Final Project Proposal

Produce Quality Assessment System

1 Executive Summary

This proposal outlines the specifications for a **Produce Quality Assessment System**. The core objective is to use images of fruits and vegetables (captured via webcam or phone) to instantly classify their ripeness or quality and predict a mock shelf-life. This project provides a practical application of computer vision focused on color and texture analysis. The system uses a lightweight MobileNetV2 model for classification, stores persistent results in an SQLite database, and presents the findings through a Streamlit dashboard for visualization.

2 Project Scope and Core Requirements

The application provides real-time quality classification and shelf-life prediction from RGB images of produce and offers structured insights via the web dashboard.

Requirement Area	Feature Description
Data Processing Engine (Input)	**Produce Preprocessing (G1):** The system must handle RGB images (JPEG/PNG). Preprocessing must focus on isolating the item via basic background subtraction/masking, followed by converting the image to the **HSV or LAB color space** for standardized color feature extraction, maintaining low latency.
Classification Engine (G2/G9)	**MobileNetV2 Feature Extraction:** The system must utilize the lightweight **MobileNetV2** model (pretrained on ImageNet) for rapid feature extraction. The final classification must identify a minimum of **five** distinct quality or ripeness classes (e.g., Unripe, Perfectly Ripe, Overripe, Bruised/Damaged, Healthy Leaf).

Custom Modeling Logic (G6)	**Feature Compression & Score:** The system implements a custom, shallow **Fully Connected Layer (FCL) classifier** built from scratch. This FCL translates MobileNetV2 features into the final quality class and a corresponding numeric **Freshness Index** (0.0 to 10.0) based on color and texture integrity.
Data Persistence Logic	**SQLite Insertion:** Upon successful classification, the system inserts the sample details into the SQLite database, storing the predicted quality, a unique image hash, and mock shelf-life metrics (e.g., predicted days until spoilage).
Reporting Dashboard (G8)	**Quality Distribution View:** The Streamlit dashboard must display overall statistics, including the **Quality Class Distribution (%)** across the five ripeness categories and the average Freshness Index for the scanned items.

3 Technical Architecture and Stack

Component	Technology	Rationale
Processing Engine	Python (3.9+), Pillow, opencv-python, NumPy	Core stack for handling image files, crucial for advanced color space manipulation and basic masking (G1).
Deep Learning Model	MobileNetV2 (PyTorch/TensorFlow)	Chosen for its lightweight architecture, enabling fast inference and transfer learning suitable for quick assessments (G2).
Custom Classifier	PyTorch/TensorFlow Custom FCL	Implementation of the custom layer for specific classification and Freshness Index prediction, built to be transparent and trainable (G6).
Data Persistence (G5)	SQLite Database	Simple, embedded database for localized data storage and efficient relational querying in a classroom setting.
Reporting Dashboard (G8)	Streamlit, pandas, plotly	Allows rapid development of an interactive web interface for displaying classification metrics and historical data.

4 SQL Database Schema Design

The relational schema defines the core tables for storing produce quality metadata and classification results, using standard SQLite data types.

Table	Purpose	Key Fields
Produce_Samples	Stores metadata for all scanned produce images.	sample_id (PK), file_hash (Unique), item_name (e.g., 'Banana'), scan_date, **image_path**
Quality_Results	Stores the classification and quality scores.	result_id (PK), sample_id (FK), quality_class (e.g., 'Perfectly Ripe'), confidence, **freshness_index**
Shelf_Life_Metrics	Stores the mock, predicted shelf life features.	metric_id (PK), sample_id (FK), predicted_storage_days, **optimal_temp_C, mock_decay_rate**

5 Testing Criteria and Validation

Rigorous testing is MANDATORY to validate the system's accuracy, stability, and data integrity. The implementation team is required to test against the following criteria:

5.1 Classification Accuracy and Robustness (G2)

- **Classification Test:** System must correctly classify all five quality classes (G9) for common produce (e.g., different ripeness stages of a banana). Target accuracy: 90%.
- **Color Consistency Test**: Verify that the HSV/LAB color space conversion normalizes the colors so that the classification is resistant to minor changes in ambient lighting conditions.

5.2 Functional and Logic Testing (G6)

- **ROI Detection Logic:** Verify that the preprocessing engine correctly isolates the produce item, excluding background elements that could skew color features.

- **FCL Classifier Test:** Ensure the custom FCL layer accurately translates the MobileNetV2 features into the specific quality class and the Freshness Index.
- **Freshness Index Logic**: Verify that the system assigns a low Freshness Index (e.g., below 3.0) to items with visible mold or heavy bruising, and a score above 9.0 for optimal items.

5.3 Data Integrity and Reporting (G5, G8)

- **De-duplication Test:** Verify that attempting to ingest the same image file twice results in a unique constraint violation based on the file_hash.
- **Dashboard Calculation:** Verify that the Quality Class Distribution accurately reflects the quality_class counts in the SQL database.
- **Search Integrity:** Verify that filtering by a specific item_name only returns records for that type of produce.

6 Conclusion and Submission

This proposal confirms the readiness for development, emphasizing image preprocessing and creating a custom model for quality scoring. The architectural requirements emphasize **modularity** and **scalability**, ensuring the final product is stable and easily maintainable.

Upon completion, the entire project codebase, including all source files, database schema definition scripts, and a detailed README.md file, **must be submitted and publicly accessible via a GitHub repository**. The submission should include a clear commit history demonstrating adherence to the implementation plan.