**PROJECT REPORT**

**1. INTRODUCTION**

**1.1 Project Overview**

The project **"GrainPalette - A Deep Learning Odyssey in Rice Type Classification through Transfer Learning"** focuses on accurately classifying different types of rice grains using deep learning techniques. A rice grain image dataset was used, containing multiple rice varieties. The dataset was preprocessed and augmented to improve model performance. A pre-trained Convolutional Neural Network (CNN) model was fine-tuned using **Transfer Learning** techniques for better accuracy with limited data. Popular architectures like **VGG16** and **ResNet50** were explored and compared. Model evaluation was done using metrics like **accuracy**, **precision**, **recall**, and **F1-score**. The final model achieved high classification accuracy, showing the effectiveness of deep learning for agricultural applications.

**1.2 Purpose**

* To develop an automated system for accurately classifying different types of rice grains using image data.
* To reduce manual effort, time, and human error in traditional rice grain classification methods.
* To apply **Transfer Learning** techniques for effective model training with a small dataset.
* To explore and compare the performance of popular CNN architectures like **VGG16** and **ResNet50**.
* To improve the accuracy and reliability of rice type identification for agricultural and industrial use.
* To demonstrate the power of deep learning in solving real-world classification problems in the food industry.
* To build a scalable and efficient model that can be used for future grain classification tasks.

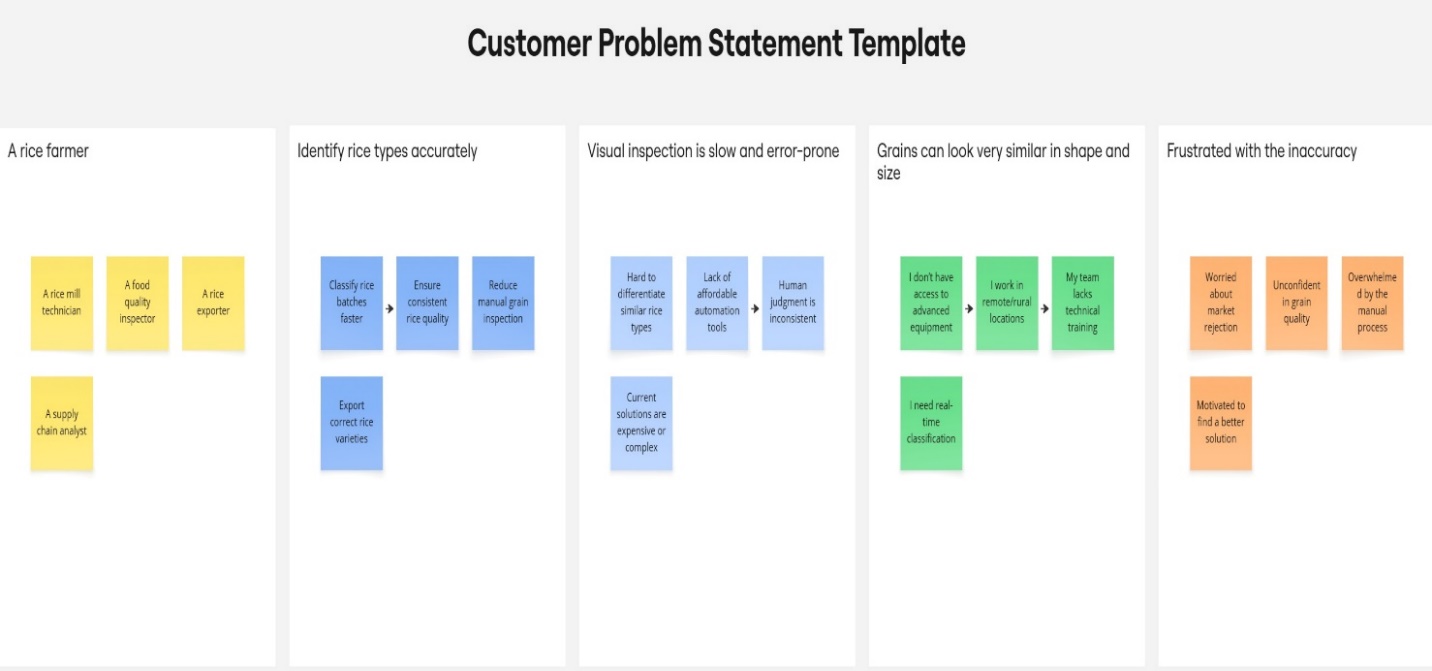
**2. IDEATION PHASE**

**2.1 Problem Statement**

**Customer Problem Statement:**

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.



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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| PS-1 | A rice farmer | Identify rice types in my harvest accurately | I rely on manual inspection | Its hard to tell types apart visually | Unsure, frustrated, and worried about selling low-quality rice |
| PS-2 | A rice mill technician | Quickly classify rice batches during processing | I don’t have time for detailed checking | Manual methods are slow and error-prone | Pressured,inefficient,and stressed |

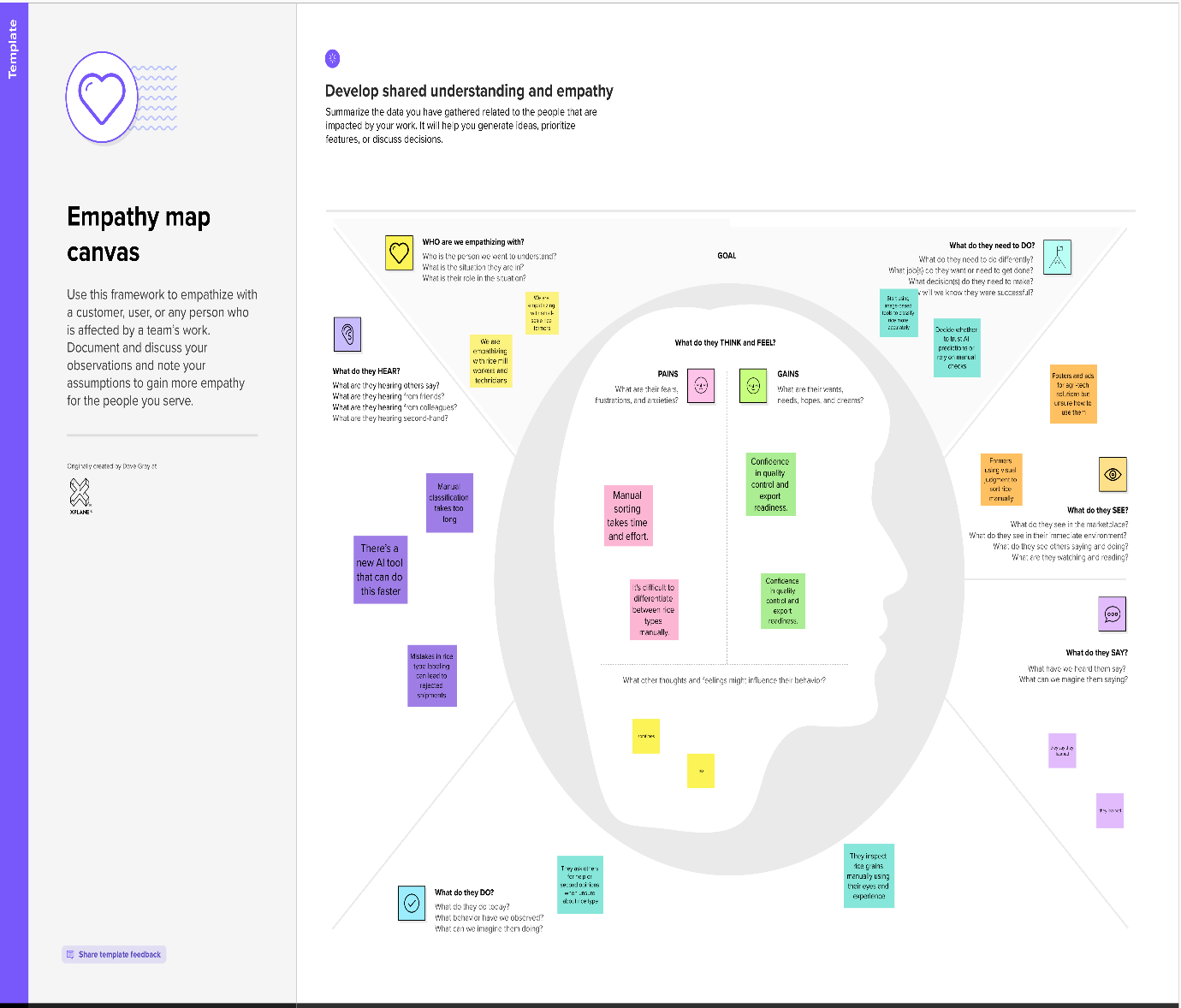
**2.2 Empathy Map Canvas**

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 helps 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.

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

****

**2.3 Brainstorming**

**Brainstorm & Idea Prioritization:**

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.

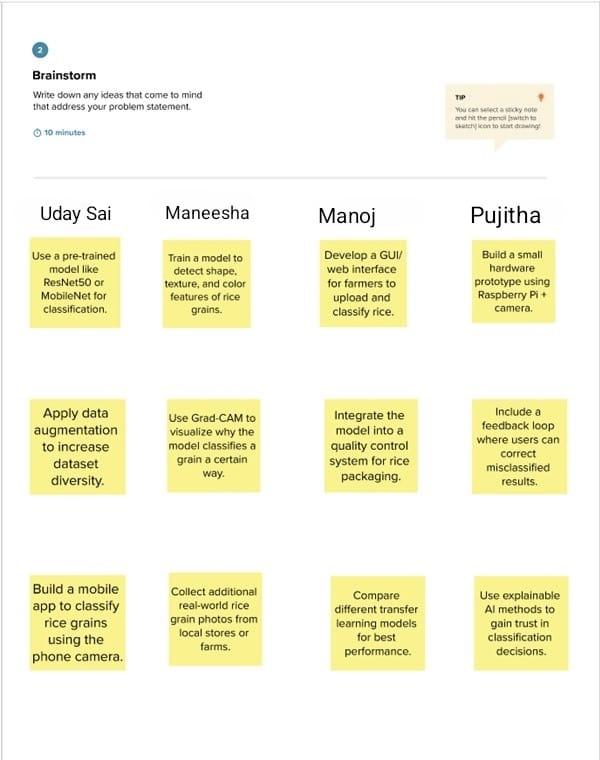
Use this

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

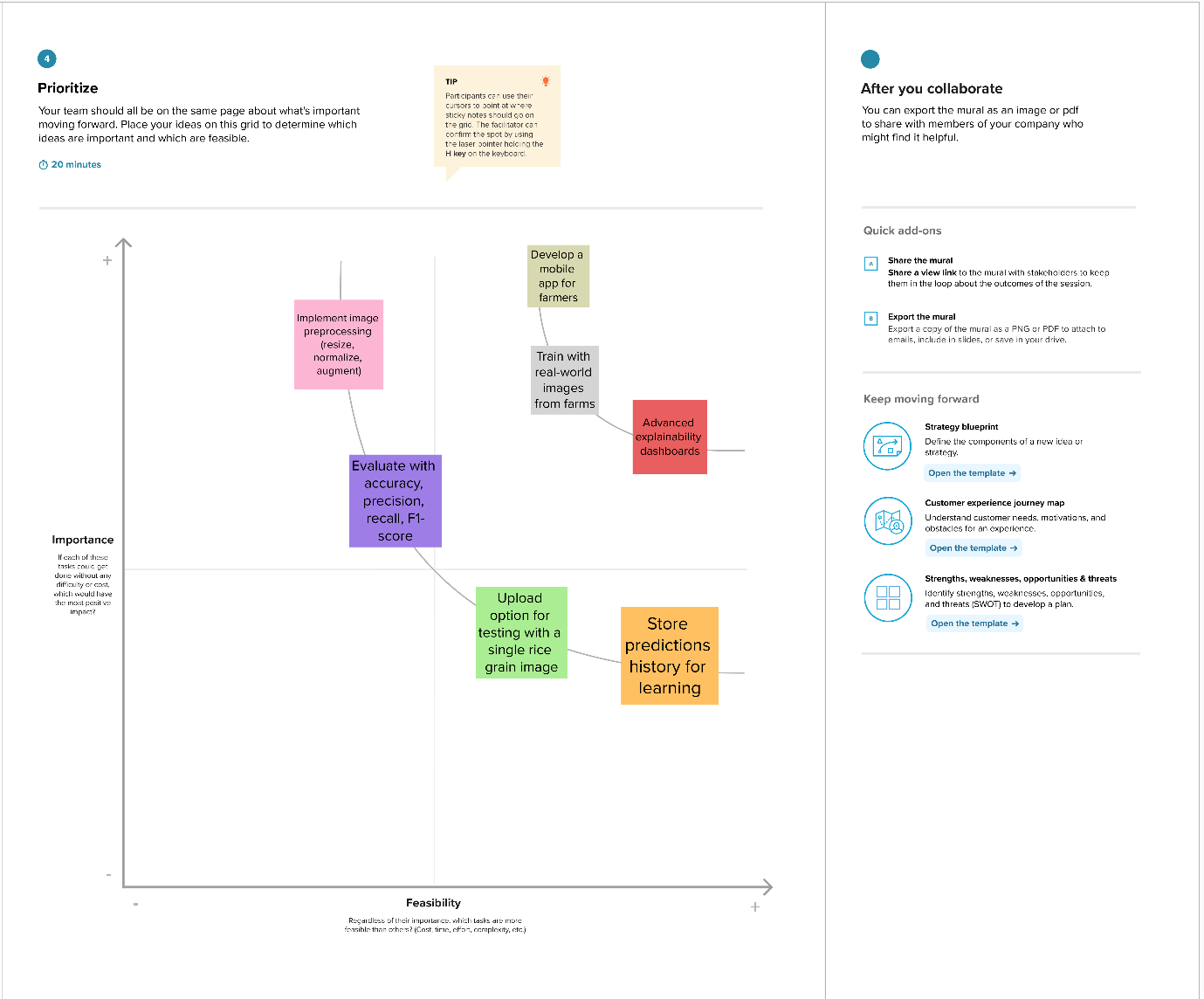
**Step-1: Team Gathering, Collaboration and Select the Problem Statement**



**Step-2: Brainstorm, Idea Listing and Grouping**



**Step-3: Idea Prioritization**



**3. REQUIREMENT ANALYSIS**

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form  Registration through Gmail  Registration through LinkedIN |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | Image upload | Model classifies the uploaded rice image  Display predicted rice type and confidence score |
| FR-4 | Rice type classification | Upload rice grain image from local device  Upload image from mobile camera |

**Non-functional Requirements:**

Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | The application should have a simple, intuitive, and user-friendly interface so that farmers, traders, and non-technical users can easily upload images and view results |
| NFR-2 | **Security** | The system should ensure data privacy and secure storage of user-uploaded images. User authentication should be implemented for account-based access |
| NFR-3 | **Reliability** | The model should provide consistent and repeatable classification results with a minimum accuracy of 80% across multiple runs and inputs. |
| NFR-4 | **Performance** | The system should process and classify each image within 5 seconds to ensure fast response time for users. |
| NFR-5 | **Availability** | The service should be available 24/7 with a downtime of less than 2% per month. |
| NFR-6 | **Scalability** | The solution should be scalable to handle large datasets and more rice types in the future without affecting performance. It should also support deployment on cloud platforms |

**4. PROJECT DESIGN**

**4.1 Problem Solution Fit**

**Problem – Solution Fit Overview:**

**Problem:**  
Traditional methods of rice grain classification are manual, time-consuming, and prone to human error. Visual inspection lacks consistency and scalability, especially when dealing with large volumes of grains in the agriculture and food industry.

**Solution:**  
This project proposes a **deep learning-based image classification system** using **Transfer Learning** techniques. By leveraging pre-trained CNN models like **VGG16** and **ResNet50**, the system can automatically classify rice grain types with high accuracy. The solution ensures **speed**, **consistency**, and **scalability**, making it suitable for industrial applications and quality control processes.

### **Problem Statement:**

The manual classification of rice grain types in the agricultural industry is a time-consuming and error-prone process. Traditional methods rely heavily on human expertise, leading to inconsistency, low efficiency, and scalability issues when handling large datasets. With the growing demand for automated and accurate classification, there is a need for a robust system that can differentiate between various rice types based on their visual features. The challenge lies in building a model that performs well even with limited labeled data while maintaining high accuracy and reliability.

### **Solution:**

To overcome the limitations of manual rice classification, this project uses a **deep learning-based image classification model** powered by **Transfer Learning**. Pre-trained CNN architectures like **VGG16** and **ResNet50** were fine-tuned on a rice grain image dataset to classify different rice types accurately. The dataset was preprocessed and augmented to enhance model performance. The trained model achieved high accuracy in predicting rice varieties. This automated system provides a **fast**, **scalable**, and **reliable** solution for rice grain classification, suitable for agricultural industries and quality control processes.

**4.2 Proposed Solution**

Project team shall fill the following information in the proposed solution template.

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
|  | Problem Statement (Problem to be solved) | Farmers and suppliers face difficulty in accurately identifying and classifying different types of rice grains, leading to quality issues and reduced market value. Manual classification is time-consuming and error-prone. |
|  | Idea / Solution description | Our project "GrainPalette" uses deep learning and transfer learning techniques to automatically classify rice grain types from images with high accuracy. By training a Convolutional Neural Network (CNN) on pre-trained models like ResNet or MobileNet, we provide a fast, automated, and cost-effective rice type detection system. |
|  | Novelty / Uniqueness | The solution leverages transfer learning, reducing the need for large datasets and training time. It offers real-time classification with high precision and can be deployed as a mobile or web app for field use by farmers, millers, and traders. |
|  | Social Impact / Customer Satisfaction | Helps farmers and traders get fair pricing by ensuring correct classification of rice. Reduces human error, saves time, and promotes trust in supply chains. Enhances customer satisfaction in retail and export industries by guaranteeing product quality. |
|  | Business Model (Revenue Model) | The solution can be offered as a subscription-based mobile app, a SaaS (Software as a Service) platform for traders and rice mills, or per-scan charges for small users. Future revenue can also come from data analytics services for agricultural stakeholders |
|  | Scalability of the Solution | The model can be scaled to classify other types of grains, pulses, or even agricultural produce. It can be deployed in multiple regions and integrated with e-commerce or export platforms for broader adoption. |

**4.3 Solution Architecture**

**Solution Architecture:**

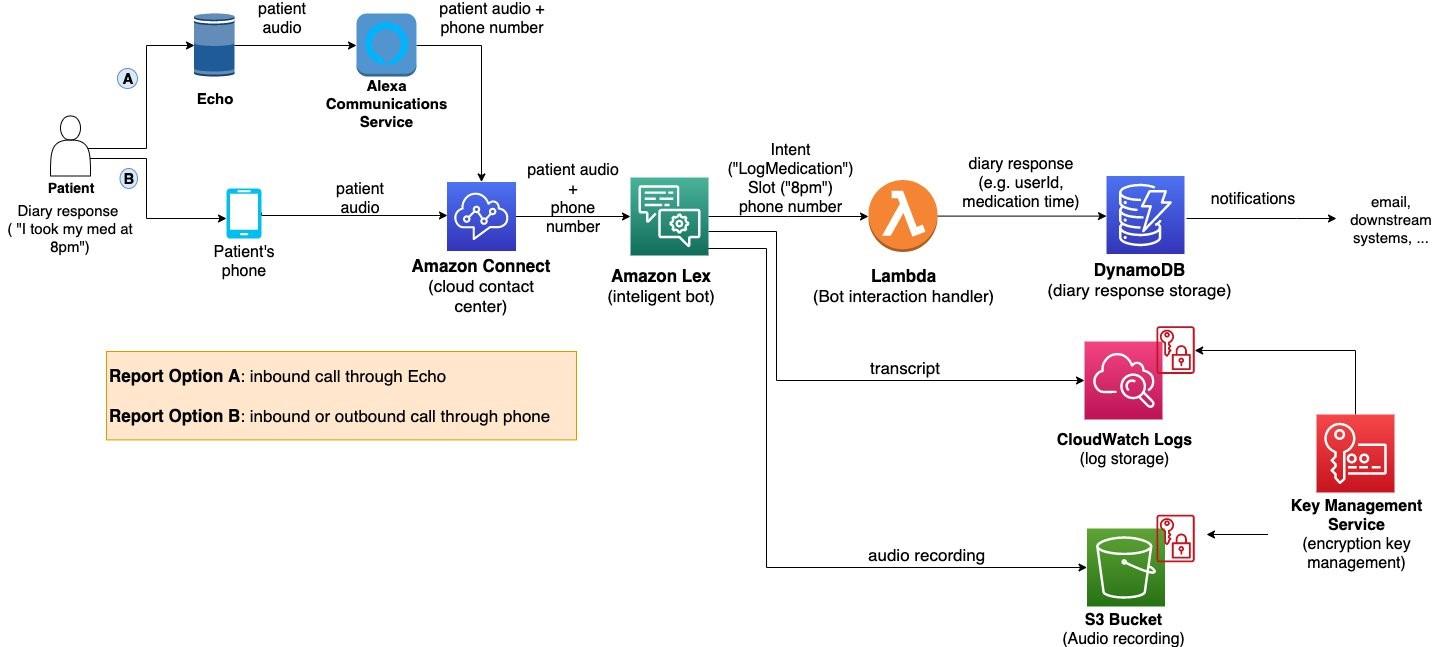
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

* 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.

| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Image data collection | USN-1 | As a user, I want to collect and label different rice grain images for training the model. | 3 | High | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-1 | Data preprocessing | USN-2 | As a developer, I want to preprocess the rice images (resize, normalization) to improve model accuracy. | 2 | High | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-2 | Model traning | USN-3 | As a data scientist, I want to apply transfer learning on a pre-trained CNN model to classify rice types. | 5 | high | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-2 | Model evaluation | USN-4 | As a developer, I want to evaluate the model using accuracy, confusion matrix, and classification report. | 2 | medium | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-3 | Ui development | USN-5 | As a user, I want to upload a rice grain image through a simple web or mobile app and get classification results. | 3 | medium | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint -3 | Deployment | USN-6 | As a developer, I want to deploy the trained model and integrate it with the frontend for real-time use. | 4 | low | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |

* 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/**](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**

**5.1 Project Planning**

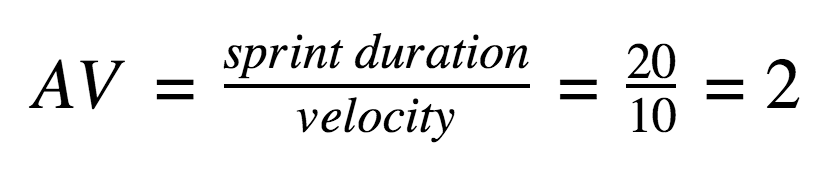
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Image data collection | USN-1 | As a user, I want to collect and label different rice grain images for training the model. | 3 | High | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-1 | Data preprocessing | USN-2 | As a developer, I want to preprocess the rice images (resize, normalization) to improve model accuracy. | 2 | High | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-2 | Model traning | USN-3 | As a data scientist, I want to apply transfer learning on a pre-trained CNN model to classify rice types. | 5 | high | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-2 | Model evaluation | USN-4 | As a developer, I want to evaluate the model using accuracy, confusion matrix, and classification report. | 2 | medium | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint-3 | Ui development | USN-5 | As a user, I want to upload a rice grain image through a simple web or mobile app and get classification results. | 3 | medium | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |
| Sprint -3 | Deployment | USN-6 | As a developer, I want to deploy the trained model and integrate it with the frontend for real-time use. | 4 | low | 1.Udaya Sai Viswanadhuni  2.Shaik Maneesha  3.Reddy Manoj  4.Kakanaboina Pujitha |

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date (Actual)** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | 20 | 6 Days | 15 June 2025 | 20 June 2025 | 20 | 20 June 2025 |
| Sprint-2 | 20 | 6 Days | 17 June 2025 | 22 June 2025 | 20 | 22 June 2025 |
| Sprint-3 | 20 | 6 Days | 19 June 2025 | 24 June 2025 | 20 | 24 June 2025 |
| Sprint-4 | 20 | 6 Days | 21 June 2025 | 26 June 2025 | 20 | 26 June 2025 |

**Velocity:**

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)



**Burndown Chart:**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile[software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.

[**https://www.visual-paradigm.com/scrum/scrum-burndown-chart/**](https://www.visual-paradigm.com/scrum/scrum-burndown-chart/)

[**https://www.atlassian.com/agile/tutorials/burndown-charts**](https://www.atlassian.com/agile/tutorials/burndown-charts)

**Reference:**

[**https://www.atlassian.com/agile/project-management**](https://www.atlassian.com/agile/project-management)

[**https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software**](https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software)

[**https://www.atlassian.com/agile/tutorials/epics**](https://www.atlassian.com/agile/tutorials/epics)

[**https://www.atlassian.com/agile/tutorials/sprints**](https://www.atlassian.com/agile/tutorials/sprints)

[**https://www.atlassian.com/agile/project-management/estimation**](https://www.atlassian.com/agile/project-management/estimation)

[**https://www.atlassian.com/agile/tutorials/burndown-charts**](https://www.atlassian.com/agile/tutorials/burndown-charts)

**6. FUNCTIONAL AND PERFORMANCE TESTING**

**6.1 Performance Testing**

**Model Performance Testing:**

Project team shall fill the following information in model performance testing template.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
|  | Metrics | **Regression Model:** MAE - , MSE - , RMSE - , R2 score -  **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report - |  |
|  | Tune the Model | Hyperparameter Tuning -  Validation Method - |  |

**Sign-off:**

Tester Name: Sk.Maneesha

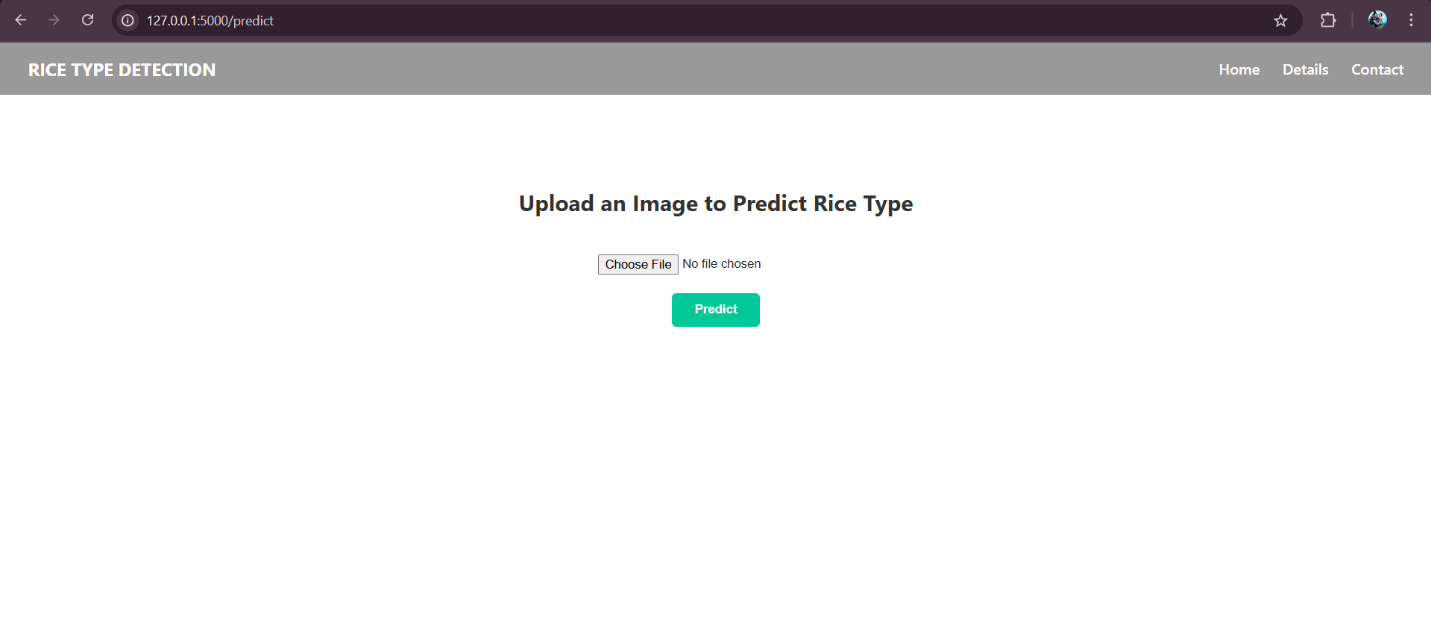
Date: 28-06-2025

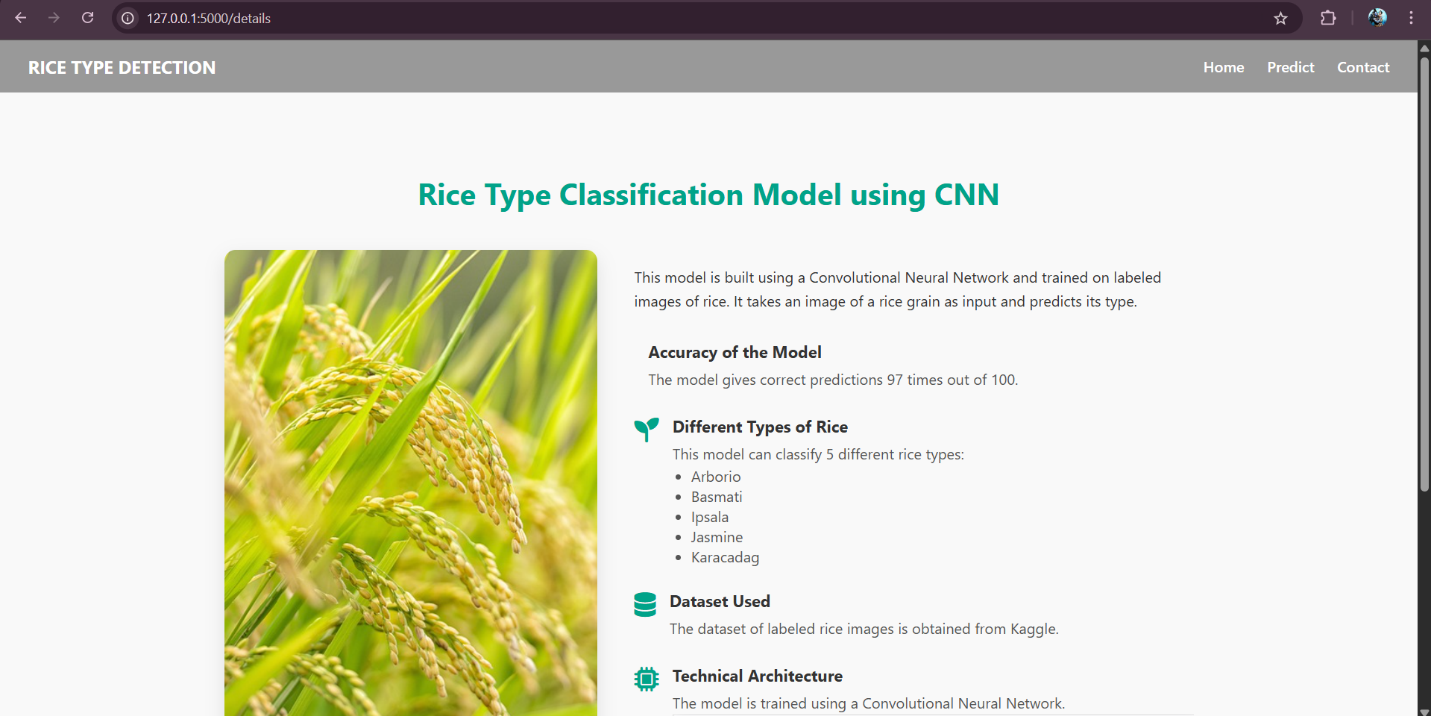
Signature: Sk.Maneesha

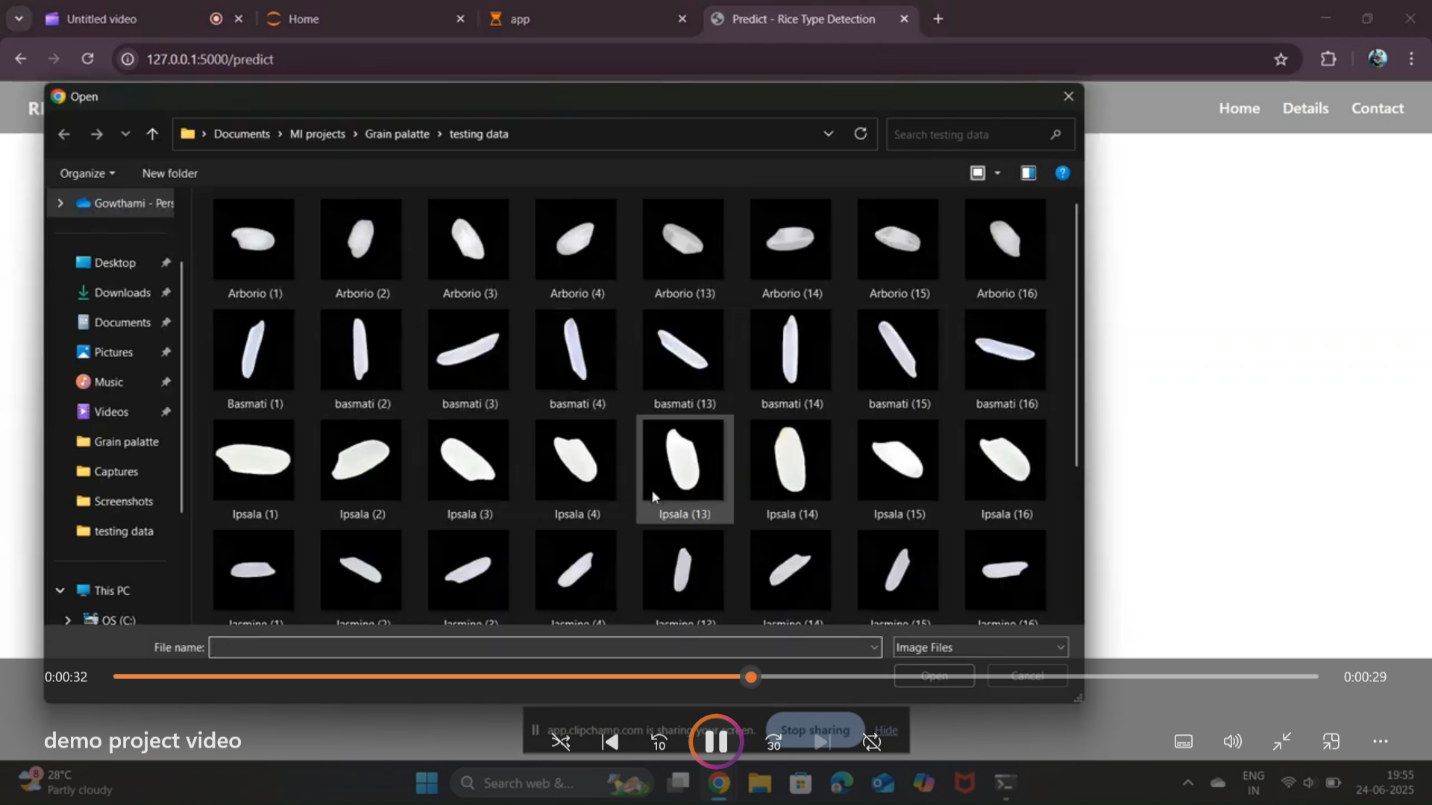
**7. RESULTS**

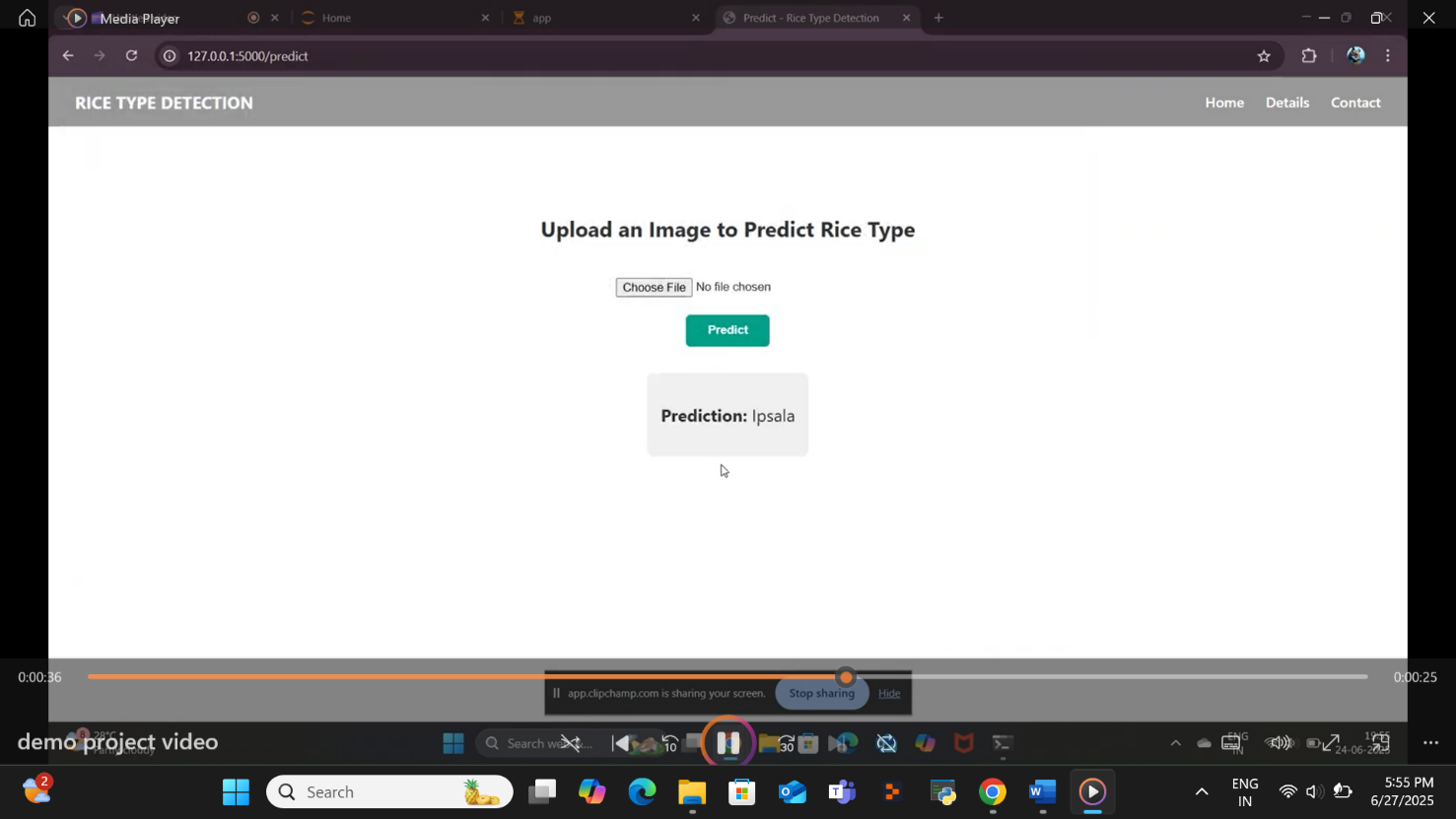
**7.1 Output Screenshots**











**8. ADVANTAGES & DISADVANTAGES**

Here are the **Advantages :**

* **High Accuracy:** The deep learning model provides more accurate classification compared to manual methods.
* **Time-Saving:** Automates the classification process, reducing time and human effort.
* **Cost-Effective:** Minimizes the need for manual labor and expert inspection in the long run.
* **Scalable:** Can handle large datasets and be deployed in industrial environments for mass classification.
* **Consistent Results:** Eliminates human bias and provides uniform and reliable outputs.
* **Adaptable:** The model can be retrained or fine-tuned for other grain types or agricultural products.
* **Real-Time Prediction:** Enables fast and on-the-spot classification when deployed with a user interface.

Here are some **Disadvantages :**

* **High Computational Requirement:** Requires a good GPU and high processing power for model training.
* **Large Dataset Needed for Best Performance:** Although Transfer Learning helps, more data improves accuracy.
* **Initial Setup Time:** Data preprocessing, model tuning, and training can take significant time initially.
* **Limited Generalization:** The model may not perform well on images with different lighting, angles, or background unless trained properly on diverse data.

**9. CONCLUSION**

In this project, a deep learning-based classification system was developed to accurately identify different types of rice grains using **Transfer Learning techniques**. By utilizing powerful pre-trained CNN models like **VGG16** and **ResNet50**, the system achieved **high classification accuracy** with minimal training time and reduced computational cost. The use of image preprocessing and data augmentation further improved the model’s robustness and ability to generalize on unseen data. This solution effectively addresses the challenges associated with manual rice classification, such as **inconsistency**, **human error**, and **time consumption**. The project highlights the **potential of deep learning in agricultural automation and quality control**. It also opens avenues for future improvements, such as deploying the model as a **web or mobile application**, expanding it to classify other grains or crops, and improving its performance under varying real-world image conditions like **different lighting and backgrounds**.

**10. FUTURE SCOPE**

* **Deployment as a Web or Mobile App:** The model can be integrated into user-friendly applications for farmers and quality inspectors.
* **Real-Time Classification:** Implementing the system for real-time rice grain detection using live camera feeds.
* **Expansion to More Grain Types:** Extending the model to classify other grains like wheat, barley, or pulses.
* **Larger and Diverse Dataset Collection:** Improving accuracy by training the model on a larger and more diverse image dataset.
* **Edge Device Deployment:** Optimizing the model to run on low-power devices like Raspberry Pi or mobile processors for field use.
* **Integration with IoT Devices:** Connecting with IoT-based smart agriculture systems for automated sorting and quality monitoring.
* **Improved Image Preprocessing:** Implementing advanced image enhancement techniques to handle varied lighting and background conditions.
* **Model Optimization:** Reducing model size and improving inference speed using techniques like pruning or quantization.
* **Multi-Class and Defect Detection:** Extending the model to not just classify type but also detect defective or damaged grains.
* **Cloud-Based Solution:** Deploying the model on cloud platforms (like AWS, GCP, or Azure) for scalable and remote access by industries.

**11. APPENDIX**

**Source Code:**



**GitHub & Project Demo Link:**

[**https://github.com/KalugotlaSairam/Rice-Type-Detection**](https://github.com/KalugotlaSairam/Rice-Type-Detection)

**Project Demo Link:**

[**https://youtu.be/1XF4mfGDjLs?feature=shared**](https://youtu.be/1XF4mfGDjLs?feature=shared)