**Rice Grain Image Classification System**

This project aims to develop a smart and accurate image classification system that can identify different types of rice grains using deep learning. The system is based on Convolutional Neural Networks (CNNs) combined with transfer learning, which helps make use of the powerful features learned from large-scale image datasets. Specifically, the model is trained to classify rice grains into five categories: Karacadag, Jasmine, Ipsala, Basmati, and Arborio.

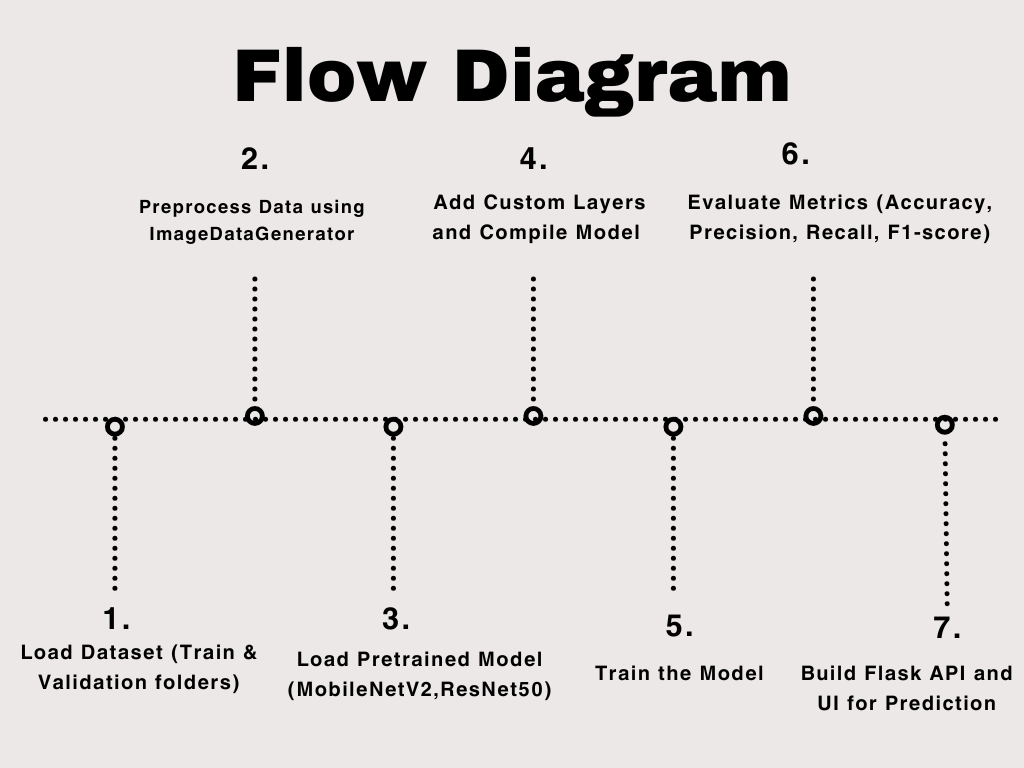
[Initially, I experimented with the VGG16 model, which is known for its deep architecture and strong feature extraction capabilities. However, due to its large size and complexity, VGG16 took a significant amount of time to train on the dataset.]

To address this, I explored more lightweight and efficient pre-trained CNN architectures such as **MobileNetV2** and **ResNet50**. Both of these models are optimized for speed and performance, making them suitable for large datasets and limited hardware resources. Out of these, **MobileNetV2** and **ResNet50** were used in this project due to their faster training times and good accuracy.

The final system not only trains the model using these pre-trained networks but also provides a user-friendly Flask-based web interface where users can upload rice grain images and receive instant predictions. This end-to-end solution integrates image processing, transfer learning, model evaluation, and real-time user interaction, making it practical for real-world applications.

**Features:**

* Transfer learning using ResNet50 and MobileNetV2 for efficient and fast feature extraction.
* Image augmentation to increase training data diversity.
* Performance metrics: Accuracy, Precision, Recall, and F1-Score.
* Flask web application with image upload and classification.
* Hyperparameter tuning for improved performance.
* Evaluation on test dataset.



**Requirements:**

* Python 3.8 or later: The programming language used for model development and API creation.
* Jupyter Notebook or VS Code: Recommended IDEs for running and editing the code.
* Keras (with TensorFlow backend): For building and training deep learning models. Make sure TensorFlow 2.9 or above is installed.
* NumPy: For numerical operations.
* scikit-learn: For model evaluation metrics like accuracy, precision, recall, and F1-score.
* Flask: To build the web-based user interface for image upload and prediction.
* Matplotlib: For plotting training curves and visualizations.

**Dependencies:**

tensorflow>=2.9

numpy

matplotlib

scikit-learn

opencv-python

flask

Pillow