**Part 1: Model Training**

1. **Set up directories**:
   * Define the paths for your training and validation datasets.
2. **Image Preprocessing**:
   * Use ImageDataGenerator for real-time data augmentation and normalization.
   * For the training dataset:
     + Rescale pixel values to the range [0, 1].
     + Apply augmentations like rotation, shifting, zoom, etc.
   * For the validation dataset:
     + Only rescale pixel values without augmentation.
3. **Load Data**:
   * Load and prepare the training and validation images using flow\_from\_directory, specifying:
     + Image size (150x150 pixels).
     + Batch size (32).
     + Class mode (binary for classification between pneumonia and normal).
4. **Build CNN Model**:
   * Construct a Sequential model with:
     + Convolutional layers (Conv2D) with ReLU activation.
     + Pooling layers (MaxPooling2D) to reduce dimensionality.
     + Flatten the output to a 1D vector.
     + Add dense layers with ReLU and a dropout layer to prevent overfitting.
     + Use a final dense layer with a sigmoid activation function for binary classification.
5. **Compile the Model**:
   * Define the optimizer (adam), loss function (binary\_crossentropy), and evaluation metric (accuracy).
6. **Set Callbacks**:
   * Use EarlyStopping to stop training when the validation loss stops improving.
   * Use ModelCheckpoint to save the best model based on validation loss.
7. **Train the Model**:
   * Train the model for 25 epochs, using the training data and validating on the validation dataset.
   * Store the best model in pneumonia\_model.h5.
8. **Save the Model**:
   * Save the trained model manually, if not saved through the checkpoint.

**Part 2: Image Annotation (Prediction)**

1. **Load the Pre-trained Model**:
   * Load the trained model using load\_model().
2. **Load the Image**:
   * Use cv2.imread() to load the chest X-ray image in grayscale format.
3. **Handle File Errors**:
   * Check if the image was loaded correctly. If not, print an error message and stop the process.
4. **Resize Image**:
   * Resize the image to match the model's input dimensions (150x150 pixels).
5. **Convert Grayscale to RGB**:
   * Convert the grayscale image to an RGB image, as the model expects 3 channels.
6. **Preprocess the Image**:
   * Convert the image to an array using img\_to\_array().
   * Normalize the pixel values to the range [0, 1].
7. **Make a Prediction**:
   * Pass the preprocessed image through the model to get a prediction.
   * Based on the prediction, classify the image as either "Pneumonia" or "Normal".
8. **Annotate the Image**:
   * Convert the grayscale image to BGR (for visualization).
   * Display the diagnosis label ("Pneumonia" or "Normal") on the image using cv2.putText().
9. **Highlight Pneumonia Symptoms**:
   * If the label is "Pneumonia", apply a thresholding operation to highlight dense regions in the lung area.
   * Draw contours around the detected regions to indicate affected areas.
10. **Display the Annotated Image**:

* Use matplotlib.pyplot to display the annotated image with the diagnosis.

**Part 3: Example Usage**

* Call the annotate\_image() function, passing the path to an X-ray image for prediction and annotation.