NAME : HARI HARAN D
ROLL no : 7376222AD140
ROLE : AI - ENGINEER

PROJECT DESCRIPTION :

The task I was given was to build an AI-based system that can classify chest X-ray images into **Normal** or **Pneumonia** using deep learning. The goal is to preprocess the dataset, train a model using CNNs and transfer learning, evaluate it with medical metrics, and finally discuss how such a model can be deployed in a real hospital PACS workflow. To solve this, I explored the dataset, applied proper preprocessing and augmentation, created patient-level splits, and implemented both a baseline CNN and a transfer-learning model using PyTorch. I trained and evaluated the model using metrics like AUC, sensitivity, specificity, and confusion matrix. I also used Grad-CAM to make the model explainable for clinical use. Finally, I designed a deployment strategy that explains how the model can fit into hospital systems while maintaining privacy and monitoring model drift over time.

1 . DATA COLLECTION & INGESTION :

- For this project, I start by collecting chest X-ray images that fall under two categories: **Normal** and **Pneumonia**.
- I prefer using the **Kaggle Chest X-Ray Pneumonia dataset** because it is widely used, clean, and already organized.

2 . DATA EXPLORATION :

- I check image sizes, resolution, channels.
- I verify class balance (this dataset usually has more pneumonia cases).
- I look for corrupted or inconsistent images.

- If patient IDs are available, I ensure that images from the same patient do not leak into multiple splits.

3 . DATA PREPROCESSING :

Handling PNG/JPG

- Convert images to grayscale for consistency
- Resize them to **224×224**
- Normalize pixel intensity values

Handling DICOM

- Load pixel arrays
- Apply windowing (if necessary)
- Convert to PIL Image for transforms

4 . DATA AUGMENTATION :

To avoid overfitting and improve generalization, I apply:

- Random rotations (up to ±15°)
- Random horizontal flips
- Random brightness/contrast adjustments
- Random resized crops

These augmentations help the model see slight variations while still keeping the clinical meaning intact.

5 . SPLITTING THE DATASET :

I strictly follow **patient-level splitting**:

- **70%** Train
- **15%** Validation

- **15%** Test

I make sure the same patient's images do not appear in multiple sets.

6 . PREPARING DATALOADERS :

I prepare PyTorch DataLoaders with:

- Train loader → with augmentation
- Validation/Test loader → only normalization and resizing

This ensures the training set is varied, while validation/testing remains consistent.

7 . MODEL DEVELOPMENT :

A. Baseline CNN

I begin with a simple CNN model to confirm my pipeline works correctly — this acts as a sanity check.

B. Transfer Learning (My main model)

For high performance, I use ResNet18, ResNet50, or DenseNet121.
I replaced the final layer with a binary classifier.

My fine-tuning strategy:

1. Freeze backbone → train classifier head
2. Unfreeze backbone → fine-tune whole model at lower LR

This approach gives strong performance with minimal training time.

8 . MODEL TRAINING :

- Loss: **BCEWithLogitsLoss**
- Optimizer: **AdamW**

- Learning rate:
    - 1e-3 for classifier head
    - 1e-4 during fine-tuning
- Batch size: **16–32**
- Epochs: **20–40**
- Early stopping using validation AUC

I save the best checkpoint based on validation metrics.

9 . MODEL EVALUATION :

After training, I compute:

- Accuracy
- AUC/ROC
- Sensitivity (recall on Pneumonia → important clinically)
- Specificity (recall on Normal)
- Confusion Matrix

These metrics give a good balance between clinical and technical evaluation.

10 . EXPLAINABILITY - Grad-CAM:

I generate Grad-CAM heatmaps to understand what parts of the lung the model is focusing on.

- I overlay the heatmaps on the original X-ray
- I check if the model is looking at the correct lung regions
- This is especially important for medical AI models

11 . DEPLOYMENT :

I convert the final model into a format suitable for deployment:

- **TorchScript**
- **ONNX**

Then I wrap the model in an inference service (REST API or DICOM-compatible service).

12 . WORKFLOW :

Here is how I integrate the model into a hospital environment:

1. X-ray is captured → stored in **PACS**
2. A DICOM router (Orthanc / DCM4CHE) sends a copy to my AI model server
3. My model processes the image
4. Output includes:
    - Prediction (Normal / Pneumonia)
    - Probability score
    - Grad-CAM heatmap
5. The result is returned to PACS as:
    - DICOM SR
    - DICOM SEG overlay
6. Radiologists view predictions inside their normal workflow

13 . MONITORING :

Input drift

Changes in image quality, resolution, or machine type.

Prediction drift

Unexpected changes in predicted class distribution.

Performance drift

Periodic evaluation using newly labeled clinical data.

When drift occurs, I:

- Gather new training samples

- Retrain or fine-tune the model
- Deploy updates safely (shadow mode → full mode)

14 . FUTURE PLANS :

1. Add segmentation (U-Net)
2. Include patient metadata (age, gender, symptoms)
3. Improve augmentations
4. Build a multi-modal model.