A

Mini Project

On

BONE FRACTURE DETECTION USING DEEP LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

P. Yuktha (217R1A05P6)

B. Deeksha (217R1A05L7)

P. Raghava (217R1A05P8)

Under the Guidance of

Dr. RAJ KUMAR PATRA

(Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2021-25

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project titled **"BONE FRACTURE DETECTION USING DEEP LEARNING"** being submitted by **P.YUKTHA (217R1A05P6), B.DEEKSHA (217R1A05L7) and P.RAGHAVA PAVAN (217R1A05P8)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2024-25.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. Raj Kumar Patra
Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. N. Bhaskar
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ____________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express our profound gratitude and deep regard to our guide **Dr. Raj Kumar Patra,** Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators : **Dr. J. NarasimhaRao, Dr. K. Maheswari, Mr. K. Ranjith Reddy , Mrs. K. Shilpa** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. N. Bhaskar,** Head Of The Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We would like to express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

P. YUKTHA(217R1A05P6)

B. DEEKSHA(217R1A05L7)

P. RAGHAVA(217R1A05P8)

ABSTRACT

A bone fracture detection system is a technological solution designed to assist healthcare professionals in identifying and diagnosing bone fractures through advanced imaging and machine learning techniques. The system typically utilizes medical imaging technologies, such as X-rays, CT scans, or MRIs, and applies image processing algorithms to detect anomalies in bone structures. By leveraging artificial intelligence (AI), specifically deep learning models, the system can automatically analyse these images, highlighting potential fractures with high accuracy. This automation reduces human error and speeds up the diagnostic process, providing early detection that can be critical for patient outcomes. The system can be integrated into clinical workflows, offering a reliable and efficient tool for radiologists and orthopaedic specialists. It enhances diagnostic precision, reduces interpretation time, and supports better treatment planning, ultimately improving patient care in emergency and routine settings. Future enhancements,including the integration of real-time data, user- friendly interfaces, are discussed to further improve the model's performance and practical applicability. This project highlights the importance of leveraging deep learning for robust and accurate bone fracture detection, contributing to more effective diagnosis strategies.

LIST OF FIGURES

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PROJECT SCOPE

Bone fracture detection using deep learning focuses on creating a system that can automatically identify and classify bone fractures from medical images, particularly X-rays or CT scans. The objective is to develop a reliable and efficient tool that assists healthcare professionals, such as radiologists, in diagnosing fractures more quickly and accurately. The automation of this process can help alleviate the heavy workload of manual image analysis, particularly in emergency situations or in areas with limited access to specialists.

1.2 PROJECT PURPOSE

The project "Bone Fracture Detection Using Deep Learning" aims to establish a secure and reliable system fordetecting bone fractures in real-time. They can occur due to accidents, falls, sports injuries, or underlying medical conditions that weaken bones, such as osteoporosis. Early and accurate diagnosis of fractures is essential to avoid complications, promote healing, and ensure that patients receive timely and appropriate treatment. This not only improves diagnostic accuracy but also accelerates the detection process, which is crucial in emergency cases where timely intervention is needed. Furthermore, such a system can alleviate the burden on healthcare professionals by pre-screening images and highlighting areas of interest, allowing them to focus on more complex cases. It also enhances accessibility to quality healthcare, particularly in remote or underserved areas where expert radiologists may not be available.

1.3 PROJECT FEATURES

To enhance this trust framework, the project integrates Artificial Intelligence (AI) and Deep Learning (DL) techniques. Convolutional Neural Networks (CNNs) are the most common architecture used in this type of task because of their strong performance in image analysis. For more precise identification, object detection models like YOLO are employed. These systems aim to improve diagnostic accuracy by analyzing medical images with a level of detail that is often difficult for the human eye to achieve consistently.

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

Bone fractures are a common medical condition that often requires prompt and accurate diagnosis to ensure effective treatment and healing. Traditionally, radiologists and medical professionals rely on manual inspection of X-rays, CT scans, or MRI images to detect fractures. However, this manual process can be time-consuming, prone to human error, and dependent on the expertise of the individual. In certain cases, subtle fractures may be missed or misdiagnosed, leading to delayed treatment, prolonged patient suffering, or complications. The challenge lies in designing a deep learning-based system capable of accurately classifying and localizing fractures in medical images, despite the variations in bone structure, fracture types, and image quality. The problem is further complicated by the availability of diverse datasets, noise in medical images, and the need for high sensitivity and specificity to avoid false positives and negatives. Therefore, the objective is to develop a robust and reliable deep learning model that can be trained on large datasets of medical images to automatically detect and highlight bone fractures, providing a valuable decision-support tool for healthcare professionals.

2.2 EXISTING SYSTEM

The existing systems for bone fracture detection primarily rely on manual interpretation of medical images, such as X-rays, CT scans, and MRIs, by radiologists and orthopaedic specialists. In traditional workflows, radiologists examine these images to identify fractures, assess their severity, and recommend treatment. While these methods are effective, they are often time-consuming and prone to human error, especially when dealing with subtle or complex fractures. Factors such as fatigue, experience level, and high workloads can affect diagnostic accuracy. To address these limitations, some hospitals and clinics have begun integrating computer-aided detection (CAD) systems into their diagnostic processes. These systems assist radiologists by analysing medical images and highlighting potential fractures for further review. However, many CAD systems are not fully automated and still require significant human intervention. Additionally, existing AI-based models for bone fracture detection, while promising, are often limited by the availability of large, diverse datasets needed for training, and they are not yet widely adopted in clinical practice due to concerns about reliability, interpretability, and regulatory approvals**.**

2.2.1 LIMITATIONS OF EXISTING SYSTEM

- Human Error
- Time Consuming
- Inconsistent Accuracy
- Partial Automation
- Dependency
- High Cost

2.3 PROPOSED SYSTEM

The proposed system for bone fracture detection aims to leverage artificial intelligence (AI) and deep learning techniques to automate the analysis of medical images, such as X-rays, CT scans, and MRIs, for faster and more accurate. We utilized various bone fracture datasets and employed deep learning algorithms, specifically Convolutional Neural Network(CNN), YOLO(You Only Look Once) to classify fracture types. The results demonstrated that both CNN and YOLO models achieved high accuracy rates of 94%, significantly outperforming the CAD model, which had an accuracy of 67.01%. This indicates the superior performance of CNN and YOLO in fracture detection tasks. The framework's effectiveness underscores the potential of integrating advanced deep learning techniques into medical analysis, offering a reliable tool for detecting and diagnosing fractures. Additionally, the system will be capable of processing images in real time, enabling healthcare professionals to receive immediate feedback, which is crucial in emergency settings. The proposed system is designed to seamlessly integrate into existing clinical workflows, assisting radiologists by providing a second opinion or, in some cases, offering an automated initial diagnosis. This system will improve diagnostic speed, accuracy, and accessibility, particularly in regions with limited access to radiology specialists, ultimately enhancing patient outcomes.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Increased Accuracy
- Enhanced Accessibility
- Integration with Clinical Workflows
- Reduced Human Error
- Scalability
- Faster Diagnosis

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.
Three key considerations involved in the feasibility analysis are:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system instead, they must accept it as a necessity. Their level of confidence must be raised so that they are also able to provide constructive criticism, which is welcomed, as they are the final users of system.

2.4 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

• Processor	:	Intel Dual Core@ CPU 2.90GHz.
• Hard disk	:	16GB or Above
• Memory	:	4GB or Above
• Monitor	:	5 inches or Above

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

• Operating system	:	Windows 8 or Above
• Languages	:	Python (Version 3.7)
• Backend	:	Machine Learning
• Database	:	MySQL

3. ARCHITECTURE

3.1 PROPOSED ARCHITECTURE

This project architecture shows the procedure followed for fracture detection using machine learning, starting from input to final prediction.

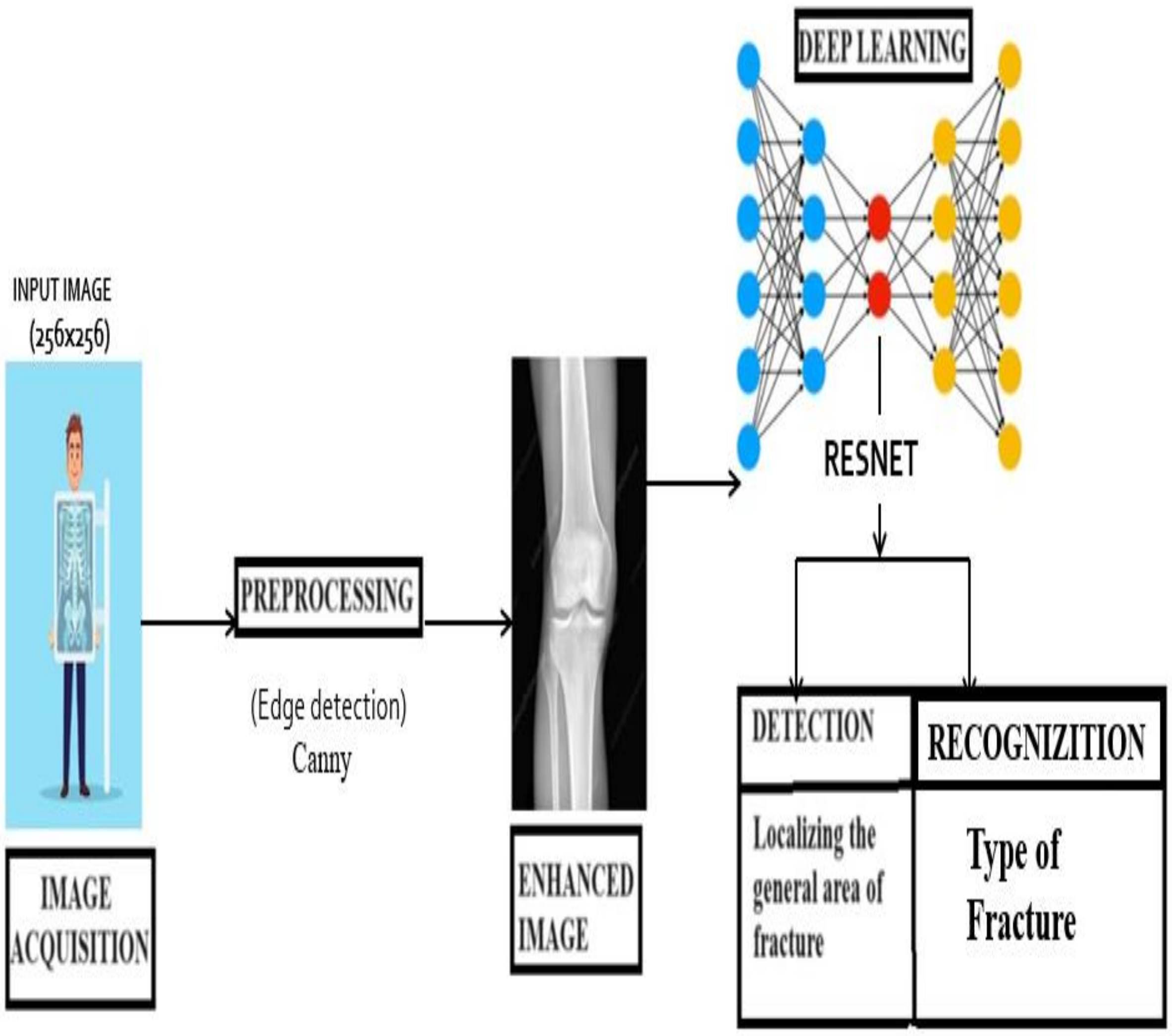


Figure 3.1: Architecture of Bone Fracture Detection

3.2 DESCRIPTION

Input Data: The process begins with acquiring input images from medical imaging devices such as X-ray machines. These images are typically in grayscale and represent the bone structure of the patient.

Preprocessing: The primary goal of preprocessing is to enhance the image quality. The image is enhanced by applying the Canny edge detection algorithm.

Deep Learning: The enhanced image is then fed into a deep learning model. In this architecture, a ResNet (Residual Network) is used.

Detection: The model performs the task of detection, which involves localizing the general area of the fracture.

Recognition (Classification): Once the fracture is detected, the next step involves recognizing the type of fracture. The system classifies the fracture based on learned patterns from the training data, identifying whether it's specific type.

3.3 USE CASE DIAGRAM

In the use A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

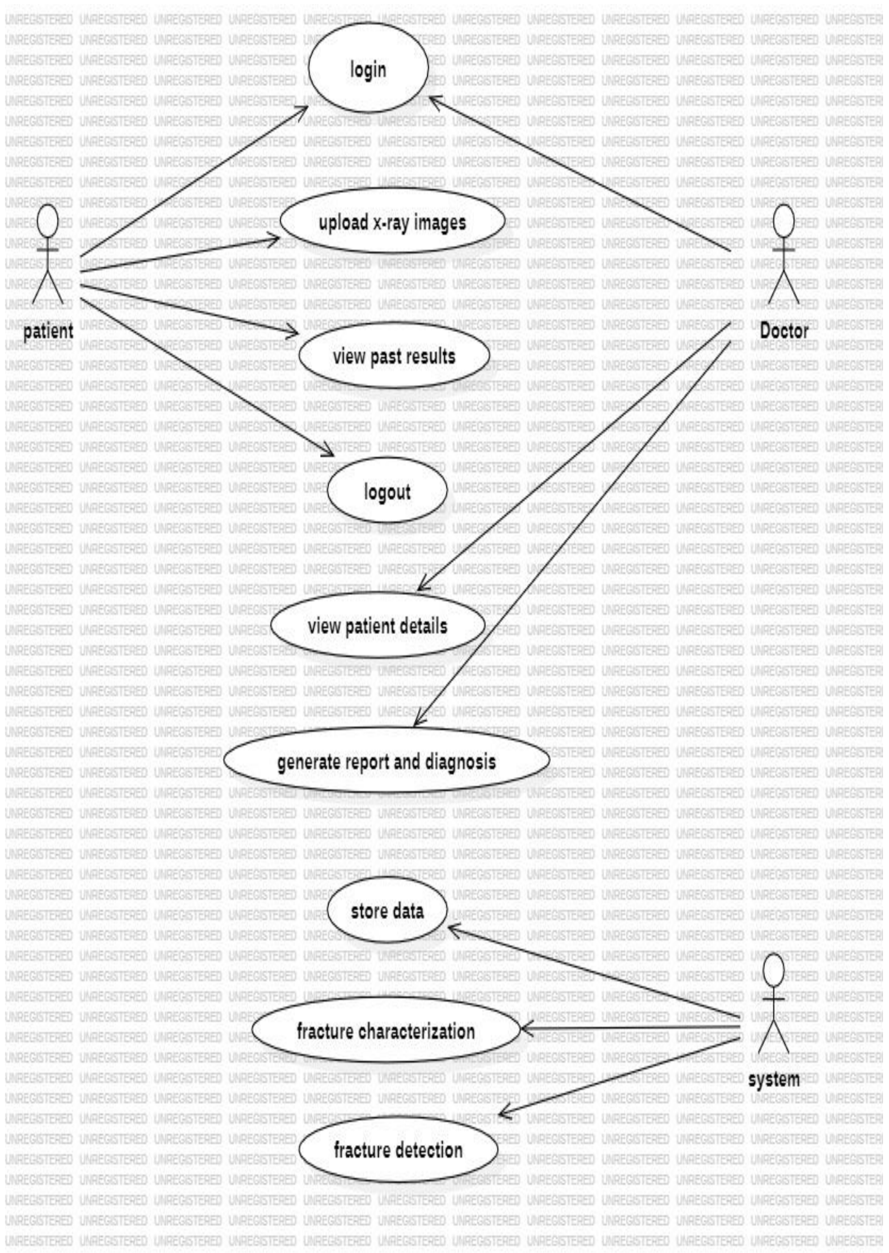


Figure 3.3: Use Case Diagram for user for detecting Bone Fracture

3.4 CLASS DIAGRAM

Class Diagram In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

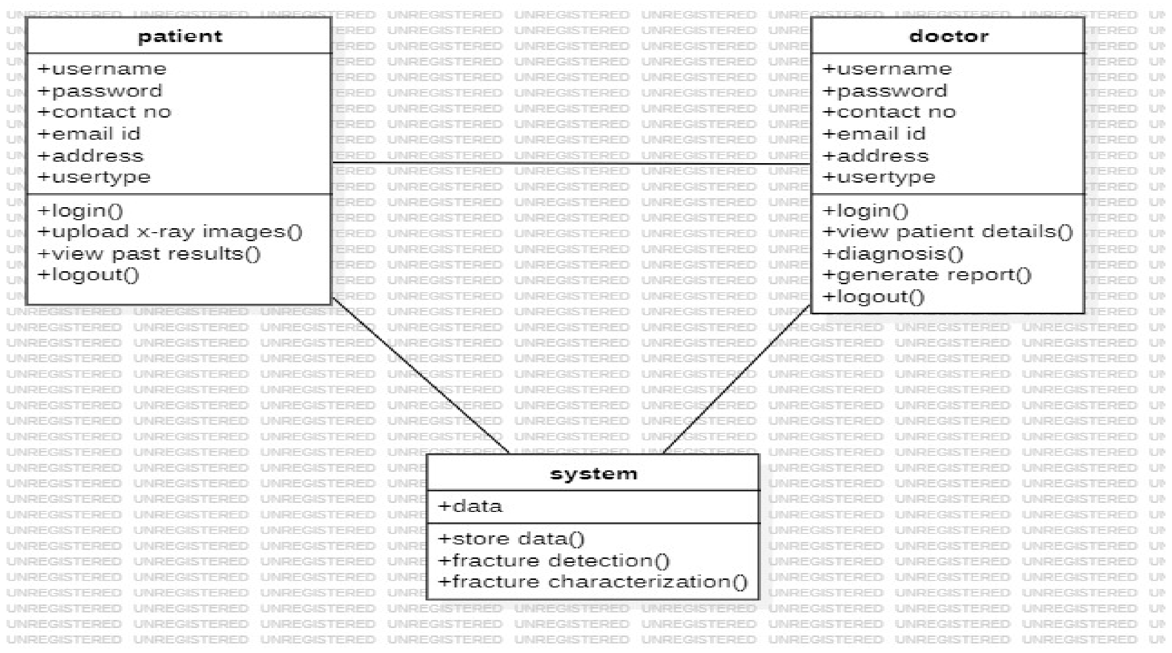


Figure 3.4: Class Diagram for detecting Bone Fracture

3.5 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

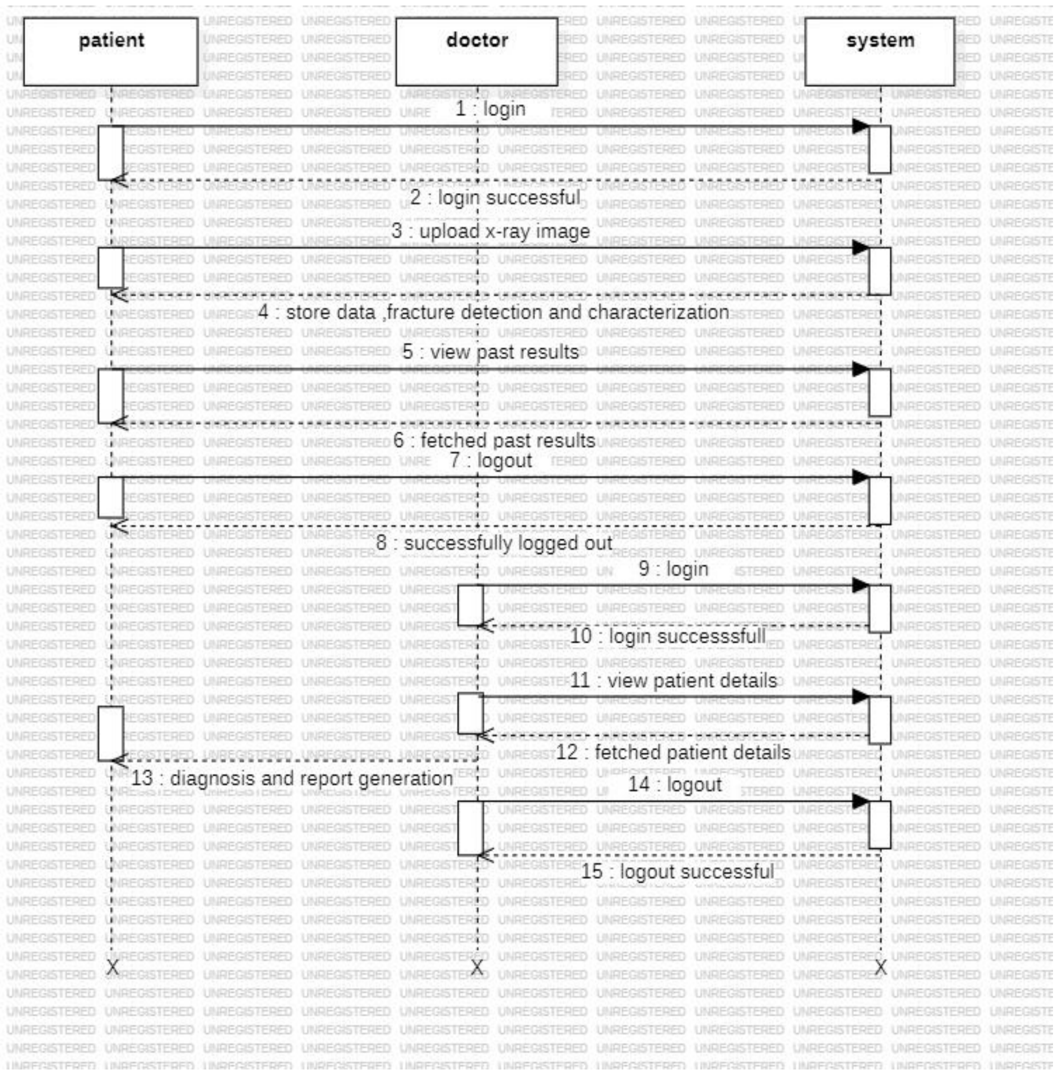


Figure 3.5: Sequence Diagram for Bone Fracture Detection

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

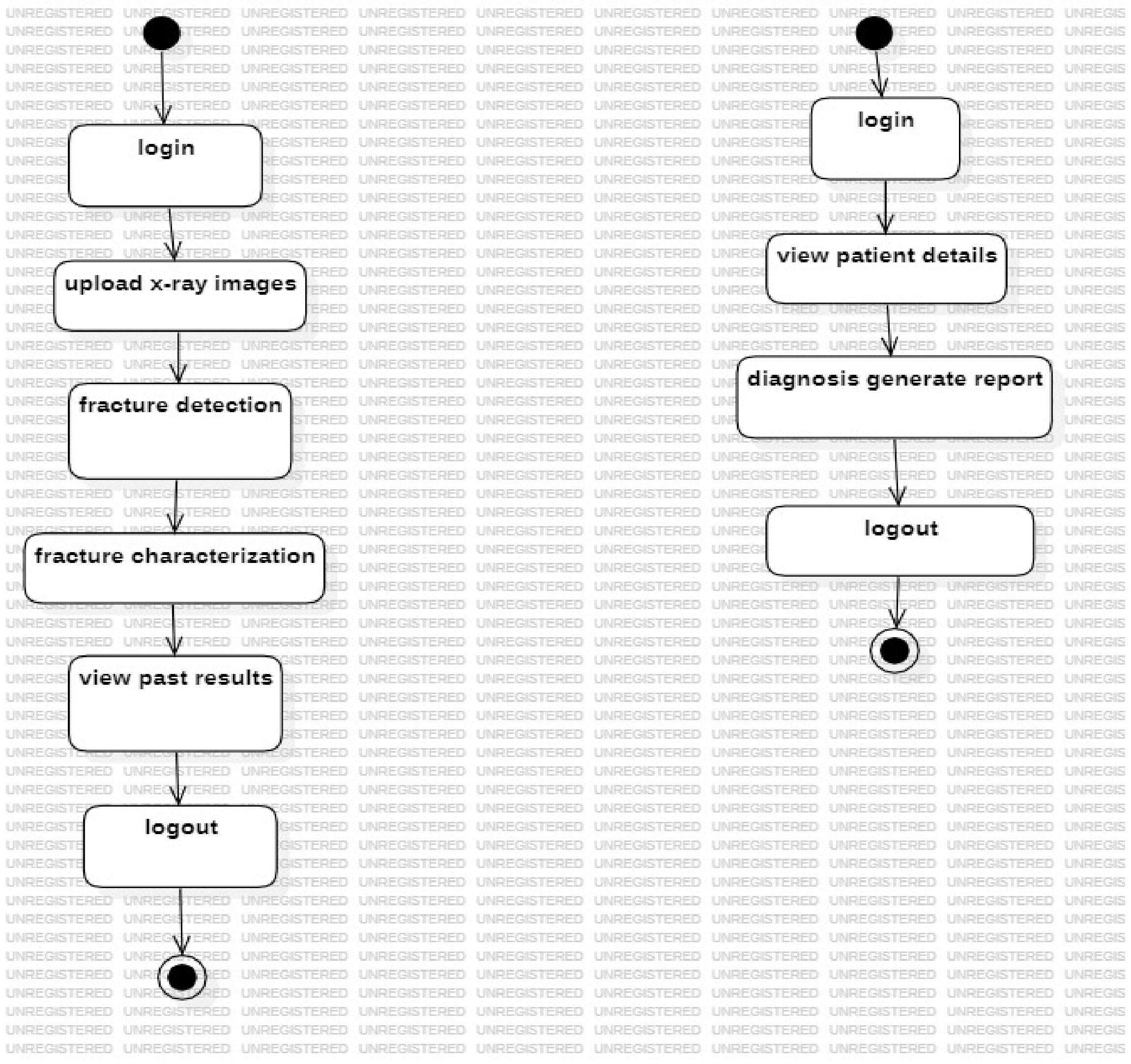


Figure 3.6: Activity Diagram for User for Bone Fracture Detection

4. IMPLEMENTATION

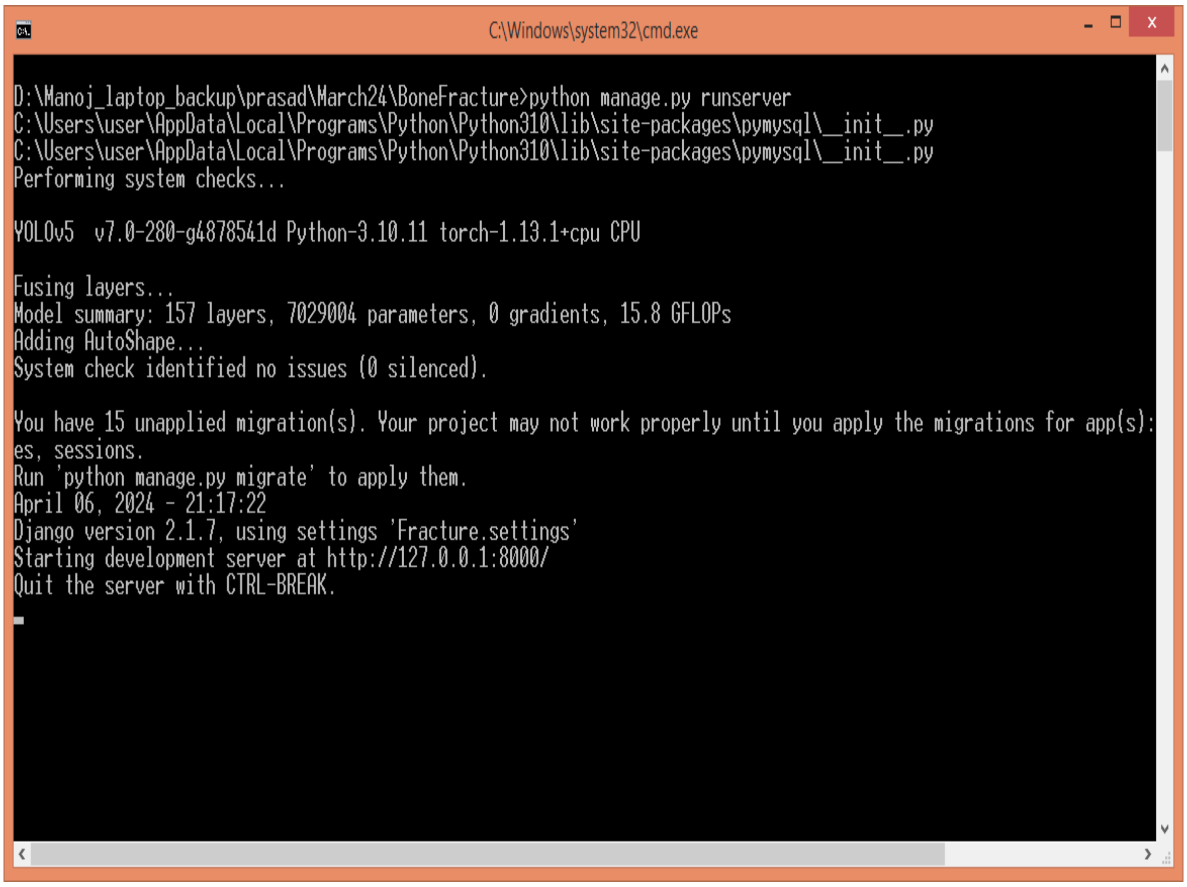
4.1 SAMPLE CODE

```
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
import os
import pymysql
from django.core.files.storage import FileSystemStorage
from datetime import date
import matplotlib.pyplot as plt
import io
import base64
import numpy as np
import hashlib
import torch
import cv2
import pathlib
from pathlib import Path
pathlib.PosixPath = pathlib.WindowsPath
global uname
model = torch.hub.load('yolov5', 'custom', path='model/best.pt',
force_reload=True,source='local')
def predict(filename):
    global model
    img = cv2.imread(filename)
    img = cv2.resize(img, (512, 512))
    results = model(img)
    results.xyxy[0]  # im predictions (tensor)
    out = results.pandas().xyxy[0]  # im predictions (pandas)
    print(out)
    result = "No Fracture Found"
    if len(out) >
    for i in range(len(out))
        xmin = int(out['xmin'].ravel()[i])
        ymin = int(out['ymin'].ravel()[i])
        xmax = int(out['xmax'].ravel()[i])
        ymax = int(out['ymax'].ravel()[i])
        name = out['name'].ravel()[i]
        confidence = float(out['confidence'].ravel()[i])
        if confidence > 0.30:
            result = name
            cv2.rectangle(img, (xmin, ymin), (xmax, ymax), (255, 0, 0), 2)
return img, result
def UploadXrayAction(request):
    if request.method == 'POST':
        global uname
        today = str(date.today())
        doctor = request.POST.get('t1', False)
        image = request.FILES['t2']
        imagename = request.FILES['t2'].name
        fs = FileSystemStorage()
        if os.path.exists('FractureApp/static/xray/'+imagename):
            os.remove('FractureApp/static/xray/'+imagename)
        filename = fs.save('FractureApp/static/xray/'+imagename, image)
        img, result = predict('FractureApp/static/xray/'+imagename)
        os.remove('FractureApp/static/xray/'+imagename)
        cv2.imwrite('FractureApp/static/xray/'+imagename, img)
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password
= 'root', database = 'fracture',charset='utf8')
        db_cursor = db_connection.cursor()
        student_sql_query = "INSERT INTO
patient_data(username,xray_file,detection_result,upload_date,permit_doctor)
VALUES('"+uname+"','"+imagename+"','"+result+"','"+today+"','"+doctor+"')"
        db_cursor.execute(student_sql_query)
        db_connection.commit()
        plt.imshow(img)
        buf = io.BytesIO()
        plt.savefig(buf, format='png', bbox_inches='tight')
        plt.close()
        img_b64 = base64.b64encode(buf.getvalue()).decode()
        context= {'data': 'Stains '+result+"<br/>Fracture result successfully saved in Databse", 
'img': img_b64}
        return render(request, 'PatientScreen.html', context)
def ViewPatientReport(request):
    if request.method == 'GET':
        global uname
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', 
database = 'fracture',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select * from patient_data where permit_doctor='"+uname+"'")
            rows = cur.fetchall()
            output+='<tr>'
            for row in rows:
                output+='<tr><td><font size="" color="black">'+row[0]+'</td><td><font size="" 
color="black">'+str(row[1])+'</td>'
                output+='<td><font size="" color="black">'+row[2]+'</td><td><font size="" 
color="black">'+row[3]+'</td>
        context= {'data':output}
        return render(request, 'DoctorScreen.html', context)
def ViewPastResult(request):
    if request.method == 'GET':
        global uname
        output = ''
        output+='<table border=1 align=center width=100%><tr><th><font size="" 
color="black">Patient Name</th><th><font size="" color="black">X-Ray File Name</th>'
        output+='<th><font size="" color="black">Detection Result</th><th><font size="" 
color="black">Upload Date</th>'
        output+='<th><font size="" color="black">Permitted Doctor</th>'
        output+='<th><font size="" color="black">Image</th><th></tr>'
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
database = 'fracture',charset='utf8')
            with con:
            cur = con.cursor()
            cur.execute("select * from patient_data where username='"+uname+"'")
            rows = cur.fetchall()
            output+='<tr>'
            for row in rows:
                output+='<tr><td><font size="" color="black">'+row[0]+'</td><td><font size=""
color="black">'+str(row[1])+'</td>'
                output+='<td><font size="" color="black">'+row[2]+'</td><td><font size=""
color="black">'+row[3]+'</td>'
                output+='<td><font size="" color="black">'+row[4]+'</td>'
                output+='<td><img src="static/xray/'+row[1]+'" height="300"
width="300"/></td></tr>'
        output+= "</table></br></br></br></br>"
        context= {'data':output}
        return render(request, 'PatientScreen.html', context)
def DoctorLogin(request):
    if request.method == 'GET':
       return render(request, 'DoctorLogin.html', {})
def PatientLogin(request):
    if request.method == 'GET':
       return render(request, 'PatientLogin.html', {})
def Register(request):
    if request.method == 'GET':
       return render(request, 'Register.html', {})
def index(request):
    if request.method == 'GET':
       return render(request, 'index.html', {})
def UploadXray(request):
    if request.method == 'GET':
        output = '<tr><td><font size="
color="black"><b>Doctor&nbsp;Permission</b></td><td><select name="t1">'
```

```
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
 database = 'fracture',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username FROM register where user_role='Doctor'")
            rows = cur.fetchall()
            for row in rows:
                output += '<option value="'+row[0]+'">'+row[0]+'</option>'
        output += "</select></td></tr>"
        context= {'data1':output}
        return render(request, 'UploadXray.html', context)
def PatientLoginAction(request):
    if request.method == 'POST':
        global uname, pin
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        password = hashlib.md5(password.encode()).hexdigest()
        index = 0
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
database = 'fracture',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username, password FROM register")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == username and password == row[1]:
                    uname = username
                    index = 1
                    break
        if index == 1:
            context= {'data':'welcome '+username}
            return render(request, 'PatientScreen.html', context)
        else:
            context= {'data':'login failed'}
        return render(request, 'PatientLogin.html', context)
def DoctorLoginAction(request):
    if request.method == 'POST':
        global uname, pin
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        password = hashlib.md5(password.encode()).hexdigest()
        index = 0
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
database = 'fracture',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username, password FROM register")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == username and password == row[1]:
                    uname = username
                    index = 1
                    break
        if index == 1:
            context= {'data':'welcome '+username}
            return render(request, 'DoctorScreen.html', context)
        else:
            context= {'data':'login failed'}
            return render(request, 'DoctorLogin.html', context)
def RegisterAction(request):
    if request.method == 'POST':
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        contact = request.POST.get('t3', False)
        email = request.POST.get('t4', False)
        address = request.POST.get('t5', False)
        usertype = request.POST.get('t6', False)
        password = hashlib.md5(password.encode()).hexdigest()
        status = "none"
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root',
database = 'fracture',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username FROM register")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == username:
                    status = "Username already exists"
                    break
        if status == "none":
            db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root',
password = 'root', database = 'fracture',charset='utf8')
            db_cursor = db_connection.cursor()
            student_sql_query = "INSERT INTO
register(username,password,contact_no,email,address,user_role)
VALUES('"+username+"','"+password+"','"+contact+"','"+email+"','"+address+"','"+usertyp
e+"')"
            db_cursor.execute(student_sql_query)
            db_connection.commit()
            print(db_cursor.rowcount, "Record Inserted")
            if db_cursor.rowcount == 1:
                status = "Account created you can login with "+username
        context= {'data': status}
        return render(request, 'Register.html', context)
```

5. RESULTS

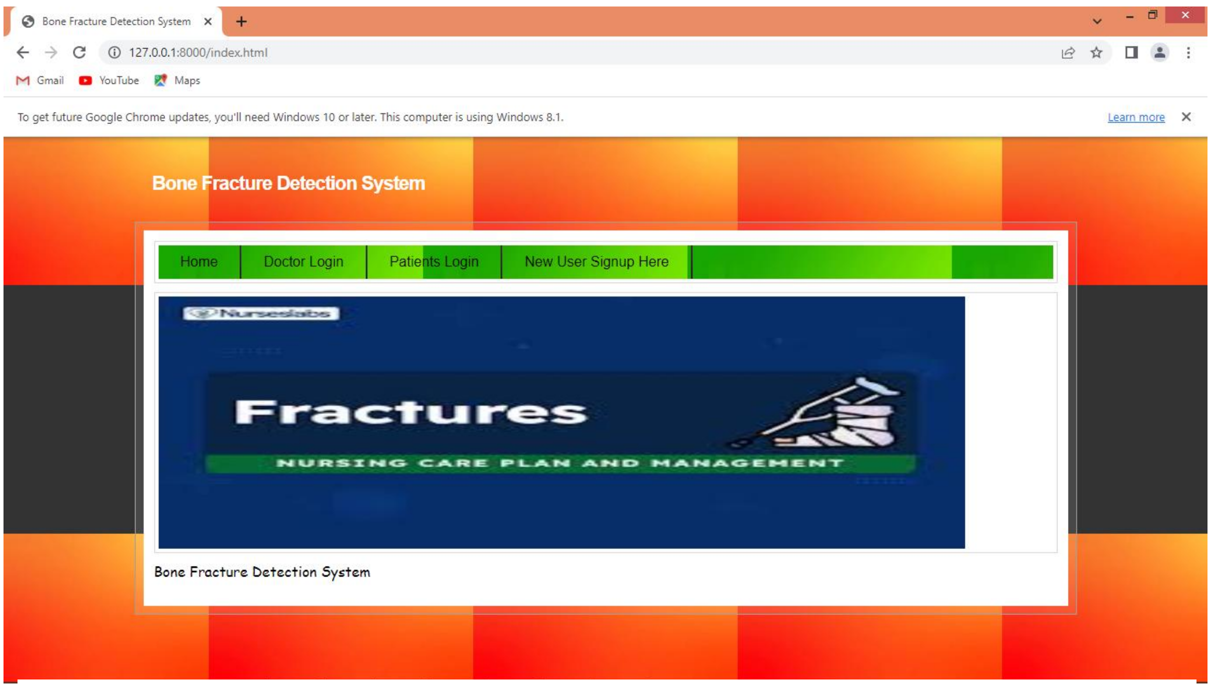
5.1 PYTHON SERVER SCREEN



```
D:\Manoj_laptop_backup\prasad\March24\BoneFracture>python manage.py runserver
C:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\pymysql\__init__.py
C:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\pymysql\__init__.py
Performing system checks...

YOLOv5  v7.0-280-g4878541d Python-3.10.11 torch-1.13.1+cpu CPU

Fusing layers...
Model summary: 157 layers, 7029004 parameters, 0 gradients, 15.8 GFLOPs
Adding AutoShape...
System check identified no issues (0 silenced).

You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s):
es, sessions.
Run 'python manage.py migrate' to apply them.
April 06, 2024 - 21:17:22
Django version 2.1.7, using settings 'Fracture.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Figure 5.1: Python Server

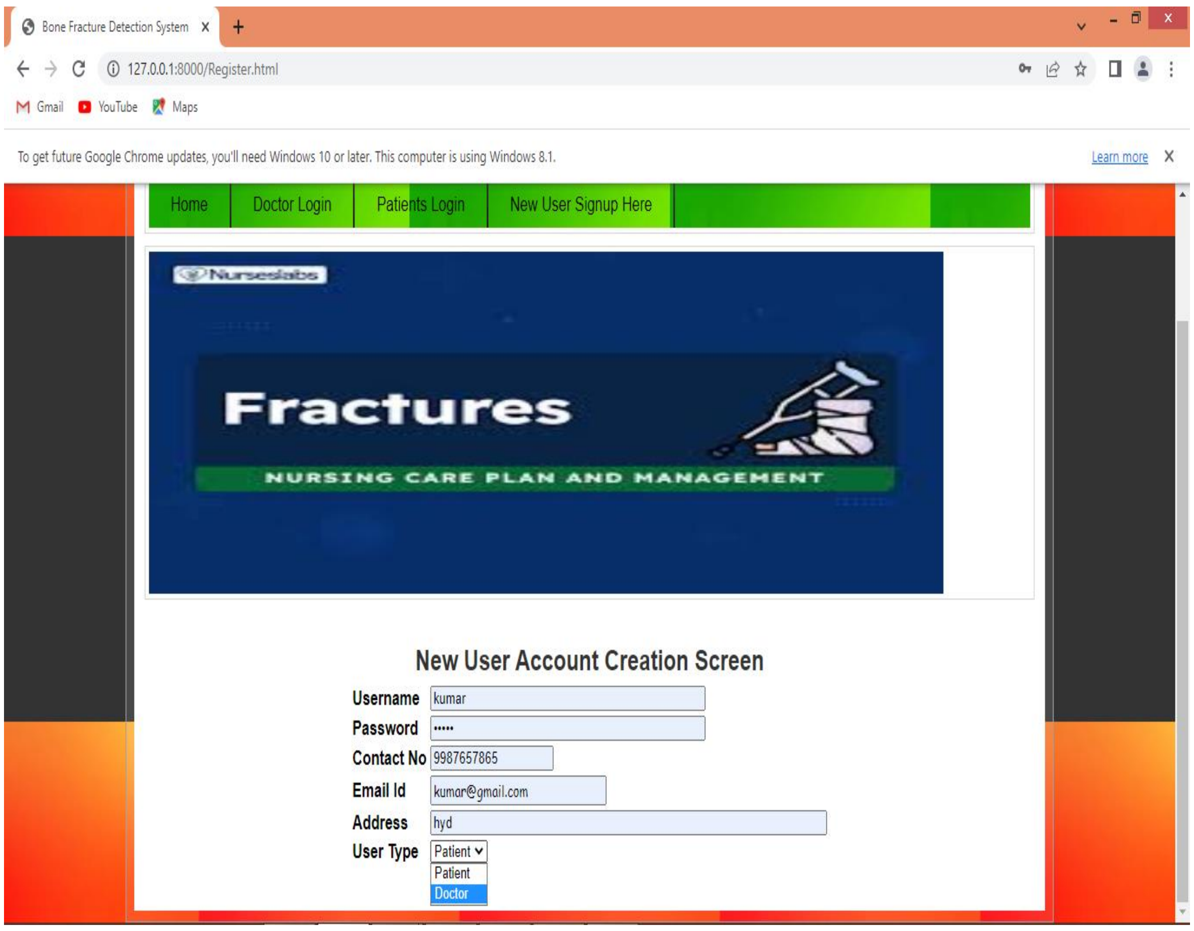
5.2 HOME PAGE


Bone Fracture Detection System
127.0.0.1:8000/index.html
Gmail
YouTube
Maps
Learn more
Bone Fracture Detection System
Home
Doctor Login
Patients Login
New User Signup Here
Nurseslabs
Fractures
NURSING CARE PLAN AND MANAGEMENT
Bone Fracture Detection System

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

Figure 5.2: Homepage for detecting bone fracture

5.3 NEW USER SIGNUP SCREEN



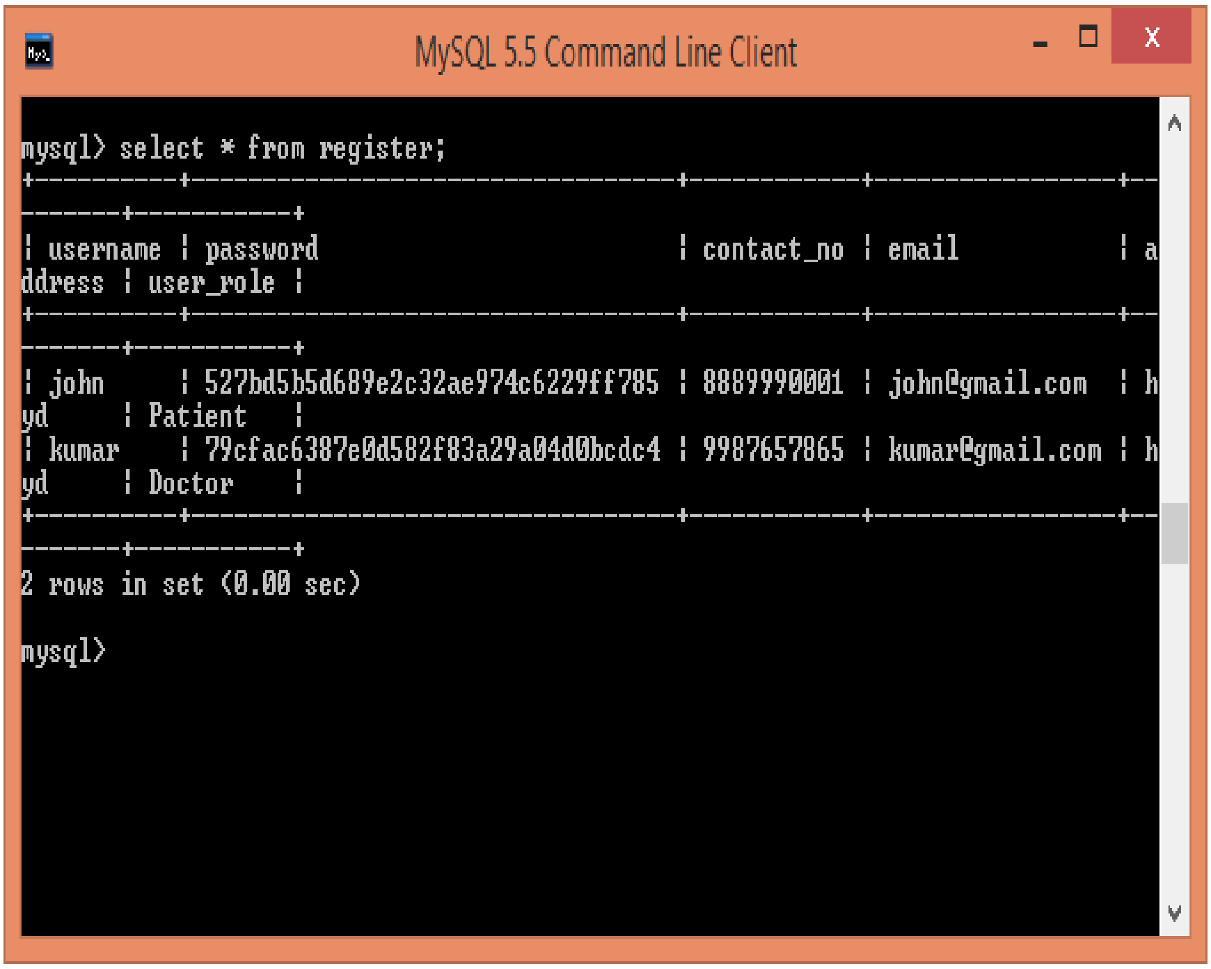
To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

New User Account Creation Screen

Username	kumar
Password	•••••
Contact No	9987657865
Email Id	kumar@gmail.com
Address	hyd
User Type	Patient

Figure 5.3: New User Sign up

5.4 DATABASE SCREEN



MySQL 5.5 Command Line Client

```
mysql> select * from register;
+----------+----------------------------------+------------+-----------------+--
-------+-----------+
| username | password                         | contact_no | email           | a
ddress | user_role |
+----------+----------------------------------+------------+-----------------+--
-------+-----------+
| john     | 527bd5b5d689e2c32ae974c6229ff785 | 8889990001 | john@gmail.com  | h
yd     | Patient   |
| kumar    | 79cfac6387e0d582f83a29a04d0bcdc4 | 9987657865 | kumar@gmail.com | h
yd     | Doctor    |
+----------+----------------------------------+------------+-----------------+--
-------+-----------+
2 rows in set (0.00 sec)

mysql>
```

Figure 5.4: Encrypted Password

5.5 PATIENT LOGIN

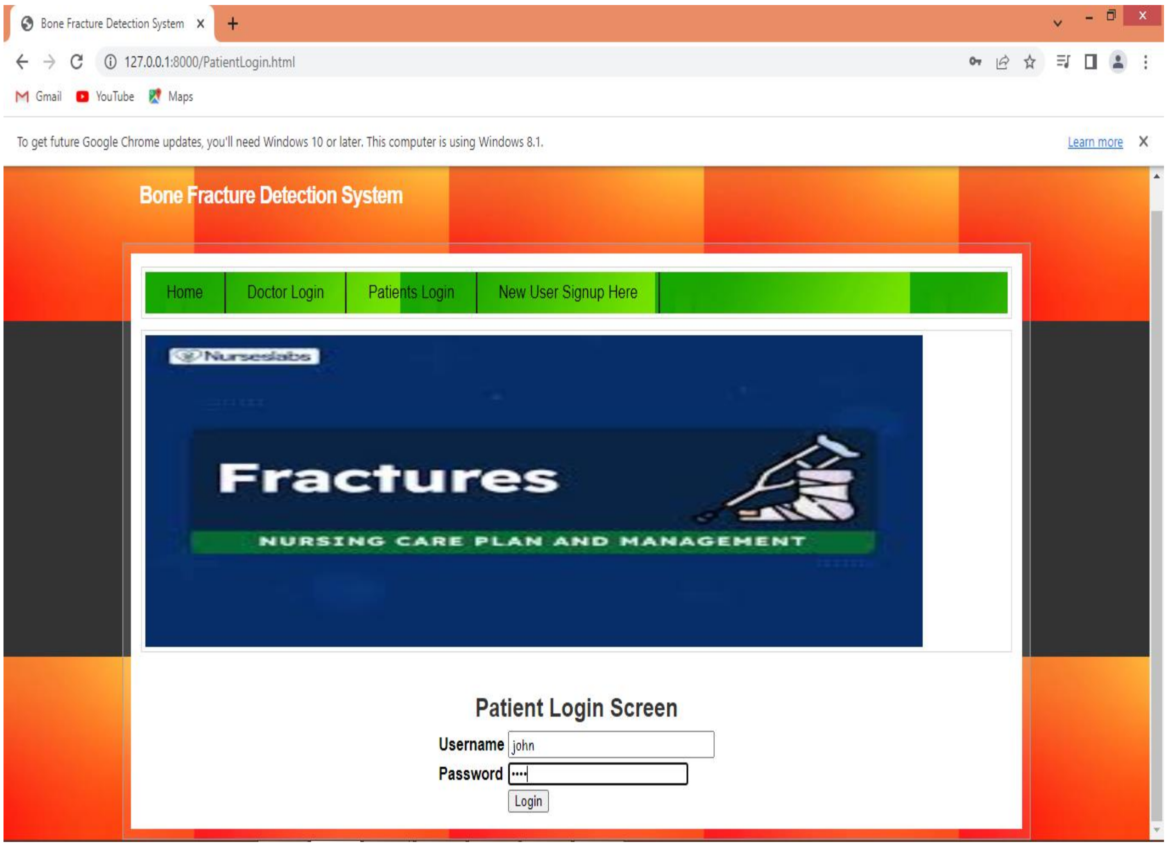
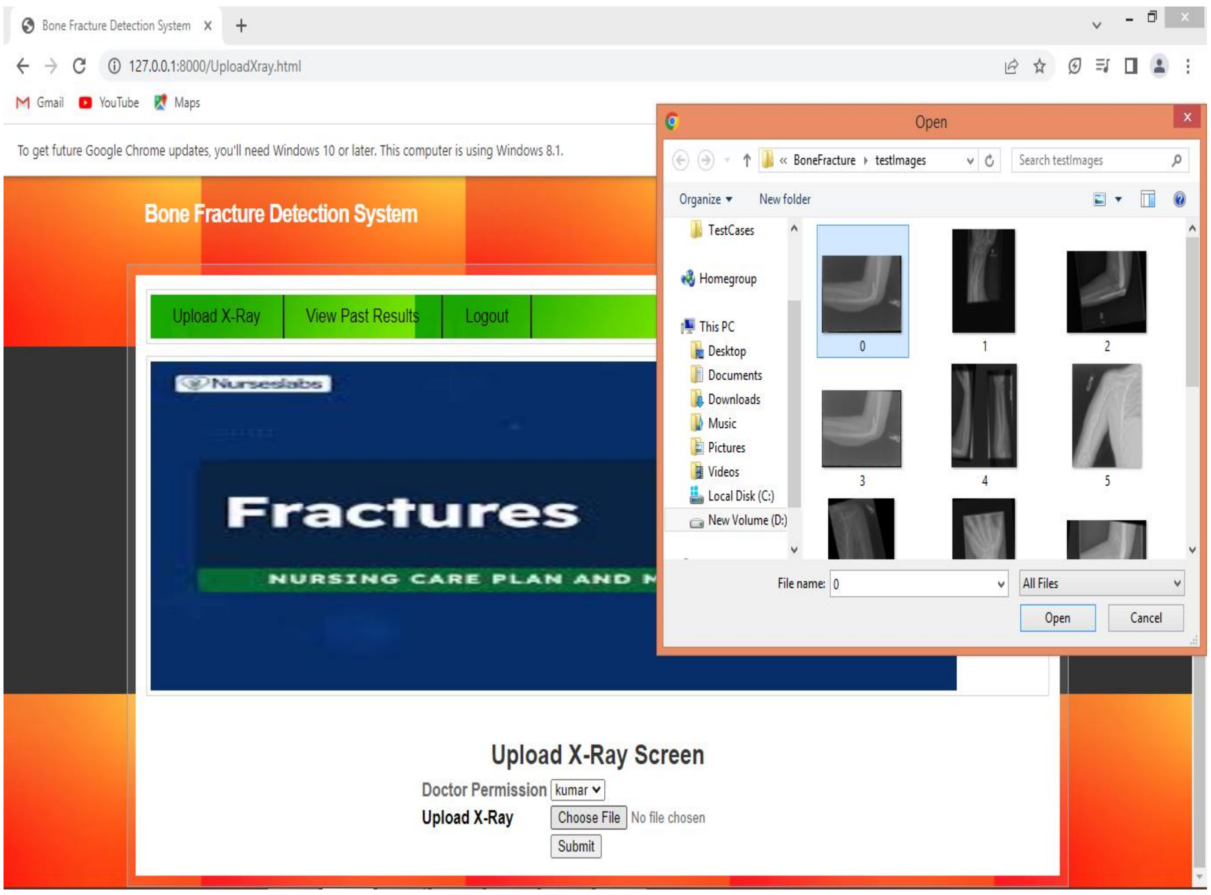

Bone Fracture Detection System
127.0.0.1:8000/PatientLogin.html
Gmail
YouTube
Maps
To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.
Learn more
Bone Fracture Detection System
Home
Doctor Login
Patients Login
New User Signup Here
Nurseslabs
Fractures
NURSING CARE PLAN AND MANAGEMENT
Patient Login Screen
Username
john
Password
Login

Figure 5.5: Patient Login

5.6 UPLOAD X-RAY IMAGE SCREEN



To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

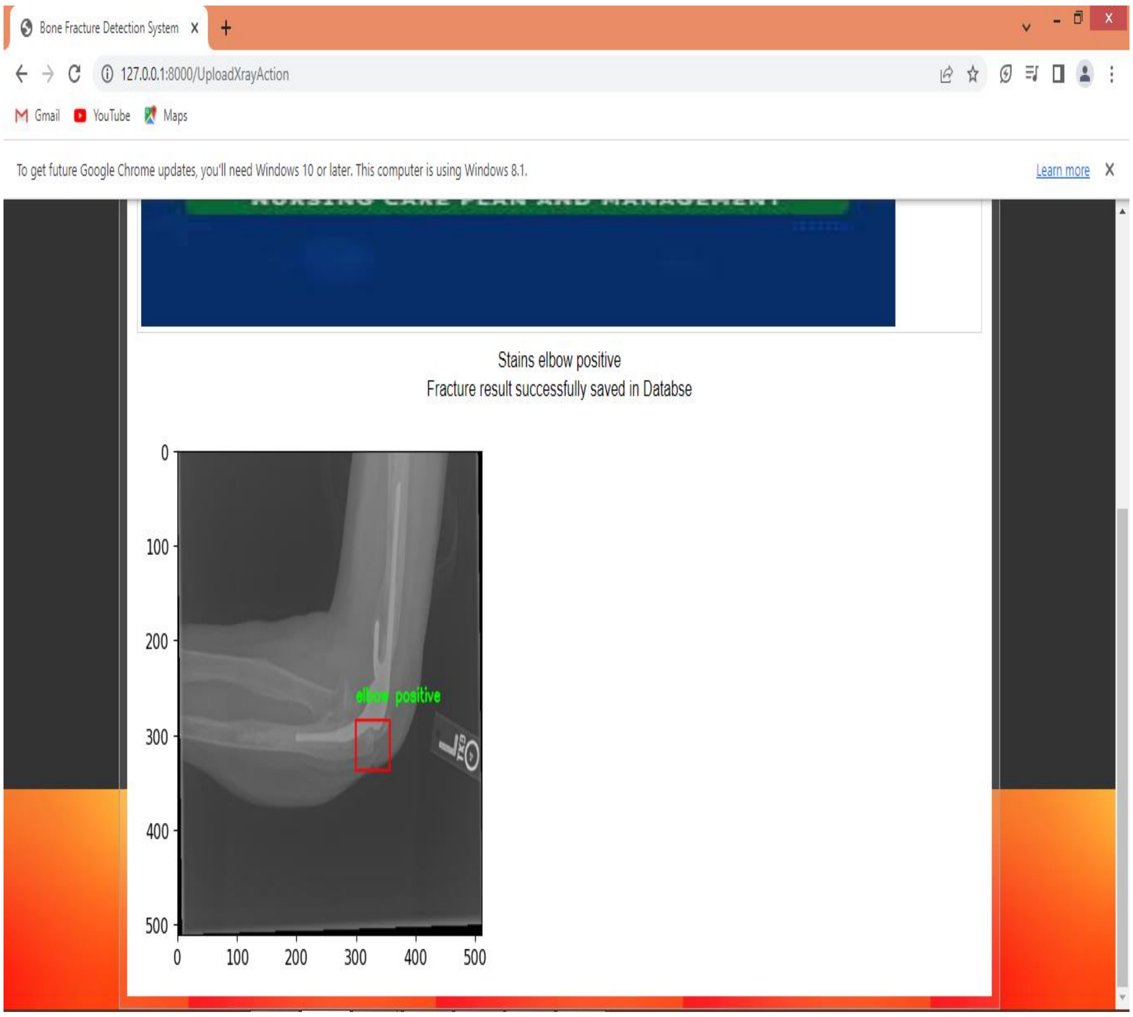
Upload X-Ray Screen

Doctor Permission: kumar

Upload X-Ray: No file chosen

Figure 5.6: Upload X-Ray

5.7 ASSISTANCE SCREEN



Bone Fracture Detection System

127.0.0.1:8000/UploadXrayAction

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

Stains elbow positive

Fracture result successfully saved in Databse

Figure 5.7: Bone Fracture Detected at Elbow

5.8 DOCTOR LOGIN

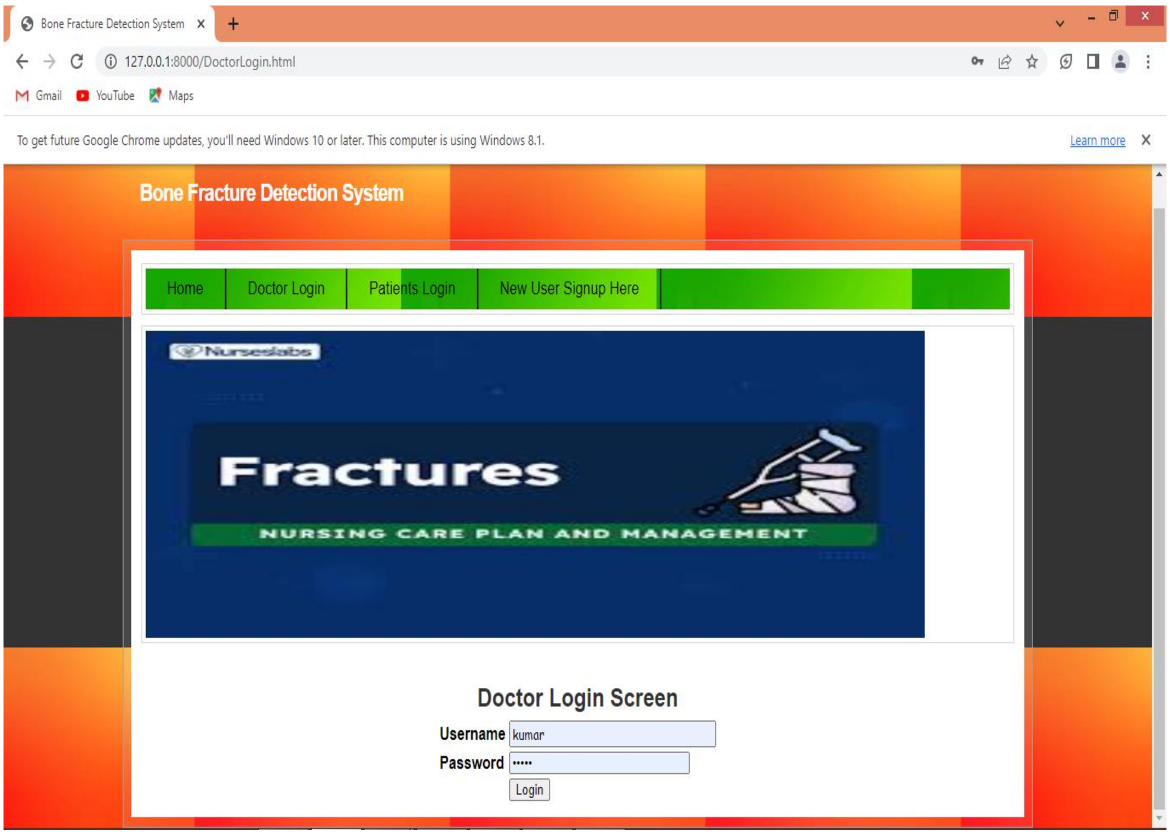
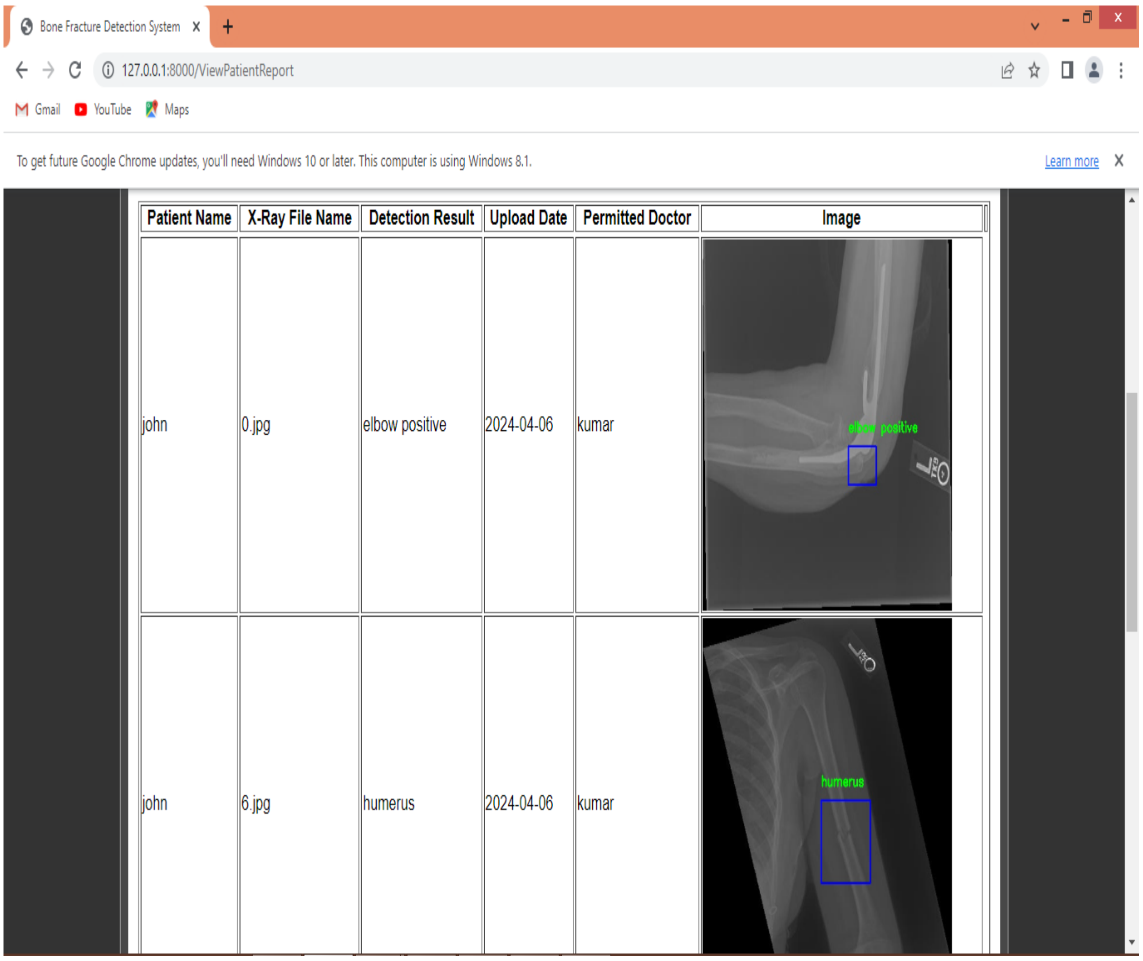

Bone Fracture Detection System
127.0.0.1:8000/DoctorLogin.html
Gmail
YouTube
Maps
To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.
Learn more
Bone Fracture Detection System
Home
Doctor Login
Patients Login
New User Signup Here
Nurseslabs
Fractures
NURSING CARE PLAN AND MANAGEMENT
Doctor Login Screen
Username
kumar
Password
•••••
Login

Figure 5.8: Doctor login

5.9 PATIENT REPORT SCREEN


Bone Fracture Detection System
127.0.0.1:8000/ViewPatientReport
Gmail
YouTube
Maps
Learn more
elbow positive
humerus

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 8.1.

Patient Name	X-Ray File Name	Detection Result	Upload Date	Permitted Doctor	Image
john	0.jpg	elbow positive	2024-04-06	kumar	
john	6.jpg	humerus	2024-04-06	kumar	

Figure 5.9: Patients report

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

6.2.4 SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.2.5 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6.2.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document. The test provides inputs and responds to outputs without considering how the software works.

6.3 TEST CASES

6.3.1 UPLOADING IMAGES

Test case ID	Test case name	Purpose	Test Case	Output
1	User uploads image	Use it for identification	The user uploads the fractured x-ray image of bone	Uploaded successfully
2	User uploads 2nd image	Use it for identification	The user uploads the non-fractured x-ray image of bone	Uploaded successfully

6.3.2 CLASSIFICATION

Test case ID	Test case name	Purpose	Input	Output
1	Classification test 1	To check if the classifier performs its task	A fractured bone image is given	Bone Fracture is detected.
2	Classification test 2	To check if the classifier performs its task	A non-fractured bone image is given	No Bone Fracture.
3	Classification test 3	To check if the classifier performs its task	A random image is given	Predicted as not a bone x-ray image.

7.CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

In conclusion, the proposed AI-driven bone fracture detection system offers a transformative solution to the limitations of traditional diagnostic methods. By leveraging advanced machine learning algorithms, the system significantly improves the accuracy, speed, and consistency of fracture detection, reducing the reliance on human interpretation and minimizing diagnostic errors. Its ability to provide real-time analysis and seamlessly integrate into clinical workflows makes it a valuable tool for radiologists, particularly in high-pressure and high-volume environments such as emergency departments. Moreover, by enhancing accessibility in regions with limited medical expertise and reducing healthcare costs, the system holds great potential for improving patient care and outcomes on a broader scale. As AI technology continues to evolve, such systems are poised to become essential components of modern healthcare, offering more efficient and reliable fracture detection.

The constraints are met and overcome successfully. The system is designed as like it was decided in the design phase. The project gives good idea on developing a full-fledged application satisfying the user requirements.

The system is very flexible and versatile. Validation checks induced have greatly reduced errors. Provisions have been made to upgrade the software. The application has been tested with live data and has provided a successful result. Hence the software has proved to work efficiently.

7.2 FUTURE SCOPE

The future scope of the proposed bone fracture detection system is vast, with potential for significant advancements and applications. As AI and deep learning technologies continue to evolve, the system could be further optimized for detecting a wider variety of bone conditions, including micro-fractures, joint dislocations, and bone abnormalities associated with diseases like osteoporosis or bone cancer. Integration with other imaging modalities such as MRI and 3D CT scans could enhance diagnostic accuracy in complex cases. Additionally, the system could be expanded to include predictive analytics, helping clinicians anticipate fracture risks based on patient history and imaging data.

8.BIBLIOGRAPHY

8.1 REFERENCES

[1] Esteva, A., et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115-118.

[2] Rajpurkar, P., et al. (2018). CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning. arXiv preprint arXiv:1711.05225.

[3] Gale, W., et al. (2020). Deep learning for detecting wrist fractures in pediatric radiographs. Medical Image Analysis, 63, 101715.

[4] Lindsey, R., et al. (2018). Deep neural networks for automatic fracture detection in radiographs. Journal of Digital Imaging, 31(6), 756-764.

[5] Olczak, M., et al. (2017). Artificial Intelligence for Fracture Detection in Skeletal Radiographs. IEEE Transactions on Medical Imaging, 36(9), 1890-1900.

[6] Kim, H., et al. (2020). Automated Hip Fracture Detection from Pelvic Radiographs Using Deep Learning. Journal of Orthopedic Surgery and Research, 15(1), 333.

8.2 WEBSITES

[1] http://127.0.0.1:8000/index.html

[2] https://github.com/peetamyuktha/Bone-Fracture-Detection