UNIVERSITY OF SRI JAYEWARDENEPURA
Faculty of Computing

FRUITOPIA

AI Intelligent Fruit Recommendation System

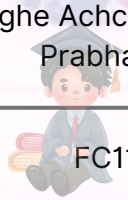
Project Report

(CCS2062 | CSE2062 - Artificial Intelligence)

Fruitopia AI is an intelligent fruit recommendation system designed to provide personalized fruit suggestions based on users' health conditions, preferences, and available inventory. The system leverages advanced AI technologies including machine learning, natural language processing, computer vision, and conversational AI to deliver a comprehensive fruit encyclopedia and interactive user experience. The project integrates multiple AI methods such as disease-based recommendations, personalized suggestions, NLP for symptom analysis, a conversational chatbot, and fruit image recognition. It features a comprehensive fruit encyclopedia with nutritional data, cultivation information, and interactive geolocation maps.

Under the guidance of:
Dr. Budditha Hettige

Elpitiya Badalge Don Sanjaya Samudra	Marasinghe Achchige Sithum Dineth Prabhashwara
FC115634	FC115551
Bachelor of Computing Honours in Software Engineering	Bachelor of Computing Honours in Computer Science



Date:October 14, 2025

Project Overview and Objectives

Project Overview

Fruitopia AI is an intelligent fruit recommendation system that leverages multiple AI technologies including machine learning, computer vision, natural language processing, and conversational AI. The system provides personalized fruit suggestions based on health conditions, interactive fruit encyclopedias with detailed nutritional data and cultivation maps, image recognition for fruit identification, and a conversational chatbot for user interactions. The project integrates a modern Angular frontend with a FastAPI backend, PyTorch-based ML models, and Rasa-powered chatbot.

Project Objectives

1 AI-Powered Recommendations

To develop an AI-powered fruit recommendation system that considers health conditions and user preferences.

2 Computer Vision

To implement computer vision capabilities for fruit identification from images.

3 Conversational AI

To create a conversational AI chatbot for interactive fruit-related queries.

4 Comprehensive Encyclopedia

To build a comprehensive fruit encyclopedia with detailed information and interactive maps.

5 User Experience

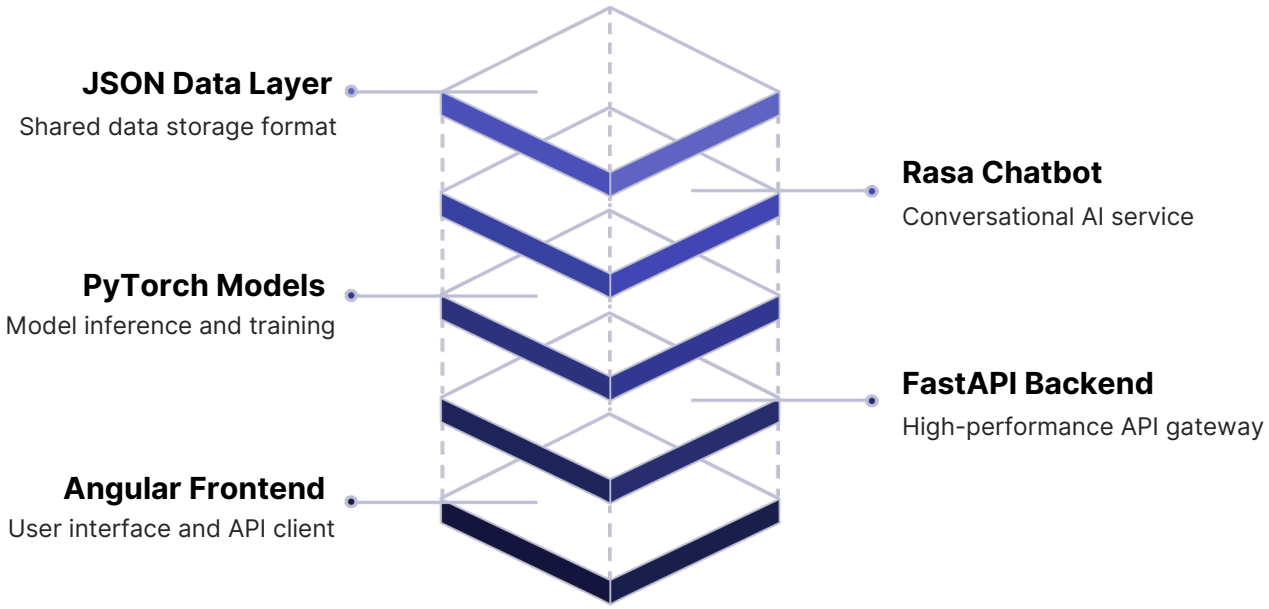
To provide a user-friendly web interface for seamless interaction.

6 Scalability

To ensure scalability, reliability, and ease of deployment.

System Architecture and Technologies Used

Thesystem follows a microservices architecture with separate frontend and backend components. The backend is built with Fast API for high-performance API endpoints, while the frontend uses Angular for a responsive web application. Machine learning models are trained separately and integrated into the backend. Key components include: " Frontend: Angular application with interactive UI components. " Backend: Fast API server handling API requests, ML inference, and data processing. " Machine Learning: PyTorch-based models for image classification and recommendation systems. " Chatbot: Rasa framework for conversational AI. " Data Layer: JSON files for disease recommendations and fruit data.

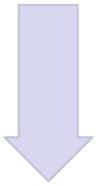


Core Technologies

Frontend Framework	Angular 19.2.0	Responsive web UI
UI Library	Angular Material 19.2.19	Material Design components
Maps	Leaflet 1.9.4	Interactive geolocation maps
Backend Framework	FastAPI	High-performance API
Machine Learning	PyTorch	Deep learning for image recognition
Computer Vision	TorchVision	Image preprocessing and models
Chatbot Framework	Rasa 3.1	Conversational AI
NLP	spaCy/NLTK	Text processing
Deployment	Uvicorn	ASGI server

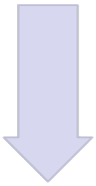
Implementation Details and Key Features

Implementation Approach



Data Collection

Gathered fruit images and nutritional data for 30+ fruit types.



Model Training

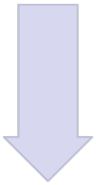
Trained a ResNet-based classifier on the fruit image dataset.

Backend Development

Built Fast API endpoints for recommendations, vision, and chatbot.

Frontend Development

Created Angular components for explore, recommend, and chat features.



Integration

Connected frontend to backend via REST APIs with CORS support.

The approach focused on modularity, with separate concerns for ML, API, and UI. Lazy loading was implemented for ML models to ensure the server starts even without PyTorch installed.

Key Features Implemented

Disease-based Recommendations	Suggests fruits based on health conditions	Rule-based ML
Personalized Suggestions	Tailored recommendations using user profiles (framework ready)	Recommendation System
NLP Symptom Analysis	Processes natural language inputs for health info	Natural Language Processing
Conversational Chatbot	Interactive AI assistant for fruit queries	Conversational AI (Rasa)
Fruit Image Recognition	Identifies fruits from uploaded images	Computer Vision (CNN)

Machine Learning and Computer Vision

The core MLcomponentis a fruit image classifier trained on a dataset of 30+ fruit classes. The model uses: "Architecture:**MobileNetV2** fine-tuned for 30 fruit classes (ResNet-based classifier was used in the initial phase) " Training Data: Custom Fruit Image Dataset with 826 images across 30 classes (acerolas, apples, apricots, avocados, bananas, blackberries, blueberries, cantaloupes, cherries, coconuts, figs, grapefruits, grapes, guava, kiwifruit, lemons, limes, mangos, olives, oranges, passionfruit, peaches, pears, pineapples, plums, pomegranates, raspberries, strawberries, tomatoes, watermelons) " Augmentation: Random crops, flips, rotations, and color jitter " Performance: Achieved high accuracy on validation set

Recommendation system uses JSON-based rules mapping diseases to beneficial fruits. Future enhancements could include collaborative filtering.

ML Project Metrics

826
Images

Total images in the custom dataset.

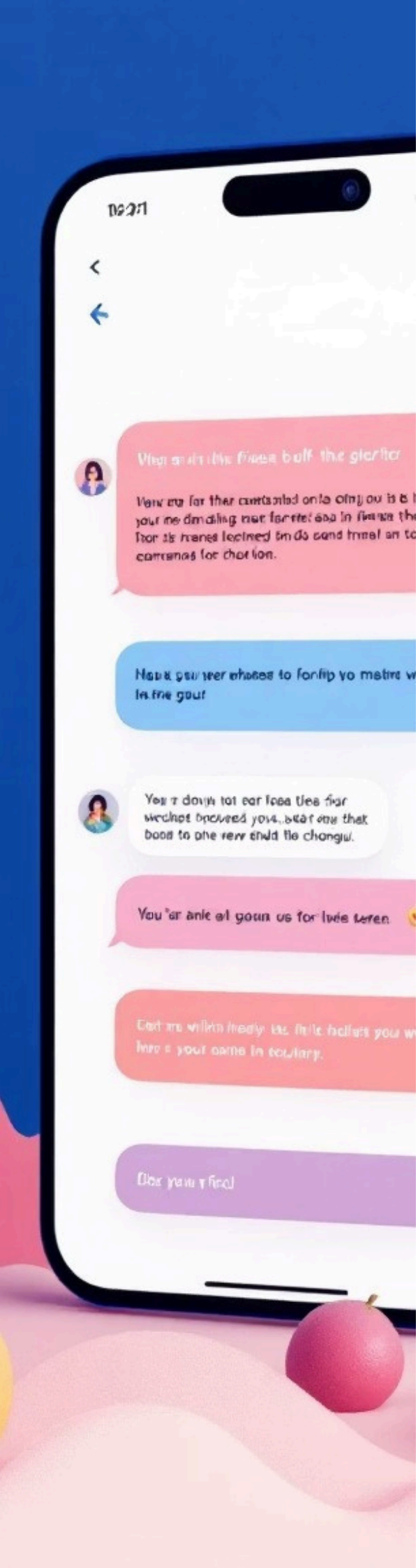
30
Fruit Classes

Classes supported by the MobileNetV2 model.

High
Accuracy

Model performance on fruit classification.

The ML integration uses lazy-loaded PyTorch models, fallback image recognition, and mock predictions for development. Training scripts include `train.py`, `evaluate.py`, `preprocess_split.py`, and `inspect_dataset.py`.



Conversational AI and Chatbot Implementation

The chatbot is built with **Rasa 3.1**, featuring intents for greetings, recommendations, fruit info,and health queries. It includes: " NLU: Natural language understanding for user intents " Dialogue Management: Context-aware conversations " Custom Actions: Python actions for dynamic responses " Integration: Connected to FastAPI backend

Chatbot Intents and Actions

Intents (16 Total)

- greet, goodbye
- ask_recommendation, inform_disease
- ask_fruit_info, ask_ health_ benefits
- ask_fruit_comparison, ask_fruit_ recipes
- ask_nutrition_info, ask_allergies, ask_warnings

Entities tracked: **disease**, **fruit**.

Custom Actions

- action_recommend_fruits
- action_get_fruit_ info
- action_get_fruit_comparison
- action_get_fruit_ recipes

The chatbot uses session management and a custom wrapper for integration with the FastAPI endpoint `/chatbot/message`.

Frontend and Backend Development

Frontend Development (Angular 19)

The Angular frontend provides a modern, responsive interface with: " ExploreComponent:6-tab encyclopedia with maps, nutrition charts " Recommend Component: Disease-based and personalized suggestions " Chat Component: Real-time chatbot interface " Vision Component: Image upload and prediction display

Uses Angular Material for consistent design and Leaflet for interactive maps. Key components include app.component.ts/html/scss (Main application shell), explore/ (Comprehensive fruit encyclopedia), recommend/, vision-upload/, and chat-widget/.

Backend Development (FastAPI)

The Fast API backend exposes RESTful endpoints via vision_api.py

/recommend

Fruit recommendations based on disease/health conditions.

/vision/predict

Image classification with PyTorch model and fallbacks.

/explore

Detailed fruit encyclopedia data and list of available fruits.

/chatbot/message

Conversational AI responses with session management.

/recipes/generate

AI-generated recipes with dietary preferences.

Includes CORS middleware for frontend integration and lazy model loading for optional dependencies.

Project Structure and Data Layer

Root Directory Structure

- `README.md` :Comprehensive setup guide (Windows PowerShell environment)
- `README_AI.md` :Detailed AI assets documentation
- `report.html` :Generated HTML report
- `response.json` :API response samples
- `test_app.py` :Flask-based test application
- `train_rasa.py` :Rasa training and server management script

Data Layer (`data/`)

The data layer is crucial for the system's functionality, providing rich information for the encyclopedia and recommendations.

Fruit Image Dataset

Organized image folders for each fruit class, used for ML training.

Explore Data (30 JSON files)

Detailed JSON files for each fruit containing botanical info, nutritional facts, health benefits, cultivation, and map coordinates etc.

Interactive Maps

Map coordinates for origins and cultivation areas, integrated via Leaflet.

Training Data

`training_data.json` for chatbot/NLP training.

Testing, Challenges, and Solutions

Testing and Validation

Testing included:

- Unit Tests: Backend API endpoints
- Integration Tests: Frontend-backend communication
- Model Evaluation: Accuracy, precision, recall on test set
- User Testing: Manual validation of features and chatbot interactions

The system was evaluated for accuracy in fruit recognition and relevance of recommendations.

Challenges and Solutions

Model Loading

Implemented lazy initialization with error swallowing to ensure server reliability, addressing the challenge of large model size.

CORS Issues

Configured middleware for cross-origin requests, enabling seamless frontend-backend communication.

Data Quality

Curated dataset and used augmentation techniques to improve model robustness and performance.

Performance

Optimized image processing and API responses to ensure a fast user experience.

Future Directions and Conclusion

Future Directions

The modular architecture provides a strong foundation for future enhancements:

Advanced ML

Implement user-based collaborative filtering for better personalization.

Nutrition Tracking

Add meal planning and calorie tracking features.

Mobile App

Develop native mobile applications for iOS and Android.

Real-time Data

Integrate with APIs for current fruit prices and availability.

Multilingual Support

Expand NLP capabilities for multiple languages.

IoTIntegration

Connect with smart refrigerators for inventory management.

Conclusion

> Fruitopia AI successfully demonstrates the integration of multiple AI technologies into a cohesive fruit recommendation system. The project achieved its objectives of providing intelligent, personalized fruit suggestions while offering a rich encyclopedia experience. The modular architecture ensures scalability and maintainability. Key achievements include high-accuracy image recognition, effective disease-based recommendations, and an engaging conversational interface. The system serves as a foundation for future enhancements in personalized nutrition and AI-driven health applications.

References

1. PyTorch Documentation
2. FastAPI Documentation
3. Angular Documentation
4. Rasa Documentation
5. Fruit nutritional data sources