Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	20 October 2023
Team ID	PNT2022TMID591889
Project Name	**Dog Breed Identification using Transfer Learning**
Maximum Marks	4 Marks

Table-1: Components and Technologies:

Component	Description	Technology
Frontend Interface	HTML, CSS, JavaScript	User interface for uploading images and displaying results.
Backend Server	Python (Flask/Django)	Handles incoming requests, image processing, and communicates with the ML model.
Image Processing	OpenCV	Library for image preprocessing tasks like resizing, normalization, and augmentation.
Deep Learning Library	TensorFlow/Keras	Used for building, training, and deploying the deep learning model.
Pre-trained Model	Pre-trained CNN (e.g., VGG16, ResNet)	Utilized for feature extraction in transfer learning.
Model Training Data	Dog Breed Dataset	Dataset containing labeled dog breed images for fine-tuning (optional).
Model Deployment	TensorFlow Serving, Flask API	TensorFlow Serving for efficient model serving. Flask for API endpoint creation

Version Control	Git	Manages codebase versions and facilitates collaborative development.
Monitoring/Logging	ELK Stack, Prometheus/Grafana	For monitoring user interactions, model performance, and system health (optional).
Continuous Integration/Continuous Deployment (CI/CD)	Jenkins, GitLab CI/CD, Travis CI	Automates testing, building, and deploying the application.

Table-2:Application Characteristics:

Characteristic	**Description**	**Technology**
Image Processing Intensive	Heavy reliance on image processing techniques for data preparation.	OpenCV
Machine Learning-Powered	Core functionality driven by a machine learning model.	TensorFlow/Keras
Transfer Learning	Utilizes pre-trained models for feature extraction.	Pre-trained CNN models
Real-Time Interaction	Provides instant feedback to users upon image upload.	JavaScript (for frontend updates)
User-Centric	Focuses on providing an intuitive, user-friendly interface	HTML, CSS, JavaScript
Feedback Loop	Incorporates a mechanism for users to report incorrect predictions.	Backend API for user feedback
Modular Architecture	Designed with modularity for potential future updates.	Microservices architecture
Scalability Considerations	Designed to handle potential high user volumes.	Cloud hosting (AWS, GCP, Azure)

Adaptability to Devices	Ensures a seamless experience across various devices.	Responsive design techniques
Security Measures	Implements measures to protect user data and ensure safe usage.	Encryption, authentication, access controls