Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	9th November 2023
Team ID	592903
Project Name	Detecting COVID-19 from chest X-rays
Maximum Marks	8 marks

Technical Architecture

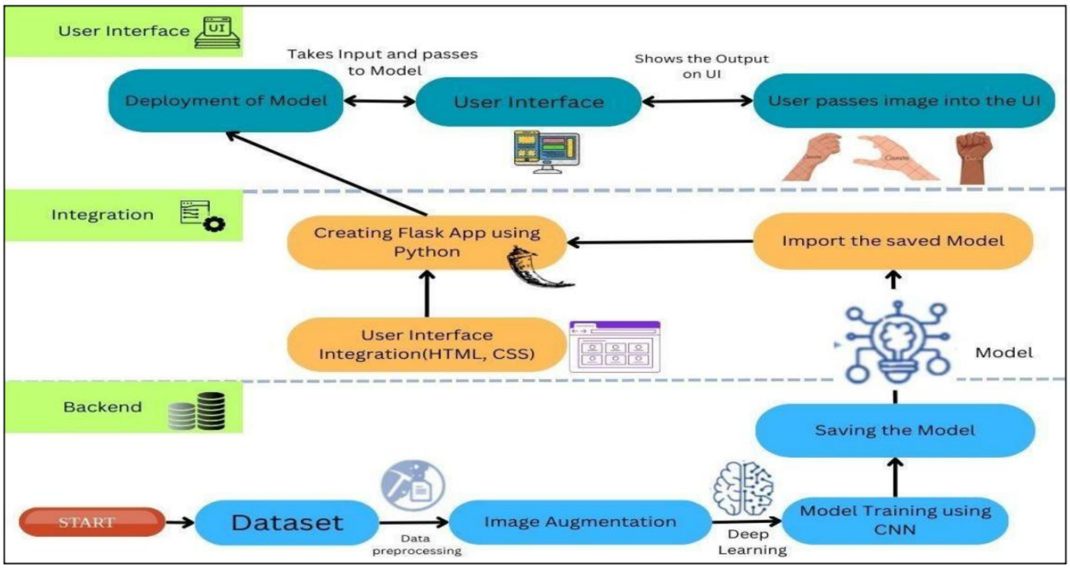

Table-1 : Components & Technologies

S.NO.	Component	Description	Technology
1.	Image Acquisition	Acquire or gather pictures depicting Human Lungs images	Image data-bases, smart phones or Digital cameras.
2.	Pre-Processing	preprocess the images for recognition by applying a range of transformations and enhancements.	open CV(Computer Vision Library),python.
3.	Image Segmentation	Isolate hand signs from the background and, if required, identify individual fingers for enhanced recognition.	open CV, Image processing Techniques.
4.	Feature Extraction	Capture relevant characteristics from the segmented images, encompassing aspects like shape, color and texture.	Feature extraction algorithms (e.g., Histogram of Oriented Gradients, Color Histograms), Python.
5.	Machine Learning Model	Develop a machine learning model capable of identifying X-ray images using features extracted from the data.	Tensor Flow, PyTorch, Scikit-Learn, Keras, or a custom model using deep learning or traditional machine learning algorithms.

6.	Training Data	A dataset of labelled X-ray images for model training	Human lungs X-ray images, data augmentation techniques.
7.	Model Evaluation	Evaluate the model's performance by examining metrics such as accuracy, precision, recall, F1 score, and other pertinent indicators	Cross-validation, evaluation metrics in Python.
8.	Model Deployment	Deploy the trained model for the real-time or batch processing of COVID-19 X-ray images in a deployed environment.	Cloud platforms (e.g., AWS, Azure, GCP), web servers, APIs.
9.	User Interface	Design a user-friendly interface that allows users to interact effortlessly with COVID-19 detection system.	Web development (HTML, CSS, JavaScript),
10.	Infrastructure (Server / Cloud)	Consistently enhance and refine the system through the collection of user feedback, ensuring regular updates to improve the model.	Agile development practices, versioncontrol (e.g., Git).

Table-2: Application Characteristics

S.NO.	Characterstics	Description	Technology
1.	open-Source Frameworks	Using open-source frameworks can greatly expedite development, cut down costs, and take advantage of a collaborative community for mutual benefits.	Python for machine learning and image processing (NumPy, OpenCV, Scikit-Learn),
2.	Security Implementations	Ensuring the security of both user data and the system is paramount. Implement a range of security measures to safeguard against data breaches and unauthorized access.	SL/TLS for secure data transmission and implement encryption for data at rest
3.	Scalable Architecture	Architecting the system to accommodate growing loads and user demands by scaling horizontally or vertically as required.	Docker for packaging applications and Kubernetes for container orchestration, Auto-scaling on cloud platforms to dynamically allocate resources based on demand, tools like Nginx or HA Proxy for distributing traffic across multiple instances.
4.	Availability	Ensuring that the system is always accessible and minimizes downtime.	Setting up failover mechanisms and replicate critical components for high availability, tools like AWS Cloud Watch to monitor system health and performance. Implement backup and recovery strategies to restore the system in case of failures. Use CDNs to distribute content and reduce latency.
5.	Performance	Enhance system performance by optimizing for rapid response times and efficient utilization of resources.	Implement caching mechanisms (e.g., Redis, Memcached) for frequently accessed data. Tools like Python's cProfile to identify bottle necks. Apache Spark or Hadoop for distributed data processing.