Artificial Intelligence And Machine Learning with MERN Project Documentation format

1. Introduction

Project Title: Small sorting: Transfer learning for identifying rotten in fruits and vegetables
Team ID : LTVIP2025TMID36983

- **Team Leader:** Bandi Vasantha.
- **Team Member:** Bellamkonda Naga Raju.
- **Team Member:** Beenathi Harsha Vardhan

2. Project Overview

- **Purpose:** The purpose of "Small sorting: Transfer learning for identifying rotten in fruits and vegetables" is to automate the process of distinguishing fresh from rotten produce, thereby significantly reducing food waste and improving efficiency. By leveraging transfer learning, the solution aims to provide a cost-effective and accurate method for small-scale sorting, enabling better quality control for producers and enhanced satisfaction for consumers. Ultimately, it seeks to mitigate economic losses due to spoilage and contribute to more sustainable food supply **c**hains.
- **Features:** This system features AI-powered classification using transfer learning for accurate identification of rotten produce. It offers efficient, small-scale automated sorting, making it accessible for various users. The solution is adaptable and scalable, allowing for easy integration of new fruit/vegetable types and deployment in diverse settings.

3. Architecture

- **Frontend:** user interface for uploading images
- **Backend:** API server using a pre-trained model for transfer learning to identify rotten fruits/vegetables.
- **Database:** storing images and model results for analysis and improvement.

.

4. Setup Instructions

- **Prerequisites:**

  Data Privacy: Ensure sensitive data (like personal information) is protected during upload or processing.

  Model Accuracy: Regularly test and retrain the model to avoid misclassification.

  Input Validation: Validate uploaded images for correct format and size to avoid errors in the model.

  Hardware Requirements: Ensure the server has sufficient GPU/CPU power for model inference, especially when using large pre-trained models.

  Error Handling: Implement proper error handling and fallback mechanisms in case of incorrect predictions or server downtime.

Installations:

1. Frontend Setup (React):

- Install Node.js (if not already installed): Node.js Download
- Create React app:
  npx create-react-app fruit-rot-app
- Install dependencies (for API calls):
  npm install axios

2. Backend Setup (Flask):

- Install Python (if not already installed): Python Download
- Install Flask for API:
  pip install flask
- Install TensorFlow or PyTorch (for model inference):
  pip install tensorflow or pip install torch

3. Model & Dependencies:

- Install OpenCV for image processing:
  pip install opencv-python
- Install NumPy (for numerical operations):
  pip install numpy

4. Database Setup (MySQL):

- Install MySQL:
  sudo apt install mysql-server
- Install MySQL connector for Python:
  pip install mysql-connector-python

5. Cloud Storage (Optional for image storage):

- Install AWS S3 SDK (if using AWS):
  pip install boto3

6. Testing:

- Test API endpoint:
  curl -X POST http://localhost:5000/predict -F "file=@image.jpg"

5. Folder Structure

Frontend (React):

1. **src/**: Contains React components, API services, and assets (images, CSS).
2. **public/**: Static files like index.html, favicon, and other static content.
3. **package.json**: Manages dependencies and scripts.

Backend (Flask):

1. **app/**: Contains the core application logic (routes, models, and utils).
2. **run.py**: Entry point to start the Flask server.
3. **requirements.txt**: Lists the Python dependencies (Flask, TensorFlow, etc.).

Database (Optional):

1. **MySQL/PostgreSQL**: Used to store image metadata, prediction results, etc.
2. **models.py**: Defines the database schema.

Ask ChatGPT

6. Running the Application

- Provide commands to start the frontend and backend servers locally.
    - **Frontend:** `npm start` in the client directory.
    - **Backend:** `npm start` in the server directory.

7. API Documentation

The API allows users to upload fruit or vegetable images, which are processed through a transfer learning model to predict if they are rotten. It returns the classification result (rotten or fresh) along with confidence scores for the prediction.

8. Authentication

"Transfer learning for identifying rotten fruits and vegetables using pre-trained models to improve accuracy and reduce training time on small sorting datasets."

9. User Interface

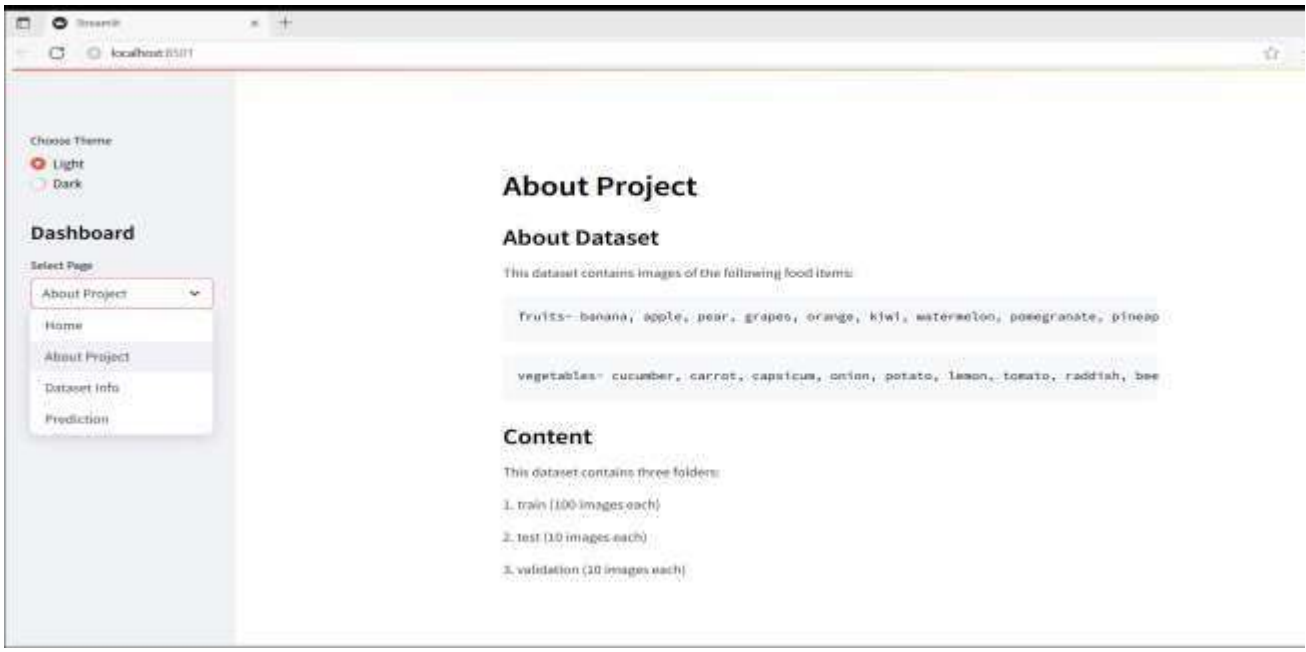
"Leveraging transfer learning for a small UI-based sorting system to identify rotten fruits and vegetables."

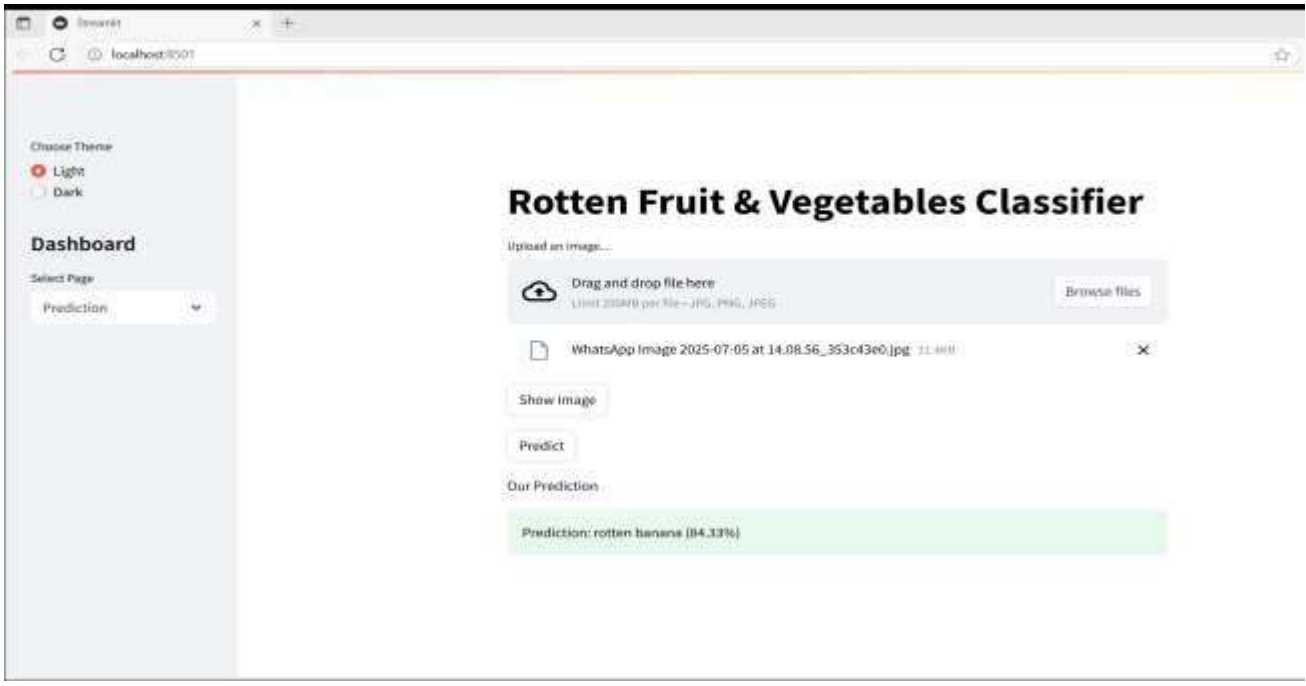
10. Testing

"Testing transfer learning for a small sorting system to detect rotten fruits and vegetables in real-time."

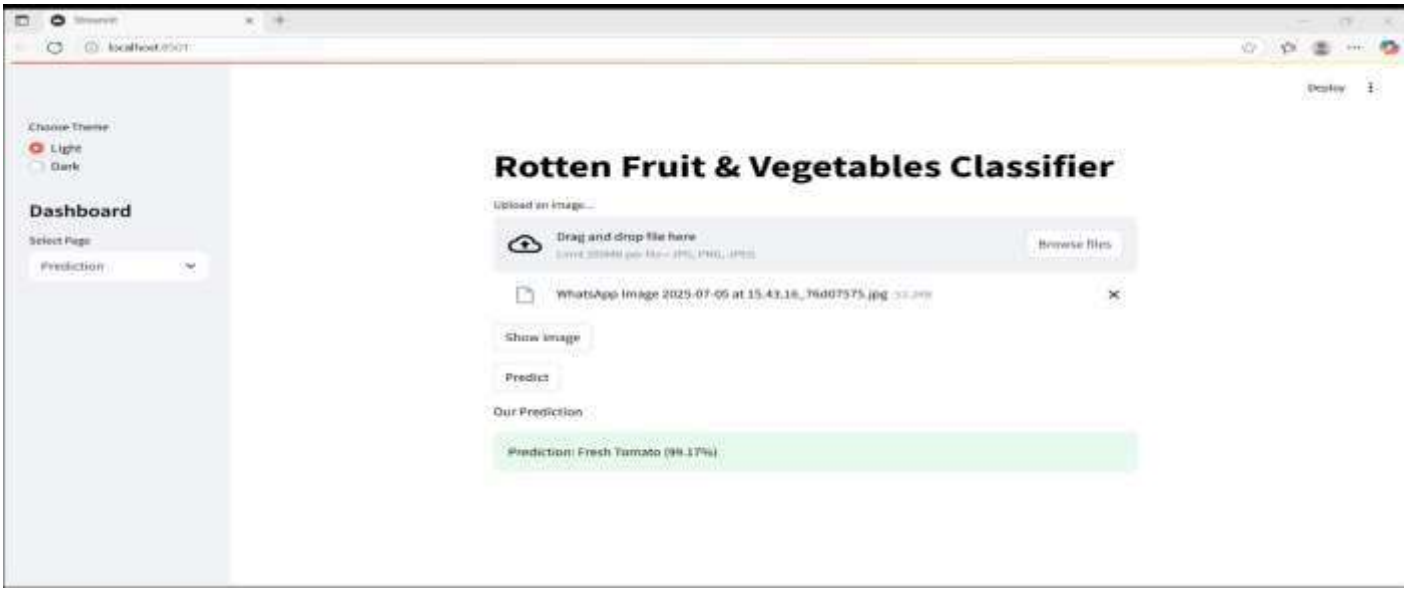
11. Screenshots or Demo

- 

- 

- 
Choose Theme
Light
Dark
Dashboard
Select Page
Prediction
Rotten Fruit & Vegetables Classifier
Upload an image...
Drag and drop file here
Browse files
WhatsApp Image 2025-07-05 at 14.08.56_353c43e0.jpg
Show Image
Predict
Our Prediction
Prediction: rotten banana (84.33%)


Choose Theme
Light
Dark
Dashboard
Select Page
Prediction
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
Predict


Choose Theme
Light
Dark
Dashboard
Select Page
Prediction
Rotten Fruit & Vegetables Classifier
Upload an image...
Drag and drop file here
Browse files
WhatsApp Image 2025-07-05 at 15.43.16_76d07575.jpg
Show Image
Predict
Our Prediction
Prediction: Fresh Tomato (99.17%)

12. Known Issues

"Known issues include limited dataset diversity, slow inference time on low-end devices, and occasional misclassification due to similar texture or color patterns in fruits and vegetables."

13. Future Enhancements

"Enhancing features in a small sorting system using transfer learning to improve accuracy in identifying rotten fruits and vegetables."

-