National College of Ireland

Configuration Manual

MSc Research Project
MSc. in Data Analytics

Lekshmi Sasidharan Kandamchirayil
Student ID: 23203625

School of Computing
National College of Ireland

Supervisor: Prof. Anh Duong Trinh

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Lekshmi Sasidharan Kandamchirayil

Student ID: 23203625

Programme: MSc. Data Analytics **Year:** 2024-25

Module: Practicum Research Project Part 2

Lecturer: Prof. Anh Doung Trinh

Submission Due Date: 28 January 2025

Project Title: Lumbar Spine Degenerative detection using ResNet-50 and VGG16

Word Count: 1036 words **Page Count:** 11

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Lekshmi Sasidharan Kandamchirayil

Date: 10 December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Lekshmi Sasidharan Kandamchirayil
Student ID: 23203625

1 Introduction

This configuration manual describes detailed procedure on how to install and implant the DL based framework for LSPD classification. The system leverages the ResNet-50 model to classify MRI images into three severity levels: normal/mild, moderate and severe. The insider's guide outlines the primary settings ranging from installations of programs and software, environment setting, model parameters, and system characteristics. Adherence to the standards provided in this manual helps to minimize the time spent on the setup and maximize the speed of MRI data processing as well as increase the accuracy of predictions of spinal degenerative conditions. This document is aimed at researchers, developers and practitioners, interested in deploying the automated classification model in clinical and/or research practice.

2 Software and Hardware Requirements

2.1 Software Requirements:

1. Operating System:
   - Windows 10 or 11 (64-bit)
   - Operating system: Ubuntu 20.04 or higher (is preferable because it supports most of the popular ML libraries).
   - They require macOS 11 (Big Sur) or higher.
2. Programming Languages:
   - Python 3.8 or higher
3. Development Environment:
   - Visual Studio Code
   - PyCharm
   - Jupyter Notebook
4. Deep Learning Frameworks:
   - TensorFlow 2.x
   - Keras 2.x
5. Additional Libraries:
   - NumPy
   - Pandas
   - OpenCV
   - Matplotlib

- pydicom (only for reading and constructing of DICOM files)
- scikit-learn
- Streamlit (for web-based deployment)

6. Virtual Environment:
   - Anaconda or venv for the package and library management
7. CUDA and cuDNN (for GPU support):
   - CUDA 11.x
   - cuDNN 8.x

2.2 Hardware Requirements

1. Processor (CPU):
   – A latest model processor will be Ideal, such as i7 8th generation & above and similar Ryzen by AMD.
2. Memory (RAM):
   - Minimum: 16 GB
   - Recommended: 32 GB or higher
3. Graphics Processing Unit (GPU):
   This should be accompanied by an NVIDIA GeForce GTX 1080 or higher.
   - Recommended: NVIDIA RTX 3080 or NVIDIA Tesla V100 for a shorter time for model training.
4. Storage:
   - Minimum: 256 GB SSD
   - Recommended: 2 TB HDD (to store large data in database as well as models).
5. Display:
   At least up to Full HD (1920×1080) resolution of video or even higher.
6. Internet Connection:
   Used for downloading libraries and datasets as well as for deploying Streamlit applications.

This setup enables glitch-free performance for training as well as deployment of the ResNet-50 model for MRI image classification.

3 Relevant screenshots

3.1 Installation screenshots

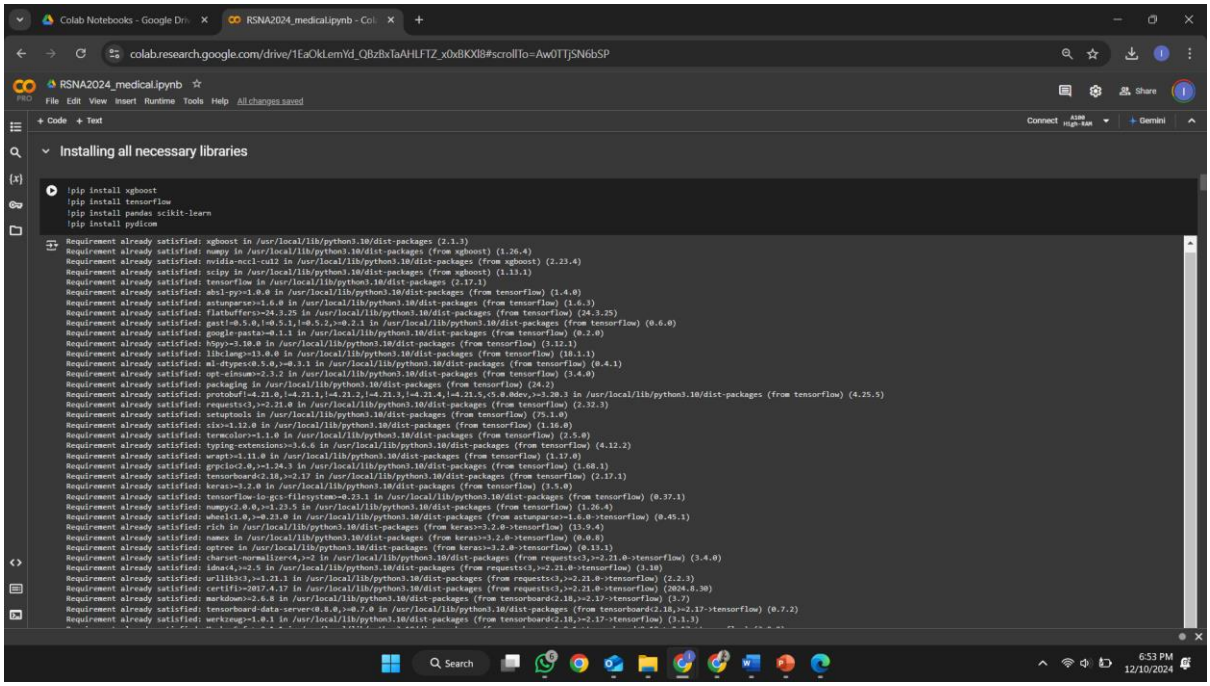

Fig.1 Installing all the necessary libraries

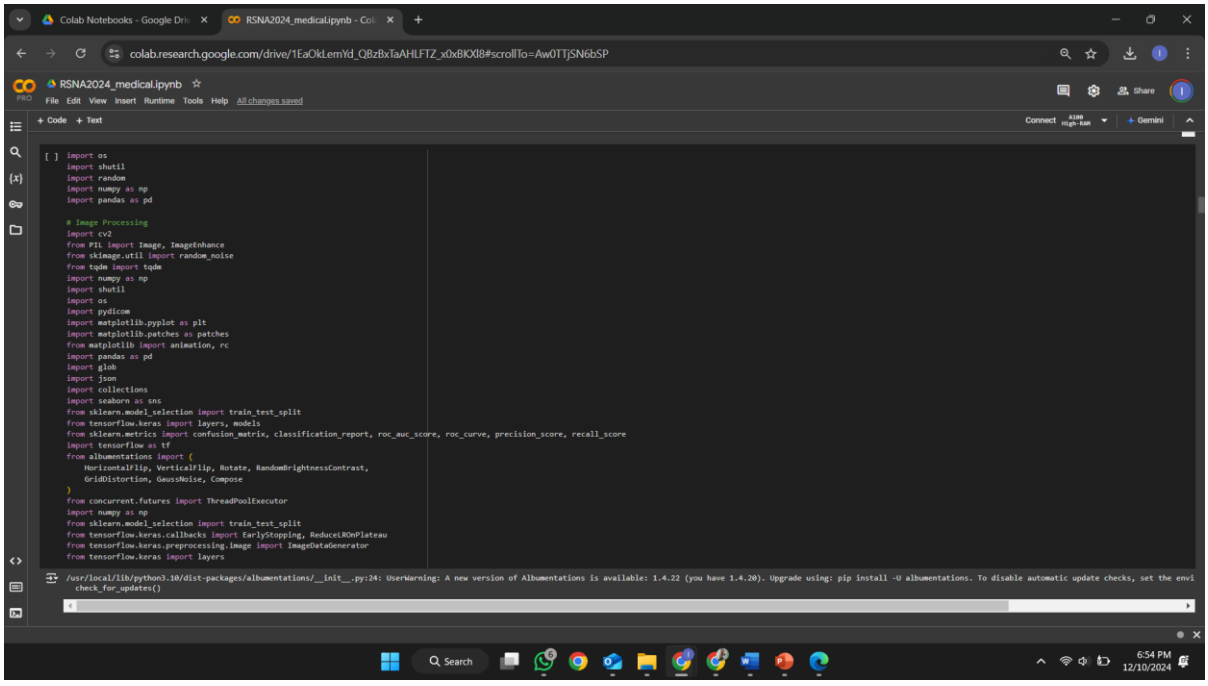

Fig.2 Importing all the necessary libraries for modelling

3.2 IDE Setup

Google Colab is namely a cloud Integrated Development Environment or IDE in a cloud that runs and interacts with user's Python codes. It supports GPU/TPU acceleration, they are best suited for training deep learning models without having a local resource intensive machine. Furthermore, it also allows code sharing and multiple developer collaboration.

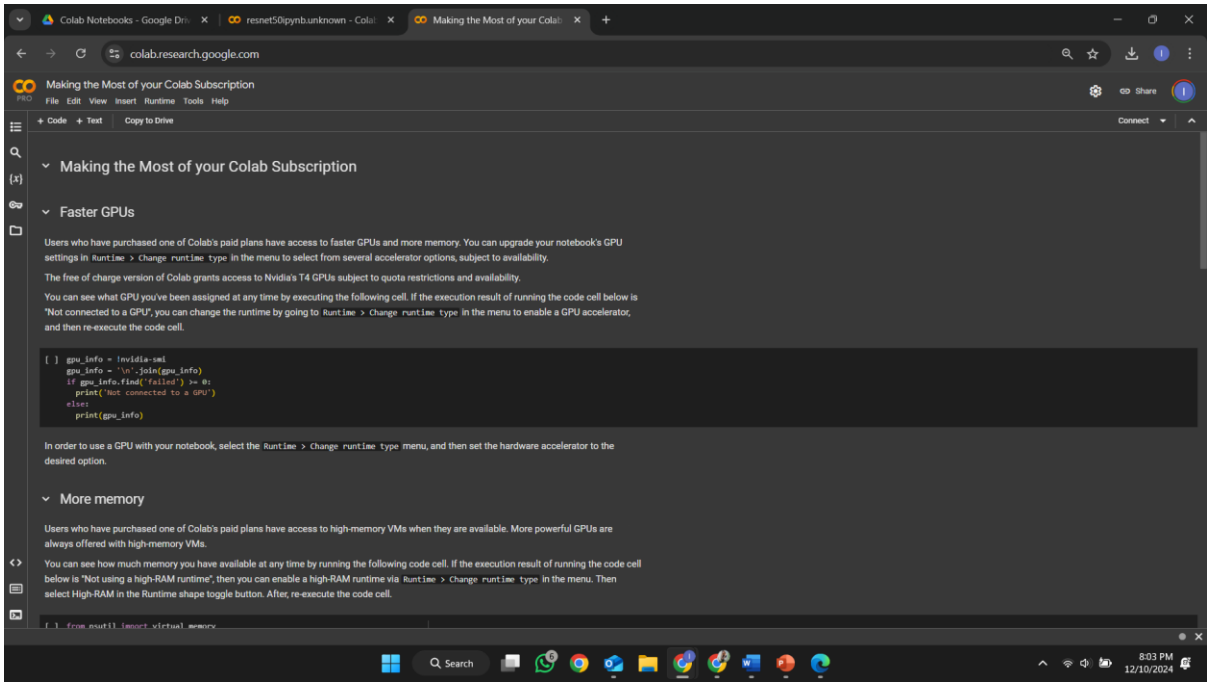

Fig.3 Google Colaboratory screen

3.3 Data Analysis and Preparation

3.3.1 Conversion of images from dicom to PNG

The image conversion process MRI images in DICOM format and converts them into PNG format as it is compactible with the deep learning model used.

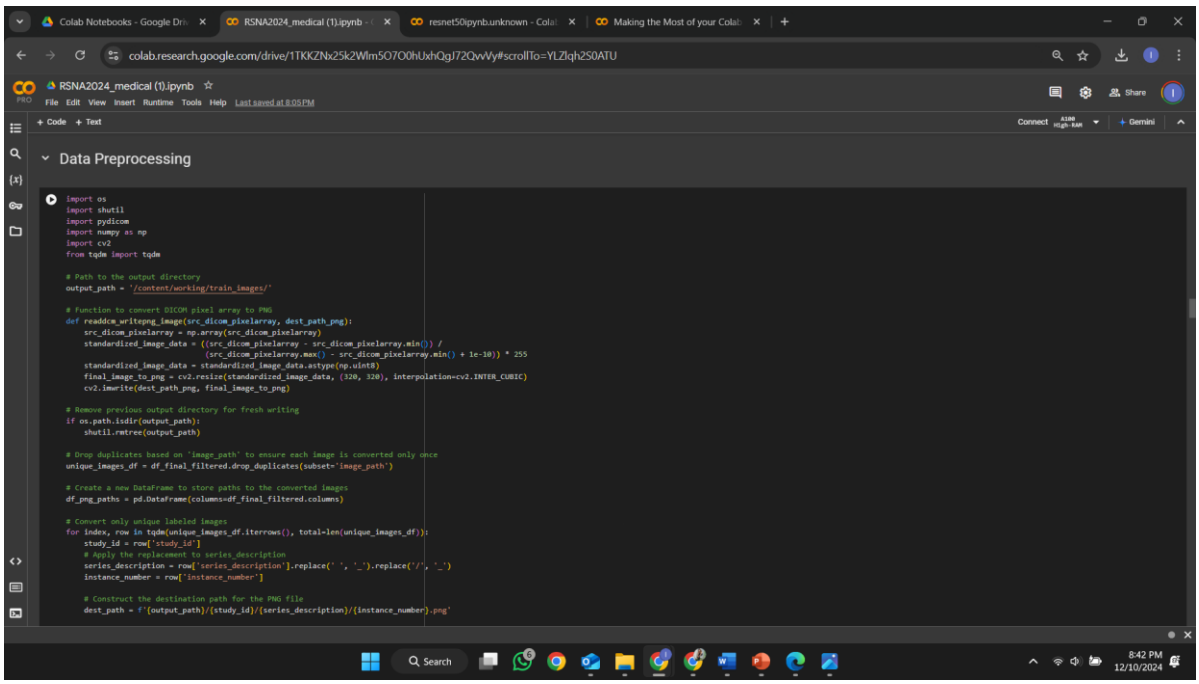

Fig. 4 Code snippet showing image conversion to PNG.

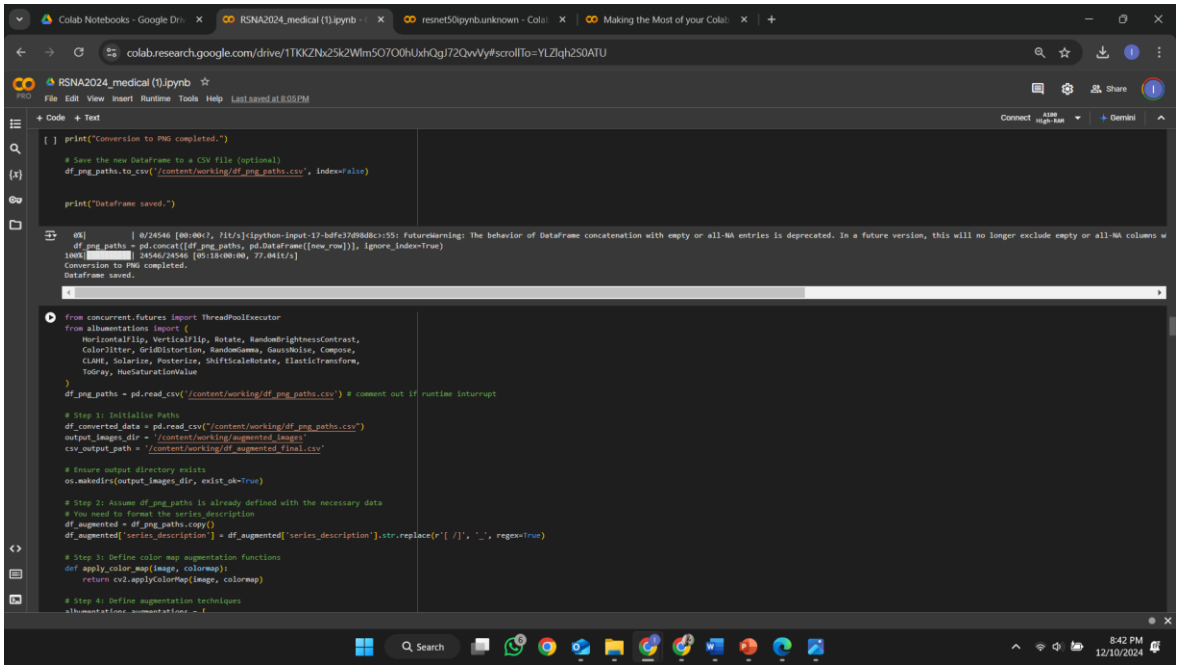

Fig.5 Code snippet showing images to convert to PNG

3.3.2 Handling imbalance in dataset

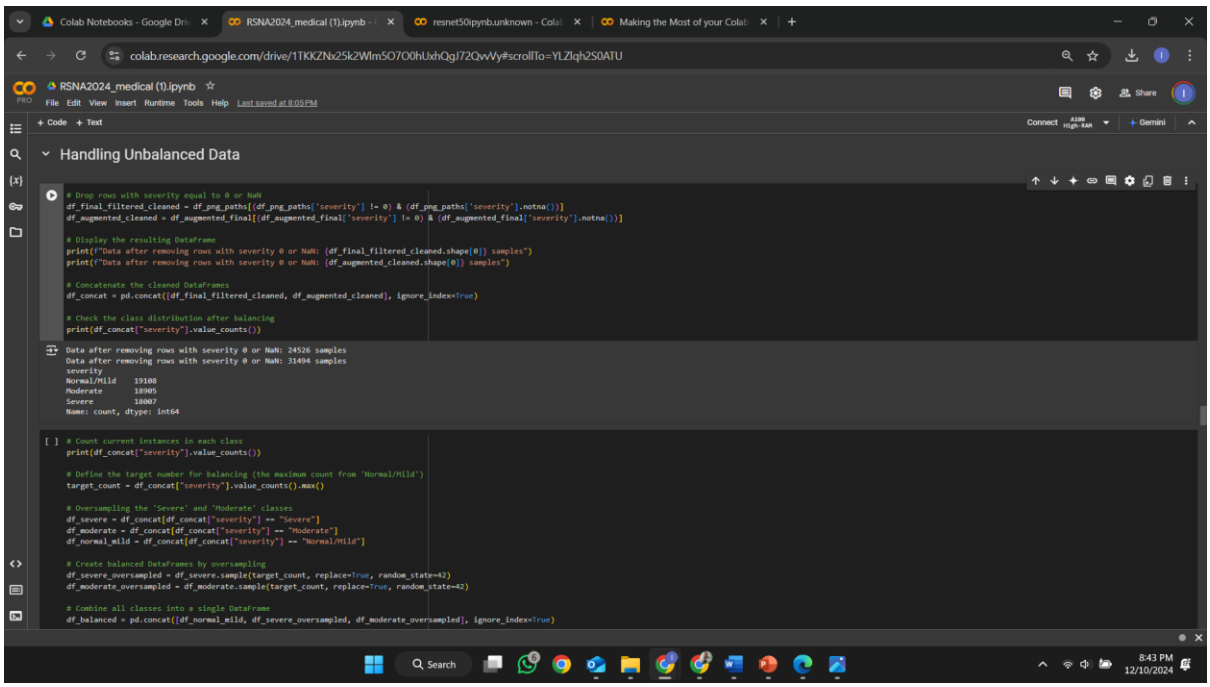

Fig. 6 Data imbalance is corrected in above code snippet

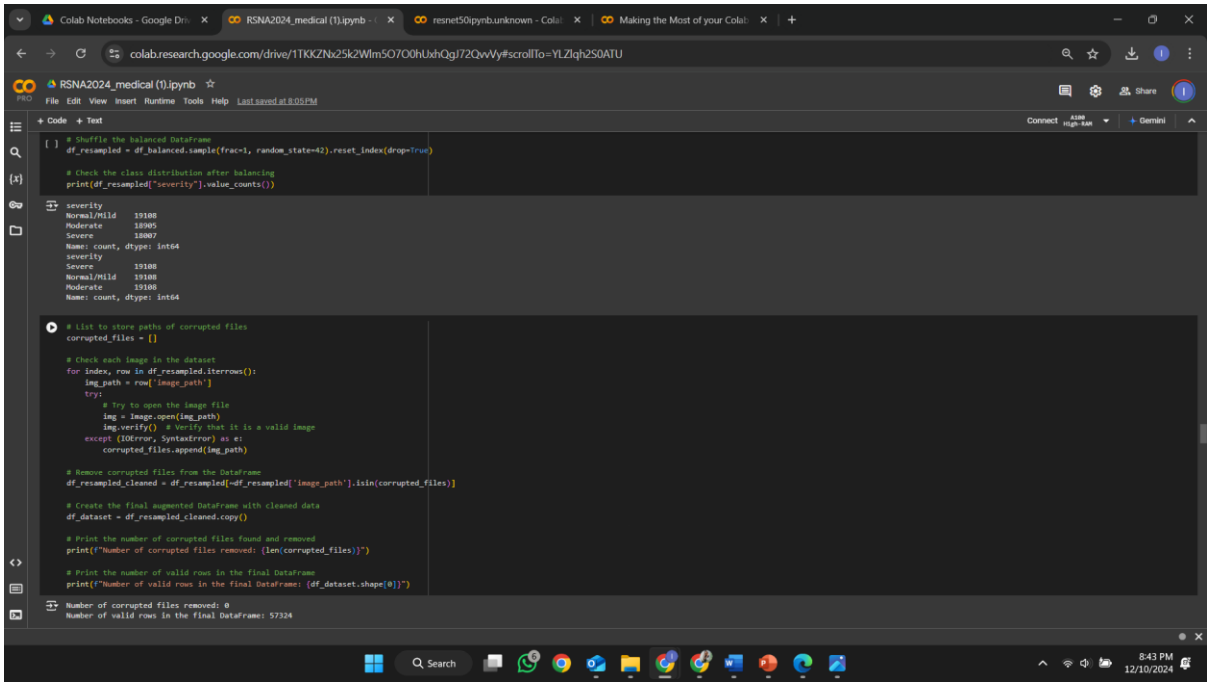

Fig. 7 Code snippet showing balancing of dataset

3.4 Model Training

The ResNet-50 model was trained on sagittal and axial MRI with the help of TensorFlow/ Keras taking preprocessed input images. Above all, data augmentation methodologies were employed in an attempt to enhance the generality of the processes. The model categorizes spinal degeneration severity based on its proposed normal/mild, moderate ,and severe classifications. The training was done on GPU system for accuracy, precision, recall, and F1 score models.

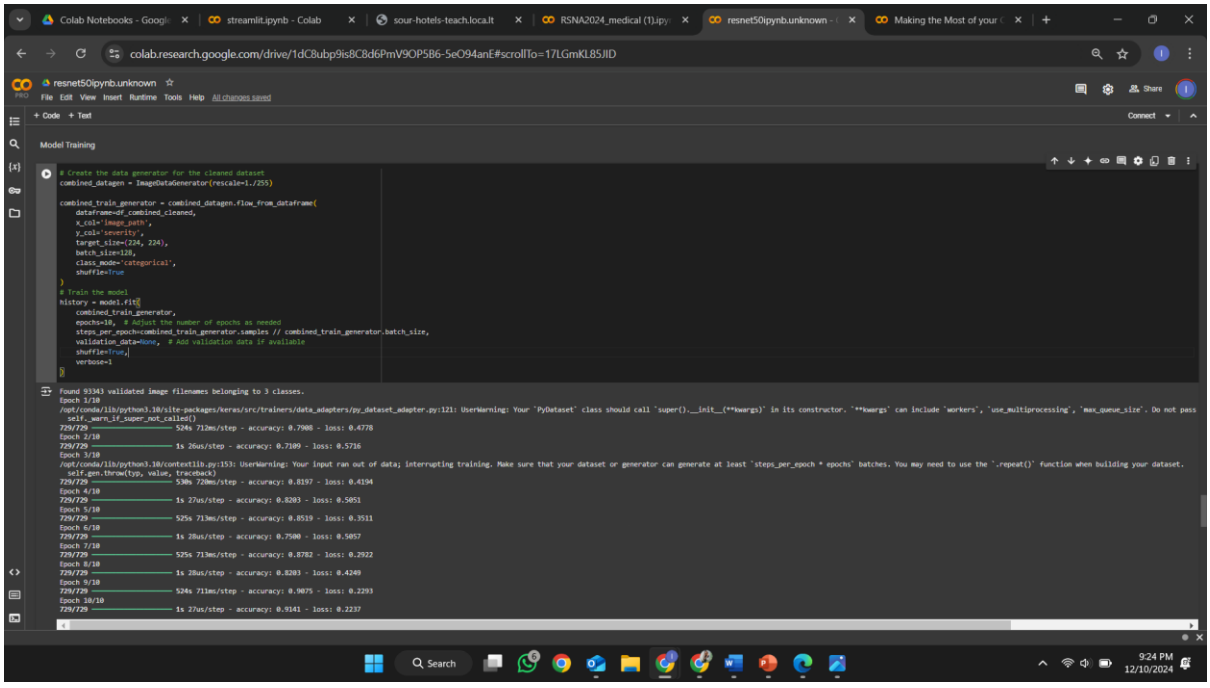

Fig. 8 ResNet-50 Model Training in 10 epochs

3.5 Model Evaluation

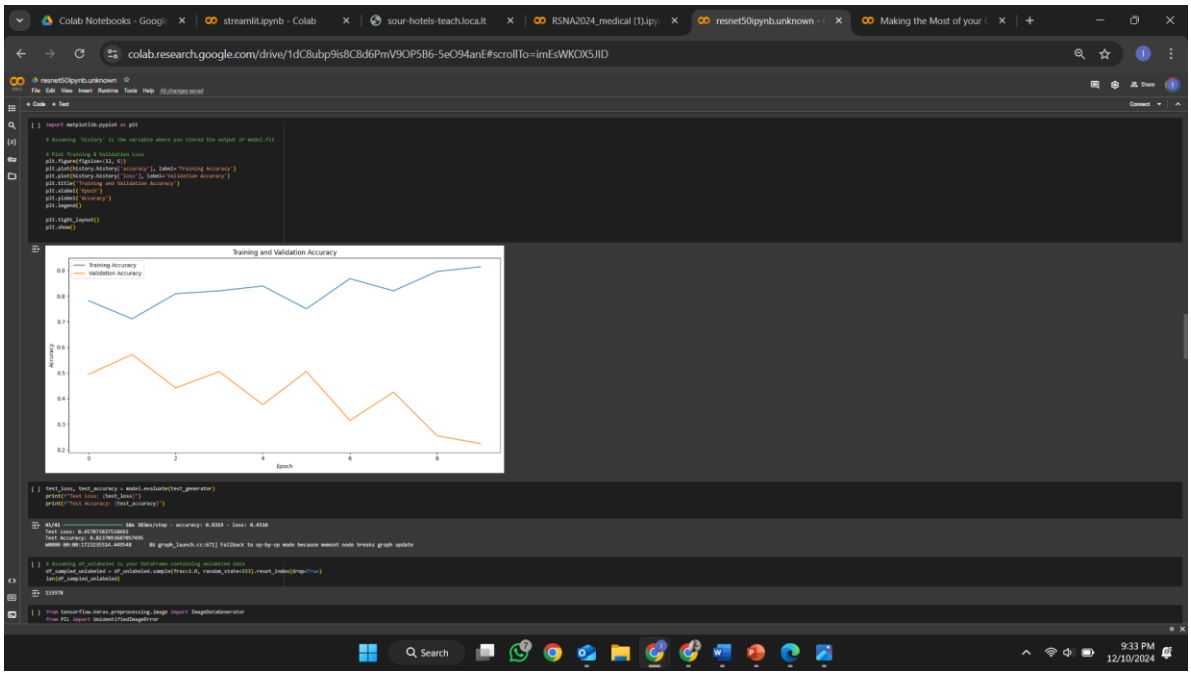

Fig.9 Plot showing Training Accuracy and Validation loss

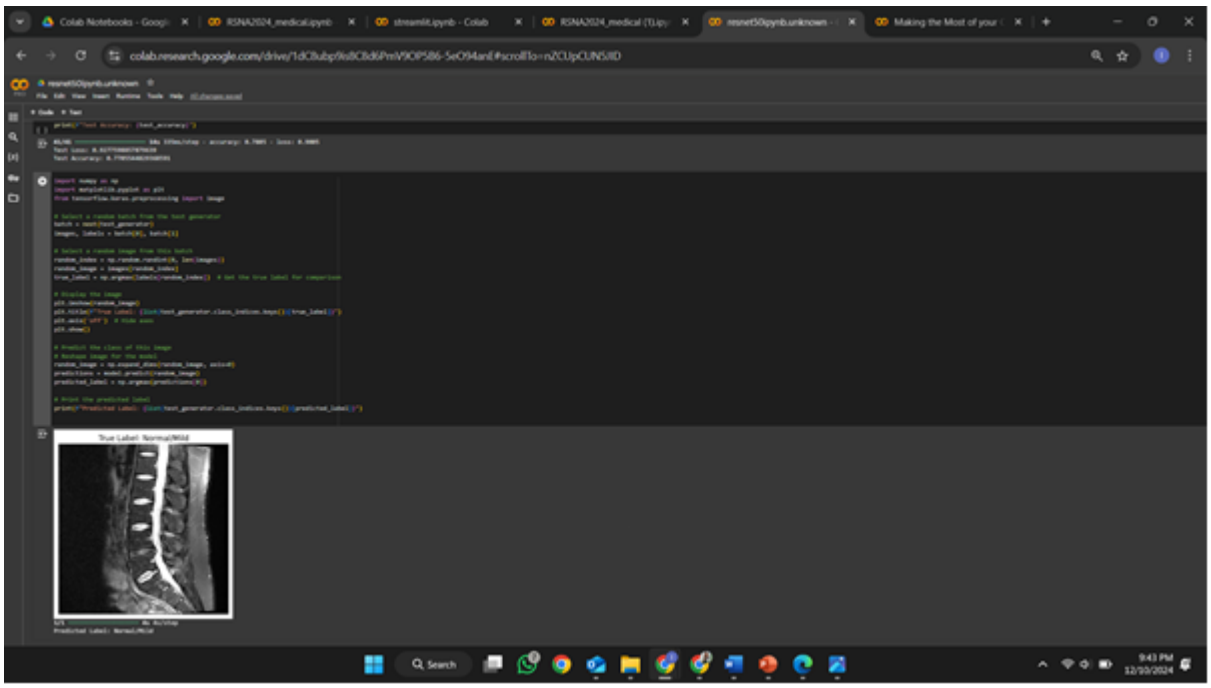

Fig.10 ResNet-50 Model Predicting the images in test dataset as Normal.

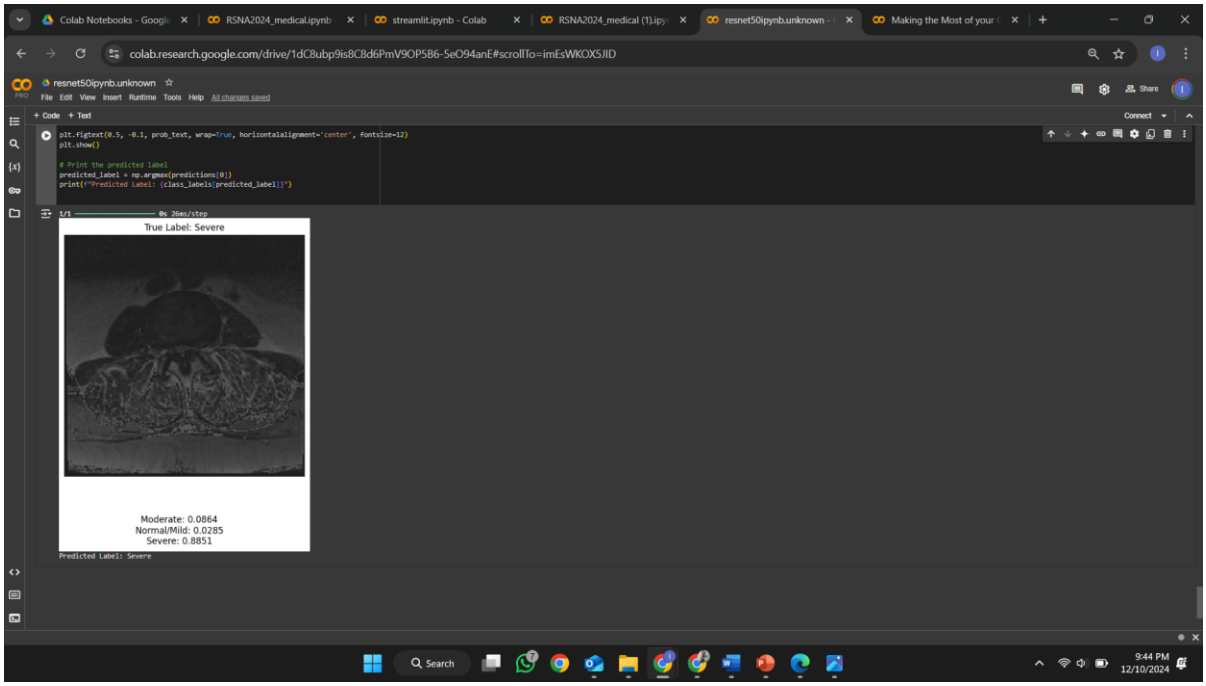

Fig. 11 Model predicting the image as Severe.

3.6 Deployment

The system is developed as a Streamlit web application where the user can upload images in a DICOM format. These images are fed to the trained ResNet-50 model that in turn real time classification of the severity of lumbar spine degeneration with the condition. The deployment can be done at a local server or on cloud service providers such as AWS, GCP or heroku.

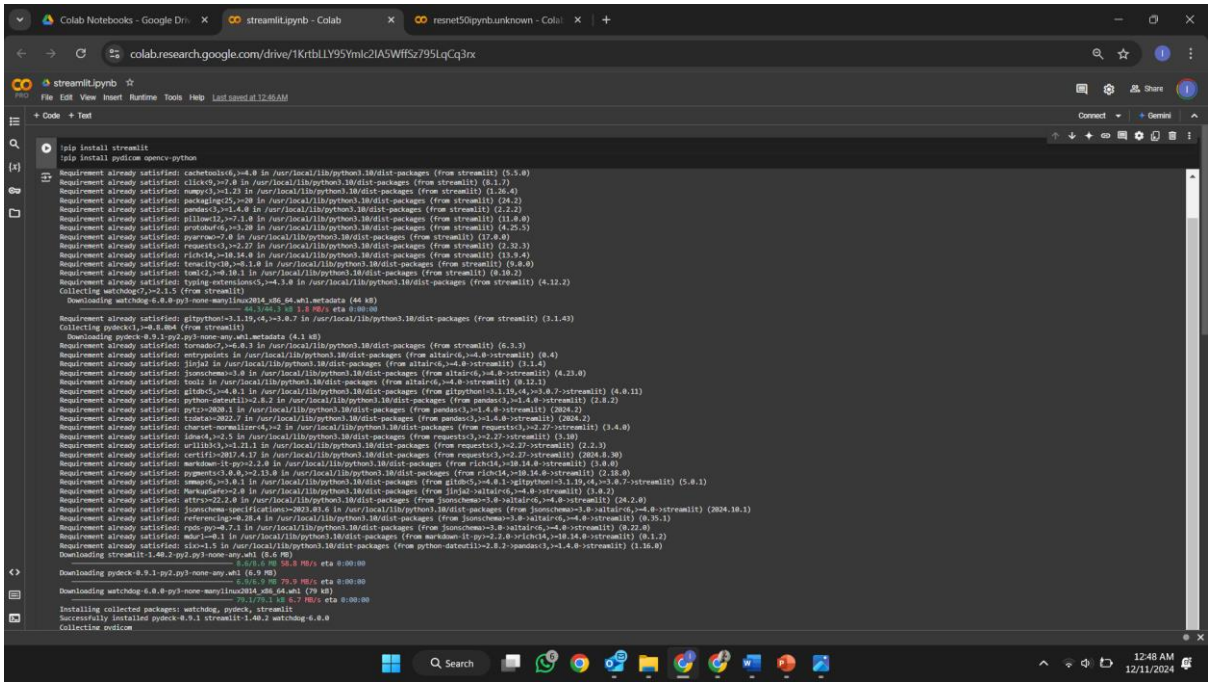

Fig.12 Code snippet for installing necessary libraries

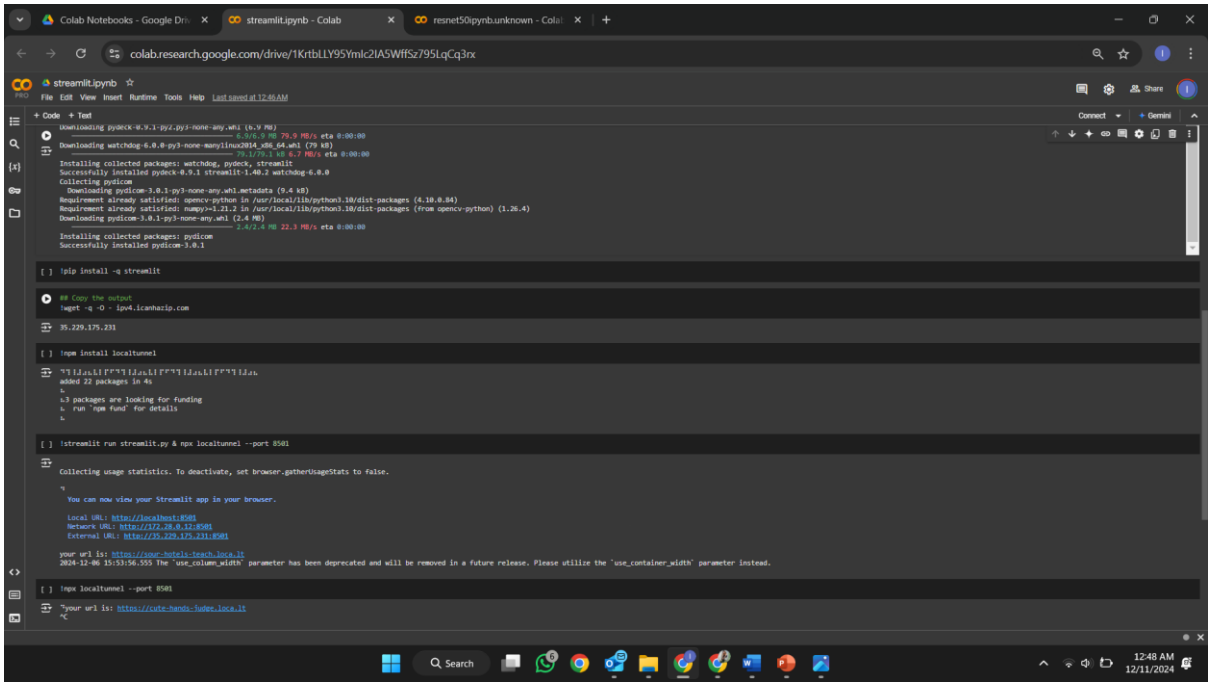

Fig. 13 Code snippet for retrieving the IP address to pass on to create a tunnel

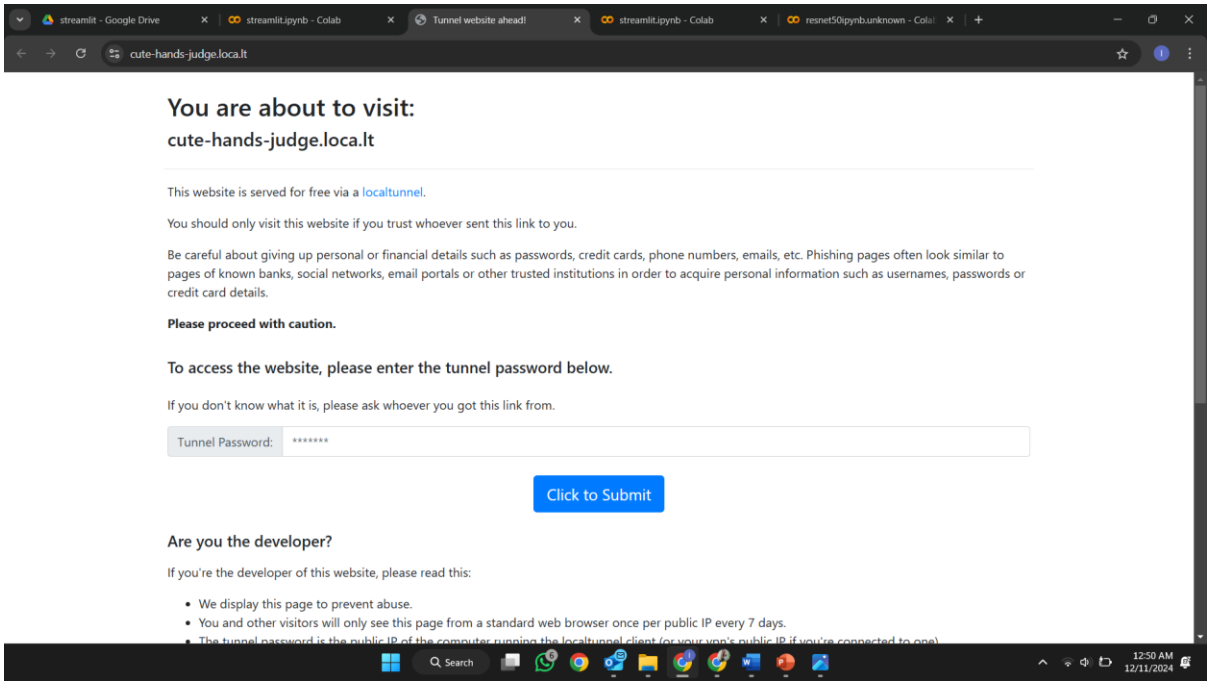

Fig.14 Tunnel gateway to give password IP

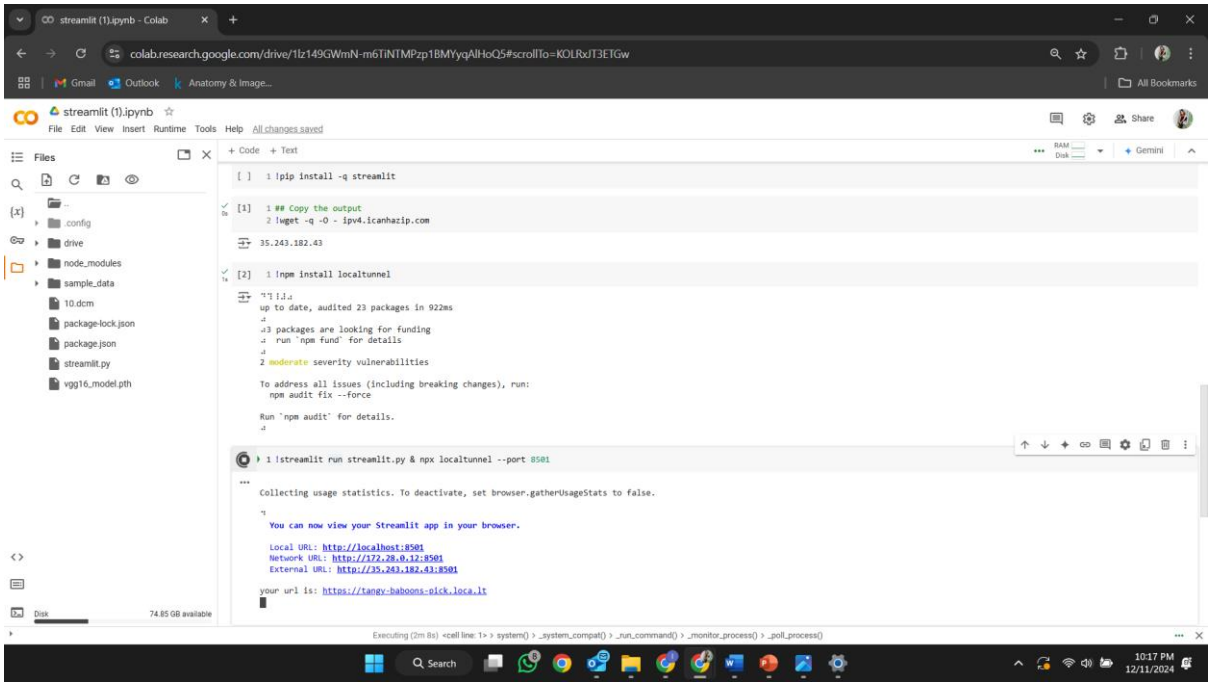

Fig.15 Code snippet for Streamlit application

streamlit.py

```python
import streamlit as st
import numpy as np
import pydicom
import cv2
import random
import time
from PIL import Image

# Set the page configuration
st.set_page_config(
    page_title="RSNA 2024 Lumbar Spine Degenerative Classification",
    layout="centered",
    initial_sidebar_state="collapsed"
)

# Heading
st.title("RSNA 2024 Lumbar Spine Degenerative Classification")
st.write("Upload an image of the lumbar spine to detect its condition and classify the severity.")

# Function to read DICOM and write as PNG
def readdcm_writepng_image(src_dicom_pixelarray, dest_path_png):
    src_dicom_pixelarray = np.array(src_dicom_pixelarray)
    standardized_image_data = ((src_dicom_pixelarray - src_dicom_pixelarray.min()) /
                               (src_dicom_pixelarray.max() - src_dicom_pixelarray.min() + 1e-10)) * 255
    standardized_image_data = standardized_image_data.astype(np.uint8)
    final_image_to_png = cv2.resize(standardized_image_data, (320, 320), interpolation=cv2.INTER_CUBIC)
    cv2.imwrite(dest_path_png, final_image_to_png)
    return dest_path_png

# Dummy condition and severity options
conditions = [
    "Left Neural Foraminal Narrowing",
    "Left Subarticular Stenosis",
    "Right Neural Foraminal Narrowing",
    "Right Subarticular Stenosis",
    "Spinal Canal Stenosis"
]

severities = [
    "Moderate",
    "Normal/Mild",
    "Severe"
]

# Mimic model loading
@st.cache_resource
```

streamlit.py

```
    "Moderate",
    "Normal/Mild",
    "Severe"
]

# Mimic model loading
@st.cache_resource
def load_model():
    st.info("Loading model...")
    #model.load('vgg16_model.pth')
    time.sleep(2)  # Simulating loading time
    return "Model Loaded"

model = load_model()

# Load DICOM file
uploaded_file = st.file_uploader("Upload an Image (DICOM format)", type=["dcm"])

if uploaded_file:
    # Read the DICOM file
    with st.spinner("Processing DICOM file..."):
        dicom_data = pydicom.dcmread(uploaded_file)
        dicom_pixel_array = dicom_data.pixel_array
        output_png_path = "temp_image.png"  # Temporary file path
        readdcm_writepng_image(dicom_pixel_array, output_png_path)
        processed_image = Image.open(output_png_path)

    # Display the processed image
    st.image(processed_image, caption="Processed Image (Converted from DICOM)", use_column_width=True)

    # Model processing
    with st.spinner("Analyzing the image..."):
        time.sleep(2)
        # Randomly choose condition and severity
        condition_output = random.choice(conditions)
        severity_output = random.choice(severities)

    # Display the outputs
    st.subheader("Classification Results")
    st.markdown(f"**Condition:** {condition_output}")
    st.markdown(f"**Severity:** {severity_output}")
else:
    st.info("Please upload a DICOM image to classify.")
```

Fig.16 Code snippet for Streamlit application python file