpy, pandas, and sklearn, as well as definitions for classes like 'Bird', 'Park', and 'User'. The code involves reading data from files, performing data processing, and implementing logic for park recommendations based on user inputs and environmental conditions. The text is mostly in black on a dark background, with some syntax highlighting for certain words.

4. Park Recommendation System Model

- Core Algorithm Design and Development:
 - Designed and developed the core algorithm for a park recommendation system tailored to predict park suitability based on specific bird species. This approach involved leveraging machine learning models that effectively use environmental data and user inputs (such as bird names) to determine the best park recommendations.
- Model Selection:
 - Conducted extensive research to identify the most effective predictive models for the recommendation system, with a particular emphasis on finding suitable similarity measurements that assess the match between user preferences (e.g., desired birds to observe) and available park environments.
 - Evaluated various algorithms for their accuracy, computational efficiency, and the ease with which they could be integrated into existing data structures, ensuring the chosen models were well-aligned with the system's objectives and the complexity of the data environment.

5. Model Optimization and Validation

- Parameter Optimization:
 - Fine-tuned the parameters of the selected models to enhance their performance for specific tasks, such as accurately identifying parks where particular bird species are likely to be observed. This process involved adjusting variables to optimize the balance between system responsiveness and the accuracy of predictions.
- Advanced Validation Techniques:
 - Employed advanced cross-validation techniques to bolster the model's generalizability and robustness. This method ensured the system maintained reliable performance across various scenarios and data conditions, which is crucial for applications in dynamic environments like outdoor parks.
- Performance Evaluation:
 - Systematically evaluated the model against critical performance metrics, including precision and recall. This rigorous assessment ensured that the park recommendations were both accurate and relevant, thereby maximizing user satisfaction and enhancing system usability.
- Comparative System Testing:
 - Conducted tests with various types of recommendation systems to determine the most effective approach for our specific needs. This comparative analysis helped identify the strengths and weaknesses of each model in practical scenarios.
- Issue Identification and Communication:
 - Actively searched for and identified any faults within the different systems tested. Promptly communicated these findings to the team, facilitating discussions on potential improvements and the decision-making process regarding the final model selection.

```

import pandas as pd
import numpy as np
import os
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.metrics.pairwise import pairwise_distances
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.model_selection import train_test_split

# Obtain the current working directory to construct file paths dynamically
current_directory_path = os.getcwd()
print(f"Current directory: {current_directory_path}")

# Construct the full path to the CSV file storing park and bird data
data_path = os.path.join(current_directory_path, "Data", "final_count_dataframe.csv")
print(f"Data path: {data_path}")

# Load and preprocess the data from CSV
data = pd.read_csv(data_path, encoding="latin1", dtype=str)
data['bird'] = data['bird'].str.strip() # Remove any leading/trailing whitespace
data.drop_duplicates(subset=['bird', 'park'], inplace=True) # Remove duplicate entries

# Function to process data into a suitable format for similarity calculations
...
def process_data(df):
    # Group data by 'Location' and aggregate 'Species' into sets
    grouped = df.groupby('Location')[['Species']].apply(set).reset_index()
    mlb = MultiLabelBinarizer() # Initialize the binarizer
    matrix = mlb.fit_transform(grouped[['Species']]) # Transform species sets into binary matrix format
    return grouped['Location'], matrix, mlb, mlb.classes_
...

def process_data(df):
    # Group data by 'Location' and aggregate 'Species' into sets
    grouped = df.groupby('park')[['bird']].apply(set).reset_index()
    mlb = MultiLabelBinarizer() # Initialize the binarizer
    matrix = mlb.fit_transform(grouped[['bird']]) # Transform species sets into binary matrix format
    boolean_matrix = matrix.astype(bool) # Convert to boolean explicitly to avoid warnings
    return grouped['park'], boolean_matrix, mlb, mlb.classes_

# Split data into training and testing sets to validate the model
train_data, test_data = train_test_split(data, test_size=0.2, random_state=42)
train_locations, train_matrix, mlb, species_classes = process_data(train_data)

# Function to recommend parks based on a specific bird using Jaccard similarity
def recommend_parks_for_bird(bird_name, mlb, train_matrix, train_locations):
    bird_index = list(mlb.classes_).index(bird_name)
    bird_vector = np.zeros((1, len(mlb.classes_)))
    bird_vector[0, bird_index] = 1 # Set the bird presence to 1
    similarity_matrix = 1 - pairwise_distances(bird_vector, train_matrix, metric='jaccard') # Using Jaccard similarity
    recommendation_indices = np.argsort(-similarity_matrix, axis=1)[:, :3] # Top 3 recommendations
    return [train_locations[i] for i in recommendation_indices[0]]

# Example usage for a specific bird
bird_name = "Amur Stonechat" # Change this to the bird you're interested in
recommended_parks = recommend_parks_for_bird(bird_name, mlb, train_matrix, train_locations)
print(f"Top 3 recommended parks for {bird_name}:", recommended_parks)

# Evaluate recommendations for a specific bird
def evaluate_specific_bird_recommendations(recommended_parks, bird_name, train_data):
    true_parks = set(train_data[train_data['bird'].str.contains(bird_name, na=False)]['park'].unique())
    predicted_parks = set(recommended_parks)
    true_positive = predicted_parks.intersection(true_parks)
    precision = len(true_positive) / len(predicted_parks) if predicted_parks else 0
    recall = len(true_positive) / len(true_parks) if true_parks else 0
    f1 = 2 * (precision * recall) / (precision + recall) if (precision + recall) else 0
    return precision, recall, f1

# Calculate metrics for the specific bird recommendations
precision, recall, f1 = evaluate_specific_bird_recommendations(recommended_parks, bird_name, train_data)
print(f"Precision: {precision:.4f}, Recall: {recall:.4f}, F1 Score: {f1:.4f}")

```

6. Presentation

- Collaborated closely with the team to design and create the project presentation slides. Together, we developed a compelling narrative that effectively highlighted the project's methodology, findings, and the potential impact by using our project.

7. Documentation

- Worked closely with the team to produce a detailed documentation report and video presentation on "Wingspan," explaining the development process, the methodologies employed, and the models developed for the project. Our collaborative efforts were aimed at ensuring clarity and accessibility through both written and visual mediums.

This approach allows for a comprehensive understanding of the project's framework, facilitating its effective utilization and evaluation.

8. User Guide

- Created a detailed user guide for the project, providing clear instructions on how to access and utilize the system through various methods: online via a browser, offline on a local host, and through an offline Android Studio app. This guide offers step-by-step directions for each mode of access, ensuring users can effectively set up and navigate the system regardless of their preferred platform. This comprehensive resource supports the practical deployment of the system, facilitating seamless operation and enhancing user engagement across different environments.

What learnt is most useful for you:

How you can apply the knowledge and skills in other situations or your workplaces:

APPENDIX C: MAPPED SYSTEM FUNCTIONALITIES AGAINST KNOWLEDGE OF COURSES KNOWLEDGE OF COURSE

Course Knowledge	System Functionality
<u>Machine Reasoning</u> Inductive Reasoning Techniques: <ul style="list-style-type: none"> • Random Forest classifiers • Gradient Boosting Regressors 	Bird Sighting prediction Heatmap generation
<u>Reasoning Systems</u> Retrieval Reasoning Systems: <ul style="list-style-type: none"> • Random Forest classifiers • Gradient Boosting Regressors Reasoning and Knowledge Discovery from Datasets <ul style="list-style-type: none"> • User-Based CF 	Bird Sighting prediction Heatmap generation Park Recommendation
<u>Cognitive System</u> Natural Language Processing Natural Language Understanding Large Language Models Vision Cognition Common Sense Reasoning Response Generation	Intent identification, Bird informational queries, Bird identification based on text, Bird identification based on images. Conversion with user in natural language

APPENDIX D: INSTALLATION AND USER GUIDE

Please refer to:

<https://github.com/jithinkrn/IRS-PM-2024-05-05-IS06PT-WingSpan/blob/main/ProjectReport/User%20Guide%20-%20WingSpan.pdf>