GrainPalette: A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning

**Team ID :** LTVIP2025TMID46630

**Team Size :** 4

**Team Leader :** G Ravali

**Team member :** K Swarnalatha

**Team member :** M Sudeepa

**Team member :** Chavva Sowmyasri

# Objective

To build a web-based deep learning model that classifies five different rice types using image inputs. The system aims to help farmers, researchers, and gardeners identify rice types effectively.

# Technologies Used

- Python  
- TensorFlow & Keras  
- Transfer Learning (MobileNetV2/V4)  
- Flask Framework  
- HTML/CSS (Frontend)  
- NumPy, OpenCV, Matplotlib

# Dataset Source

Rice Image Dataset from Kaggle: https://www.kaggle.com/datasets/muratkokludataset/rice-image-dataset

# Rice Varieties Covered

- Arborio  
- Basmati  
- Ipsala  
- Jasmine  
- Karacadag

# Project Workflow

1. Data Collection & Preprocessing  
2. Model Building  
3. Flask Web Application  
4. Testing & Evaluation

# Use Case Scenarios

- Farmers: Identify rice seeds before cultivation  
- Researchers: Quickly classify samples during field trials  
- Home Gardeners: Understand seed packets and crop type

# Results

Model Accuracy: ~95%  
Prediction Time: < 2 seconds per image

# Screenshots Included

Model training graphs  
Flask web interface (upload & prediction result)  
Sample predictions of 5 rice types

# Conclusion

This project demonstrates the effectiveness of transfer learning for agricultural applications. By integrating a deep learning model with a user-friendly Flask web interface, we created a powerful tool for real-world rice classification. This system is scalable and can be extended to more crops or plant disease detection in the future.

# References

- TensorFlow Hub MobileNetV2 Docs  
- Kaggle Rice Dataset  
- Flask Documentation

# Phase-1: Brainstorming & Ideation

Objective:  
Identify the problem statement.  
Define the purpose and impact of the project.

Key Points:  
1. Problem Statement: Visual rice classification is a tedious and error-prone task for non-experts.  
2. Proposed Solution: An AI-driven image classifier that instantly identifies rice varieties.  
3. Target Users: Farmers, agricultural researchers, home gardeners.  
4. Expected Outcome: Quick, accurate rice variety prediction with minimal user effort.

# Phase-2: Requirement Analysis

Objective:  
Define technical and functional requirements.

Key Points:  
1. Technical Requirements: Python, TensorFlow, Keras, Flask, OpenCV.  
2. Functional Requirements: Image upload, classification, UI for prediction result.  
3. Constraints & Challenges: Model generalization, real-time inference, limited training data.

# Phase-3: Project Design

Objective:  
Create the architecture and user flow.

Key Points:  
1. System Architecture Diagram: Included in solution architecture section.  
2. User Flow: Upload image → Image processed → Model prediction → Result shown.  
3. UI/UX Considerations: Simple upload interface, clear prediction feedback.

# Phase-4: Project Planning (Agile Methodologies)

Objective:  
Break down the tasks using Agile methodologies.

Key Points:  
1. Sprint Planning: Weekly sprints with model tuning, UI development.  
2. Task Allocation: Backend (Daniel), Frontend (Anusha), Model (Kavyasri), Testing (Priyanka).  
3. Timeline & Milestones: 4-week plan from data preprocessing to deployment.

# Phase-5: Project Development

Objective:  
Code the project and integrate components.

Key Points:  
1. Technology Stack Used: Python, Flask, TensorFlow, Bootstrap, OpenCV.  
2. Development Process: Collected data, preprocessed, trained model, developed UI, integrated backend.  
3. Challenges & Fixes: Overfitting (solved via data augmentation), Flask image loading errors (fixed via preprocessing pipeline).

# Phase-6: Functional & Performance Testing

Objective:  
Ensure the project works as expected.

Key Points:  
1. Test Cases Executed: Each rice type tested individually.  
2. Bug Fixes & Improvements: Fixed UI responsiveness, improved confidence display.  
3. Final Validation: Met >95% accuracy requirement.  
4. Deployment: Flask app hosted locally for demo.

# Final Submission

1. Project Report Based on the templates  
2. Demo Video (3-5 Minutes)  
3. GitHub/Code Repository Link  
4. Presentation.