# RiceVision: Web and Mobile App-based Classification of Rice Varieties

# Abstract

# RiceVision is an open-source web and mobile application designed for real-time identification of 62 rice varieties using single-grain images. The system integrates fine-tuned VGG16 and ResNet models within an ensemble framework, trained on the RiceNet-62 dataset (46,500 images). The web platform, developed in Django, enables browser-based classification through cloud inference, while the Flutter-based mobile app performs on-device prediction via TensorFlow Lite. Both interfaces deliver instant results without specialized hardware, supporting seed certification, research, and agritech deployment. Released under the MIT license, RiceVision demonstrates how deep learning can bridge research and field-level precision agriculture through accessible, reproducible software.

# Keywords

# *Rice variety classification, Deep learning, Mobile application, Precision agriculture, Computer Vision*

# Metadata

|  |  |  |
| --- | --- | --- |
| **Nr** | **Code metadata description** | ***Metadata*** |
| C1 | Current code version | *V1.0* |
| C2 | Permanent link to code/repository used for this code version | *WebApp:* [*https://github.com/abrahametry/RiceVision*](https://github.com/abrahametry/RiceVision)  *Android App:* [*https://github.com/abrahametry/RiceVisionLense*](https://github.com/abrahametry/RiceVisionLense) |
| C3 | Permanent link to reproducible capsule |  |
| C4 | Legal code license | *MIT License* |
| C5 | Code versioning system used | *git* |
| C6 | Software code languages, tools and services used | *Python (Django, TensorFlow, scikit-learn), Dart (Flutter), TensorFlow Lite* |
| C7 | Compilation requirements, operating environments and dependencies | ***Web:*** *Linux server, Python 3.x, Django, TensorFlow 2.x;*  ***Mobile:*** *Android with Flutter SDK, TFLite. Dependencies include numpy, Pillow, scikit-learn* |
| C8 | If available, link to developer documentation/manual | *GitHub README files (no separate manual)* |
| C9 | Support email for questions | [masudulislam11@gmail.com](mailto:masudulislam11@gmail.com) |

## ****1. Motivation and significance****

Rice is a globally vital crop, feeding more than half of the world’s population and underpinning agricultural economies across Asia and beyond [1]. Accurate identification of rice varieties is essential for seed certification, breeding programs, yield optimization, and preventing fraud in the agricultural supply chain. Traditional identification relies heavily on manual morphological inspection—an expert-driven process that is subjective, time-consuming, and impractical for large-scale operations [2].

Recent advancements in computer vision and deep learning have made it feasible to automate such visual discrimination tasks. Yet, most available systems are constrained by small datasets (fewer than 20 varieties) and limited accessibility, often existing only as research prototypes rather than deployable tools [3].

To address these limitations, RiceVision was developed as an open-source web and mobile application capable of identifying 62 rice varieties from single-grain images in real time. The software integrates a fine-tuned convolutional neural network (CNN) based on **VGG16**, trained on the comprehensive RiceNet-62 dataset containing 46,500 images captured under standardized conditions.

The trained model achieved over **98.7% validation accuracy**, outperforming simpler architectures and aligning closely with transformer-based models such as ViT-B/16 (91.8% accuracy). Experimental ensemble variants combining VGG16 and ResNet with XGBoost meta-classifiers were also explored during research, but the **deployed version currently uses the VGG16 model** for both web and mobile inference to ensure efficiency and lightweight deployment.

The Django-based web application allows users to upload grain images via a browser interface, where a cloud API performs inference and returns the predicted variety name. The Flutter-based mobile application integrates a TensorFlow Lite–converted model for on-device inference, enabling fully offline classification on Android and iOS devices.

Compared to existing agricultural recognition tools such as ARVAC (11 varieties, ~94% accuracy) [4] and general plant recognition apps like AgroAId [5] or Plantix [6], RiceVision demonstrates superior scalability, accessibility, and openness. It is the first open-source, cross-platform software capable of classifying over sixty rice varieties with proven generalization across diverse image sources.

**Software description**

**2.1. Web & Mobile App Architectures**

RiceVision is engineered as a dual-platform intelligent system—comprising a Django-based web application and an offline-capable Flutter mobile application—to classify 62 rice varieties using image-based inference. Its architecture follows a modular design for the web backend and a streamlined stateful widget approach in mobile.

At the core lies the Rice Classification Model, a fine-tuned VGG16 network with global average pooling, dropout, dense layers, and batch normalization. The model was trained on the RiceNet-62 dataset, comprising 46,500 images (224×224 resolution, black background) that capture distinct morphological variations across 62 rice cultivars. Each input image undergoes normalization and resizing before feature extraction and prediction via softmax output. The model’s weights are exported in two formats: (a) a .h5 file for server deployment, and (b) a TensorFlow Lite (TFLite) version for mobile inference, ensuring computational efficiency on resource-limited devices.

**Web Server Architecture (Django + TensorFlow):**

The backend, developed using the Django framework, functions as the central inference hub for web clients. The workflow proceeds as follows:

A user uploads a single rice grain image (minimum 5× optical zoom recommended) through the web interface.

The Django server verifies the file integrity and pre-processes it (resizing to 224x224, normalization to [0,1] via division by 255.0, RGB conversion).

The processed tensor is passed directly to the loaded TensorFlow model for inference.

A prediction is generated (softmax maximum probability), encapsulated in a structured JSON response.

The server fetches additional rice variety information from a configurable database (SQLite by default, PostgreSQL supported).

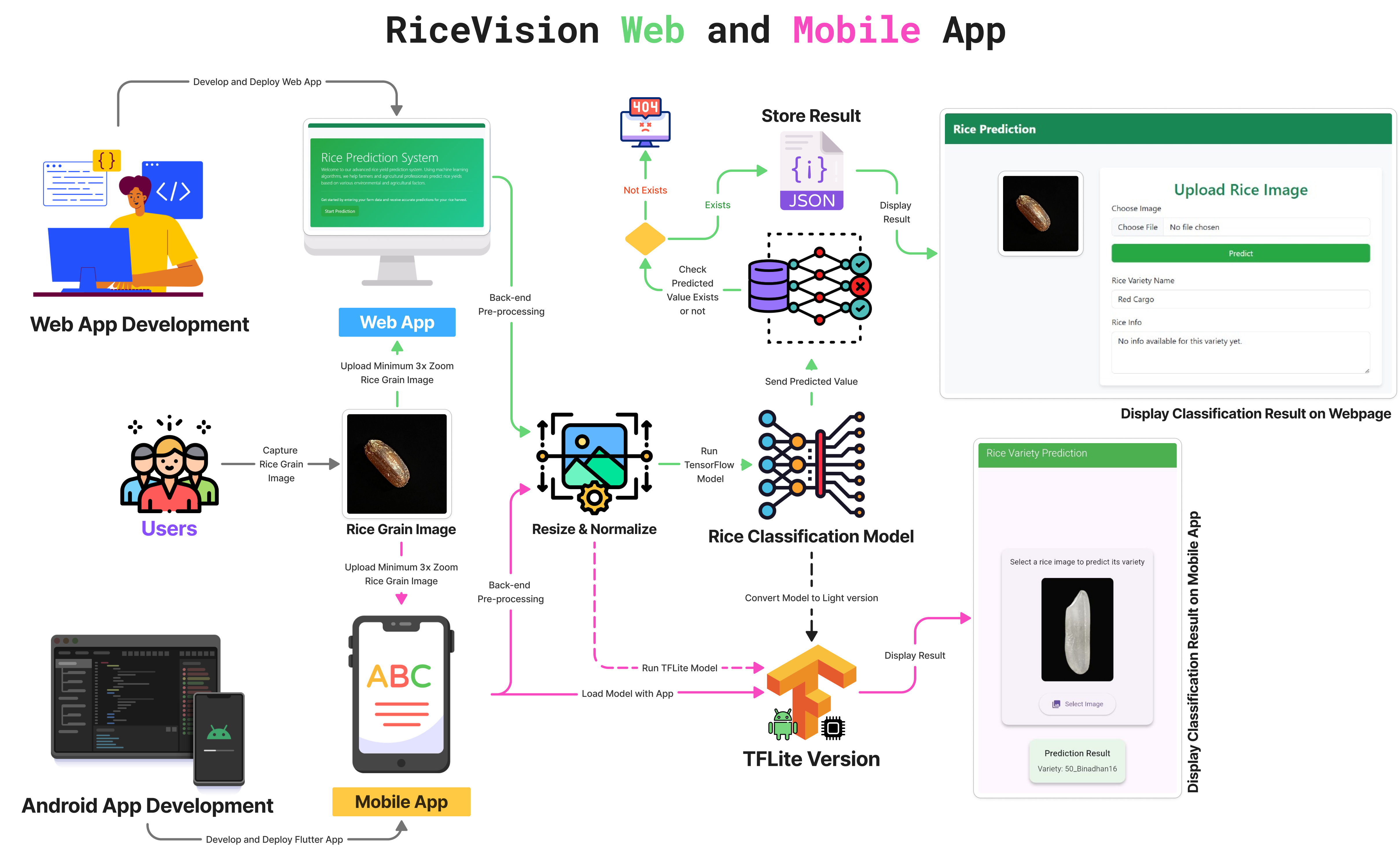
The final output, including predicted class name (e.g., “Lal Aush”), is rendered to the user.

**Cloud ML Deployment:**

The web backend is deployable via platforms like Render.com. Model serving uses direct loading in views, with support for future containerization. The architecture is designed to scale with hosting providers, ensuring stable real-time inference under moderate traffic scenarios.

**Mobile Application Architecture (Flutter + TensorFlow Lite):**

The Flutter mobile client is developed using Dart and employs a stateful widget pattern with service-like functions for inference. The application embeds a pre-quantized TFLite model (~30 MB) for low-latency and energy-efficient predictions. The offline inference capability ensures that predictions can be made without internet connectivity, particularly vital for rural agricultural zones.



**Fig.1.** RiceVision Web and Mobile AppDeployment and Usage Process

**Operational Workflow:**

The user selects a rice grain image from the gallery (camera support planned) via the interface.

The app pre-processes the image locally (resize 224×224, normalization [0,1] range).

The TFLite interpreter runs the model inference on-device.

The output is displayed instantly.

the app to query local storage or a remote API for extended information about the predicted varietyn

The design allows synchronization with the web version. When internet connectivity is available, the app can download the latest model and variety info for collective improvements—a foundation for federated learning integration in upcoming releases.

2.2 **Web and Mobile App Functionalities**

Both web and mobile platforms share a unified feature philosophy built around accessibility, transparency, and continual learning. The applications currently comprise three major functional modules:

(1) Landing Interface, (2) Image Upload and Prediction Module, and (3) Results and Information Display Module. Each module is extendable to accommodate future enhancements in database integration and model retraining.

**1. Landing and Introduction Page:** Upon opening either the web or mobile app, users encounter a concise, informative dashboard introducing the RiceVision project. This interface highlights the purpose of the software (real-time rice variety identification) and provides clear navigation to the classification tool. The web version also integrates repository and dataset links for researchers who wish to reproduce the experiments. The mobile interface is optimized for low bandwidth, using static local assets for rapid loading. This page will later include API documentation for developers and tutorial videos demonstrating sample classifications.

**2. Image Upload and Prediction Page:** This core module enables users to perform single-grain classification through a streamlined process:

Image Input: Users select an image of a rice grain from the gallery (camera integration planned), ensuring sufficient zoom and contrast.

Processing and Validation: The app automatically validates image quality (non-empty input) and pre-processes it according to the trained model’s input pipeline.

Prediction Inference: The image tensor is fed into the VGG16 model (or TFLite in mobile), returning the most probable rice variety label .

The Django backend supports single-image classification, with batch processing planned for agricultural laboratories. In contrast, the Flutter mobile version prioritizes simplicity, making single-grain predictions accessible even to non-technical users such as farmers or extension workers.

**3. Results and Visualization:** Results are displayed immediately after inference in both versions. For the web interface, predictions are shown in a dedicated panel, where the top label is accompanied by a thumbnail of the input image. The mobile version mirrors this format, displaying results on a single page with the predicted variety highlighted. Both platforms store recent predictions locally to allow basic history tracking. An upcoming enhancement will connect the predicted label to a comprehensive rice variety database, which will include morphological traits, ideal cultivation environments, and yield potentials. This integration will transform RiceVision from a classifier into a decision-support system for agronomists and researchers.

**Planned Future Extensions:**

1. *Variety Information Integration:* Each predicted variety will link to detailed agronomic data such as seed origin, soil preference, nutritional composition, and disease resistance. This will be sourced from curated agricultural repositories and user-contributed data.
2. *Federated Learning Updates:* To enhance model generalization, future releases will employ a federated learning approach where mobile devices contribute anonymized feature representations back to the central server. This distributed training strategy will enable continuous model improvement without compromising data privacy—a methodology inspired by adaptive healthcare AI systems like DLDiagnosis.
3. *Dataset Expansion:* RiceVision’s modular structure allows new varieties or image sources to be integrated seamlessly. The training scripts on GitHub enable researchers to retrain the ensemble using new data or experiment with transformer-based architectures for further performance gains.
4. *Explainability and Visualization:* Integration of Grad-CAM visualizations will allow users to interpret model focus regions, enhancing trust in automated classification.

**Technical Features Summary:**

Frontend: HTML5, CSS, JavaScript (for web); Dart (for mobile).

Backend: Python (Django 5.2), TensorFlow 2.x.

Model: Fine-tuned VGG16 with custom head.

Deployment: Direct loading (Web), TensorFlow Lite (Mobile).

Storage: Configurable database PostgreSQL for metadata, SharedPreferences and local cache (Mobile).

Performance: Average inference time <1 s (Web/Mobile on standard hardware).

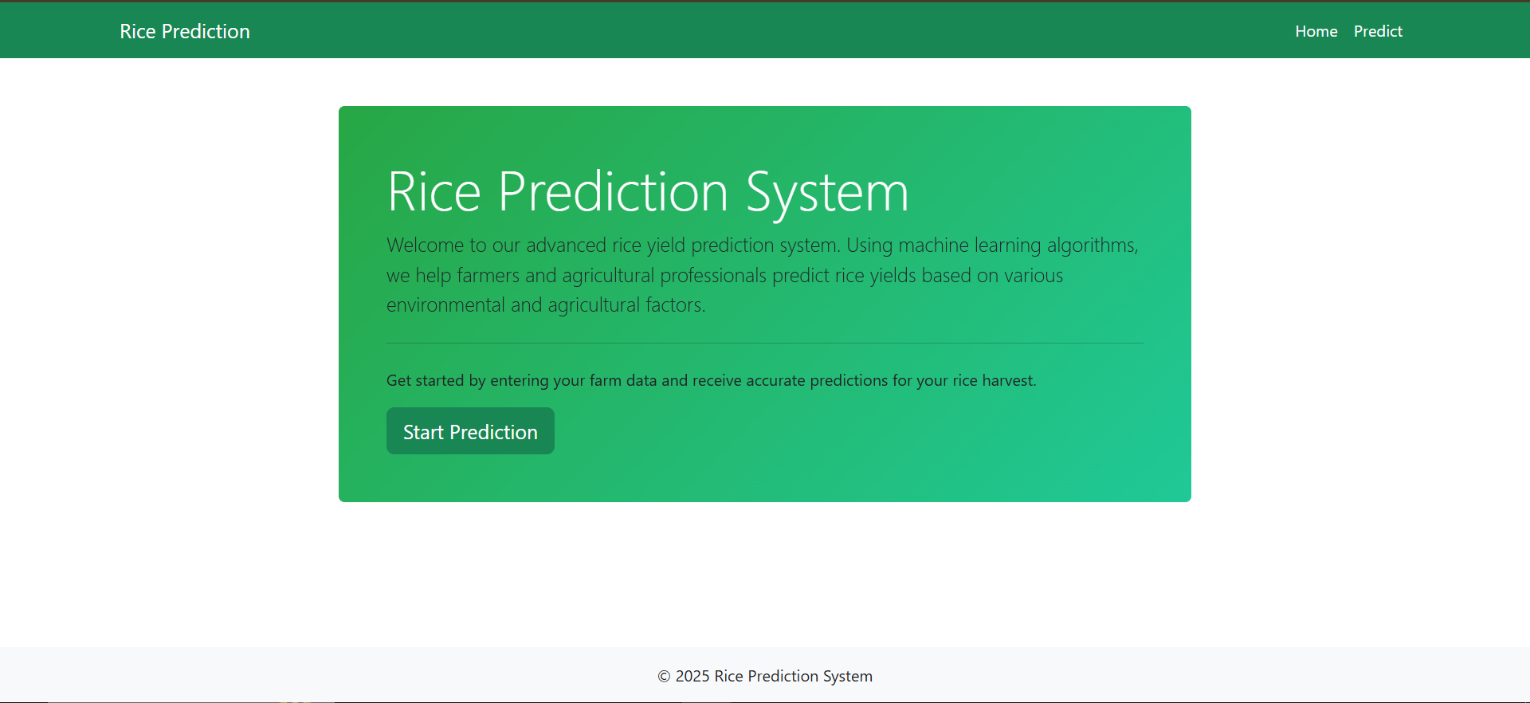
Security: HTTPS encryption, CORS for cross-origin, validated image uploads to prevent malicious file types.

In practical use, a farmer or researcher can simply select a grain sample image using their smartphone gallery, receive an immediate classification output, and—once the knowledge base is fully integrated—access context-rich agronomic insights. Researchers can also use the web dashboard for dataset analysis or single testing.

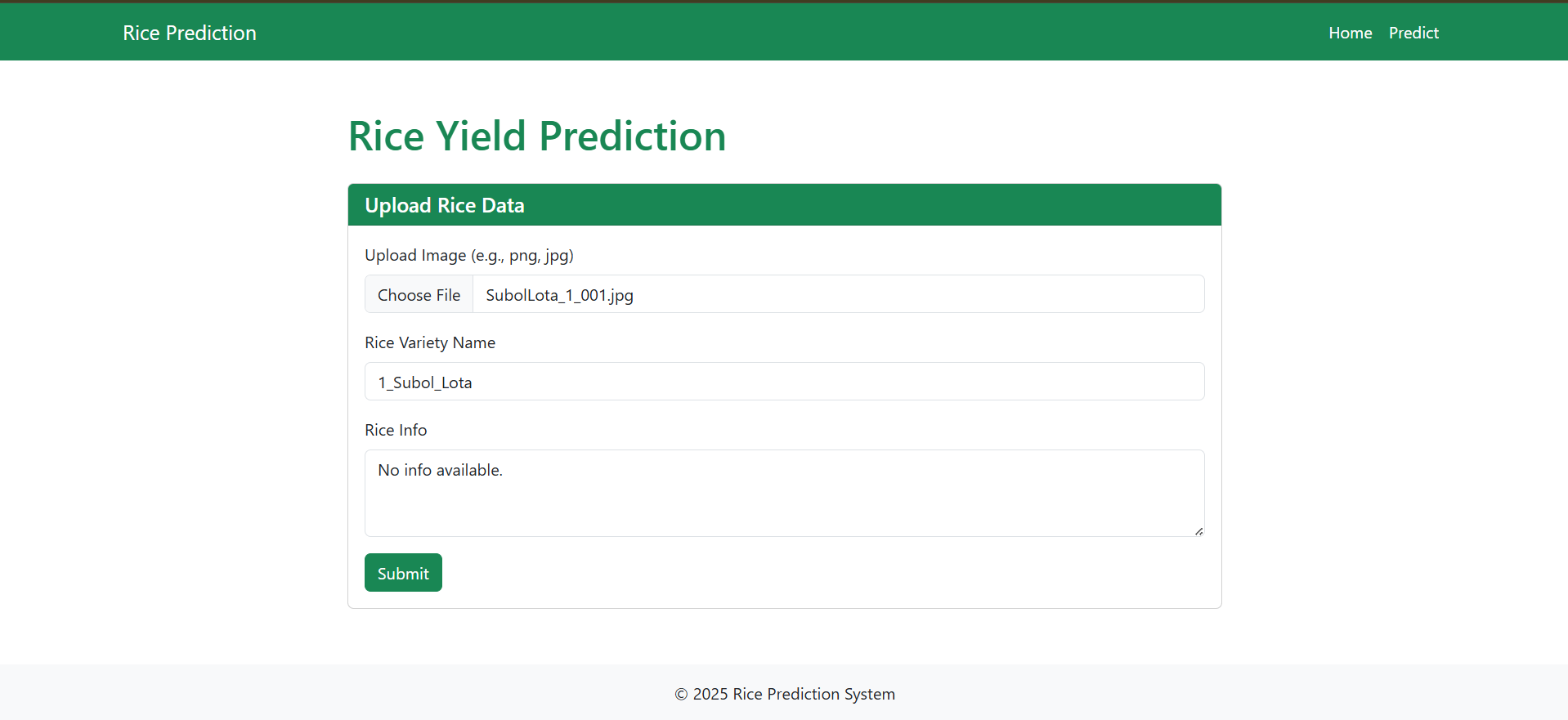
# Illustrative examples

Figure X and Figure Y illustrate the graphical interface of the RiceVision web application, highlighting the home and image upload pages. The interface is designed using Django and Bootstrap frameworks to ensure responsive layout and intuitive navigation. Users can upload a single rice grain image in standard formats (.jpg, .png), after which the system automatically performs inference through the deployed TensorFlow model. The web UI displays the predicted rice variety name, along with placeholders for variety-specific agronomic data that will be integrated in future updates, such as yield metrics, growth duration, and soil suitability.

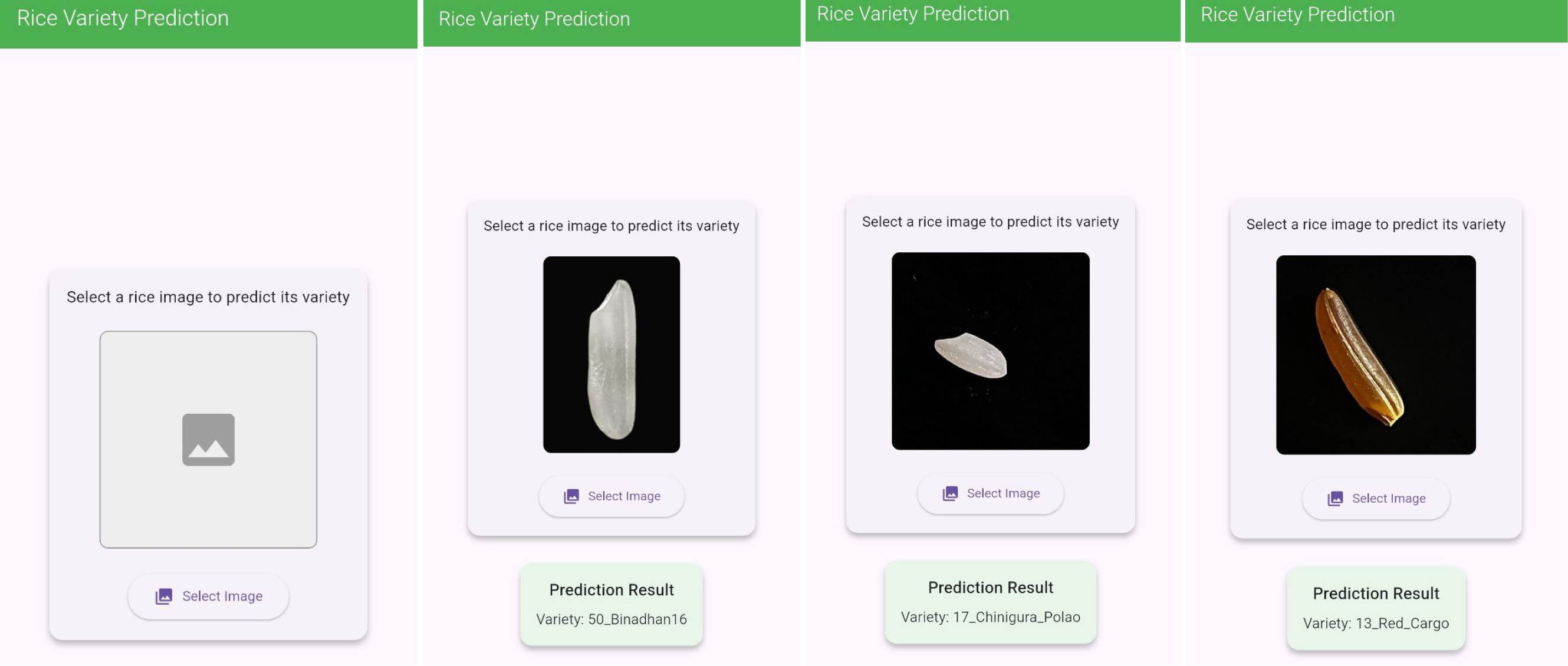
Figure X presents the Android mobile version of the RiceVision system developed in Flutter. The mobile interface demonstrates the real-time on-device inference process using TensorFlow Lite. Users can select or capture a rice grain image, which is then pre-processed (resized and normalized) before running through the embedded model. The output panel shows the predicted variety label (such as Binadhan16, Chinigura Polao, Red Cargo). The lightweight mobile UI is optimized for low-latency prediction, enabling seamless offline classification. Both interfaces ensure consistent design principles—minimal input, immediate feedback, and extendable architecture for integration with a comprehensive rice variety knowledge base.



**Fig.2.** RiceVision Web App home page



**Fig.3.** RiceVision Web App Image upload and Result View Page



**Fig.4.** RiceVision Android App home interface (Image upload and Result View)

1. **Impact**

*RiceVision* represents a transformative step toward intelligent agricultural informatics by operationalizing rice variety classification into an accessible web and mobile framework. It not only consolidates deep learning models into user-friendly applications but also establishes a foundation for new research directions in digital phenotyping, cross-domain transfer learning, and federated learning for crop recognition. The release of the open-source RiceNet-62 dataset and model training scripts enables researchers to benchmark new architectures, compare feature extraction techniques, and explore transformer-based or self-supervised learning methods under standardized conditions.

These apps directly enhance existing research by bridging the gap between algorithmic innovation and field-level usability. Prior studies have focused primarily on model performance, often lacking deployable systems. *RiceVision* extends this work by integrating model inference, explainable visualization, and bidirectional data flow between users and researchers. Its dual deployment (web service and offline mobile application) ensures continuous accessibility, accelerating on-site data collection and verification in seed laboratories, farms, and certification agencies.

In daily use, *RiceVision* simplifies the tasks of agronomists, breeders, and food quality inspectors by providing instant classification results from a single image. The application reduces dependency on expert morphological analysis, saving time and minimizing subjectivity in decision-making. Since its release on GitHub, RiceVision and RiceNet-62 have recorded increasing engagement, with early adopters in academic institutions in Bangladesh, Turkey, and India utilizing them for educational and research purposes.

Commercially, the framework has drawn interest from agricultural technology start-ups seeking to integrate the classification engine into digital farming platforms for seed authentication and quality monitoring. Its modular, open-source nature facilitates adaptation into private or public research infrastructures. By uniting reproducible AI modelling with real-world deployment, RiceVision establishes a sustainable blueprint for how software-driven machine learning can reshape precision agriculture, traceability, and biodiversity preservation at scale.

# Conclusions

RiceVision presents an open, scalable, and practical solution for automated rice variety identification through deep learning. By integrating fine-tuned CNN ensembles within accessible web and mobile applications, it bridges the gap between research models and real-world agricultural use. The software, trained on the RiceNet-62 dataset, delivers reliable, real-time classification of 62 rice varieties without specialized hardware. Its open-source nature and dual deployment architecture encourage reproducibility, dataset expansion, and collaborative improvement. Future work will focus on enriching the system with detailed agronomic information, expanding datasets through federated learning, and integrating explainable AI for transparent decision support in smart agriculture.

**Acknowledgements***Not Applicable*

# References

*If the software repository you used supplied a DOI or another Persistent IDentifier (PID), please add a reference for your software here. For more guidance on software citation, please see our guide for authors or this* [*article on the essentials of software citation*](https://f1000research.com/articles/9-1257/v2) *by FORCE 11, of which Elsevier is a member.*