# GrainPalette – A Deep Learning Odyssey in Rice Type Classification through Transfer Learning

Team ID: LTVIP2025TMID32428

## 1. Introduction

Project Title: GrainPalette – A Deep Learning Odyssey in Rice Type Classification through Transfer Learning

Team Members:  
1. [Matha Nani] – **Frontend Developer**

## 2. Project Overview

Purpose: The purpose of GrainPalette is to build an AI-powered system capable of accurately classifying rice grain types using transfer learning techniques. This project aims to assist agricultural industries, quality control centers, and researchers in efficient rice type identification.

Features:  
• Upload rice grain images for classification  
• AI-powered prediction using a transfer learning model  
• User-friendly interface for results visualization  
• Secure authentication system  
• Database integration for storing predictions and user data

## 3. Architecture

Frontend: Built with React, providing an interactive UI for uploading images, displaying predictions, and viewing history.  
Backend: Developed using Node.js and Express.js to handle API requests, AI model integration, and authentication.  
Database: MongoDB stores user profiles, classification results, and uploaded image metadata.

## 4. Setup Instructions

Prerequisites:  
• Node.js  
• MongoDB  
• Python (for AI model training)  
• Git

Installation:  
1. Clone the repository  
2. Navigate to the project folder  
3. Install dependencies using npm install  
4. Configure environment variables (.env file)  
5. Start the servers

## 5. Folder Structure

Client: React frontend organized into components, pages, and services.  
Server: Node.js backend with routes, controllers, and services for AI model interaction.

## 6. Running the Application

Frontend: npm start (inside client folder)  
Backend: npm start (inside server folder)

## 7. API Documentation

POST /api/classify – Upload an image and get predicted rice type  
GET /api/history – Retrieve past classifications  
POST /api/register – User registration  
POST /api/login – User login

## 8. Authentication

Authentication handled via JWT tokens. Passwords stored securely using bcrypt hashing. Tokens validated for protected routes.

## 9. User Interface

The UI includes an image upload page, prediction results display, and classification history page.

## 10. Testing

Testing is conducted using Jest for backend and React Testing Library for frontend components.

## 11. Screenshots or Demo

[Insert screenshots or provide demo link]

## 12. Known Issues

• Model accuracy may vary with low-quality images  
• Large image sizes increase prediction time

## 13. Future Enhancements

• Expand dataset for more rice varieties  
• Deploy as a mobile app  
• Add real-time camera-based classification