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