**RiceScan: A Python-Based Mobile App with Camera API for Rice Recognition**

An

Application Development Project

Presented to the Faculty of

**Mindoro State University Calapan City Campus**

Masipit, Calapan City

Oriental Mindoro

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Science in Information Technology

by:

**Cordero, Jamir Lander**

**Arellano, Princess Maan**

**Reyes, Keziah Julien**

**Victa, Glen Matthew**

October 7, 2024

**TABLE OF CONTENTS**

**TABLE OF CONTENTS**

Title page ………………………………………………………………………………………………………………… 1

Table of Contents………………………………………………………………………………………………… 2

**CHAPTER**

1. INTRODUCTION

Project Context

Objectives

Scope and Limitations

Definitions of Terms

1. REQUIREMENTS SPECIFICATION

Hardware and Software Requirements

Functional Requirements

Non-Functional Requirements

Operational Requirement

Performance Requirement

Security Requirement

1. DESIGN AND DEVELOPMENT METHODOLOGIES

System Design

Architectural Diagram/Block Diagram

DFD Level 0

Sample Mock Up

REFERENCES

APPENDICES

**CHAPTER I**

**INTRODUCTION**

**Project Context**

Rice is one of the most important food crops worldwide, feeding more than half of the global population. Accurate identification and classification of rice varieties are crucial for agricultural production, quality control, and consumer trust. Traditionally, these tasks have been performed manually, relying on human expertise, which is both time-consuming and prone to error. Recently, machine learning and computer vision have emerged as effective methods for automating agricultural tasks, including rice variety recognition and quality assessment. Studies have demonstrated the potential of image processing techniques for identifying grains with high precision when using computer-based systems (Choudhury et al., 2022).

Despite advancements in machine learning and computer vision, there remains a lack of accessible, mobile-based applications that leverage these technologies for rice recognition. While research has explored the use of standalone computer systems for classifying grains, the

integration of these systems into mobile platforms remains underexplored. The challenge lies not only in developing high-performing algorithms but also in optimizing them for mobile environments, which involves constraints such as limited processing power and varying camera quality. There is a need to investigate how machine learning models can be applied in real-time on mobile devices using camera APIs (Tan & Kumar, 2023).

To bridge this gap, we propose RiceScan, a Python-based mobile application that integrates a Camera API to capture images of rice grains and applies machine learning algorithms to identify and classify them in real-time. This solution aims to make rice classification more accessible, especially for farmers and traders who lack access to expensive laboratory equipment. By offering a portable and affordable alternative, RiceScan could enhance quality control practices and streamline agricultural processes. Furthermore, integrating such technology into mobile devices would empower users to quickly and accurately assess rice quality and type from virtually anywhere, reducing reliance on traditional methods and minimizing human error (Patel & Verma, 2021). The

democratization of this technology could lead to improved market transparency and consumer confidence.

**Objectives**

The developers aim to create a comprehensive mobile application, RiceScan, to address inefficiencies in the manual identification and classification of rice varieties. By integrating smart technologies such as machine learning and real-time image recognition, the app seeks to enhance the efficiency of rice classification, improve accuracy, and provide a user-friendly solution accessible to farmers, traders, and consumers alike. Additionally, the app aims to enable predictive analysis of rice quality and characteristics using advanced algorithms, making it a valuable tool for improving quality control and decision-making in agricultural processes.

Specifically, the objective of this study aims to:

1. Develop a mobile application that integrates a Camera API to capture high-quality images of rice grains for real-time recognition and classification.

2. Implement machine learning algorithms within the app to accurately classify various rice varieties with a minimum accuracy rate of 90%.

3. Ensure a user-friendly interface that allows farmers, traders, and consumers to easily operate the app without specialized technical knowledge.

4. Enable offline functionality to ensure that users in remote or low-connectivity areas can still utilize the app for rice recognition and classification.

5. Provide predictive analytics based on rice variety data, allowing users to forecast trends in rice quality and characteristics using historical data and machine learning models.

**Scope and Limitations**

The study aims to develop RiceScan, a Python-based mobile application that leverages a Camera API for real-time rice grain recognition and classification, along with machine learning algorithms to enhance accuracy and efficiency. The app is designed to assist farmers, traders, and consumers by offering an accessible tool for identifying rice varieties,

improving quality control, and providing predictive analytics on rice characteristics and trends.

The scope of this study focuses on the app’s core features, including real-time image capture of rice grains via the Camera API, the integration of machine learning algorithms for rice variety classification, and a user-friendly interface to make rice identification accessible to non-experts. Additionally, the app will include offline functionality, allowing users in low-connectivity areas to classify rice without the need for internet access. Predictive analytics based on rice data will be included, using historical patterns to forecast trends in rice quality and characteristics, thus aiding decision-making in agricultural practices. The app will also provide data storage, enabling users to securely store and access their classification history.

The limitations of the study include the dependence on high-quality images for accurate classification; inconsistent lighting or poor camera quality may affect the app’s ability to correctly identify rice. Additionally, the accuracy of

predictive analytics will rely on the quantity and quality of historical data available, which may limit the precision of forecasts for users with minimal or inconsistent data records. While the app will include offline functionality, features requiring data synchronization, such as the sharing of classification history, will need internet access. Security measures will be implemented to protect user data, but as with any digital system, it will remain vulnerable to external cybersecurity threats.

**Definition of Terms**

**RiceScan** – A Python-based mobile application that utilizes a Camera API and machine learning algorithms to identify and classify rice varieties in real-time through image recognition. It aims to streamline the rice classification process for farmers, traders, and consumers.

**Camera API** – An interface integrated into the RiceScan application that allows the mobile device's camera to capture high-resolution images of rice grains for processing and classification.

**Machine Learning Algorithms** – Computational models used by RiceScan to analyze captured images of rice grains, learning

from patterns in the data to classify various rice types with high accuracy.

**Real-time Recognition** – The capability of the RiceScan app to classify rice varieties instantly as the user captures an image, without significant delay, ensuring efficient and timely identification.

**Predictive Analytics** – A feature of the RiceScan app that uses historical data to forecast trends in rice quality and

characteristics, enabling users to make informed decisions based on patterns and future predictions.

**Data Synchronization** – The process of keeping the user's classification data consistent across multiple devices or platforms, allowing secure access and backup once the device is connected to the internet.

**Classification Accuracy** – The percentage of correct rice variety identifications made by the RiceScan app, with the goal of achieving at least 90% accuracy in classification based on machine learning models.

**Rice Grain Image Recognition –** The technology used by RiceScan to analyze visual characteristics of rice grains, such as shape, size, and texture, for the purpose of identifying different rice varieties.

**User Interface (UI)** – The design and layout of the RiceScan mobile application that ensures ease of use, enabling users with minimal technical expertise to effectively classify rice grains.

**CHAPTER II**

**REQUIREMENTS SPECIFICATION**

**Software Requirements**

Software Specifications refer to the representation of the software used by the system. Table 1 below presents the software specifications to be used by the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Type** | **Minimum Specification** | **Recommended Specification** |
| OpenCV/Pillow | Image Processing | Python version 3.6 or higher | Python version 3.8 or higher |
| TensorFlow/Keras | Machine Learning | Python version 3.6 or higher | Python version 3.8 or higher |
| Vue js | Web Framework | Node js version 10.0 or higher | Node js version 14.0 or higher |

**Functional Requirements**

* **Rice Variety Recognition**

- Image capture through the device’s camera.

Machine learning model to classify rice varieties.

- Display the identified rice variety with relevant details.

* **Image Capture and Processing**

- High-resolution image capture.

- Pre-processing of images (e.g., cropping, filtering) to enhance recognition accuracy.

* **User Interface (UI)**

- Simple navigation with simple steps for scanning.

- Clear display of recognition results and reports.

- Responsive design compatible with various mobile devices.

**Non-Functional Requirements**

* **Performance Requirements**

- The system will have a simple and easy-to-use interface, enabling users to perform tasks with little training.

- The model that will be used for testing is separated from the model that will be used for input to ensure the accuracy of the image recognition.

* **Operational Requirements**

- The system shall support Android phones for an easy-to-use and optimal user experience.

- The system shall operate on Python or Java for backend development, OpenCV and Pillow for image processing, TensorFlow/Keras for machine learning framework, and Vue js for web frameworks.

* **Security Requirements**

- The model and data for training will only be accessible to the developer to ensure data safety.

* **Cultural and Political Requirements**

- The system shall support multiple languages for users.

- The system shall comply with relevant local and international regulations, such as data protection laws.

- Comprehensive user and technical documentation shall be provided for system maintenance and support.

**Chapter III**

**Design and Development Methodologies**

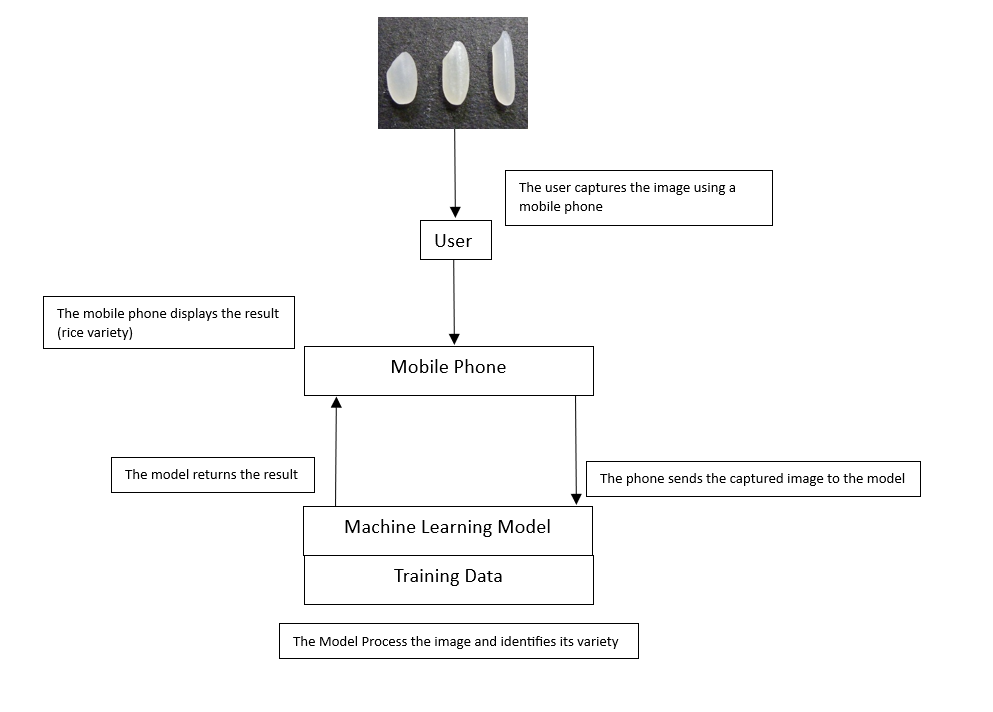
**System Design**

RiceScan: A Python-based mobile app with Camera API for Rice Recognition represents its architecture including mobile phone, machine learning model, and training model. The user captures the image using the mobile phone, and the phone sends the captured image to the model. After that, the model processes the image and identifies the variety of rice grains then the model returns the result to the phone to display the result.

It showcases how the user interacts with the mobile application to capture rice images, which are then processed by the server, while the database stores user profiles and analysis results. This application offers functionalities to improve operations in image capture and automated quality analysis of rice samples using machine learning. It also provides user management features for secure registration and access.

**Architectural Diagram/ Block Diagram**

In this section, system architecture was designed to define the flow and behavior of the system’s functionalities to execute its high-quality performance.

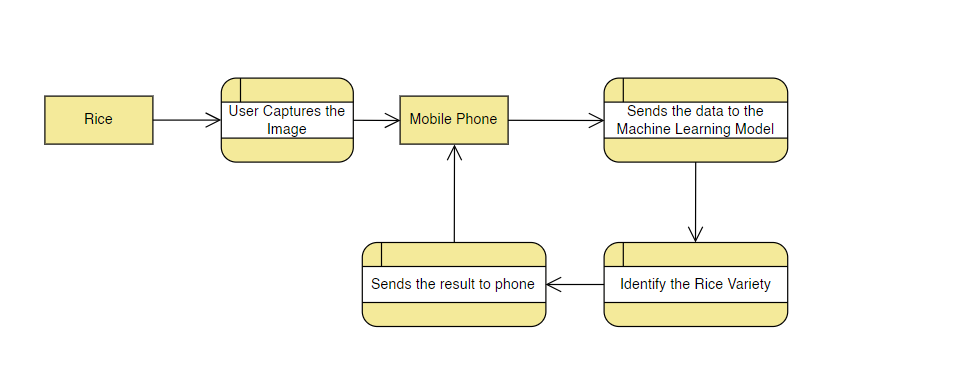
****

**Figure 1: System Architecture of RiceScan**

Figure 1 shows the process which begins with a user capturing an image of a rice grain using their mobile phone. This image is then sent to a machine learning model, which has been trained on a dataset of various rice varieties. The model analyzes the image, extracting features like shape, color, and texture. It then compares these features to its knowledge base to identify the most likely rice variety. Once the model determines the variety, it sends this information back to the

mobile phone. The phone then displays the identified rice variety to the user.

**DFD Level 0**

****

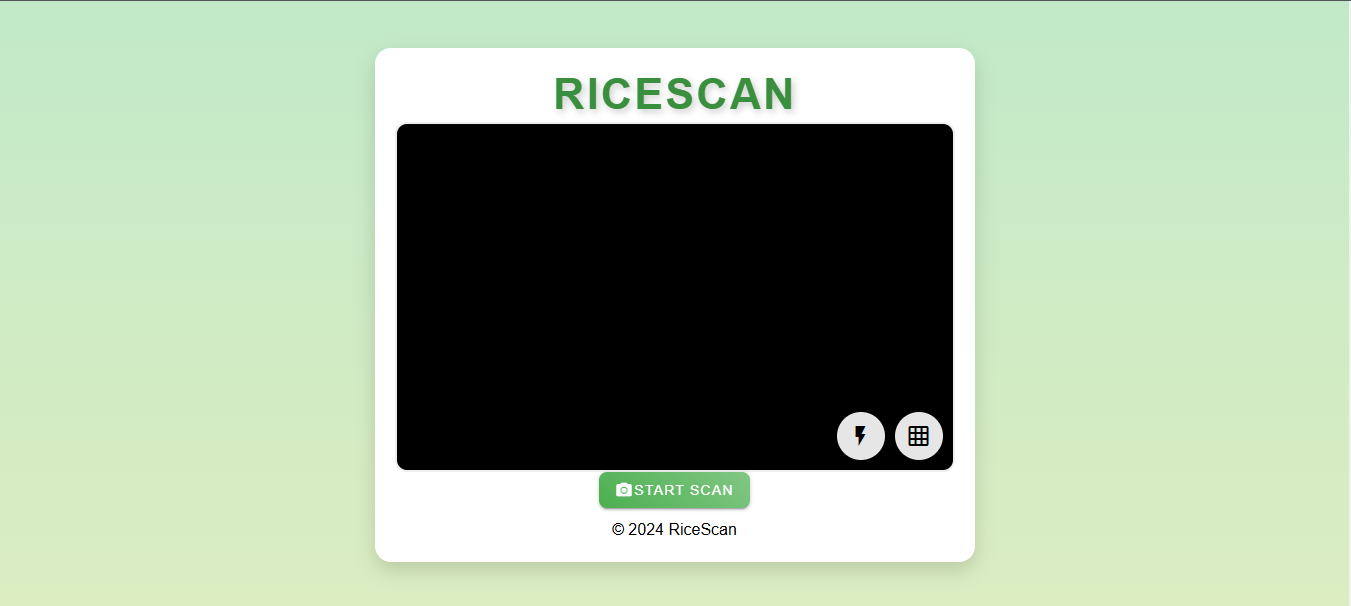
**Figure 2: Data Flow Diagram Level 0**

Figure 2 The image illustrates a process for identifying rice varieties using a machine learning model. A user begins by capturing an image of a rice grain using a mobile phone. This captured image is then transmitted to a machine learning model. The model, which has been trained on a dataset of various rice varieties, analyzes the image to extract key features such as shape, color, and texture. By comparing these features to its knowledge base, the model identifies the most likely rice variety. Once the identification is complete, the model sends the result back to the mobile phone. The user

then receives the identified rice variety on their phone's display.

**Sample Mock-up**

It is a visual representation of a design concept, often used to showcase how a product or design will look in a real-world setting.



**Figure 3: RiceScan Interface**

**Development Method**

In developing RiceScan, we use Agile methodology which is particularly suitable RiceScan due to its inherent flexibility and adaptability. This approach allows for the accommodation of evolving requirements, especially in a field like machine learning where advancements are rapid. By

prioritizing customer collaboration, the team can ensure that the app aligns with the specific needs of farmers, traders, and consumers. Agile's iterative nature facilitates continuous improvement, enabling the team to learn from mistakes and incorporate new insights. Agile empowers the RiceScan team to build a robust, user-friendly, and effective app that meets the evolving demands of its target users.



**Planning:** In this phase, our group define the project's scope, goals, and timeline. We break down the project into

smaller, manageable tasks and create a plan for completing them within a specific timeframe.

**Requirements Analysis:** In this phase, we gathered and analyzed the specific requirements for the project. This involves identifying the features to be implemented, and defining the acceptance criteria for each feature.

**Design:** In this phase, we design the overall architecture and user interface. This phase involves creating detailed design documents, and mockups to visualize the final product.

**Coding:** In this phase, the backend developer write the actual code to implement the designed features. They follow coding standards and best practices to ensure the code is clean, efficient, and maintainable. And the other member starts with collecting the datasets.

**Unit Testing:** In this phase, Developers write and execute tests to verify the correctness of individual code units. This helps identify and fix bugs early in the development process.

**Acceptance Testing:** In this phase, we conducted tests to ensure that the completed features meet the defined

functional requirements. This involves testing the functionality, performance, and usability of the software.

**Chapter IV**

**REFERENCES**

B. Arora, N. Bhagat, L. R. Saritha and S. Arcot, "Rice Grain Classification using Image Processing & Machine Learning Techniques," 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2020, pp. 205-208, doi: 10.1109/ICICT48043.2020.9112418.

V. E. Nirmalan, R. D. Nawarathna and S. Manivannan, "Comparative Analysis of Different Features and Encoding Methods for Rice Image Classification," 2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS), Colombo, Sri Lanka, 2018, pp. 1-5, doi: 10.1109/ICIAFS.2018.8913380.

B. Arora, N. Bhagat, L. R. Saritha and S. Arcot, "Rice Grain Classification using Image Processing & Machine Learning Techniques," 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2020, pp. 205-208, doi: 10.1109/ICICT48043.2020.9112418.

Choudhury, B., Sharma, P., & Singh, R. (2022). Machine learning approaches for rice grain classification: A

review. International Journal of Computer Applications in Agriculture, 9(3), 65-72.

Tan, L., & Kumar, R. (2023). Mobile applications for real-time grain classification using camera APIs. Journal of Agricultural Informatics, 11(1), 34-50. Patel, S., & Verma, M. (2021). The role of mobile technology in improving agricultural quality control. International Journal of Agricultural Technology, 8(2), 55-67.