Final Project Report

Dog Breed Identification using Transfer Learning

**1. Introduction**

**1.1. Project Overview**

The Dog Breed Classification project aims to leverage machine learning and advanced transfer learning techniques to accurately identify the breed of a dog from an image. By utilizing a Kaggle dataset containing thousands of labeled dog images, the project will train and evaluate models to ensure robust breed identification. The ultimate goal is to deploy the model as a user-friendly web application, making it accessible for veterinarians, rescue organizations, and pet owners.

**1.2. Objectives**

The primary objective of the Dog Breed Classification project is to develop a machine learning model capable of accurately identifying the breed of a dog from an image using advanced transfer learning techniques.

**2.**

**Milestone 1: Project Initialization and Planning Phase**

**Activity 1: Define Problem Statement**

With the increasing popularity of pet ownership, accurately identifying dog breeds can assist veterinarians, rescue organizations, and pet owners. Manual identification can be error-prone and time-consuming, especially with mixed breeds or less common breeds.

**Activity 2: Project Proposal (Proposed Solution)**

The project employs transfer learning with pretrained models on a Kaggle dataset of 10,222 dog images. Data preprocessing includes standardization and augmentation. Model performance is evaluated using metrics like accuracy, aiming for robust breed identification. Deployment as a user-friendly web app enhances accessibility and usability.

- Use of multiple pre-trained models for robust feature extraction.

- Web-based deployment for user-friendly interaction.

**Activity 3: Initial Project Planning**

| **Sprit** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Data Collection | DBI-4 | Collection of Data | 2 | High | Diya |
| Sprint-1 | Data Collection | DBI-5 | Loading the Data | 4 | Low | Diya |
| Sprint-2 | Data Preprocessing | DBI-7 | Understanding the data | 3 | Medium | Diya |
| Sprint-2 | Data Preprocessing | DBI-8 | Classifying the data | 5 | Medium | Diya |
| Sprint-3 | Model Building | DBI-10 | Importing the Model Building Libraries | 2 | High | Akshay |
| Sprint-3 | Model Building | DBI-11 | Importing the Models | 3 | High | Akshay |
| Sprint-3 | Model Building | DBI-12 | Initialization the Models | 3 | High | Tanmay |
| Sprint-3 | Model Building | DBI-13 | Adding Fully Connecting Layers | 5 | Medium | Tanmay |
| Sprint-3 | Model Building | DBI-14 | Configure the Learning Process | 5 | Medium | Tanmay |
| Sprint-3 | Model Building | DBI-15 | Train the Models | 8 | High | Swarit, Tanmay |
| Sprint-3 | Model Building | DBI-16 | Save the Models | 2 | Medium | Swarit |
| Sprint-3 | Model Building | DBI-17 | Test the Models | 5 | High | Swarit, Tanmay |
| Sprint-4 | Application Building | DBI-19 | Create HTML Pages | 3 | Low | Akshay |
| Sprint-4 | Application Building | DBI-20 | Build Python Code | 5 | High | Diya |
| Sprint-4 | Application Building | DBI-21 | Importing Libraries | 2 | High | Akshay |
| Sprint-4 | Application Building | DBI-22 | Creating Our Flask Application and Loading Our Model by Using Load\_model Method | 8 | High | Swarit |
| Sprint-4 | Application Building | DBI-23 | Routing to the Html Page | 3 | Medium | Swarit |
| Sprint-4 | Application Building | DBI-24 | Run the Application | 2 | Medium | Swarit |
| Sprint-5 | Documentation | DBI-26 | Project Documentation | 3 | Medium | Diya |

**3.**

**Milestone 2: Data Collection and Preprocessing Phase**

**Activity 1: Data Collection Plan and Raw Data Sources Identified**

Data for this project will be sourced primarily from publicly available datasets. The main dataset will be obtained from the Kaggle competition "Dog Breed Identification," which provides a large collection of labeled dog images. This dataset is ideal for transfer learning as it contains a diverse set of images across various dog breeds.

**Raw Data Sources Identified**

This dataset consists of images of dogs labeled with their corresponding breeds. The dataset includes a training set with labeled images and a test set with unlabeled images. The training set will be used to train the model, while the test set will be used to evaluate its performance.

- Contents:

- train/: A folder containing training images.

- test/: A folder containing test images.

- labels.csv: A CSV file containing image IDs and their corresponding breed labels.

**Activity 2: Data Quality Report**

The Kaggle Dog Breed Identification dataset, used for this project, presents several data quality issues that need to be addressed. Firstly, the dataset may have missing labels, which is a moderate severity issue. To resolve this, we will verify the integrity of the dataset by checking for any missing or null values in the labels file. Using pandas, we can identify and handle missing labels by either removing the corresponding images or imputing the missing values if applicable.

Secondly, there is variability in image sizes, which is a low severity issue. To ensure uniform input dimensions for the neural network, we will resize all images to a consistent size of 224x224 pixels using the `ImageDataGenerator`.

Thirdly, the dataset shows variability in image quality, a moderate severity issue. To address this, we will apply image preprocessing techniques such as normalization to scale pixel values and possibly use denoising filters to improve image quality.

Fourthly, the high dimensionality of the images poses a high severity issue. We will use feature extraction from pre-trained models such as VGG19, EfficientNetB7, and MobileNetV2 to reduce the dimensionality of the images and focus on the most informative features.

Lastly, the dataset is imbalanced, which is the least severe issue. To counter this, we will use data augmentation techniques such as rotation, shifting, zooming, and flipping to artificially increase the number of images for underrepresented classes. This will help in creating a more balanced dataset for training the model.

**Activity 3: Data Preprocessing**

The dataset used in this project comprises images of dogs labeled with their corresponding breeds. The primary dataset is sourced from the Kaggle Dog Breed Identification competition. It includes a diverse collection of dog images spanning multiple breeds, which will be used for training and evaluating the dog breed identification model.

- Dimension:

- Training Set: 10,222 images

- Test Set: 10,357 images

- Number of Classes: 120 dog breeds

- Feature Dimension: Images resized to 224x224 pixels with 3 color channels (RGB)

**4.**

**Milestone 3: Model Development Phase**

**Activity 1: Model Selection Report**

**Model 1: MobileNetV2**

MobileNetV2 is a deep convolutional neural network architecture designed for mobile and embedded vision applications. It emphasizes efficiency and small model size while maintaining good accuracy in various computer vision tasks like image classification and object detection.

**Model 2: VGG19**

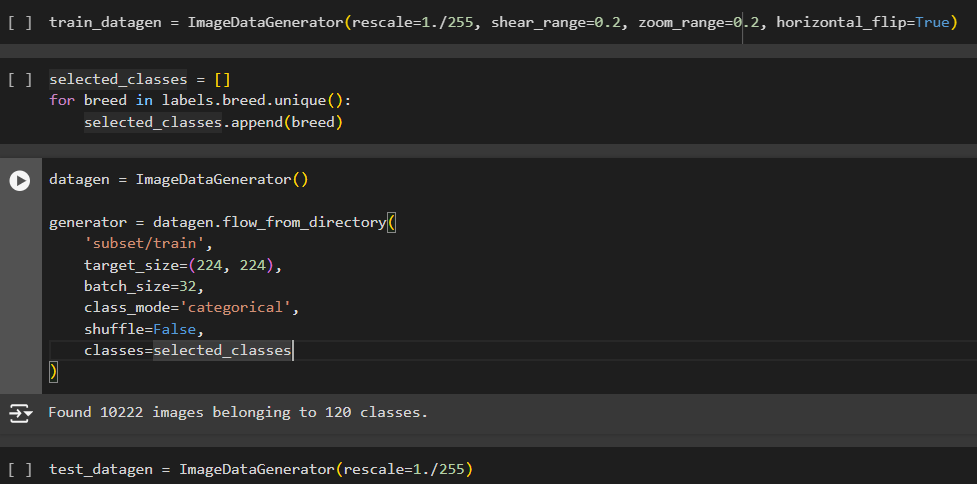
VGG-19 is a deep convolutional neural network architecture known for its simplicity and effectiveness. It consists of 19 layers (16 convolutional and 3 fully connected layers) and has been widely used as a benchmark in image classification tasks due to its strong performance and straightforward architecture.

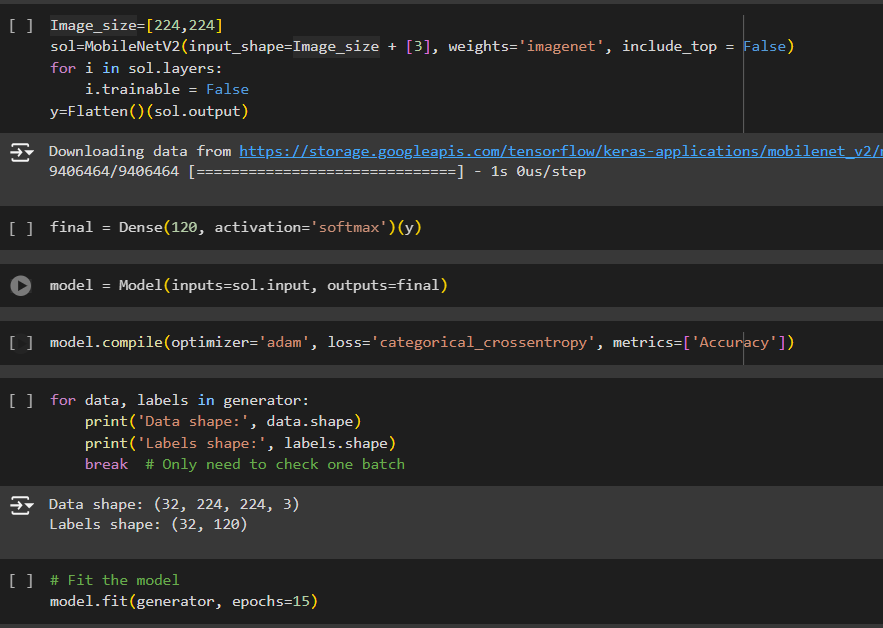
**Model 3: EfficientNetB7**

EfficientNet-B7 is part of the EfficientNet family of neural network architectures, which are designed to achieve state-of-the-art accuracy with increased efficiency. EfficientNet-B7 specifically is one of the largest models in the family, capable of achieving high accuracy while being computationally efficient, making it suitable for a wide range of tasks including image classification and object detection.

**Activity 2: Initial Model Training Code, Model Validation and Evaluation Report**

**Initial Model Training Code (5 marks):**





**Model Validation and Evaluation Report (5 marks):**

|  |  |  |
| --- | --- | --- |
| **Model** | **Summary** | **Training and Validation Performance Metrics** |
| Model 1  MobileNetV2 |  |  |
| Model 2  VGG19 |  |  |
| Model 3  EfficientNetB7 |  |  |

**5.**

**Milestone 4: Model Optimization and Tuning Phase**

**Activity 1: Tuning Documentation**

|  |  |
| --- | --- |
| **Model** | **Tuned Hyperparameters** |
| **EfficientNetB7** | 1. Rescale: 1./255    * Normalizes the image pixel values to the range [0, 1]. 2. Shear Range: 0.2    * Randomly applies shearing transformations to the images. 3. Zoom Range: 0.2    * Randomly zooms inside images. 4. Horizontal Flip: True    * Randomly flips half of the images horizontally. 5. Target Size: (224, 224)    * The size to which all images are resized. 6. Batch Size: 32    * The number of images to be used in each batch for training. 7. Optimizer: Adam    * Optimizer used to minimize the loss function. 8. Loss Function: categorical\_crossentropy    * Loss function used for multi-class classification problems. 9. Epochs: 15    * The number of times the entire dataset is passed through the network during training. 10. **Activation**: softmax     * Activation function used in the output layer for multi-class classification. |
| **vgg19** | 1. Learning Rate: Defined by the Adam optimizer with default settings. 2. Batch Size: 32 (used in ImageDataGenerator). 3. Epochs: 15 (defined in the model.fit function). 4. Image Size: 224x224 pixels (used in ImageDataGenerator and load\_img). 5. Pre-trained Model: VGG19 (imported and used as base model). |
| **MobileNet** | 1. Image Size: The target size of the input images used for the VGG19 model. 2. Pre-trained Weights: Specifies that the model uses pre-trained weights from ImageNet. 3. Trainable Layers: Sets all layers of MobileNet to be non-trainable to use the pre-trained features. 4. Output Layer Activation: The activation function used in the final output layer, set to 'softmax' for multi-class classification |

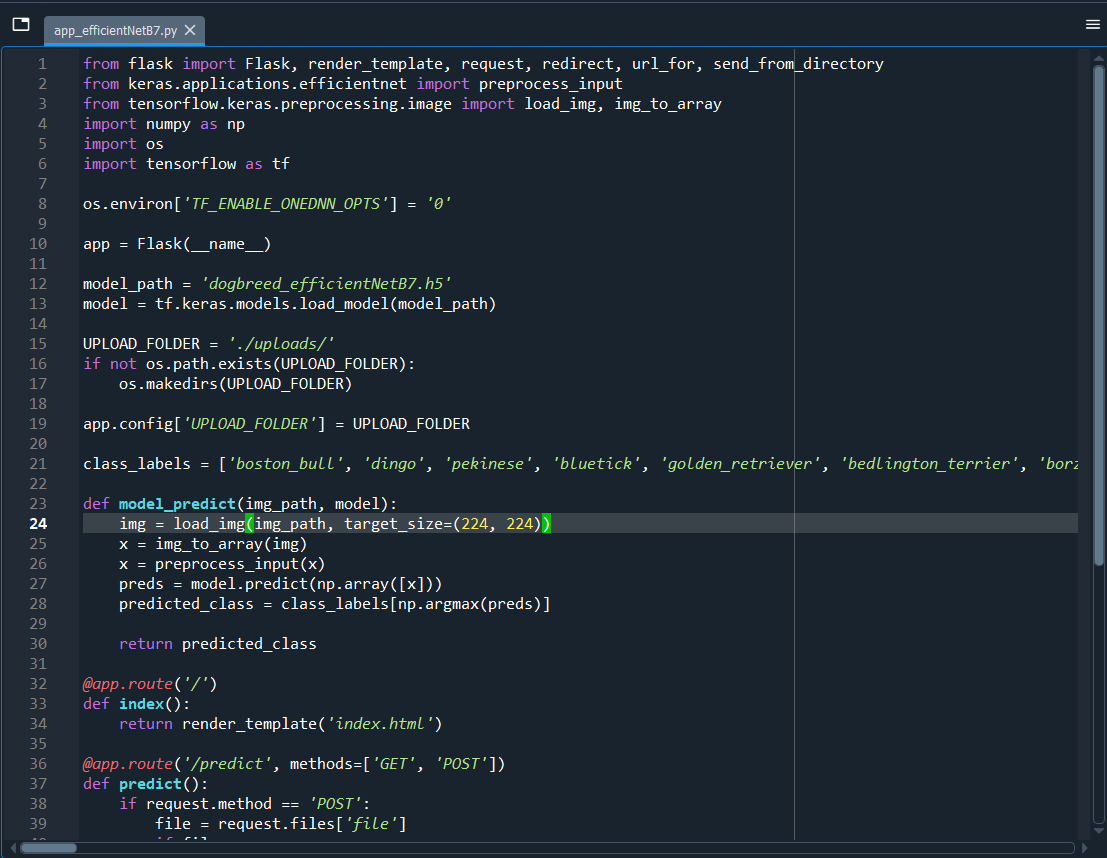
**Activity 2: Final Model Selection Justification**

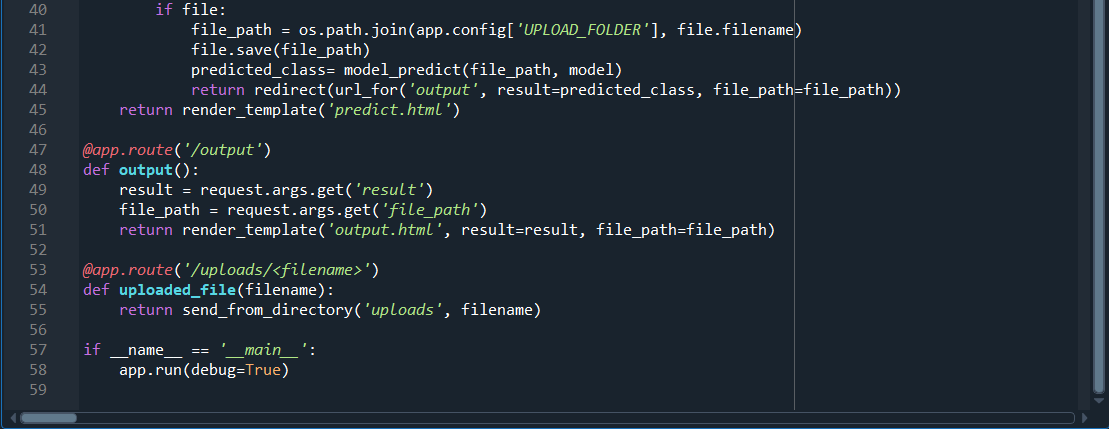
|  |  |
| --- | --- |
| **Final Model** | **Reasoning** |
| **EfficientNetB7** | * EfficientNetB7 is known for its superior performance on a variety of image classification benchmarks. It achieves high accuracy with relatively fewer parameters compared to other models. * The EfficientNet architecture scales efficiently in terms of depth, width, and resolution, allowing it to perform well on large datasets while maintaining computational efficiency. * EfficientNetB7 comes pre-trained on ImageNet, providing a strong feature extraction capability which is beneficial for fine-tuning on the dog breed identification dataset. * Despite being a deep model, EfficientNetB7 is optimized for both speed and memory usage, making it practical for real-world applications where computational resources might be limited. * **We chose the model because for specific images nearly every prediction was correct for this model which was not much accurate in other models.** |

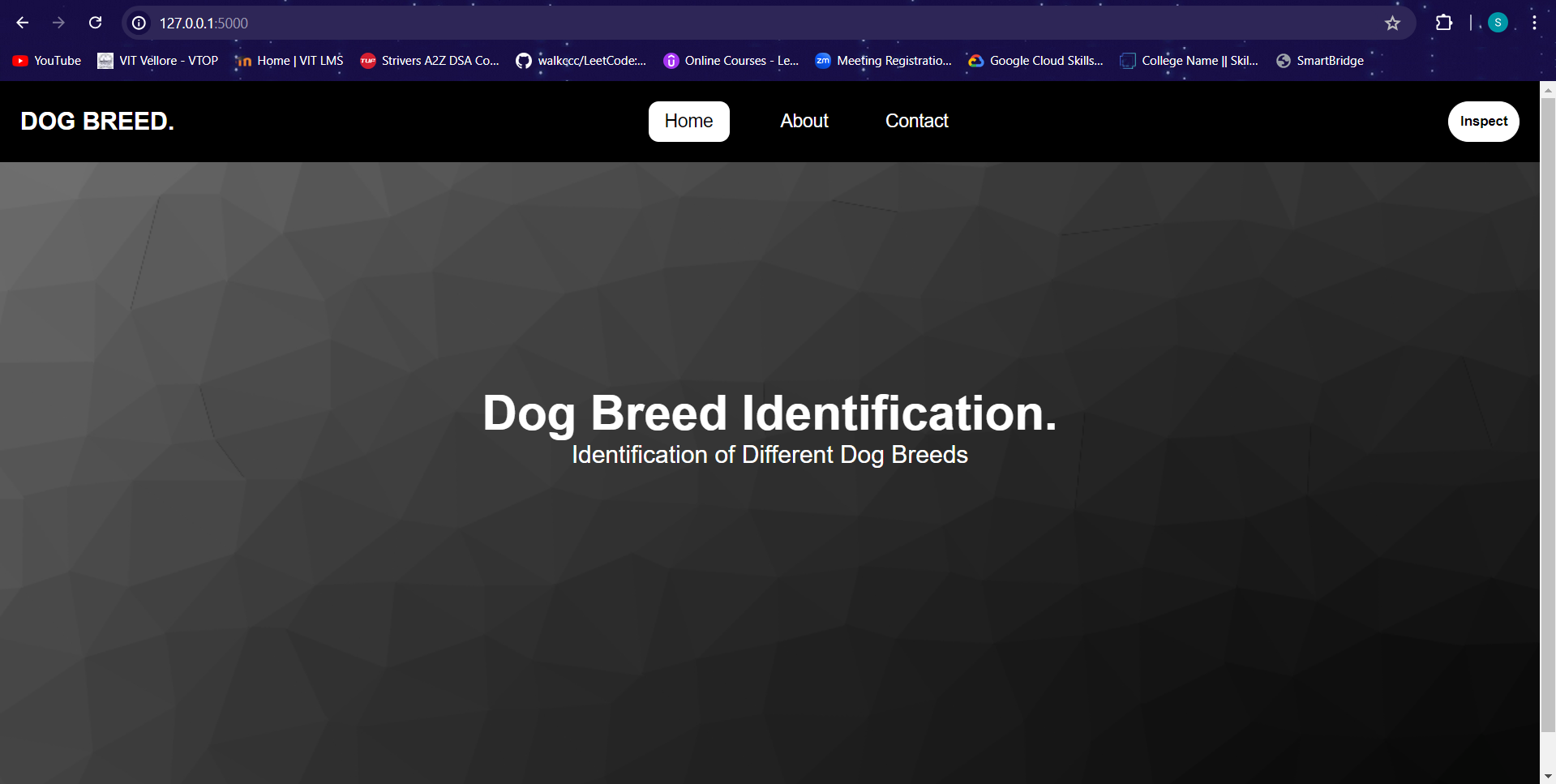
**6.**

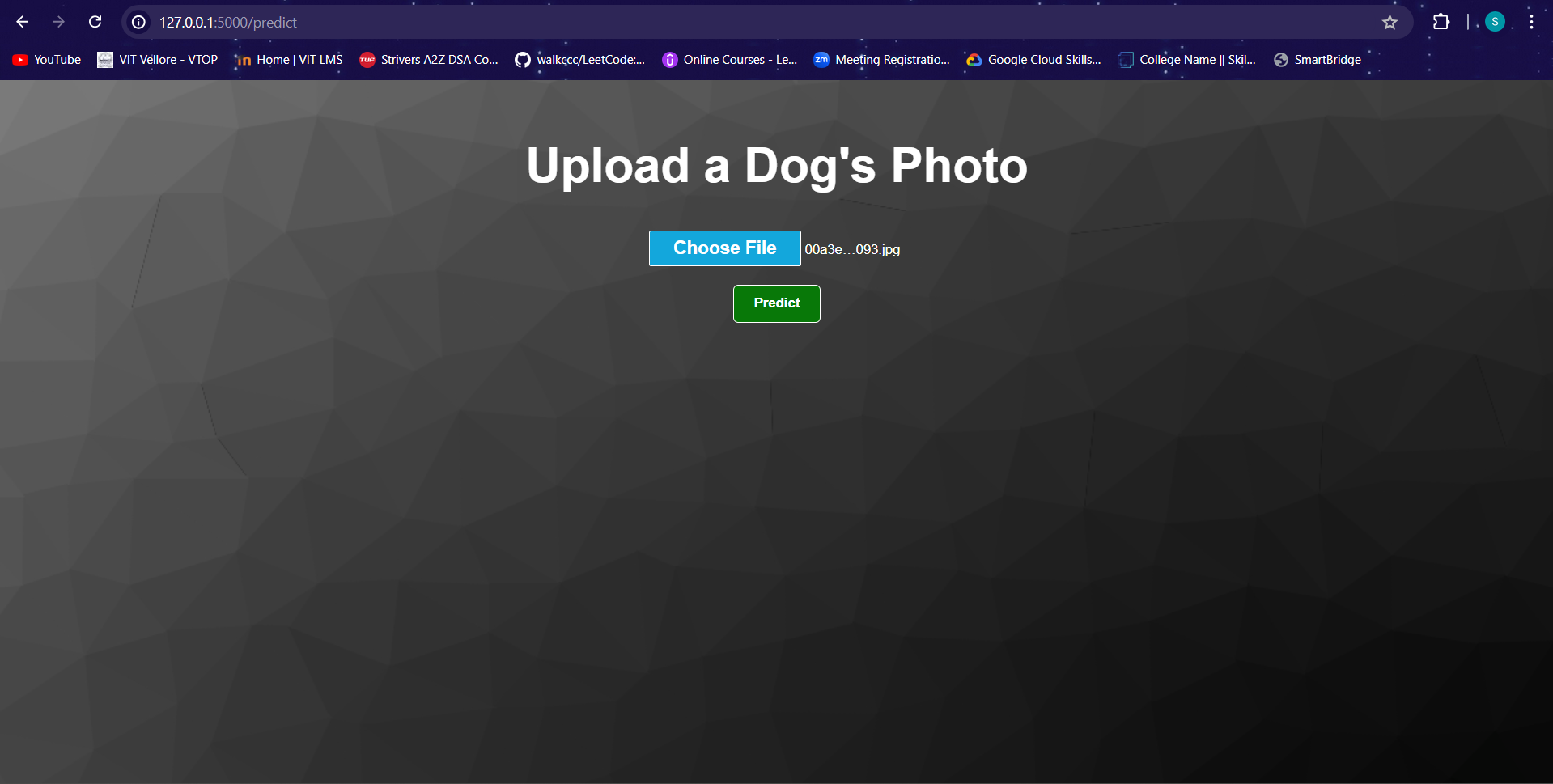
**Milestone 5: Results**

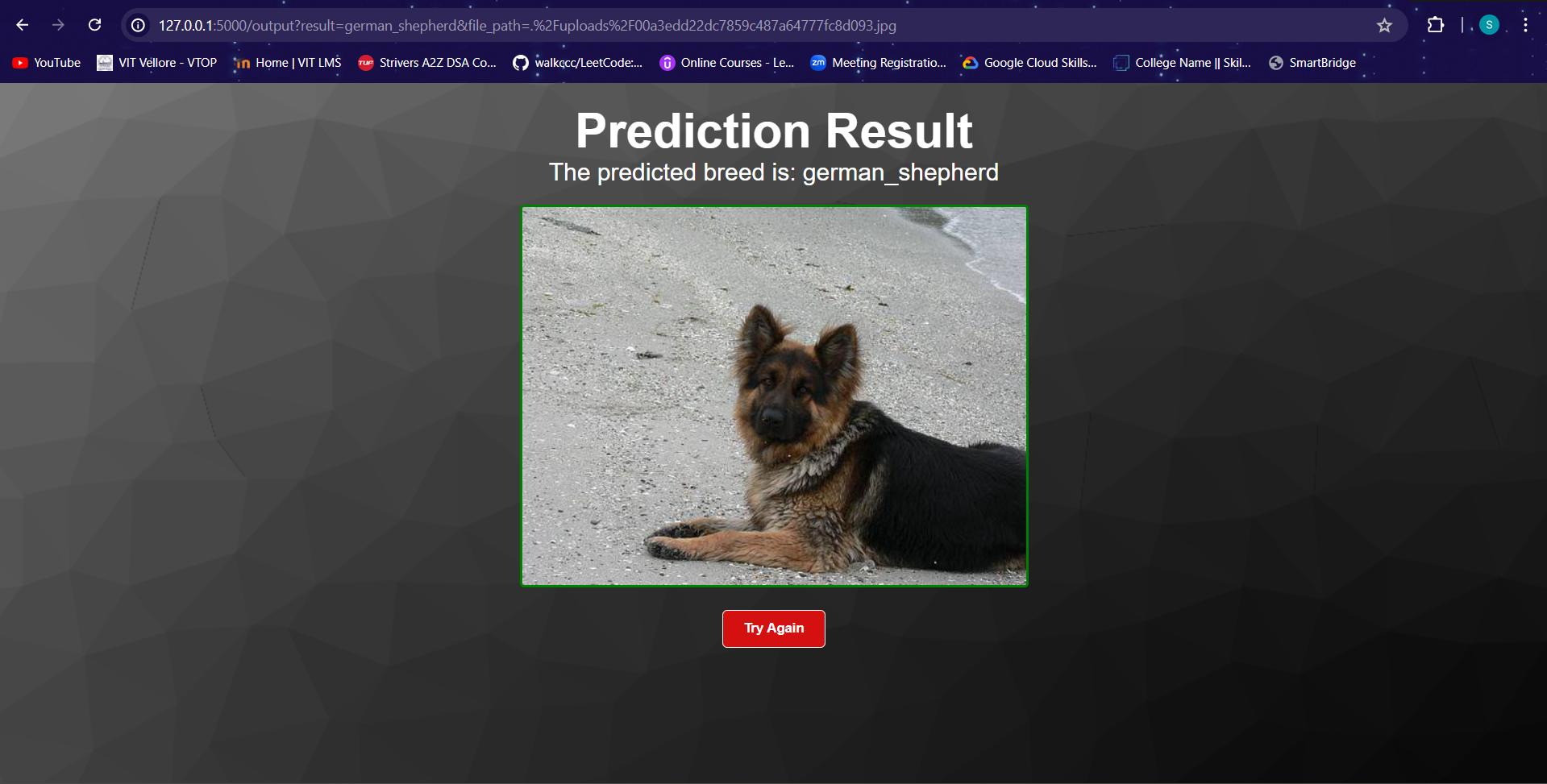
(Screenshots for the output of the Flask App attached below)











**7.** **Advantages & Disadvantages**

**Advantages:**

- Accuracy: Utilizing advanced transfer learning techniques with pre-trained models ensures high accuracy in breed identification.

- Efficiency: Pre-trained models like MobileNetV2 and EfficientNetB7 offer efficient computation, making them suitable for deployment on various devices.

- User-Friendly: The web-based deployment provides an accessible interface for users to identify dog breeds easily.

**Disadvantages:**

- Data Dependency: The model’s performance is heavily reliant on the quality and diversity of the training data.

- Resource Intensive: Training deep learning models can be resource-intensive, requiring significant computational power and memory.

- Maintenance: Keeping the model updated with new data and breeds can be challenging and time-consuming.

**8.** **Conclusion**

The Dog Breed Classification project successfully demonstrates the application of advanced machine learning techniques to accurately identify dog breeds from images. By leveraging pre-trained models and a robust dataset, the project provides a reliable tool for veterinarians, rescue organizations, and pet owners. The deployment as a web application ensures accessibility and usability.

**9.** **Future Scope**

- Model Improvement: Continuously improving the model by incorporating new data and exploring more advanced architectures.

- Mobile Application: Developing a mobile application version for easier access and usability on handheld devices.

- Integration with Veterinary Systems: Integrating the model with veterinary systems to assist in clinical decision-making and pet care management.

- Real-time Identification: Enhancing the model to support real-time breed identification through live camera feeds.

**GitHub & Project Demo Link**

- GitHub Repository: <https://github.com/sjainpropy9724/Dog-Breed-Identification-using-Transfer-Learning>

- Project Demo: <https://drive.google.com/file/d/1JLaK8uAg_qrk32SacO3nw8unJ6niehCO/view?usp=sharing>