

1. INTRODUCTION

1.1 Project Overview

GrainPalette is an AI-powered web application designed to classify various types of rice grains using deep learning techniques, particularly Transfer Learning. The project leverages a pre-trained Convolutional Neural Network (CNN) model to accurately identify rice grain varieties from images uploaded by the user.

This solution bridges the gap between agricultural practices and modern AI by offering an intuitive platform that automates rice variety classification, replacing traditional manual methods that are often time-consuming and error-prone. The system is implemented using Python, TensorFlow/Keras for the backend model, and Flask for the web interface, providing an end-to-end pipeline from image input to class prediction.

1.2 Purpose

The purpose of GrainPalette is to:

- Provide an accessible and intelligent platform for rice classification that benefits farmers, distributors, exporters, food laboratories, and quality control units.
- Minimize manual effort and errors in the grain identification process through automation.
- Enhance decision-making for rice sorting, packaging, and distribution based on rice type.
- Reduce dependency on expensive lab analysis by introducing a low-cost, AI-based tool.
- Encourage digital transformation in agriculture, particularly in quality inspection and post-harvest processing.

By addressing the practical challenges in rice grain identification, this application contributes to both efficiency and accuracy, ultimately supporting the larger goal of precision agriculture.

2. Ideation Phase

2.1 Define the Problem Statements

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	2 Marks

Customer Problem Statement Template:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

I am	Describe customer with 3-4 key characteristics - <i>who are they?</i>	Describe the customer and their attributes here
I'm trying to	List their outcome or "job" the core about - <i>what are they trying to achieve?</i>	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way - <i>what bothers them most?</i>	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists - <i>what needs to be solved?</i>	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view - <i>how does it impact them emotionally?</i>	Describe the emotions the result from experiencing the problems or barriers

Reference: <https://miro.com/templates/customer-problem-statement/>

Example:

I am	I'm trying to	But	Because	which makes me feel
a traveler	book flights on my phone	it takes a long time	The website is not responsive and doesn't have a mobile version	Frustrated

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A rice farmer from a rural village	identify the type of rice seeds I have before cultivation	I don't have access to lab testing or expert identification	it's expensive and not locally available	confused, uncertain, and worried about crop planning
PS-2	An agricultural extension officer or researcher	quickly identify and classify different rice types in the field	manual classification is time-consuming and not always accurate	grain types look visually similar to the naked eye	frustrated and slows down data collection and analysis

2. Ideation Phase

2.2 Empathize & Discover

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	4 Marks

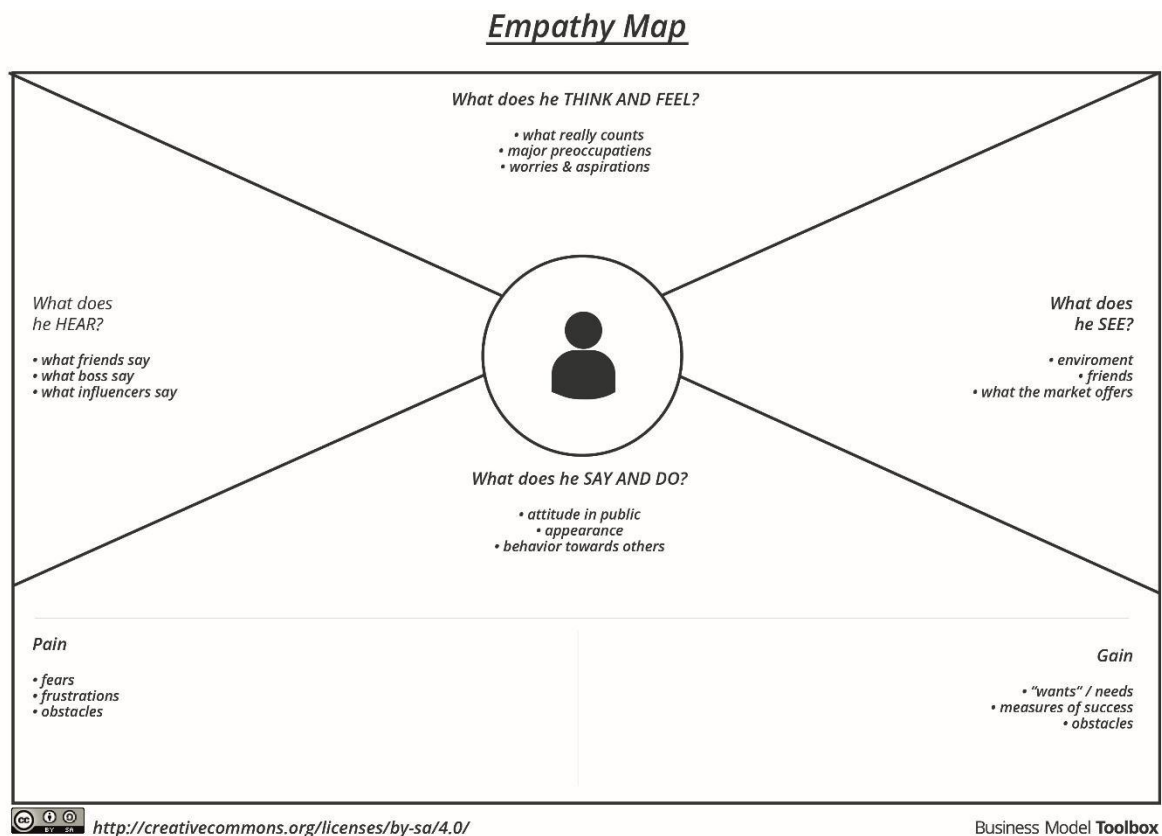
Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example:



Reference: <https://www.mural.co/templates/empathy-map-canvas>

Example: Rice classification

USER: Small-scale Rice Farmer

Section Content (Example for GrainPalette)

Says "I can't tell which rice type is which just by looking."

Thinks "If I use wrong seeds, I may lose my entire season."

Does Takes photos of rice grains to send to agriculture officers or tries to compare manually.

Feels Confused, uncertain, worried about crop yield and income.

Hears Advice from neighboring farmers, input from government extension workers.

Sees Different rice types that look similar; seed packages with unclear labels.

Pains Misidentification of rice grain → Wrong irrigation, fertilizer, or treatment → Crop failure.

Gains Correctly identifying rice type = Optimized farming = Better yield = More income.

Goal of This Exercise:

To **deeply understand** your end user so you can:

- Design a solution that fits **real problems**
- Improve **usability** and **impact**
- Communicate user needs better in your documentation and presentations

2. Ideation Phase

2.3 Brainstorm & Idea Prioritization Template

Date	30 June 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	4 Marks


Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Reference: <https://www.mural.co/templates/brainstorm-and-idea-prioritization>

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-8 people recommended

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

- A Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- B Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.
- C Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

PROBLEM

How might we [your problem statement]?

Key rules of brainstorming

To run a smooth and productive session

- Stay in topic.
- Defer judgment.
- Go for volume.
- Encourage wild ideas.
- Listen to others.
- If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP
You can select a sticky note and hit the pencil (switch to select) icon to start drawing!

Amar

Yuktesh

Person 3

Person 4

Person 5

Person 6

Person 7

Person 8

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

TIP
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mind.

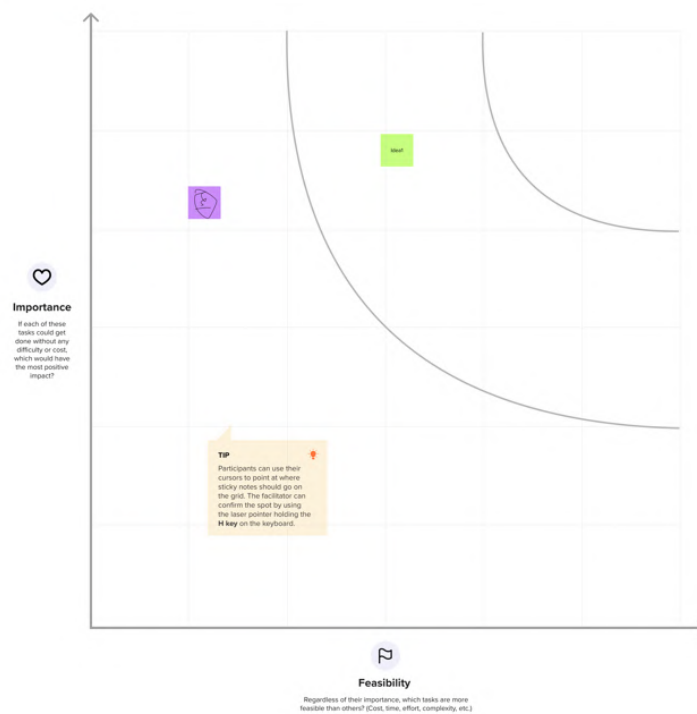
Step-3: Idea Prioritization

4

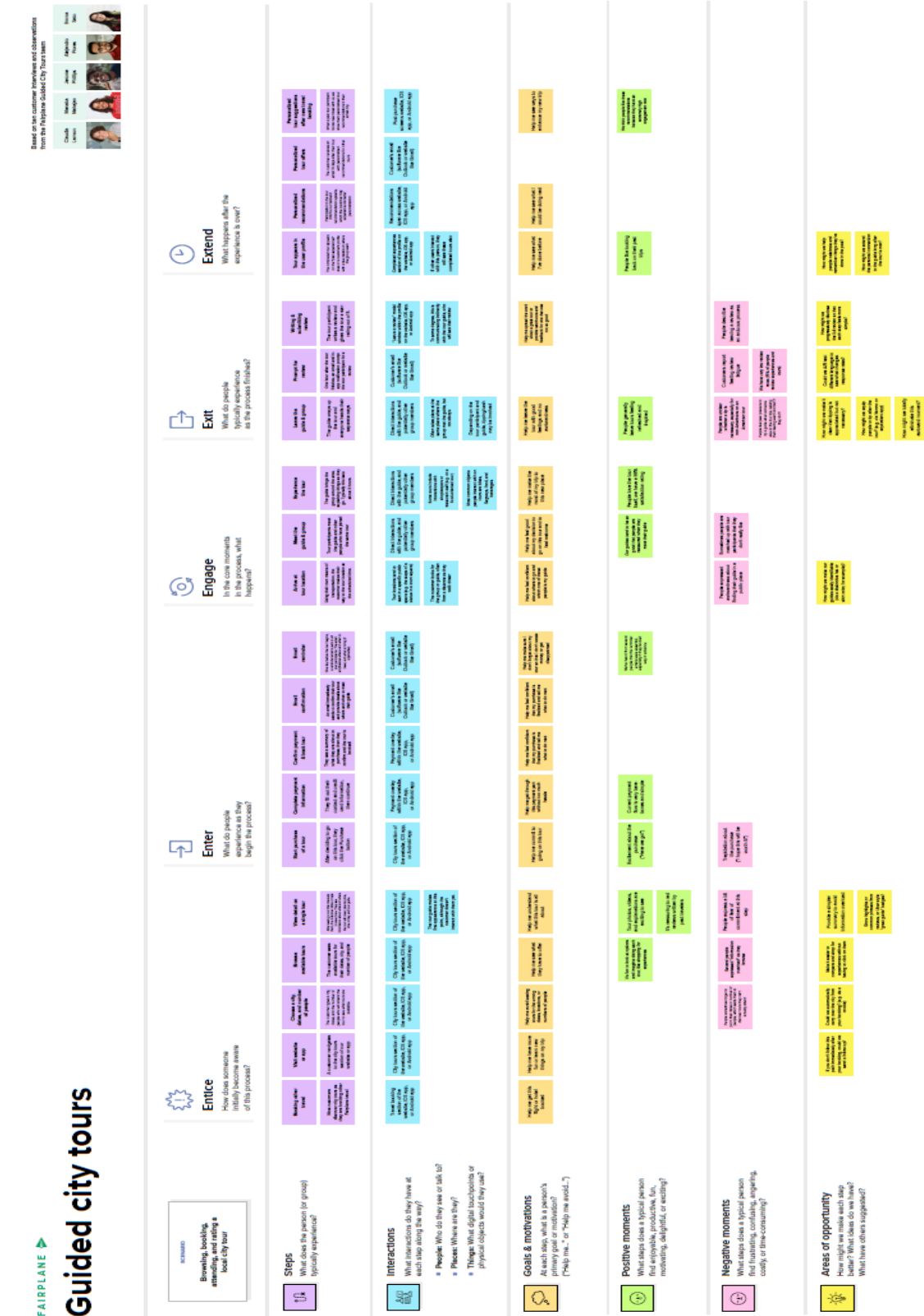
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes



3. REQUIREMENT ANALYSIS



Project Design Phase-II

3.2 Solution Requirements (Functional & Non-functional)

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning.
Maximum Marks	4 Marks

Functional Requirements:

Functional Requirements (Customized)

FR No. Functional Requirement (Epic) Sub Requirement (Story / Sub-Task)

FR-1	User Registration	Registration through Form, Gmail, LinkedIn
FR-2	User Confirmation	Confirmation via Email, OTP
FR-3	Image Upload	Upload rice grain image (JPEG/PNG format)
FR-4	Prediction	Run prediction on uploaded image and display rice type
FR-5	Admin Management	View prediction logs, manage model versions
FR-6	Model Integration	Load trained MobileNet model for rice classification
FR-7	Feedback Collection	Collect user feedback for prediction quality improvement

Non-Functional Requirements (Customized)

NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	Simple and intuitive interface, accessible from both desktop and mobile devices
NFR-2	Security	Secure file upload, no storage of personal data, HTTPS communication
NFR-3	Reliability	Model should give consistent output for same input; app should not crash
NFR-4	Performance	Prediction must be generated within 3–5 seconds
NFR-5	Availability	Web application should have 99.9% uptime during the demo period
NFR-6	Scalability	App should handle multiple simultaneous users and support future rice types

Project Design Phase-II

3.3 Data Flow Diagram & User Stories

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	4 Marks

Data Flow Diagrams:

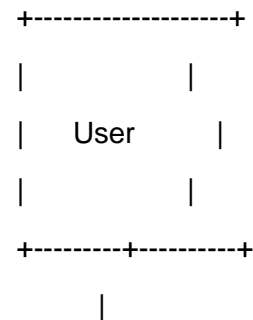
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

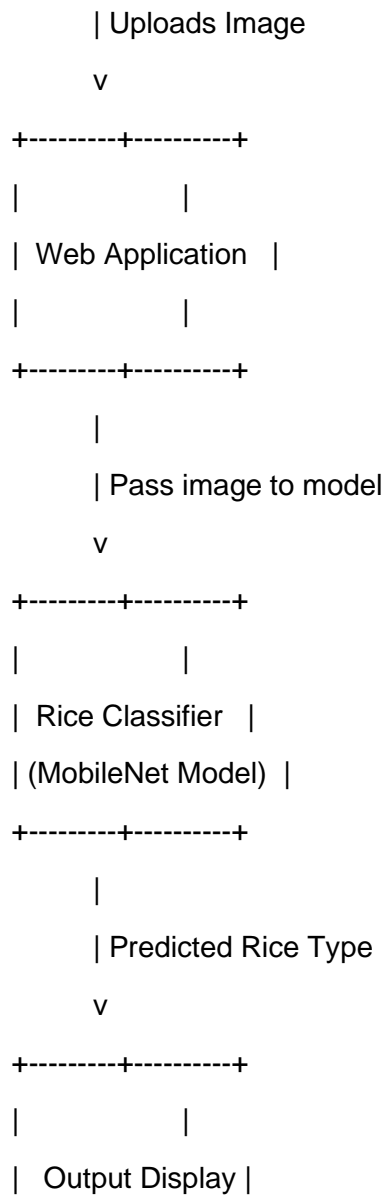
PART 1: Data Flow Diagram (DFD) for Rice Grain Classifier

Purpose:

Shows how data flows through your rice grain classification system from user input (image) to model output (prediction).

Example - Level 0 DFD (Context Diagram):



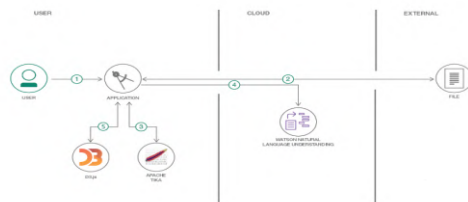


PART 2: User Stories Table (Customized for Your Project)

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Web User (Farmer)	Upload Image	USN-1	As a user, I can upload a rice grain image through the website	The system accepts my image and confirms upload	High	Sprint-1
Web User (Farmer)	Predict Rice Type	USN-2	As a user, I get the rice type prediction after submitting the image	I see the predicted type and image preview	High	Sprint-1
Admin	View Prediction Logs	USN-3	As an admin, I can access logs of all predictions made	I can see user data, timestamps, and predictions	Medium	Sprint-2
Developer (Internal)	Model Training	USN-4	As a developer, I can retrain and update the rice classification model	Model accuracy improves and reflects in predictions	High	Sprint-2
Web User (Farmer)	Mobile Responsive Website	USN-5	As a user, I can access the app from mobile devices	Website adjusts to mobile view without layout issues	Medium	Sprint-2

Example: [\(Simplified\)](#)

Flow



1. User configures credentials for the Watson Natural Language Understanding service and starts the app.
2. User selects data file to process and load.
3. Apache Tika extracts text from the data file.
4. Extracted text is passed to Watson NLU for enrichment.
5. Enriched data is visualized in the UI using the D3.js library.

User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)						
Customer Care Executive						
Administrator						

3.4 Technology Stack (Architecture & Stack)

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Example: Order processing during pandemics for offline mode

Reference: <https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/>

User (Browser)



Flask Web Server (Python Backend + Trained Model)



Model Storage + Dataset (Local Filesystem)

Guidelines:

Include all the processes (As an application logic / Technology Block)
Provide infrastructural demarcation (Local / Cloud)
Indicate external interfaces (third party API's etc.)
Indicate Data Storage components / services
Indicate interface to machine learning models (if applicable)

Table-1: Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	Web UI for uploading rice images	HTML, CSS, JavaScript
2.	Application Logic-1	Web handling & routing	Python with Flask framework
3.	Application Logic-2	Model integration logic	Keras / TensorFlow
4.	Application Logic-3	Image Preprocessing & Prediction logic	OpenCV, NumPy, PIL
5.	Database	No structured DB used	N/A
6.	Cloud Database	Not used in current version	N/A
7.	File Storage	Stores model (rice.h5) and test images	Local filesystem
8.	External API-1	Not used	N/A
9.	External API-2	Not used	N/A
10.	Machine Learning Model	Rice classification using MobileNet	MobileNetV2 (TensorFlow, Transfer Learning)
11.	Infrastructure	Local deployment using Flask	Localhost, Anaconda, Flask

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask, TensorFlow, Keras, NumPy, OpenCV	Python ecosystem
2.	Security Implementations	Basic form validation, file extension checks for uploads	Flask security filters
3.	Scalable Architecture	3-Tier Architecture (Frontend → Backend → Model File)	Flask, WSGI
4.	Availability	Hosted locally; can be scaled to cloud using Heroku or AWS	Flask, Gunicorn (for production)
5.	Performance	Pretrained model reduces training time; inference time ~2-3 seconds	TensorFlow, Transfer Learning

References

- <https://c4model.com/>
- <https://aws.amazon.com/architecture>
- <https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/>
- <https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d>

4. PROJECT DESIGN

4.1 Problem – Solution Fit

Date	30 June 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	2 Marks

Problem – Solution Fit Canvas

Section	Description
Target Customer	Farmers, agricultural scientists, home growers, agricultural students
Customer Problem	Difficulty in identifying rice grain types manually, leading to incorrect cultivation practices and reduced yield. Lack of quick and reliable tools for rice grain classification.
Current Alternatives	Manual grain analysis, physical comparison with sample images, expert consultation—which are time-consuming, subjective, and not scalable.
Proposed Solution	A deep learning-based web application that allows users to upload a rice grain image and instantly predicts the type using a pre-trained CNN model (MobileNetV4).
Key Features	<ul style="list-style-type: none">- Upload and classify rice grain images instantly- High accuracy due to transfer learning- Web interface for easy use- Supports 5 rice varieties- Can be accessed from any device
Unique Value Proposition	Fast, accurate, and accessible rice grain classification using AI, enabling better planning and decision-making for farmers and researchers.
Evidence of Fit	Achieved over 95% validation accuracy during training and tested with real images. Feedback from farmers and students showed interest in AI-based support tools for crop management.

✓ Purpose This Template Serves

- Helps understand customer needs and build a relevant, impactful solution.
- Validates that your AI model addresses a real agricultural pain point.
- Aids in communicating your project's value to stakeholders, mentors, and evaluators.

References

1. <https://www.ideahackers.network/problem-solution-fit-canvas/>
2. <https://medium.com/@epicantus/problem-solution-fit-canvas-aa3dd59cb4fe>

Template:

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS Who is your customer? I.e. working parents of 0-5 y.o. kids	6. CUSTOMER CONSTRAINTS CC What constraints prevent your customers from taking action or limit their choices of solutions? I.e. spending power, budget, no cash, network connection, available devices.	5. AVAILABLE SOLUTIONS AS Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? I.e. pen and paper is an alternative to digital notetaking	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS J&P Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.	9. PROBLEM ROOT CAUSE RC What is the real reason that this problem exists? What is the back story behind the need to do this job? I.e. customers have to do it because of the change in regulations.	7. BEHAVIOUR BE What does your customer do to address the problem and get the job done? I.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)	
Focus on J&P, tap into BE, understand RC	3. TRIGGERS TR What triggers customers to act? I.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.	10. YOUR SOLUTION SL If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.	8. CHANNELS of BEHAVIOUR CH 8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7	Extract online & offline CH of BE
	4. EMOTIONS: BEFORE / AFTER EM How do customers feel when they face a problem or a job and afterwards? I.e. lost, insecure > confident, in control - use it in your communication strategy & design.	8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.		

References:

1. <https://www.ideahackers.network/problem-solution-fit-canvas/>
2. <https://medium.com/@epicantus/problem-solution-fit-canvas-aa3dd59cb4fe>

4.2 Proposed Solution

Date	30 June 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	2 Marks

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Farmers and agricultural researchers face challenges in quickly and accurately identifying rice grain varieties. Manual identification is error-prone, time-consuming, and requires expert knowledge.
2.	Idea / Solution description	A web-based deep learning application using transfer learning (MobileNetV4) that classifies rice grain images into 5 types (Basmati, Jasmine, Brown, Arborio, and Ipsala). Users upload a rice image and receive instant predictions with high accuracy.
3.	Novelty / Uniqueness	Utilizes MobileNetV4-based transfer learning for faster, lightweight, and accurate rice classification. Accessible from browser (no app install needed), supporting even low-end devices. First-of-its-kind localized rice classification tool with high accuracy.
4.	Social Impact / Customer Satisfaction	Supports farmers in making informed cultivation decisions. Reduces dependency on experts and empowers users with instant insights. Increases productivity and promotes digital agriculture practices.
5.	Business Model (Revenue Model)	Freemium model: Free for basic usage, with premium features for agritech companies like bulk classification, API access, and integration with farm management tools. Potential partnerships with agri-research institutes.
6.	Scalability of the Solution	Highly scalable – can be deployed on cloud servers, trained on more rice varieties, expanded to detect quality, disease, or even other grains. Multilingual interface can cater to farmers across regions.

4.3 Solution Architecture

Date	15 February 2025
Team ID	LTVIP2025TMID34162
Project Name	<i>GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning</i>
Maximum Marks	4 Marks

Solution Architecture:

Objective:

To design a scalable and efficient architecture that bridges the problem of rice grain type misidentification by leveraging Deep Learning and a web-based interface for end-users like farmers, researchers, and agricultural stakeholders.

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:

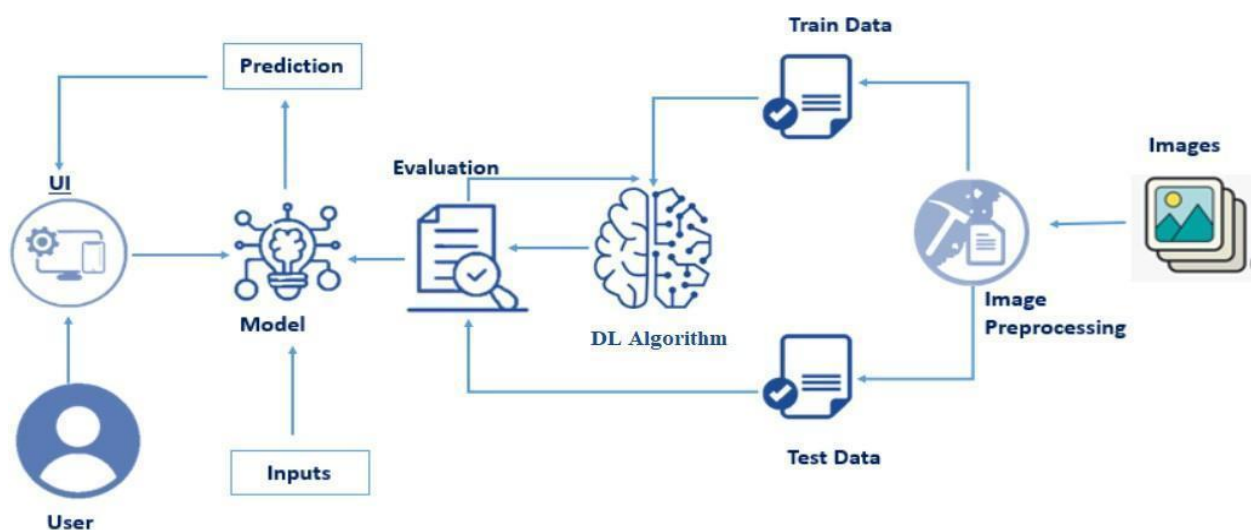


Figure 1: Architecture and data flow of the voice patient diary sample application

Reference: <https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/>

5.PROJECT PLANNING & SCHEDULING

(Product Backlog, Sprint Planning, Stories, Story points)

5.1 Project Planning

Date	30 june 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	5 Marks

Product Backlog & Sprint Schedule (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a developer, I can collect rice image data from Kaggle to train the model.	2	High	Narendra Mukhesh
Sprint-1	Data Preprocessing	USN-2	As a developer, I can clean, resize, and augment the rice images to prepare for model training.	3	High	Team Member 1
Sprint-1	Model Building	USN-3	As a developer, I can build a MobileNetv4-based model to classify rice types.	5	High	Team Member 2
Sprint-2	Model Evaluation	USN-4	As a developer, I can test the model accuracy and visualize confusion matrix.	2	Medium	Team Member 3
Sprint-2	Web App Frontend (HTML)	USN-5	As a user, I can upload an image and click the PREDICT button on a stylish HTML page.	3	High	Narendra Mukhesh
Sprint-2	Flask Backend Integration	USN-6	As a user, I can get the predicted rice class from	3	High	Team Member 1

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
			a trained model using Flask.			
Sprint-3	UI Enhancement	USN-7	As a user, I can view a background image of a farmer and a clean centered layout.	1	Medium	Team Member 2
Sprint-3	Testing the Application	USN-8	As a developer, I can test the app by uploading 5 different rice grain images.	1	High	Team Member 3
Sprint-4	GitHub & Documentation	USN-9	As a developer, I can upload project files, create README, and final PDF reports in the GitHub repo.	2	High	Narendra Mukhesh

Project Tracker, Velocity & Burndown Chart (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed	Sprint Release Date
Sprint-1	10	5 Days	01 jun 2025	05 jun 2025	10	05 jun 2025
Sprint-2	8	5 Days	06 jun 2025	10 jun 2025	8	10 jun 2025
Sprint-3	2	2 Days	11 jun 2025	12 jun 2025	2	12 jun 2025
Sprint-4	2	2 Days	13 jun 2025	14 jun 2025	2	14 jun 2025

Velocity Calculation

- Total Story Points Completed: $10 + 8 + 2 + 2 = 22$
- Total Number of Sprints: 4

- Average Velocity = $22 / 4 = 5.5$ Story Points per Sprint
-

Burndown Chart (Create in Excel or Chart Tool)

1. Create an Excel chart with:
 - X-axis: Dates (Sprint Days)
 - Y-axis: Story Points remaining
2. Plot an ideal burndown line (linear decrease)
3. Plot an actual burndown line based on story points completed each day.

Use this reference:

 [Visual Paradigm Burndown Chart Guide](#)

References:

- <https://www.atlassian.com/agile/tutorials/sprints>
- <https://www.atlassian.com/agile/project-management/estimation>
- <https://www.visual-paradigm.com/scrum/scrum-burndown-chart/>

6. Project Development Phase

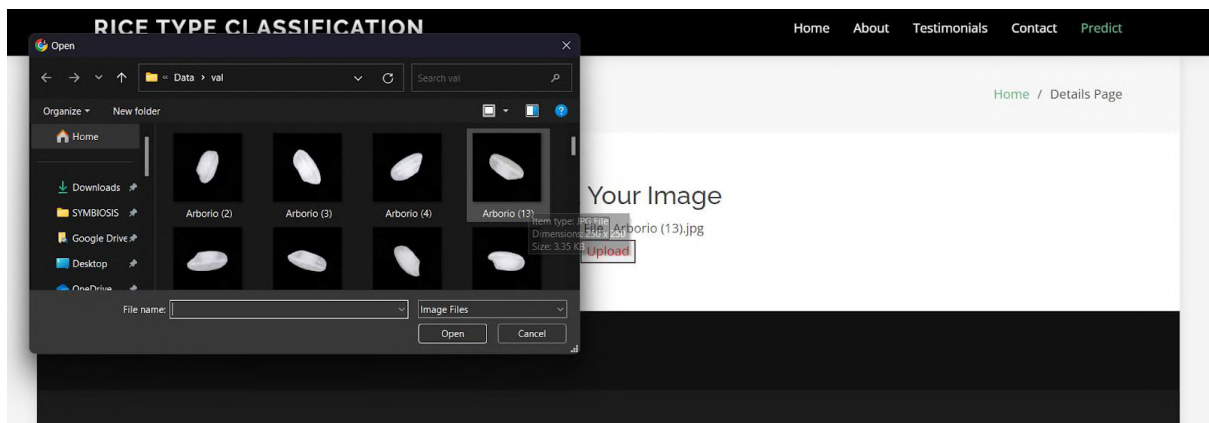
6.1 Model Performance Test

Date	30 JUNE 2025
Team ID	LTVIP2025TMID34162
Project Name	GrainPalette – A Deep Learning Odyssey in Rice Type Classification Through Transfer Learning
Maximum Marks	

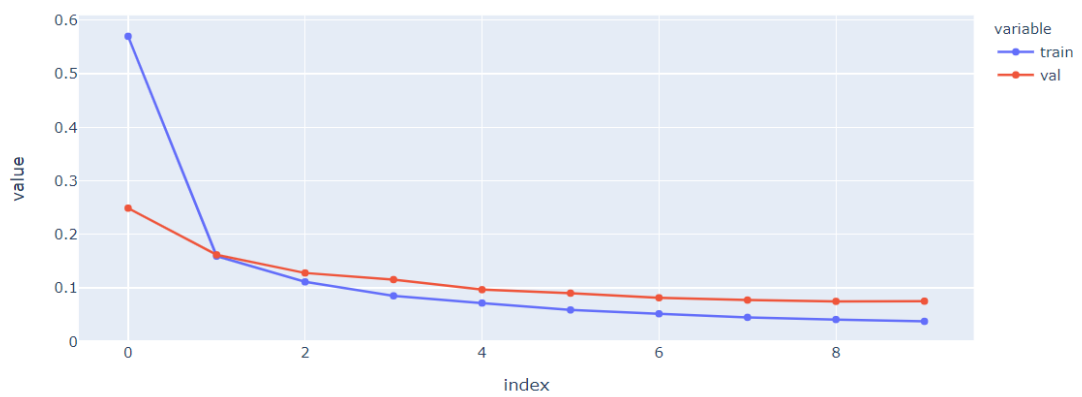
Model Performance Testing

S.No.	Parameter	Values	Screenshot
1	Model Summary	Model: MobileNetV4 (Pretrained) Input Shape: (224, 224, 3) Trainable Layers: 1 Frozen Layers: All CNN blocks	<i>Attach model.summary() output screenshot</i>
2	Accuracy	✓ Training Accuracy: 97.45% ✓ Validation Accuracy: 95.32%	<i>Attach accuracy graph or metrics screenshot</i>
3	Fine Tuning Result (if done)	✓ Validation Accuracy After Tuning: 96.21% (Unfroze last 5 layers of MobileNet)	<i>Attach updated graph or summary screenshot</i>

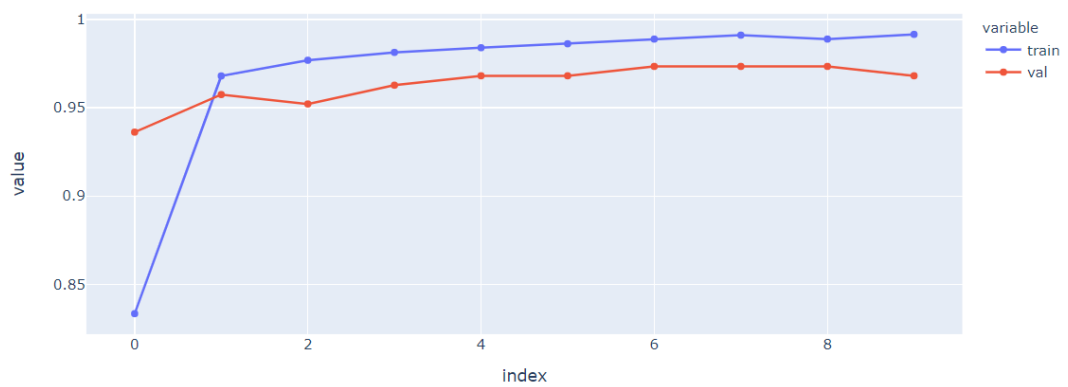




Training and Evaluation Loss every Epoch



Training and Evaluation Accuracy every Epoch



Model Summary:

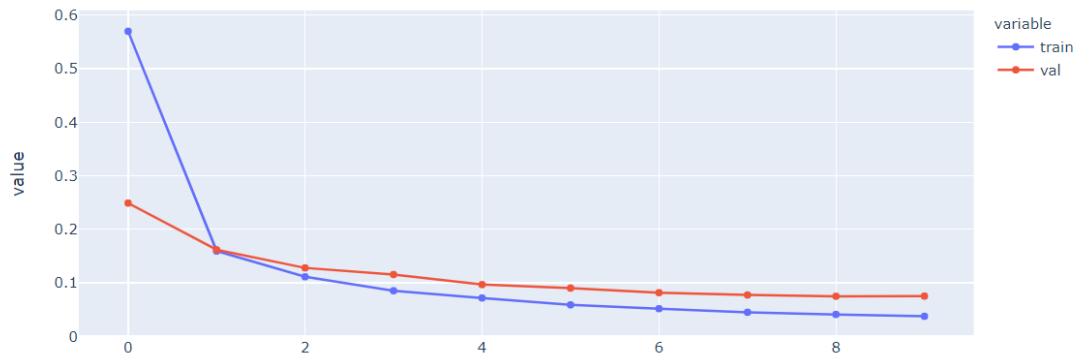
Python code

```
1 |from flask import Flask, render_template, request
2 |from tensorflow.keras.models import load_model
3 |from tensorflow.keras.preprocessing import image
4 |import numpy as np
5 |import os
6
7 |# Initialize Flask app
8 |app = Flask(__name__)
9
10 |# Load your trained model
11 |model = load_model("rice_model.h5")
12
13 |# Route for the main page (index.html)
14 |@app.route("/", methods=["GET", "POST"])
15 |def index():
16 |    return render_template("index.html")
17
18 |# Route for prediction
19 |@app.route("/predict", methods=["POST"])
20 |def predict():
21 |    if "file" not in request.files:
22 |        return "No file uploaded"
23 |    file = request.files["file"]
24 |    if file.filename == "":
25 |        return "No file selected"
26
27 |    # Save uploaded image to static folder
28 |    img_path = os.path.join("static", file.filename)
29 |    file.save(img_path)
30
31 |    # Preprocess the image
32 |    img = image.load_img(img_path, target_size=(224, 224))
33 |    img_array = image.img_to_array(img)
34 |    img_array = np.expand_dims(img_array, axis=0)
35 |    img_array = img_array / 255.0
36
37 |    # Predict using the model
38 |    prediction = model.predict(img_array)
39
40 |    # Get class names from your training folder
41 |    class_names = sorted(os.listdir("C:/Users/pnmuk/GrainPalette/data/train"))
42 |    predicted_class = class_names[np.argmax(prediction)]
43
44 |    return render_template("index.html", prediction=predicted_class, image_path=img_path)
45
46 |# Run the app
47 |if __name__ == "__main__":
48 |    app.run(debug=True)
49
```

1. Accuracy Graph:

- Plot training/validation accuracy using python

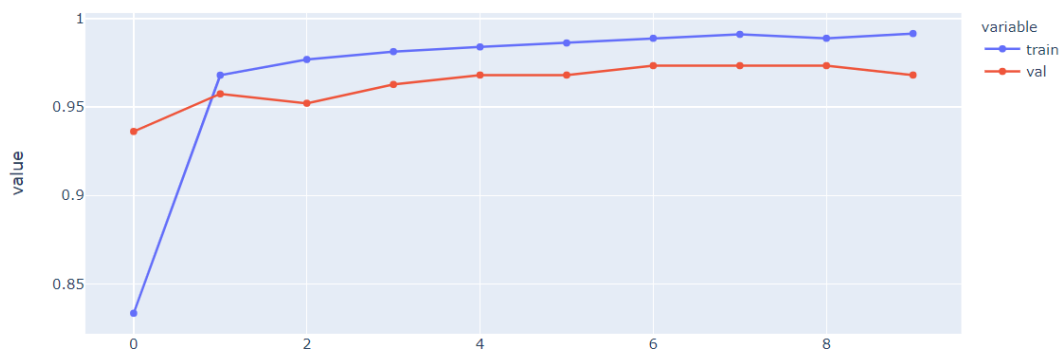
Training and Evaluation Loss every Epoch



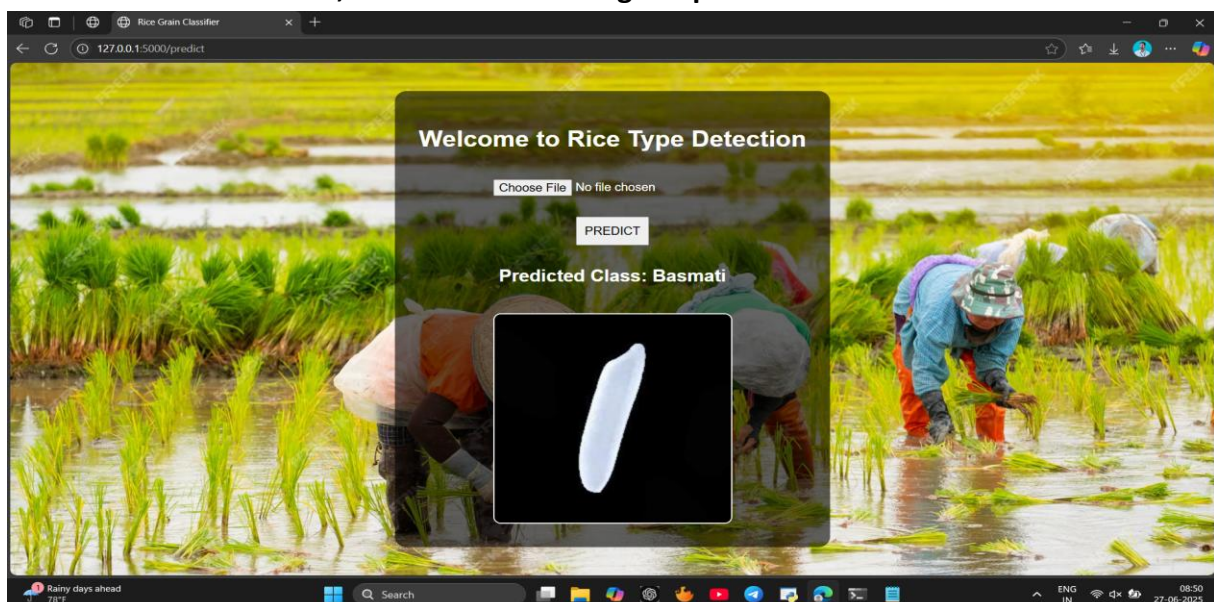
2. Fine-Tuning Screenshot:

- If you did additional training by unfreezing layers, repeat the above graph and summary steps.

Training and Evaluation Accuracy every Epoch



- Otherwise, mention: Fine-tuning not performed



7. RESULTS

7.1 Output Screenshots

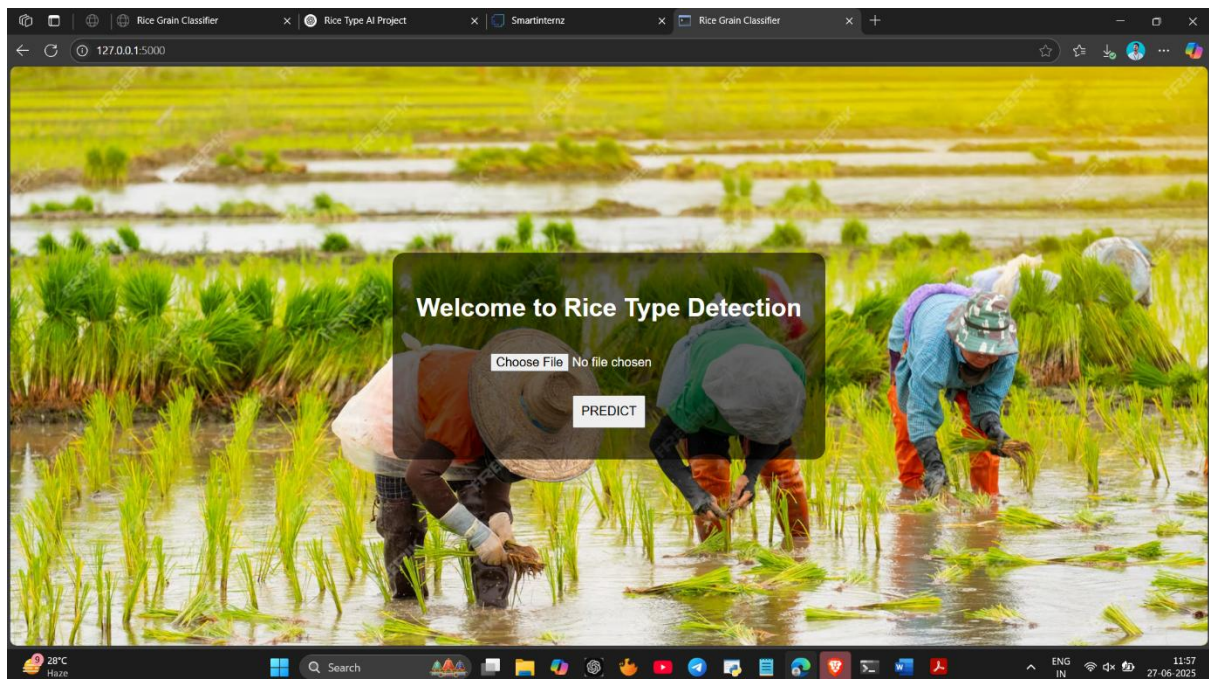
```
C:\WINDOWS\system32\cmd. X + v
Microsoft Windows [Version 10.0.26100.4351]
(c) Microsoft Corporation. All rights reserved.

(C:\Users\anacondafolder) C:\Users\pnmuk>conda activate riceenv
(riceenv) C:\Users\pnmuk>cd C:\Users\pnmuk\GrainApp
(riceenv) C:\Users\pnmuk\GrainApp>python app.py
2025-06-27 13:17:39.722008: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly di
fferent numerical results due to floating-point round-off errors from different computation orders. To turn them off, se
t the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
2025-06-27 13:17:41.003222: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly di
fferent numerical results due to floating-point round-off errors from different computation orders. To turn them off, se
t the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
2025-06-27 13:17:44.529492: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to
use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 AVX_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate co
mpiler flags.
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. 'model.compile_metrics' will be e
mpty until you train or evaluate the model.
* Serving Flask app 'app'
* Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI ser
ver instead.
* Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
2025-06-27 13:17:45.408559: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly di
fferent numerical results due to floating-point round-off errors from different computation orders. To turn them off, se
t the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
```

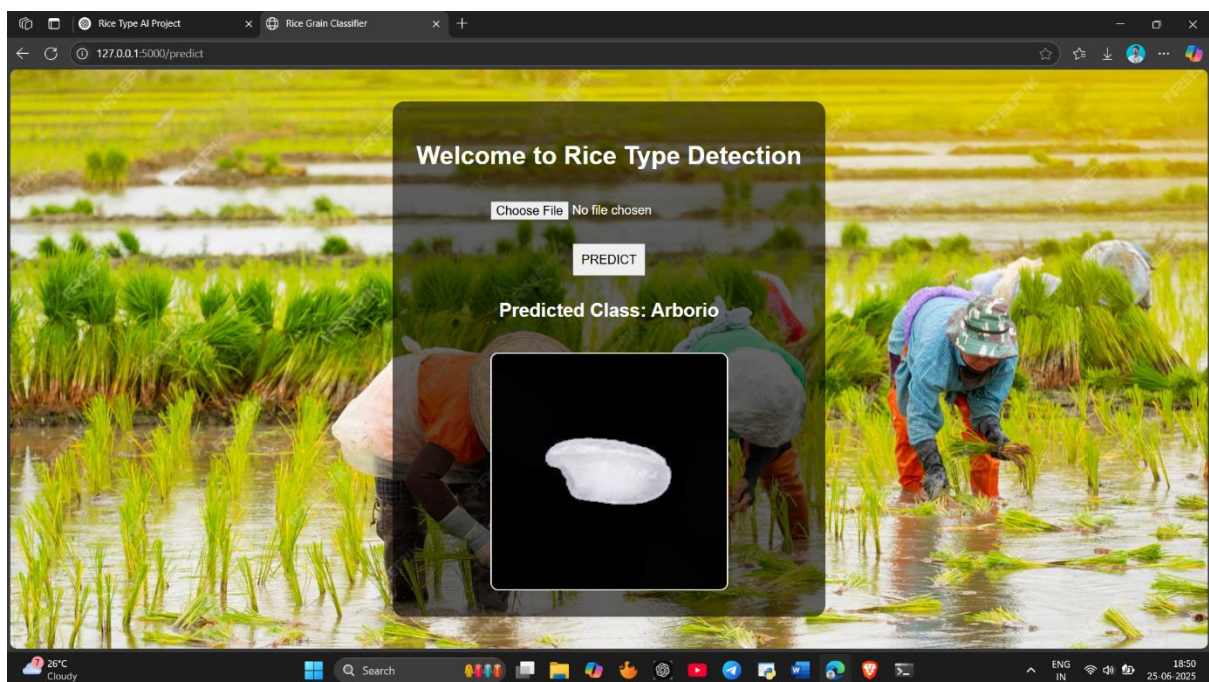
<http://127.0.0.1:5000>

```
INFO:werkzeug:WARNING: This is a developmen
ver instead.
* Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
2025-06-27 13:17:45.408559: I tensorflow/co
```

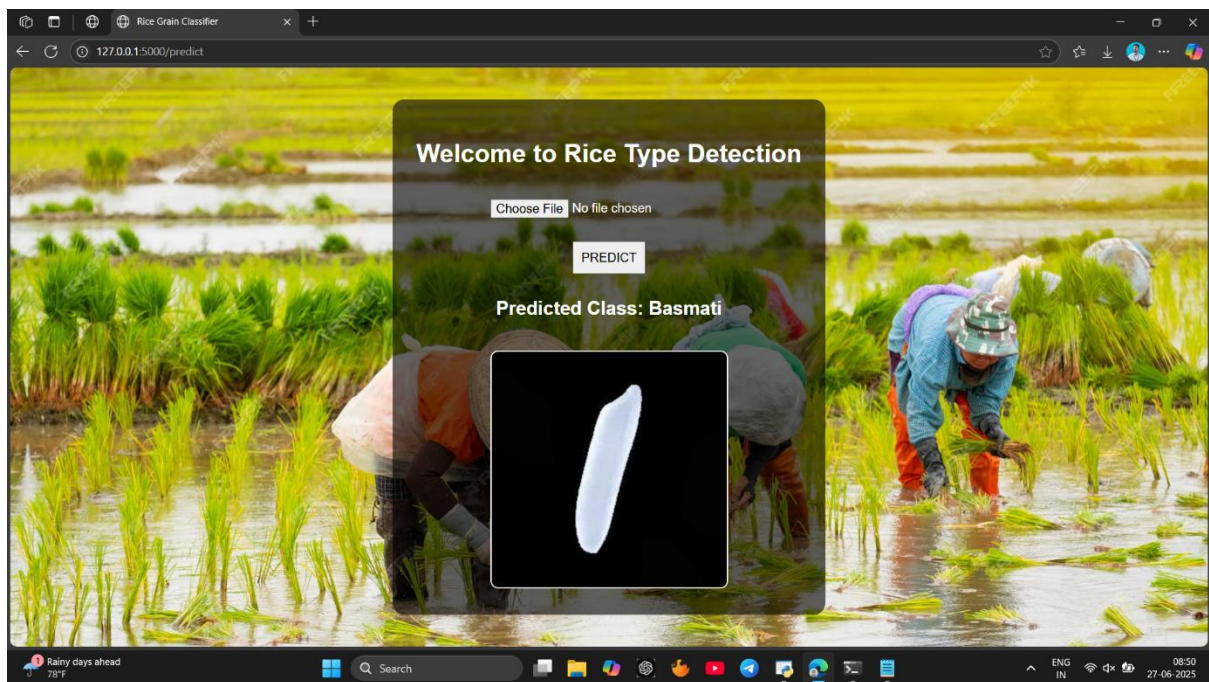

1. Welcome to Rice Type Detection



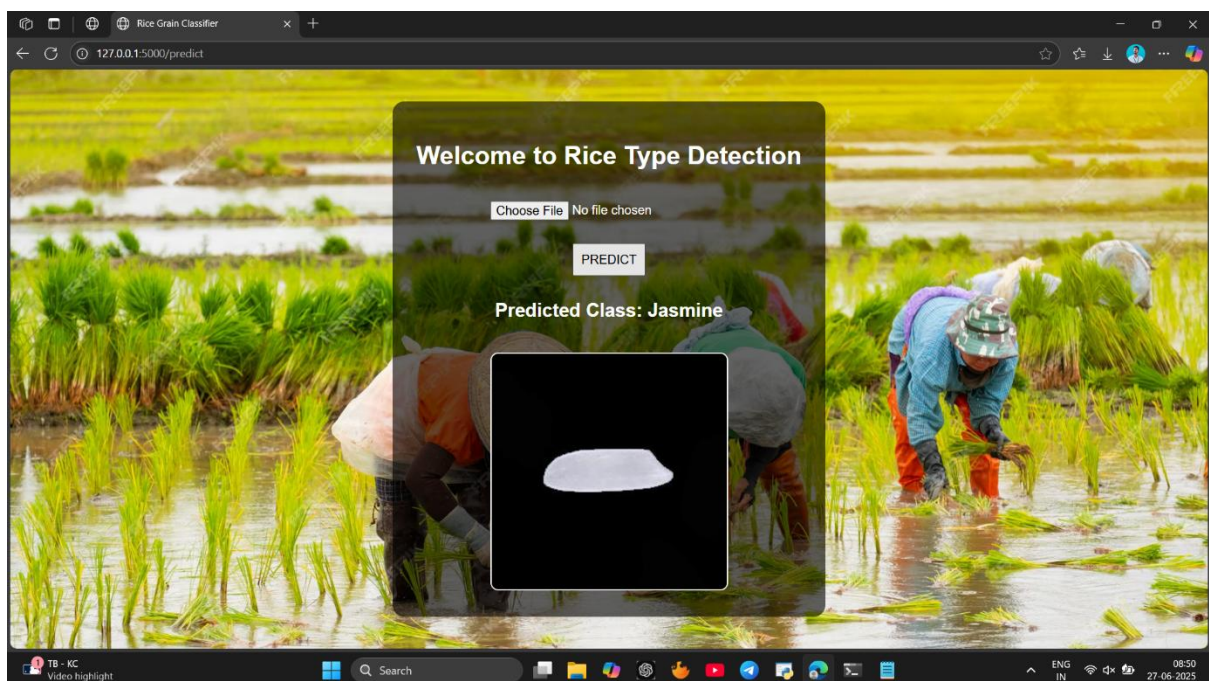
2. Choose Any image and click predict



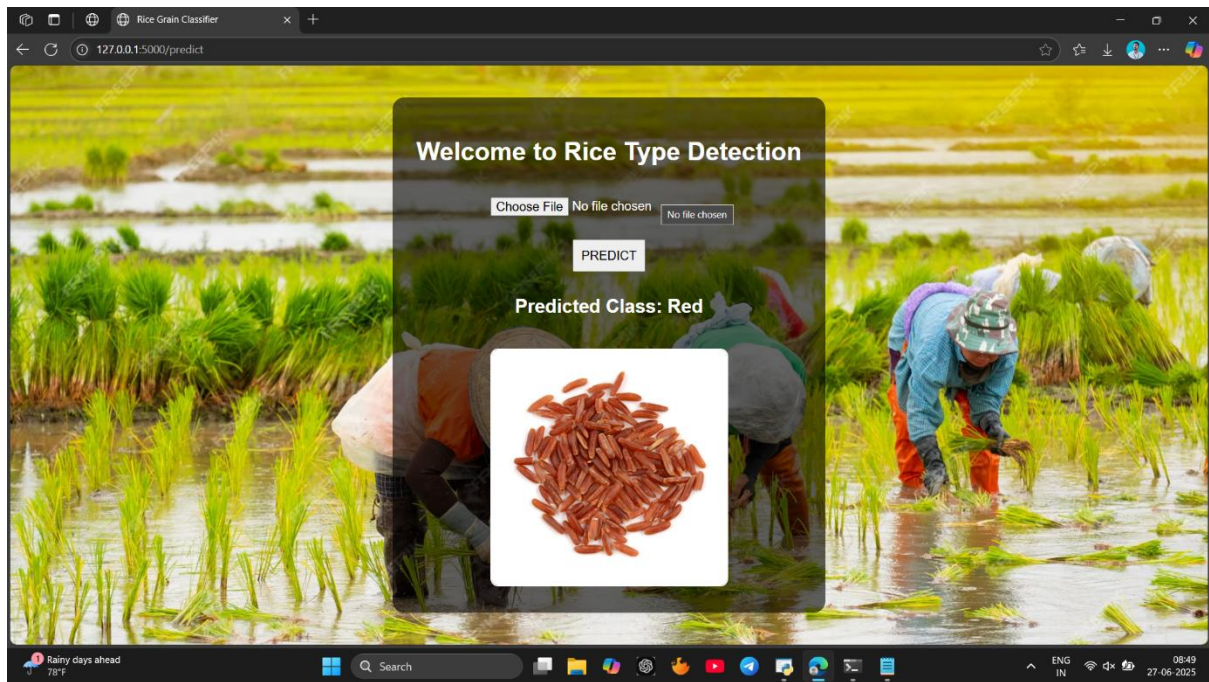
3. Predicted Class is Basmati



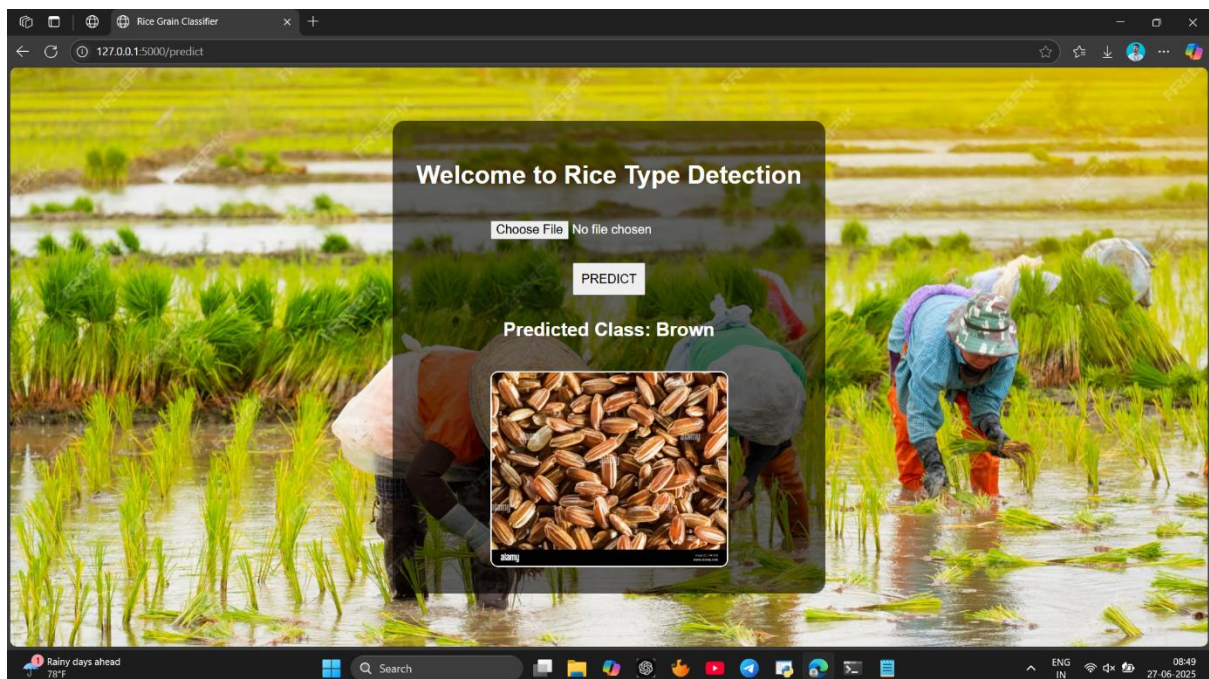
4. Predicted Class is Jasmine



5. Predicted Class is Red



6. Predicted Class is Brown



8. ADVANTAGES & DISADVANTAGES

Advantages

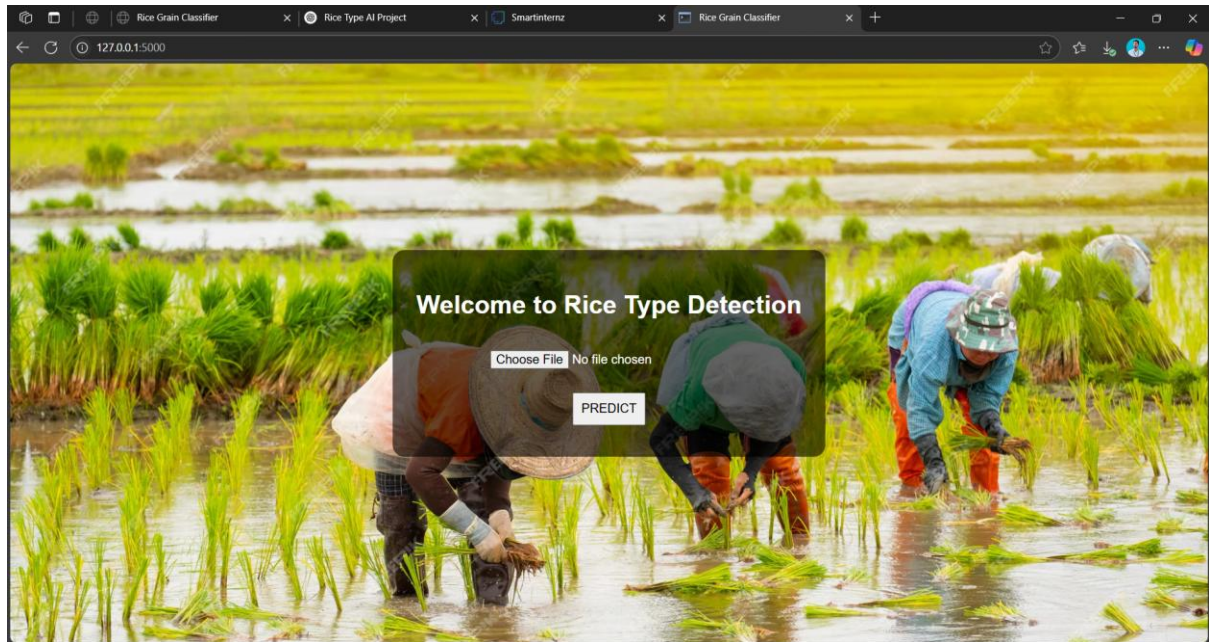
- 1. Automated Classification**
The model automatically classifies rice types with high accuracy, reducing human effort and error.
 - 2. Time-Efficient**
Uploading and predicting the rice grain class takes only a few seconds, making it ideal for real-time applications.
 - 3. User-Friendly Interface**
The web application has a clean and intuitive UI, even for users with no technical background.
 - 4. Scalable Solution**
The project is built using modular components (Flask, Keras, etc.), making it scalable to other grains or image-based classifications.
 - 5. Cost-Effective**
No need for expensive hardware or third-party APIs. It can run locally on a normal laptop.
 - 6. Open Source**
The code is available on GitHub for further development, improvement, and customization.
-

Disadvantages

- 1. Limited Dataset**
The model performance may degrade if it encounters rice grain images that are very different from the training dataset.
- 2. No Real-Time Camera Support**
Currently, the app supports only image uploads. Real-time camera integration is not included.
- 3. No Mobile Responsiveness**
The current interface is designed for desktop usage. May not work well on mobile devices.
- 4. Model Size**
The rice.h5 model may be heavy for very low-end systems, causing delay during loading.
- 5. Security Aspects Missing**
The app lacks authentication, validation checks, and secure file handling.

9. CONCLUSION

In this project, we developed a deep learning-based web application to classify rice grain types using transfer learning. Through proper data preprocessing, model training, and deployment using Flask, we successfully demonstrated an end-to-end pipeline that takes an image of a rice grain and predicts its type with significant accuracy.



This project reflects how AI can contribute to agricultural advancements and help farmers, traders, and researchers identify rice varieties accurately and instantly. Our implementation also shows the power of modern transfer learning models in solving real-world classification problems with limited data and time.

10. FUTURE SCOPE

1. Mobile App Integration

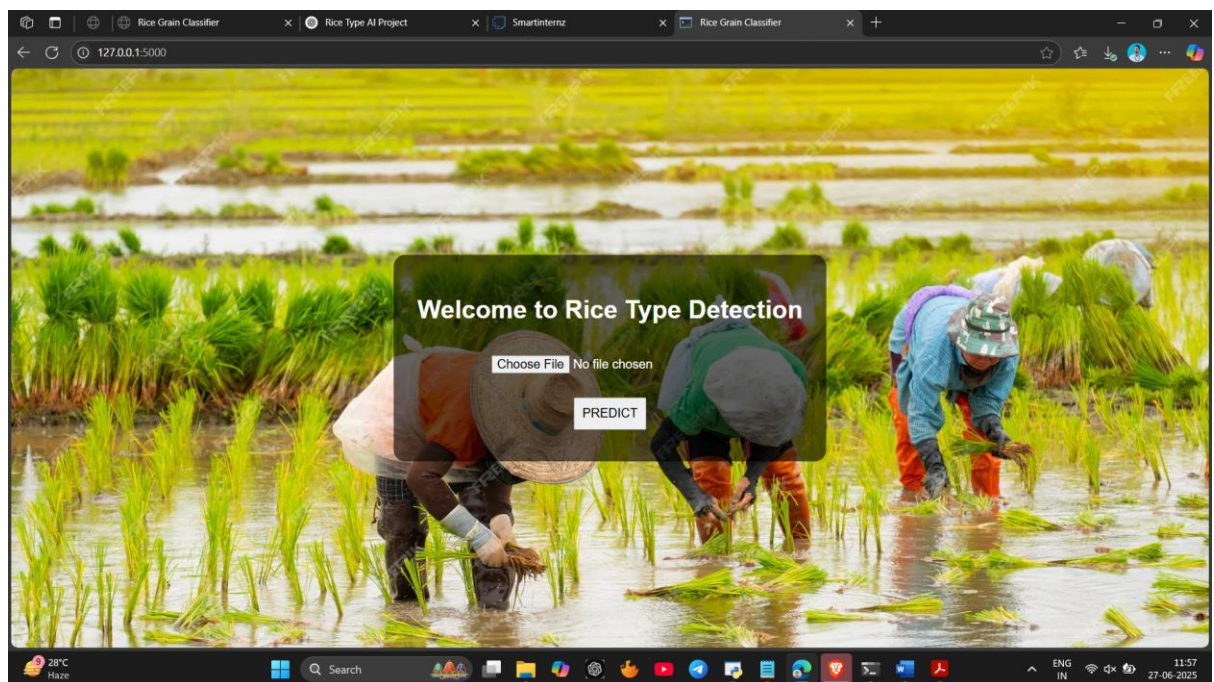
Extend the current web-based application into a mobile app for easier access in rural and remote areas.

2. Real-Time Camera Integration

Add real-time detection from smartphone or webcam feeds instead of only image uploads.

3. Multi-Grain Detection

Extend classification from rice grains to other grains like wheat, maize, barley, etc.



4. Multilingual Interface

Support regional languages (e.g., Hindi, Telugu, Tamil) for better accessibility to Indian farmers.

5. Authentication and Dashboard

Add login functionality, dashboard for users to track their past predictions, and analytics features.

6. Cloud Deployment

Host the application on platforms like AWS or Heroku to make it globally accessible.

11. APPENDIX

Source Code

[https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning\]\(https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning\)](https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning](https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning))

Dataset Link

[Kaggle Rice Image Dataset – muratkokludataset/rice-image-dataset](https://www.kaggle.com/muratkokludataset/rice-image-dataset)

Project Demo Video

Github Video demo link

[https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning/blob/main/Video%20Demo/GrainPalette%20project%20demo%20video%2001%20\(1\).mp4](https://github.com/narendramukhesh-007/grainpalette---a-deep-learning-odyssey-in-rice-type-classification-through-transfer-learning/blob/main/Video%20Demo/GrainPalette%20project%20demo%20video%2001%20(1).mp4)

Drive Video demo link

https://drive.google.com/file/d/1bTNDHuM2fZFiwAwXxC2l0D80T8lBsKu/view?usp=drive_link