**Project Title:**

**Smart Cashew Sorter** – *Real-time Classification & Deployment using CNN/YOLO, Streamlit, Docker, and OpenCV*

**Project Overview**

The goal of this project is to build an end-to-end computer vision system for cashew nut classification using deep learning (CNN or YOLO). The system should accept live images from a camera (via OpenCV), classify different types of cashew nuts (e.g., whole, broken, discolored), and present the result through a user-friendly Streamlit app. The final app should be containerized and deployed using Docker.

**Task-wise Specification**

**Task 1: Dataset Understanding and Preparation**

* **Description**: Explore the cashew image dataset provided. Each image belongs to one of several classes (e.g., whole, broken, discolored).
* **Deliverables**:
  + Class distribution report
  + Image samples per class
  + Image dimension analysis
* **Tools**: Pandas, matplotlib, seaborn, OpenCV
* **Duration**: 1 day

**Task 2: Exploratory Data Analysis (EDA)**

* **Description**:
  + Visualize image sample counts per class
  + Analyze lighting, orientation, and background quality
  + Compute pixel intensity histograms
* **Deliverables**:
  + Jupyter Notebook report with visuals and observations
* **Duration**: 1 day

**Task 3: Data Preprocessing & Augmentation**

* **Description**:
  + Resize and normalize images
  + Apply augmentation techniques like rotation, flip, zoom, brightness
* **Deliverables**:
  + Script for preprocessing pipeline
  + Visualization of augmented samples
* **Tools**: TensorFlow/Keras ImageDataGenerator or Albumentations
* **Duration**: 1 day

**Task 4: Model Building – CNN or YOLO**

* **Option A – CNN**:
  + Build a simple CNN model using TensorFlow/Keras or PyTorch
* **Option B – YOLO**:
  + Annotate dataset using LabelImg or Roboflow
  + Train YOLOv5 on annotated data using Ultralytics or equivalent
* **Deliverables**:
  + Model training code
  + Accuracy, precision, recall metrics
  + Confusion matrix
* **Duration**: 2 days

**Task 5: Model Testing with Live Camera**

* **Description**:
  + Integrate OpenCV to capture image/video from webcam
  + Run trained model to classify cashews in real-time
* **Deliverables**:
  + Python script that detects and classifies cashew in a camera frame
  + Bounding boxes (YOLO) or class labels (CNN)
* **Duration**: 1 day

**Task 6: Streamlit Application**

* **Description**:
  + Build a Streamlit web application with the following features:
    - Upload image or use live webcam
    - Display predicted class with confidence
    - Visual output of bounding boxes (YOLO)
* **Deliverables**:
  + app.py with UI and inference code
  + requirements.txt for dependencies
* **Duration**: 1 day

**Task 7: Dockerization and Deployment**

* **Description**:
  + Write a Dockerfile for containerizing the application
  + Ensure the container runs the app with webcam input
  + Optional: Deploy on a local server or edge device
* **Deliverables**:
  + Dockerfile
  + Docker run/test instructions
* **Duration**: 1 day

**Bonus Task (Optional)**

* **Objective**:
  + Create a basic REST API using FastAPI or Flask to accept images and return classification
  + Integrate with Streamlit for API testing
* **Duration**: 1 day

**Appendix**

**A. Sample Dataset Sources**

* Cashew image dataset (labelled) – provided separately or collected from:
  + Kaggle
  + Roboflow Open Datasets

**B. Tools and Libraries**

* Python 3.10+
* TensorFlow / PyTorch
* YOLOv5 (Ultralytics)
* OpenCV
* Streamlit
* Docker
* Albumentations
* LabelImg

**C. Expected Skills Gained**

* Deep learning (CNN/YOLO)
* Real-time computer vision
* App development with Streamlit
* Docker-based deployment
* Data augmentation & annotation
* Integration of ML with front-end