



People's Democratic Republic of Algeria  
Ministry of Higher Education and Scientific Research  
University of Mohamed Khider – BISKRA  
Faculty of Exact Sciences, Science of Nature and Life  
Computer Science Department

Number Of Order : ../../2025

## End of study project report

Submitted in partial fulfillment of the requirements for the Master's degree  
in Information Systems, Optimization and Decision

---

# Medico Plus: An AI-powered Digital Healthcare Platform

---

Presented by:

**Barkat Mayar Wijdane**

**Lanani Ghoufrane**

**Merzoug Djihane**

Supervisor: **Dr Mokhtari Bilal**

Co-supervisor: **Dr Merizig abdelhak**

**Session : 2024/2025**

# Acknowledgments

---

We would like to express our deepest gratitude and appreciation to **Allah**, the Most Merciful and the Most Gracious, for His guidance and countless blessings throughout this journey.

We also extend our heartfelt thanks to our supervisor, **Dr. Mokhtari Bilal**, and our co-supervisor, **Dr. Mrezig Abdelhak**, for their invaluable guidance, unwavering support, and constant encouragement. Their expertise, patience, and dedication were instrumental in shaping this thesis and in helping us overcome numerous challenges along the way. We are truly grateful for their mentorship and the knowledge they have generously shared with us.

Our sincere appreciation also goes to the **Department of Computer Science** and to all the teachers who have accompanied us throughout our academic journey. Their support, teachings, and encouragement have laid the foundation for our growth and success.

In conclusion, this thesis is the result of collective efforts, and we are truly grateful to each and every person who played a part, directly or indirectly, in its completion. May Allah bless you all abundantly for your kindness, support, and belief in us.

Barkat Mayar Wijdane

Lanani Ghoufrane

Merzoug Djihane

# Contents

Abbreviations	V
List of Figures	VI
List of Tables	VIII
Abstract	1
General introduction	2
<b>1 AI in Healthcare: Focus on Medical Image Processing</b>	<b>4</b>
1.1 Introduction . . . . .	4
1.2 Definition of Artificial intelligence . . . . .	4
1.3 Definition of Medical Imaging . . . . .	5
1.4 Role of AI in Medical Imaging . . . . .	5
1.5 Historical Evolution of AI in Medical Imaging . . . . .	6
1.6 Applications of AI in Medical Imaging . . . . .	7
1.6.1 Disease diagnosis and prognosis . . . . .	7
1.6.2 Radiomics and predictive modeling . . . . .	8
1.6.3 Image segmentation and organ localization . . . . .	8
1.7 Challenges and Ethical Considerations of AI in Medical Imaging . . . . .	8
1.7.1 Data privacy and security . . . . .	8
1.7.2 Validation and standardization . . . . .	9
1.7.3 Interpretable AI in clinical practice . . . . .	9
1.8 Future of AI in Medical Imaging . . . . .	9
1.9 The Integration of CNN, XAI, and NLG in Medical Imaging: Methodology and Implementation . . . . .	10
1.9.1 Convolutional Neural Network (CNN) . . . . .	11
1.9.2 Explainable Artificial Intelligence (XAI) . . . . .	14
1.9.3 Natural Language Generation (NLG) . . . . .	16
1.9.3.1 Importance and relevance of NLG . . . . .	17
1.9.3.2 Evolution of NLG techniques and algorithms . . . . .	17
1.9.3.3 NLG in Healthcare . . . . .	18

<b>2</b>	<b>Web Application</b>	<b>21</b>
2.1	Introduction . . . . .	21
2.2	Definition of Web Application . . . . .	21
2.3	Web Application vs Website . . . . .	22
2.4	Architectural Layers of Web Development . . . . .	24
2.4.1	Front-End Devolopment . . . . .	24
2.4.2	Back-End Devolopment . . . . .	25
2.4.2.1	Technologies and Programming Languages . . . . .	25
2.4.3	Full Stack Devolopment . . . . .	26
2.4.4	Databases . . . . .	27
2.4.4.1	The main differences between SQL and NoSQL . . . . .	27
2.5	Client Server Architecture . . . . .	28
2.5.1	Definition of Client-Server Architecture . . . . .	28
2.5.2	Types Of Client-Server Architectures . . . . .	29
<b>3</b>	<b>Conception</b>	<b>32</b>
3.1	Introduction . . . . .	32
3.2	Principal Actors . . . . .	32
3.3	Functional and Non-Functional Requirements . . . . .	33
3.3.1	Functional requirements . . . . .	33
3.3.2	Non Functional requirements . . . . .	34
3.4	Unified Modeling Language (UML) . . . . .	35
3.4.1	Use Case Diagram . . . . .	35
3.4.1.1	Use Case Diagram of the Web Application . . . . .	35
3.4.2	Class Diagram . . . . .	37
3.4.3	Sequence Diagram . . . . .	38
<b>4</b>	<b>Implementation and GUI of The Web Application</b>	<b>44</b>
4.1	Introduction . . . . .	44
4.1.1	The Material Environment: . . . . .	44
4.1.2	The Software Environment : . . . . .	45
4.2	Graphical User Interface (GUI) . . . . .	46
4.2.1	Login Interface . . . . .	46
4.2.2	Sign-Up Interface . . . . .	47
4.2.3	Landing page Interface . . . . .	48
4.2.4	Doctor Profile Settings Interface . . . . .	49
4.2.5	Doctor Notification Interface . . . . .	49
4.2.6	User Profile Settings Interface . . . . .	50
4.2.7	User Notification Interface . . . . .	50
4.2.8	Appointment Scheduling Interface . . . . .	51
4.2.9	Doctor Appointment List Interface . . . . .	52
4.2.10	Doctor Analytics Interface . . . . .	52
4.2.11	Doctor Payment Interface . . . . .	53

4.2.12	User Home Page Interface . . . . .	54
4.2.13	User Search Doctor Interface . . . . .	56
4.2.14	Doctor Profile: Search Result Interface . . . . .	56
4.2.15	User Book Appointment Interface . . . . .	57
4.3	User Appointment List Interface . . . . .	58
4.3.1	Available Ambulance List Interface . . . . .	59
4.3.2	Register Ambulance Service Interface . . . . .	59
4.3.3	Ambulance Tracking Interface . . . . .	60
4.3.4	AI Assistant Interface . . . . .	60
4.4	Conclusion . . . . .	64
<b>General Conclusion</b>		<b>65</b>
<b>Bibliography</b>		<b>67</b>

# Abbreviations

<b>AI</b>	Artificial Intelligence
<b>XAI</b>	Explainable Artificial Intelligence
<b>NLG</b>	Natural Language Generation
<b>CNN</b>	Convolutional Neural Network
<b>ML</b>	Machine Learning
<b>DL</b>	Deep Learning
<b>NLP</b>	Natural Language Processing
<b>X-ray</b>	X-radiation
<b>MRI</b>	Magnetic Resonance Imaging
<b>CT</b>	Computed Tomography
<b>PET</b>	Positron Emission Tomography
<b>CAD</b>	Computer-Aided Diagnosis
<b>ANNs</b>	Artificial Neural Networks
<b>GPUs</b>	Graphics Processing Units
<b>GANs</b>	Generative Adversarial Networks
<b>RNNs</b>	Recurrent Neural Networks
<b>LLM</b>	Large Language Model
<b>HTML</b>	HyperText Markup Language
<b>CSS</b>	Cascading Style Sheets
<b>JS</b>	JavaScript
<b>CRUD</b>	Create, Read, Update, Delete Operations
<b>PHP</b>	Hypertext Preprocessor
<b>CGI</b>	Common Gateway Interface Function
<b>ORM</b>	Object-Relational Manager
<b>RDBMS</b>	Relational Database Management System
<b>UML</b>	Unified Modeling Language

# List of Figures

1.1	Medical imaging modalities . . . . .	5
1.2	CNN Architecture [20]. . . . .	12
1.3	EfficientNet Architecture [24]. . . . .	13
1.4	Interpretability map using Grad-CAM on a brain MRI scan . . . . .	16
2.1	Full Stack Development . . . . .	26
2.2	Client-server architecture . . . . .	28
3.1	Use Case Diagram of the Web Application . . . . .	36
3.2	Class Diagram of the Web Application . . . . .	37
3.3	Log-In Sequence Diagram of the Web Application . . . . .	38
3.4	Ai Assistant Sequence Diagram of the Web Application . . . . .	39
3.5	Doctor's Appointment List Viewing Sequence Diagram of the Web Application	40
3.6	User Appointment Booking Sequence Diagram of the Web Application . .	41
3.7	Patient Searches for An Ambulance Sequence Diagram of the Web Application	42
4.1	Login Interface. . . . .	46
4.2	Sign-Up Interface. . . . .	47
4.3	Landing Page Interface. . . . .	48
4.4	Doctor Profile Settings Interface. . . . .	49
4.5	Doctor Notification Interface. . . . .	49
4.6	User Profile Settings Interface. . . . .	50
4.7	User Notification Interface. . . . .	50
4.8	Appointment Scheduling Interface. . . . .	51
4.9	Doctor Appointment List Interface. . . . .	52
4.10	Doctor Analytics Interface. . . . .	53
4.11	Doctor Payment Interface. . . . .	54
4.12	User Home Page Interface. . . . .	55
4.13	User Search Doctor Interface. . . . .	56
4.14	Search Result Interface. . . . .	57
4.15	Search Result Interface. . . . .	57
4.16	User Book Appointment Interface. . . . .	58
4.17	User Appointment List Interface. . . . .	58
4.18	User Available Ambulance List Interface. . . . .	59
4.19	Register Ambulance Service Interface. . . . .	59

4.20 Ambulance Tracking Interface. . . . .	60
4.21 AI Assistant Interface -Part 01. . . . .	61
4.22 AI Assistant Interface -Part 02. . . . .	61
4.23 Medical Imaging Diagnostic Report -Part 01 . . . . .	62
4.24 Medical Imaging Diagnostic Report -Part 02 . . . . .	63



# List of Tables

Accuracy of CNN Models on Different Medical Image Datasets . . . . .	14
Comparison between 1-tier, 2-tier, 3-tier, and N-tier Architectures. . . .	30
Comparison between Websites and Web Applications [34]. . . . .	23
Hardware environment used during development . . . . .	45

# Abstract

The swift development of artificial intelligence has opened the door for intelligent systems that have the potential to revolutionize a number of industries, with the healthcare industry leading the way. The Algerian healthcare system still faces enduring issues like delayed medical diagnosis, traditional phone appointment scheduling, and restricted access to emergency services, even with technology developments. This project addresses these problems by introducing Medico Plus, a digital healthcare platform driven by AI that offers precise, timely, and accessible medical assistance.

To enhance diagnostics and overall user experience, the platform integrates Convolutional Neural Networks (CNNs) for automatic classification of medical images, Explainable AI (XAI) techniques such as Grad-CAM for decision transparency, and Natural Language Generation (NLG) with TinyLLaMA to produce human-readable medical reports. A user-friendly web interface enables real-time appointment booking, medical image uploads, and emergency ambulance access. Through the integration of these intelligent components, Medico Plus seeks to empower patients and healthcare professionals with an easily navigable digital environment that is adapted to the Algerian context, modernize healthcare service delivery, and decrease diagnostic errors.

# General introduction

Algeria's healthcare system is currently grappling with a multitude of pressing challenges that hinder its ability to deliver timely, efficient, and high-quality medical care. Among the most critical issues are inadequate patient care supervision, inefficient appointment scheduling mechanisms, limited emergency response coordination, and a general lack of intelligent decision-support tools to assist healthcare professionals. As Algeria's population continues to grow and the demand for medical services escalates, these shortcomings have become increasingly detrimental, revealing a significant gap between existing healthcare infrastructure and the dynamic needs of patients and healthcare providers alike.

One of the most glaring deficiencies in the current system is the absence of integrated digital solutions capable of streamlining clinical workflows, reducing the administrative burden on healthcare staff, and ensuring rapid and accurate patient management. In an era marked by rapid advancements in digital health technologies and artificial intelligence, Algeria's healthcare sector remains in urgent need of innovative tools that can support clinical decision-making, optimize resource allocation, and improve the overall patient experience.

This project addresses these challenges through the development of a smart, web-based healthcare management platform tailored specifically to the Algerian healthcare context. The proposed system consolidates multiple essential services into a single, user-friendly digital interface. These include AI-assisted diagnostic tools, real-time ambulance dispatch and tracking, and an efficient appointment scheduling system. By integrating cutting-edge technologies, the platform aims not only to modernize healthcare delivery but also to alleviate the burdens faced by both medical personnel and patients.

At the core of the system lies a suite of advanced artificial intelligence models capable of accurately identifying potential illnesses through the analysis of medical images. To ensure transparency and trustworthiness in the diagnostic process, Explainable AI (XAI) tech-

niques—such as Gradient-weighted Class Activation Mapping (Grad-CAM)—are utilized to generate intuitive heatmaps that highlight the most diagnostically significant regions of each image. This empowers healthcare professionals to understand and validate the AI’s decisions more effectively.

Additionally, the platform employs Natural Language Generation (NLG) algorithms to convert complex diagnostic outputs into clear, coherent, and accessible medical reports. This feature is particularly valuable for enabling patients, caregivers, and non-specialist staff to comprehend the diagnostic findings without requiring deep medical expertise.

The structure of this report is organized into four comprehensive chapters. The first chapter explores the role of artificial intelligence in modern healthcare systems, with a special emphasis on medical image analysis. The second chapter provides an in-depth description of the architecture and key functionalities of the developed web application. The third chapter presents the system’s design through various diagrams, illustrating the technical and logical structure of the platform. Finally, the fourth chapter focuses on the graphical user interface (GUI) and outlines the steps taken during the system’s implementation.

Together, these components represent a holistic and practical solution aimed at transforming Algeria’s healthcare landscape through the power of artificial intelligence and digital innovation.

# Chapter 1

## AI in Healthcare: Focus on Medical Image Processing

### 1.1 Introduction

With the rise of artificial intelligence (AI), healthcare has entered a new era of innovation that has transformed diagnostics, decision-making, and patient care. This chapter explores the wide-ranging effects of artificial intelligence, with an emphasis on its innovative uses in medical imaging. We'll look at the evolution of artificial intelligence (AI) in healthcare, along with its current obstacles and promising future prospects. This chapter's primary goal is to examine the methods and resources that underpin our project's AI-driven medical imaging. From Natural Language Generation (NLG) for automated reporting and Explainable AI (XAI) techniques for model transparency to Convolutional Neural Networks (CNNs) for image identification, these technologies are revolutionizing this sector.

### 1.2 Definition of Artificial intelligence

Artificial intelligence (AI) is the theory and development of computer systems capable of performing tasks that historically required human intelligence, such as recognizing speech, making decisions, and identifying patterns. AI is an umbrella term that encompasses a wide variety of technologies, including machine learning (ML), deep learning (DL), and natural language processing (NLP) [\[1\]](#).

## 1.3 Definition of Medical Imaging

Medical imaging involves the application of various technologies and techniques to create visual representations of the internal structures and functions of the human body for the purposes of diagnosis and treatment. It is an essential tool in modern medicine, enabling doctors and other healthcare professionals to detect and diagnose a wide range of diseases and conditions. There are several types of medical imaging, including X-ray, Computed Tomography (CT), Magnetic Resonance Imaging (MRI), ultrasound imaging, optical imaging and nuclear medicine imaging. Each of these techniques has its own advantages and limitations and may be used for various imaging investigations, depending on the specific needs of the patient and the healthcare provider [2].

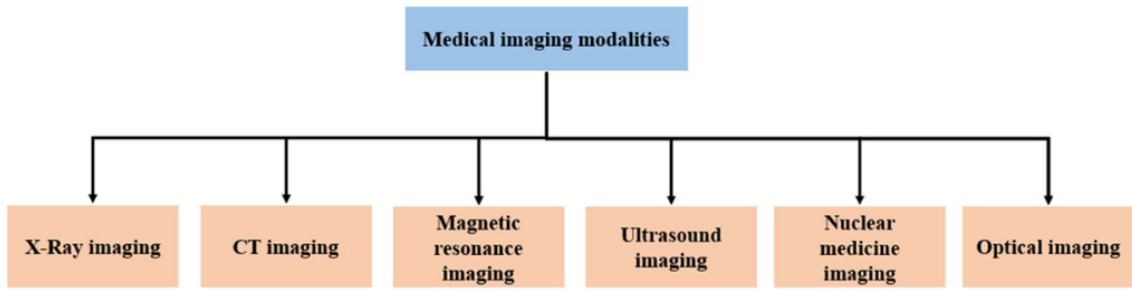


Figure 1.1 – Medical imaging modalities

Figure 1.1 depicts the most frequent types of medical imaging, which include X-rays, CT scans, MRIs, ultrasound imaging, nuclear medicine imaging, optical imaging. Each of these procedures provides a distinct sort of anatomical or functional information and is used according to the clinical requirement.

## 1.4 Role of AI in Medical Imaging

Medical imaging techniques such as CT, MRI, and positron emission tomography (PET) play a pivotal role in providing clinicians with detailed and comprehensive visual information about the human body. These imaging modalities generate large volumes of data, which artificial intelligence (AI) helps analyze and interpret efficiently [3]. AI, particularly deep learning algorithms, has demonstrated remarkable capabilities in extracting valuable insights from medical images [4].

Deep learning models trained on large datasets are capable of :

- Recognizing complex patterns and features that may not be readily discernible to the human eye [5, 6].
- Provide a new perspective about what image features should be valued to support decisions [7].
- Enhancing the accuracy and efficiency of disease diagnosis [4, 8].
- Assist healthcare professionals in detecting abnormalities, identifying specific structures, and predicting disease outcomes [8, 9].

## 1.5 Historical Evolution of AI in Medical Imaging

The evolution of Artificial Intelligence (AI) in medical imaging has been a transformative journey. AI has consistently pushed the limits of what is feasible in medical imaging, starting with rule-based systems and continuing with today's complex deep learning models.

### Early Beginnings: 1960s–1980s

The application of AI in medical imaging began in the 1960s with the advent of computer-aided diagnosis (CAD) systems. These early systems helped radiologists analyze medical images. Although promising, these approaches were constrained by their dependence on preset rules, which limited their accuracy and scalability, and their incapacity to adjust to new data [10].

### Emergence of Machine Learning: 1980s–1990s

The introduction of machine learning (ML) and artificial neural networks (ANNs) in the 1980s and 1990s marked a significant step forward. These techniques allowed for more adaptive image analysis, but their potential was constrained by limited computing power and the lack of large, annotated datasets [10].

## **The Deep Learning Revolution: Early 2010s**

The early 2010s marked a breakthrough in the application of deep learning algorithms, particularly Convolutional Neural Networks (CNNs), in medical imaging. These models demonstrated exceptional capabilities in recognizing patterns and features within images, enabling tasks such as segmentation, classification, and anomaly detection. The availability of large-scale datasets like ImageNet and advancements in Graphics Processing Units (GPUs) played a pivotal role in accelerating progress. This period saw the successful application of AI in clinical settings, including diabetic retinopathy detection and breast cancer screening, showcasing its transformative potential in healthcare practices [11, 12].

## **Recent innovations: Late 2010s–2020s**

In recent years, AI-driven medical imaging has continued to evolve with several significant innovations such as

- Generative Adversarial Networks (GANs)
- Explainable AI (XAI)
- Pre-trained Models

These developments have not only increased the precision and dependability of AI systems but have also opened the door for wider implementation in clinical settings.

## **1.6 Applications of AI in Medical Imaging**

Advances in medical imaging and artificial intelligence have opened up new avenues of possibility in the realm of healthcare. The combination of these two realms has transformed many aspects of medical practice.

### **1.6.1 Disease diagnosis and prognosis**

AI-enabled medical imaging has significantly contributed to the early detection and accurate diagnosis of diseases, including cancer, neurological disorders, and cardiovascular conditions [13].

An artificial intelligence (AI)-powered mammography system was created by McKinney et al. that employs deep learning to evaluate pictures and identify breast cancer with



accuracy on par with or better than that of human radiologists. Clinical validation across a range of populations is still necessary, even though the approach successfully lowers false positives and negatives [14].

### **1.6.2 Radiomics and predictive modeling**

Discuss how AI is leveraged to extract quantitative features from medical images (radiomics) and how these features aid in developing predictive models for patient outcomes and treatment responses [13].

Aerts et al. draw attention to the application of radiomics-based predictive modeling, which forecasts patient outcomes and treatment responses by using artificial intelligence to extract high-dimensional quantitative information from medical images. These traits are able to identify tumor heterogeneity and phenotypic patterns that are not readily apparent to the naked eye. Although radiomics has demonstrated promise in non-invasive prognosis and therapeutic guiding, its clinical application is still constrained by issues with interpretability and consistency[15].

### **1.6.3 Image segmentation and organ localization**

AI-based image segmentation techniques have proved invaluable in identifying and delineating anatomical structures, enabling precise treatment planning and surgery [13].

Convolutional neural networks (CNNs) are used in deep learning-based image segmentation and organ localization to automatically recognize and differentiate anatomical characteristics in medical pictures, according to Hesamian et al. In identifying intricate regions of interest, this method has demonstrated excellent accuracy and resilience, assisting with activities like diagnosis and treatment planning [16].

## **1.7 Challenges and Ethical Considerations of AI in Medical Imaging**

### **1.7.1 Data privacy and security**

The integration of AI in medical imaging raises concerns about patient data privacy and data security [13].

The ongoing difficulties in protecting private medical picture data from manipulation and illegal access are examined by Mokkadem et al. Data privacy and integrity are raised by their work, which reveals vulnerabilities that occur when such photographs are kept on cloud-based platforms or transported over networks [17].

### 1.7.2 Validation and standardization

The need for robust validation protocols and standardized datasets to evaluate AI algorithms in medical imaging is explored, ensuring reliable and reproducible results [13]. The significance of consistent methodologies and benchmark datasets for accurately assessing AI algorithms is emphasized in Sourlos et al.’s study on Validation and Standardization in AI Medical Imaging. Variations in data labeling, preprocessing, and reporting hinder repeatability and clinical applicability, as their findings show [18].

### 1.7.3 Interpretable AI in clinical practice

The interpretability of AI models is critical to gain clinician trust and facilitate their adoption into clinical workflows [13].

## 1.8 Future of AI in Medical Imaging

- **Artificial and augmented intelligence** While true artificial intelligence emulates human-like thinking without intervention, augmented intelligence systems in health-care care aim to improve human processes by improving monotonous or burdensome physician workflows. In medical imaging, augmented intelligence improves the collaboration between radiologists and oncologists by reducing repetitive data entry tasks and automatically submitting electronic health records (EHR) with imaging and diagnostic data. Augmented intelligence can also be used to improve image quality, recommend customized imaging protocols based on patient history, and generate automated reports [19].
- **Virtual and augmented reality 3D medical imaging** As powerful as MRIs and CT scans are now, their 2D renders demand physicians to imagine a spatial dimension that they can’t actually see. New augmented reality technologies, like

EchoPixel True 3D, make it possible for physicians to create a 3D image of MRIs. They can then examine the image with a typical VR headset. Physicians can rotate, zoom in, or make cross-sections of the 3D image using peripheral pointing devices and other tools. This enables better visualization and planning before a procedure. Some VR platforms even allow physicians to practice surgeries using these images, or create physical models using 3D printers. AR is like VR in that it creates three-dimensional images but projects them into the real world using spatial computing and a special headset. Companies like Proprio are using machine learning and AR to help surgeons see through obstacles and blockages that may impede a high-risk operation [19].

- **Nuclear imaging** In nuclear imaging, a patient ingests radioactive materials called radiotracers (or radiopharmaceuticals) before a medical imaging scan. During a scan, a camera focuses on where the radioactive material concentrates. These types of scans are particularly helpful when diagnosing internal conditions like thyroid disease, cancer, and Alzheimer’s disease [19].

## 1.9 The Integration of CNN, XAI, and NLG in Medical Imaging: Methodology and Implementation

By improving the speed and precision of medical imaging diagnoses, artificial intelligence (AI) is transforming the healthcare industry. Large volumes of visual data may be quickly analyzed by AI algorithms, which can then identify minute patterns and irregularities that the human eye could miss. Through accurate and dependable image analysis, this skill improves patient outcomes by facilitating the early detection and treatment of diseases including cancer and cardiovascular disorders. AI also makes precision medicine easier by combining genetic data, imaging data, and medical histories to create thorough profiles for individualized treatment regimens. Healthcare providers may decrease diagnostic errors, optimize workflows, and provide more effective and efficient care by integrating AI into medical imaging.

### 1.9.1 Convolutional Neural Network (CNN)

CNN is a deep learning model that uses a grid pattern to interpret data, such as photographs. It is inspired by animal visual cortex and learns spatial hierarchies of information from low to high level patterns. CNN consists of three layers: convolution, pooling, and fully connected. The first two layers, convolution and pooling, extract features, while the third layer, a fully connected layer, transfers them into the final output, such as classification. CNN relies heavily on a convolution layer, which is made up of mathematical procedures like convolution, a specialized linear operation. CNNs are highly efficient for image processing as they store pixel values in a two-dimensional (2D) grid and apply an optimizable feature extractor at each image position. This allows for features to occur anywhere in the image. Extracted features might get increasingly complex as they pass from one layer to the next. Training involves improving parameters, such as kernels, to minimize the gap between outputs and ground truth labels using algorithms like backpropagation and gradient descent [20].

A thorough analysis of convolutional neural networks (CNNs), following their development from early designs to sophisticated models, is given by Li et al. Their overview emphasizes the uses of CNNs in fields like computer vision and natural language processing and examines the basic ideas behind them, such as their ability to acquire hierarchical spatial data through convolutional layers [21].

Figure 1.2 shows a typical Convolutional Neural Network (CNN) architecture for image processing. The procedure starts with an input image, then moves on to consecutive layers of convolution (with ReLU activation) and max pooling operations to extract hierarchical features using trained kernels. The process culminates in backpropagation, which updates the model's weights to improve performance. The diagram accurately depicts the main components of a CNN; however, labeling specific parameters (e.g., kernel sizes, stride) or adding a fully linked layer for classification would improve comprehension for novices.

In this study, we used EfficientNet as the basic deep learning architecture for classification because of its high efficiency and scalability. EfficientNet's compound scaling strategy enables us to balance network depth, width, and resolution to maximize performance while minimizing computing cost. This makes it ideal for medical picture analysis, where high accuracy and efficiency are required.

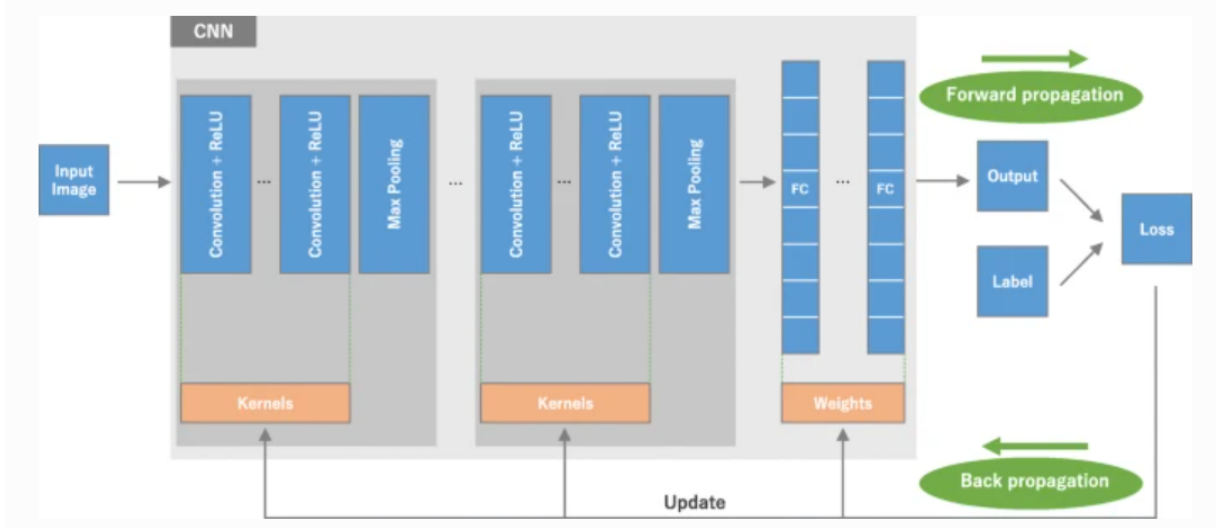


Figure 1.2 – CNN Architecture [20].

## EfficientNet

EfficientNet is a convolutional neural network design and scaling technique that uses a compound coefficient to scale all depth, breadth, and resolution dimensions equally. For example, we may easily expand the depth of the network by  $\alpha^N$ , the breadth by  $\beta^N$ , and the size of the picture by  $\gamma^N$  if we wish to employ  $2^N$  times more computational resources. In this case,  $\alpha$ ,  $\beta$ , and  $\gamma$  are constant coefficients that were found via a small grid search on the initial small model. In a rational manner, EfficientNet scales network breadth, depth, and resolution uniformly using a compound coefficient  $\phi$  [22].

The idea that a larger input image requires more layers to expand the receptive field and more channels to capture finer-grained patterns on the larger image justifies the use of the compound scaling method [22].

The EfficientNet family of models, presented by Tan and Le (2020), is intended to improve accuracy and processing efficiency. In order to concurrently modify depth, breadth, and resolution, they suggest a compound scaling technique. In addition to lowering processing costs, this balanced scaling approach produces models (B0 to B7) that perform better on test datasets like ImageNet than other CNN designs [23].

Figure 1.3 depicts a convolutional neural network (CNN) architecture that begins with an input image processed by an initial 3x3 convolution, followed by several MBConv (Mobile Inverted Bottleneck Convolution) blocks. These blocks, designated 1–7, are made up of MBConv layers with expansion factors of 1 or 6 and kernel sizes of 3x3 or 5x5, which are

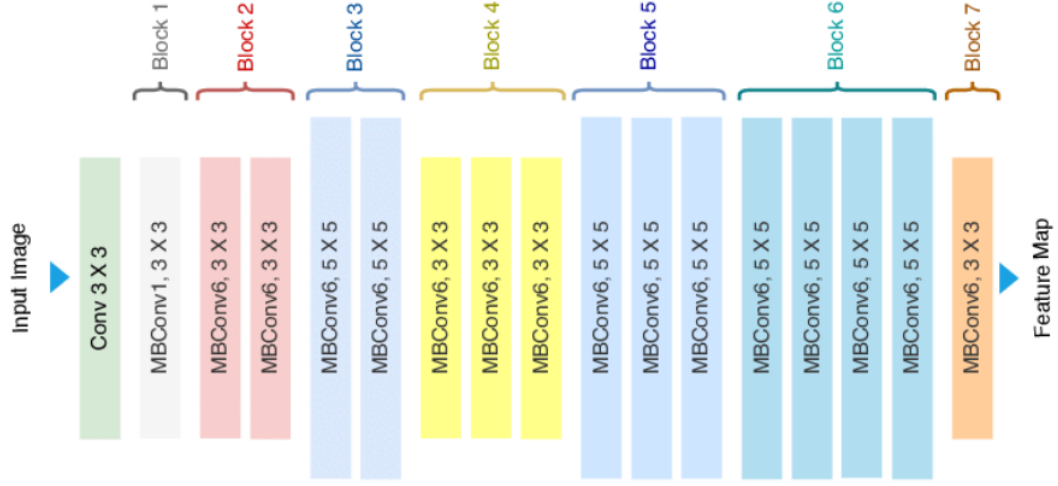


Figure 1.3 – EfficientNet Architecture [24].

designed to efficiently extract multi-scale features. The network finally produces a feature map, which represents the final high-level features extracted from the input image. This topology is typical of efficient CNN architectures, such as MobileNet or EfficientNet, which balance performance and computational cost.

## Model Comparison and Justification of EfficientNet Selection

In order to determine and choose the best convolutional neural network (CNN) architecture for medical image classification, we experimented with three distinct medical image datasets that represented kidney problems, lung disorders, and brain tumors. Approximately 6,000 brain images, 4,026 lung images, and 3,800 kidney images made up the datasets, which were all taken from publicly available medical imaging repositories. All images were preprocessed before training, such as resizing to fit the expected input dimensions of each model (e.g.,  $224 \times 224$  for VGG16,  $240 \times 240$  for EfficientNetB1, and  $299 \times 299$  for InceptionV3), normalization, and augmentation methods. We precisely chose a batch size for each model that was appropriate for its computational cost and architectural depth, guaranteeing training stability and effective memory use. The classification accuracy attained by each model on the corresponding organ dataset is summarized in the table below.

Table 1.1 – Accuracy of CNN Models on Different Medical Image Datasets

Model	Organ	Accuracy (%)
InceptionV3	Brain	95.00
InceptionV3	Kidney	88.89
InceptionV3	Lung	92.93
VGG16	Brain	78.00
VGG16	Kidney	67.00
VGG16	Lung	85.46
DenseNet121	Brain	91.92
DenseNet121	Kidney	89.93
DenseNet121	Lung	92.71
EfficientNetB1	Brain	99.89
EfficientNetB1	Kidney	99.79
EfficientNetB1	Lung	99.77

EfficientNetB1 was selected as the primary model for medical image classification because of its higher accuracy, consistency across organs, and computational efficiency, as shown by the comparison result shown in Table 1.1. Achieving impressive accuracies of 99.89% for brain and 99.79% for kidney images, EfficientNetB1 surpassed DenseNet121 and InceptionV3, which both performed well, especially on the brain and lung datasets.

### 1.9.2 Explainable Artificial Intelligence (XAI)

Explainable artificial intelligence (XAI) is a collection of procedures and techniques that allows human users to comprehend and have faith in the output and outcomes produced by machine learning algorithms. The term "Explainable AI" refers to an AI model's anticipated effects and possible biases. In AI-powered decision making, it aids in describing model correctness, fairness, transparency, and results. When implementing AI models in production, a company needs explainable AI to foster confidence and trust. An organization can also embrace a responsible approach to AI development with the aid of AI explainability.

The increasing sophistication of AI makes it harder for humans to understand and trace

the algorithm's path to a conclusion. The entire computation process is transformed into what is often called a "black box" that is incomprehensible. The data is used directly to generate these black box models. Furthermore, not even the data scientists or engineers who developed the algorithm are able to comprehend or describe what is going on inside of them or how the AI algorithm came to a particular conclusion [25].

Key ideas and terminology in Explainable AI (XAI) are presented in a thorough literature analysis by Mersha et al., which also highlights the importance of interpretability, names its beneficiaries, classifies XAI methodologies, and illustrates how they are used in various fields [26].

### **Benefits of XAI**

- **Operationalize AI with trust and confidence**

Increase trust in production AI. Rapidly deploy your AI models into production. Ensure that AI models are interpretable and explainable. Reduce the complexity of model assessment while boosting model transparency and traceability [25] .

- **Speed time to AI results**

Systematically monitor and manage models to optimize business outcomes. Continue to assess and enhance the model's performance. Adjust model development initiatives in light of ongoing assessment [25] .

- **Mitigate risk and cost of model governance**

Keep your AI models understandable and transparent. Manage regulatory, compliance, and risk-related obligations. Reduce the costs associated with human inspection and mistakes. Reduce the possibility of inadvertent bias [25] .

Grad-CAM (Gradient-weighted Class Activation Mapping) was used in our study as an important component of our Explainable AI (XAI) architecture. Grad-CAM helped us view and analyze our deep learning model's decision-making process by highlighting the regions of the input picture that had the most influence on the model's predictions.

### **Gradient-weighted Class Activation Mapping (Grad-CAM)**

The gradient-weighted class activation map (Grad CAM) generates a heat map that emphasizes significant portions of an image based on the final convolutional layer's target gradients. The Grad CAM method is a common visualization tool for understanding



how a convolutional neural network was trained to reach a classification judgment. It is class-specific, which means that it may generate a unique visualization for each class in the image. In the case of a classification error, this approach can be quite valuable in determining where the fault sits within the convolutional network. It also makes the algorithm more understandable [27]. The approach known as Gradient-weighted Class Activation Mapping (Grad-CAM), which was first presented by Selvaraju et al., uses localization maps to illustrate the decisions made by convolutional neural networks (CNNs). The gradients of a target class that flow into the last convolutional layer are used to emphasize areas of the picture that are most important for the model's prediction [28].

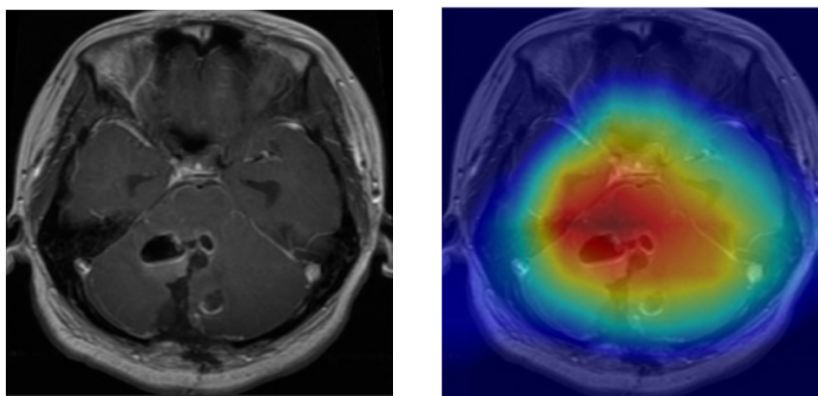


Figure 1.4 – Interpretability map using Grad-CAM on a brain MRI scan

Figure 1.4 shows how Grad-CAM highlights the regions that contributed the most to the decision of classification.

### **1.9.3 Natural Language Generation (NLG)**

Natural Language Generation (NLG) is the process of automatically creating human-readable text or voice from structured data or other types of input. NLG systems examine incoming data, comprehend its meaning, and produce coherent and context-appropriate language output. NLG includes a variety of techniques, ranging from rule-based systems to complex deep learning models, all targeted at converting structured data into natural

language form. NLG has received a lot of attention in academia and industry because of its ability to automate content generating processes, improve human-computer interaction, and facilitate communication across multiple fields [29].

Gatt and Krahmer’s thorough review of Natural Language creation (NLG) research during the previous 20 years examines deep learning approaches for data-to-text and text-to-text creation. Key NLG tasks and datasets are described in the paper, along with assessment issues and how NLG might be integrated with other AI domains like computer vision and computational creativity [30].

#### **1.9.3.1 Importance and relevance of NLG**

NLG is extremely important and relevant in a variety of disciplines, including as journalism, healthcare, business intelligence, and virtual assistants. NLG systems have been used in journalism to generate news items automatically from structured data, improving content production and distribution efficiency. Similarly, in healthcare, NLG is used to generate patient reports and individualized health recommendations based on clinical data, hence increasing communication between physicians and patients.

Furthermore, NLG contributes significantly to business intelligence and analytics by automatically creating textual summaries and insights from big datasets, allowing stakeholders to make educated decisions. NLG supports natural and engaging interactions with virtual assistants and chatbots by generating human-like responses to user questions, hence increasing user happiness and usability [29].

#### **1.9.3.2 Evolution of NLG techniques and algorithms**

The progress of NLG techniques and algorithms has been marked by a shift toward increasingly complex, data-driven approaches. Deep learning has recently transformed NLG, with models such as recurrent neural networks (RNNs) and transformers demonstrating cutting-edge performance in a variety of NLG applications [29].

Furthermore, integrating NLG with other AI technologies, such as natural language understanding (NLU) and dialogue management, has resulted in the creation of more interactive and context-aware NLG systems. These developments have made NLG applicable in a variety of sectors, including virtual assistants and chatbots, as well as automated

content generation and data analytics [29].

### 1.9.3.3 NLG in Healthcare

NLG is utilized in healthcare to generate medical reports, discharge summaries, and other documentation, lowering the workload for healthcare providers. Patient communication and education: NLG can be used to create patient-friendly explanations of medical diseases, treatments, and procedures, hence increasing patient comprehension and compliance. NLG can be used to provide tailored healthcare recommendations based on patient data and medical criteria, hence improving patient outcomes [29].

A Natural Language Generation (NLG) system called SemScribe was created by Dušek et al. with the purpose of creating medical reports, namely physician letters that provide an explanation of cardiological data [31].

## Natural Language Generation with TinyLLaMA

We used TinyLLaMA, a cutting-edge lightweight Large Language Model (LLM), to automatically generate medical explanation reports. Meta’s LLaMA architecture served as the basis for TinyLLaMA, a tiny, open-source language model. Compared to larger models, its compact and efficient design allows deployment on devices with limited processing capabilities without significantly compromising performance [32].

## How TinyLLaMA Works

TinyLLaMA uses a scaled-down and accelerated version of the LLaMA model to comprehend and produce text that is human-like. Despite its portability, it is capable of carrying out critical functions, such as producing precise and understandable medical explanations. It is made to function well on low-power devices while maintaining high-quality output. Its construction and text processing are described in the following sections. **Architecture and Tokenization:** TinyLLaMA is compatible with current LLaMA-based systems because it uses the same architecture and tokenizer as LLaMA 2. This design decision keeps the model size smaller while enabling TinyLLaMA to take use of the improvements and optimizations made for LLaMA 2 [32]. **Training Process:** Over three epochs, TinyLLaMA was pretrained on around 1 trillion tokens. The following were part of the training process [32]:

- **Pretraining Phase:** To build basic language comprehension, the model was first trained on a sizable corpus [32].
- **Continual Pretraining:** The model was further trained on domain-specific datasets to improve its performance in specific domains [32].
- **Cooldown Phase:** To increase model convergence and stabilize the training process, a cooling phase was introduced [32].

**Optimizations For Efficiency:** TinyLLaMA uses a number of optimizations to provide good performance with low computing requirements [32]:

- **FlashAttention:** This method lowers computational cost and memory consumption by speeding up the attention process [32].
- **Lit-GPT:** An efficient GPT implementation that helps with quicker training and inference times [32].
- **Grouped-Query Attention:** a technique that reduces the number of processes by grouping attention heads, improving efficiency without sacrificing performance [32].

TinyLLaMA was chosen for our web application’s NLG component because of its competitive performance and effective architecture. Due to its small size, which enables quicker inference and lowers processing overhead, it can be used to generate medical imaging reports in real time without compromising accuracy. TinyLLaMA’s effective natural language generation capabilities make it ideal for producing medical reports and explanations in AI-powered healthcare technologies, even if it hasn’t been used specifically in medical imaging literature yet.

## Conclusion

In terms of enhancing medical imaging, optimizing healthcare processes, and increasing patient care, AI has shown great promise. The accuracy and efficiency of image-based diagnostics have been greatly improved by artificial intelligence (AI), especially in the form of Convolutional Neural Networks (CNNs), which can process large and complicated medical data. CNNs have emerged as the mainstay of intelligent imaging systems, from radiology tumor identification to MRI and CT scan anomaly recognition. Moreover, developments in Explainable AI (XAI) and Natural Language Generation (NLG) have enabled transparent interpretation and communication of AI judgments in addition to high-performance

categorization. By bridging the gap between medical practitioners and black-box models, these technologies promote confidence and comprehension in AI-assisted diagnosis. Despite its achievements, AI in medical imaging still confronts several significant obstacles, such as system standardization, openness, and data protection. Better interpretability, consistent standards, and adherence to data regulations are necessary to ensure ethical and trustworthy adoption. By facilitating more precise diagnoses, individualized therapies, and better patient outcomes, the strategic application of AI technologies has the potential to revolutionize modern medicine.

# Chapter 2

## Web Application

### 2.1 Introduction

In the current digital age, where practically every part of life is interconnected with the Internet, web applications have become essential and potent instruments. Web applications have changed dramatically over time in terms of usability, performance, and capability. They might be anything from straightforward webpages with static content to intricate and dynamic platforms like cloud-based services, social networks, online banking, and project management applications. Web apps are unique in that they can be utilized from almost any device, including computers, tablets, and smartphones, provided that there is a web browser and an internet connection. As the need for quick, effective, and intuitive digital experiences keeps growing, web apps are becoming increasingly important in determining how we use technology. They affect many different areas, which encourages creativity and transforms the way we live and work.

### 2.2 Definition of Web Application

A web application is an interactive software created with web technologies (HTML, CSS, JS, etc.) that allows a team or individual to complete activities online by storing data (files, databases) and manipulating data through CRUD actions (Create, Read, Update, and Delete).

These tools are used by users in their web browsers. As everything is managed on the server side instead of the client side, we do not rely on specific operating systems or

programming languages to be installed on the user's device, as would be the case with local software or a mobile app.

Viewed from a higher level, a web app is a software application accessed via a web browser. This encompasses tools such as Figma, Canva, Hubspot, or Trello. It excludes standard websites, the majority of locally installed applications, and native mobile apps. Generally, web apps are cloud-based. However, there are several significant caveats to this. One option is self-hosted web applications that operate on local devices or on-premises infrastructure, yet are accessible via a browser [33].

## **2.3 Web Application vs Website**

Websites and web applications are similar in many basic ways, including being accessed through web browsers and often built using similar web technologies such as HTML, CSS, and JavaScript. Using a client-server architecture, both act as platforms for the online delivery of services and content, allowing users to access resources stored on distant servers. Their user interfaces and functionalities, however, differ greatly. Websites are mostly used to present people with text, graphics, and multimedia as primarily static or educational content. They emphasize the clean and appealing presentation of information, frequently requiring no user engagement other than navigation. Web applications, on the other hand, are made to offer dynamic and interactive experiences that let users carry out particular tasks like completing forms, organizing data, or gaining access to customized material. In order to provide secure access, this interaction necessitates more intricate programming, backend database integration, and frequently authorization and authentication procedures. Web apps are more like traditional software programs but can be accessed through a browser since they have to manage user inputs, business logic, and real-time updates. Furthermore, in order to support numerous users at once and safeguard private information, web applications need constant maintenance, scalability solutions, and strong security measures. Notwithstanding these distinctions, websites and web applications are both crucial elements of the contemporary digital environment, each fulfilling distinct functions—from straightforward information sharing to intricate user-driven features—and satisfying a range of user demands and corporate objectives.

Table 2.1 – Comparison between Websites and Web Applications [34].

Basis	Website	Web Application
1. Purpose	Companies use websites to share information and provide contact details. Users merely consume the content here, without engaging with it.	It is designed to address specific tasks via interactive functionalities and dynamic information access.
2. Development process	To develop a website, one must establish its architecture, design an appealing interface, and create content. Testing guarantees correct page displays and functionality of links/forms. The deployment process encompasses file uploads and domain configuration.	Creating a web application is more challenging, necessitating various user types, functions, and security measures. Deployment frequently entails server infrastructure, databases, and load balancers.
3. Authentication	Usually, users do not need to authenticate themselves to view content, but it is necessary for them to leave a comment or sign up for a newsletter.	As web applications deal with user data, they need authentication to block any unauthorized access.
4. Interactivity	The majority of websites feature static text and visual elements, with interactivity typically provided through a widget or a contact form.	Users engage with the content on the page to achieve the desired outcome.
5. Scalability	Websites need straightforward scaling options, like increasing server resources.	Advanced scaling solutions like automatic scaling or load balancing across multiple servers are necessary for web applications.

The table 2.1 offers a concise and organized comparison of websites and web apps in



a number of important areas, such as scalability, purpose, development process, authentication, and interactivity. It draws attention to the fact that websites are largely used for information sharing and provide simple interaction, frequently without needing user identification. Web apps, on the other hand, are made for specialized purposes and provide dynamic, interactive experiences as well as more complicated development and deployment procedures that call for user identification. In addition, web apps require sophisticated scalability solutions to accommodate numerous users and preserve speed, while webpages often have less complex scaling requirements. Overall, this comparison highlights how the two differ fundamentally in terms of utility and technical complexity.

## 2.4 Architectural Layers of Web Development

### 2.4.1 Front-End Development

Front-end development is an important part of web application development since it is the face of the application. Front-end development technology is the process of creating the interface of a web application. It primarily consists of three elements: hypertext markup language (HTML), cascade stylesheet (CSS), and JavaScript (JS). These technologies are also implemented in our project [35, 36]

#### HTML

Hypertext Markup Language is a standard markup language, not a programming language, as it is commonly used to develop static pages as it lacks dynamic functionality. It is a case-insensitive language that doesn't need a compiler. It is directly interpreted by the browser, which then shows the data that the user requested. HTML basically defines a webpage's structure. When combined with CSS and JavaScript, it can achieve styling, responsiveness, and interactive features [35].

#### CSS

Cascading Style Sheet is a language used to style the appearance of a website or web application. It enhances the quality and presentation of the content. CSS is a popular language for website styling. It is typically incorporated with HTML files internally (via style tags at the top of the page), externally, and via inline properties [35].

## JAVASCRIPT

It is a scripting language that adds interactivity to web pages beyond what standard HTML can deliver. This language can be used on both the client (front-end) and the server (back-end), with client-side development being the most common. JavaScript is not a programming language, although it employs conventions and syntax comparable to programming languages. As a client-side scripting language, it is a key component of the World Wide Web [35, 37]

### 2.4.2 Back-End Development

Back-end web development is the backbone of web applications, Providing the data and logic necessary for their functionality. It consists of three main components: the server, the application, and the database. They work together to handle requests, manage data, and guarantee that the front-end has the information it needs to display [38]. The backend consists of three parts :

- A server : is a device or computer application that handles requests and manages network resources.
- A database : is a collection of data.
- Back-End programming language or framework : developers can choose any programming language based on the sort of application to build.

#### 2.4.2.1 Technologies and Programming Languages

To create a dynamic website, we cannot rely exclusively on HTML, CSS, and Javascript; instead, we require a backend role (backend programming languages such as PHP, Python, Java, or C) [39] To develop the back-end of our project, we used :

#### **Hypertext Preprocessor (PHP)**

is a popular open-source general-purpose scripting language that is well-suited to web development and can be embedded in HTML. PHP is primarily focused on server-side scripting, therefore it can do any CGI function, such as collecting form data, generating dynamic page content, or sending and receiving cookies etc [40].

## Python

Python is a high-level, interpreted language that prioritizes code readability and clarity. It supports several programming paradigms, making it ideal for backend development, data science, and machine learning [41]. Python is an important tool for data processing and visualization [42].

In our project, we used Python to classify the medical images with Python and we used it also in the Explainable AI (XAI) approaches to show how the model makes decisions.

## Flask

Flask is a Python module and web framework that makes it simple to create web applications without having to worry about low-level issues like thread management and protocol. It is a microframework with a minimal and easily extensible core that lacks features like an object relational manager (ORM).

### 2.4.3 Full Stack Development

Full-stack development refers to the full process of creating web applications. It combines front-end and back-end development [43]. It involves operating on both the front-end and back-end of a program. This term is commonly used for persons working in the web [44]. Figure 2.1 shows the three main components of a web application: frontend

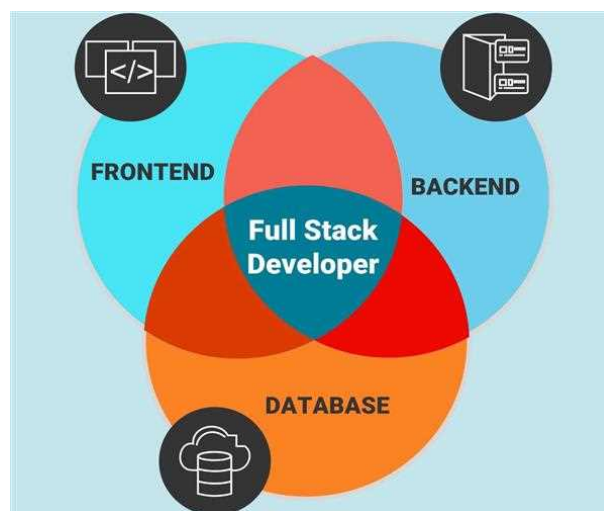


Figure 2.1 – Full Stack Development

end, back-end, and database. Together, they illustrate how these layers combine to form a full-stack system.

The major stacks of full-stack web development that are often used are: [\[44\]](#)

- Linux Apache MySQL PHP (LAMP)
- Cross-Platform Apache MariaDB PHP (XAMP)
- MongoDB Express Angular Node.js (MEAN)
- Windows Apache MySQL PHP (WAMP)
- Apache MySQL PHP PERL Softaculous (AMPPS)

These software stacks are collections of technology used in the development, hosting, and operation of web applications and websites. They offer the database, web server, operating system, and programming languages required to build and administer dynamic websites and apps.

## 2.4.4 Databases

Databases are an important aspect in building web applications. It's used to store and manage data making it easily accessible when needed. The type that is most commonly used is the Relational Database Management System (RDBMS), which divides data into tables. MySQL, PostgreSQL, SQLite, Oracle Database, and Microsoft SQL Server are some of the most frequently utilized relational databases. These systems are perfect for applications that demand structured data as well as the ability to perform complex queries utilizing organized Query Language.

NoSQL databases, on the other hand, non-relational and unstructured or semi-structured data. Common examples are MongoDB (document-based), Cassandra (wide column store), Redis (key-value store), and Neo4j (graph-based database) [\[45\]](#) .

### 2.4.4.1 The main differences between SQL and NoSQL

- SQL databases are relational, while NoSQL are non-relational [\[45\]](#) .
- SQL databases utilize a structured query language with a predetermined schema. NoSQL databases provide dynamic schemas for unstructured data [\[45\]](#) .
- SQL databases are table-based, whereas NoSQL databases are document, key-value, graph, or broad column stores [\[45\]](#) .

## 2.5 Client Server Architecture

With the developments in technology, the web is becoming more and more important in our daily lives, with almost everything we do now involving the use of the web. Furthermore, the application of Web is not confined to computers, but it is accessible to a variety of intelligent electronics. To make possible this widespread accessibility and interactivity, most web applications adopt a client-server architecture, which is a key model in web development [46, 47].

### 2.5.1 Definition of Client-Server Architecture

According to the client-server architecture, a client computer uses a network to transmit a resource request to a server [48]. The server receives this request, processes it, and then responds appropriately. A single server can receive service requests from several clients at once. The server usually keeps a database that holds information and performs the operations required to handle and complete these client requests [49].

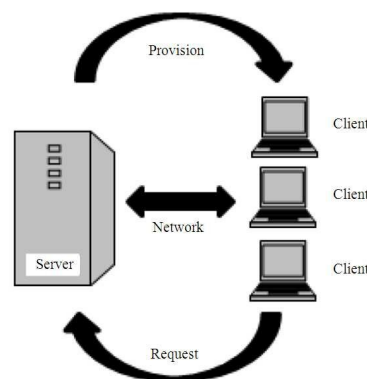


Figure 2.2 – Client-server architecture

Figure 2.2 illustrates the client-server architecture, showing how clients send requests to a server and how the server processes and responds to these requests

## 2.5.2 Types Of Client-Server Architectures

There are several client-server architectures that depend upon the number of servers participating in the implementation. Some common architectures include :

### Two-Tier Architecture

This approach has only two tiers of computing devices: client and server. Workload is distributed across these systems, with the client hosting user interfaces and sending requests to the server, and the server hosting the system that processes user requests [50] .

### Three-Tier Architecture

This architecture presents an additional tier to the two-tier architecture to help overcome challenges with execution. In this approach, the business logic, such as information access, data storage, and user interfaces, are all housed differently. The additional tier houses the application system, while the other houses the database. This indicates that the application system is hosted on a different server than the database or storage system [50] .

### N-Tier Architecture

Also known as multi-tier architecture, this concept separates presentation, processing, and data management logic both logically and physically. In this architecture, numerous servers work together to process a request. The architecture is divided into clearly defined levels, with each one which performs a specific functionality. N-tier architecture is more scalable than other architectures and contributes to improved data integrity. This architecture makes it easy to create reusable components while additionally enhancing security. Though this architecture has plenty of benefits, it can be challenging to create applications in this architecture since programming, deployment, and maintenance require a significant amount of effort [50] .

Table 2.2 – Comparison between 1-tier, 2-tier, 3-tier, and N-tier Architectures.

Architecture	Description	Advantages and Disadvantages
1-Tier	There is one system for all the components—data, logic, pre-sentation.	Simple, inexpensive, and easy to get set up. However, scalability is restricted and performance may suffer as the system expands.
2-Tier	The client-server model separates the presentation layer from the data layer.	Provides better separation of concerns and reasonable scalability. However, flexibility remains limited, and performance may suffer as user or data volume increases.
3-Tier	Separates the system into three layers: display, logic, and data.	improves scalability, security, and maintainability. However, it is more difficult to set up, and performance overhead may occur.
N-Tier	An extension of three-tier architecture with several distributed layers over multiple servers.	Scalable, flexible, and adaptive to complicated systems. However, it is more complex and expensive to design and maintain, necessitating close coordination among the layers.

## Conclusion

In this chapter, we explored the fundamental concepts that govern modern web application development. Beginning with the architectural layers such as front-end, back-end, and full-stack. We investigated how they work together to build robust and scalable applications. Then We looked at different sorts of web application architectures, demonstrating the various ways developers can structure their apps. Any developer or engineer working on the design and development of web applications will benefit greatly from having an adequate knowledge of these architectural concepts as web technologies advance. When combined, these fundamental ideas offer an in-depth understanding of the architecture and security of contemporary web applications, giving developers the skills they need to create efficient and safe systems.



# Chapter 3

## Conception

### 3.1 Introduction

To ensure efficient organization and a clear specification of the activities to be completed, a strict approach is necessary when developing a web application. The design phase of any software project is essential, it must be completed precisely since it serves as an accurate representation of the system even before it is put into use. This chapter introduces the design of our web application using UML (Unified Modeling Language), a popular tool for effectively and clearly modeling a system's many components. The goal is to convert functional and technical requirements into organized, understandable diagrams. Using Modelio, we create sequence diagrams, use case diagrams, and class diagrams that give a comprehensive picture of the system.

### 3.2 Principal Actors

An actor in UML modeling is any external entity (user or system) that communicates with the application. Every actor has distinct goals they hope to accomplish through the system [51].

#### **Admin**

The web application administrator is responsible for keeping an eye on the platform statistics, including the number of registered doctors, new users, and overall user activity. The administrator has direct access to the database and can examine and manage user,

doctor, and patient accounts if necessary, even though they do not manage users via the application interface.

## **User**

The user is a key actor in the system, mainly engaged with the platform. An AI-generated diagnosis, along with a comprehensive report and an explanation of the results, can be obtained by the user by uploading images of X-rays, MRIs, or other medical scans. Beyond uploading images for diagnosis, the user can book appointments and, if needed, emergency service requests from ambulance drivers. Also, the user can offer an ambulance service.

## **Doctor**

Another key actor in the system, the doctor, benefits from an integrated decision support system that assists in evaluating medical images submitted by patients. This technology enables the doctor to make well-informed clinical decisions using AI-generated diagnoses and detailed feedback. Additionally, the doctor has access to and control over appointments booked by patients through the platform. The doctor can also offer an ambulance directly through the platform, enhancing the overall care and accessibility provided.

## **3.3 Functional and Non-Functional Requirements**

### **3.3.1 Functional requirements**

Functional requirements are product features or functions that developers must include to allow users to complete their tasks. As a result, they must be clearly communicated to both the development team and the stakeholders. In general, functional requirements specify how a system will behave under specified scenarios [52].

### **User Authentication**

Users, doctors, and administrators must all be able to safely register and log in using the system. Only features pertinent to their roles should be accessible to each sort of user.

**User Functionality**

- Upload medical images (e.g., X-rays, MRIs).
- View AI-generated diagnostic reports and explanatory feedback.
- Book appointments with doctors.
- Request emergency ambulance service.
- If registered as a service provider, offer ambulance services through the platform.

**Doctor Functionality**

- Upload medical images (e.g., X-rays, MRIs).
- View AI-generated diagnostic reports and explanatory feedback.
- View and manage patient-booked appointments.
- Offer an ambulance service if available.

**Ambulance Operator Functionality**

- Receive emergency alerts from patients with their information.

**Administrator Functionality**

- Manage user accounts (update, or delete patients, doctors, and ambulance operators).
- Verify and approve doctor credentials.
- Access and manage patient inquiries submitted through the contact form.

**3.3.2 Non Functional requirements**

NFRs, or non-functional requirements, are a collection of specifications that outline the limitations and operational capabilities of the system. These are essentially the specifications that define how well it functions [53].

**Performance**

The system must respond quickly to the users actions [53].

## Scalability

A rising user base and several concurrent requests must be supported by the system without causing performance issues [53].

## Usability

Users with basic computer abilities should be able to easily navigate and utilize the interface [53].

## Compatibility

The program must be adaptable and compatible with common web browsers like Chrome, Firefox, and Edge [53].

# 3.4 Unified Modeling Language (UML)

the Unified Modeling Language (UML) is a standardized modeling language is used to define, illustrate, and document models of software systems, including their design and structure. Software system standards, construction, and documentation are made easier by UML, which offers a collection of graphical notation tools for building abstract models of particular systems [54].

## 3.4.1 Use Case Diagram

Often called behavior diagrams, use case diagrams are used to illustrate a series of actions (use cases) that a system or systems should or are capable of acting in cooperation with one or more external users of the system (actors). Every use case should to yield a tangible useful outcome for the system's actors or other stakeholders [55].

### 3.4.1.1 Use Case Diagram of the Web Application

The web application's primary features are depicted in this use case diagram, which also demonstrates how different actors—including administrators, physicians, and users—interact with the system. Determining user demands, understanding the application's scope, and guiding the development process are all made easier with the help of this representation.

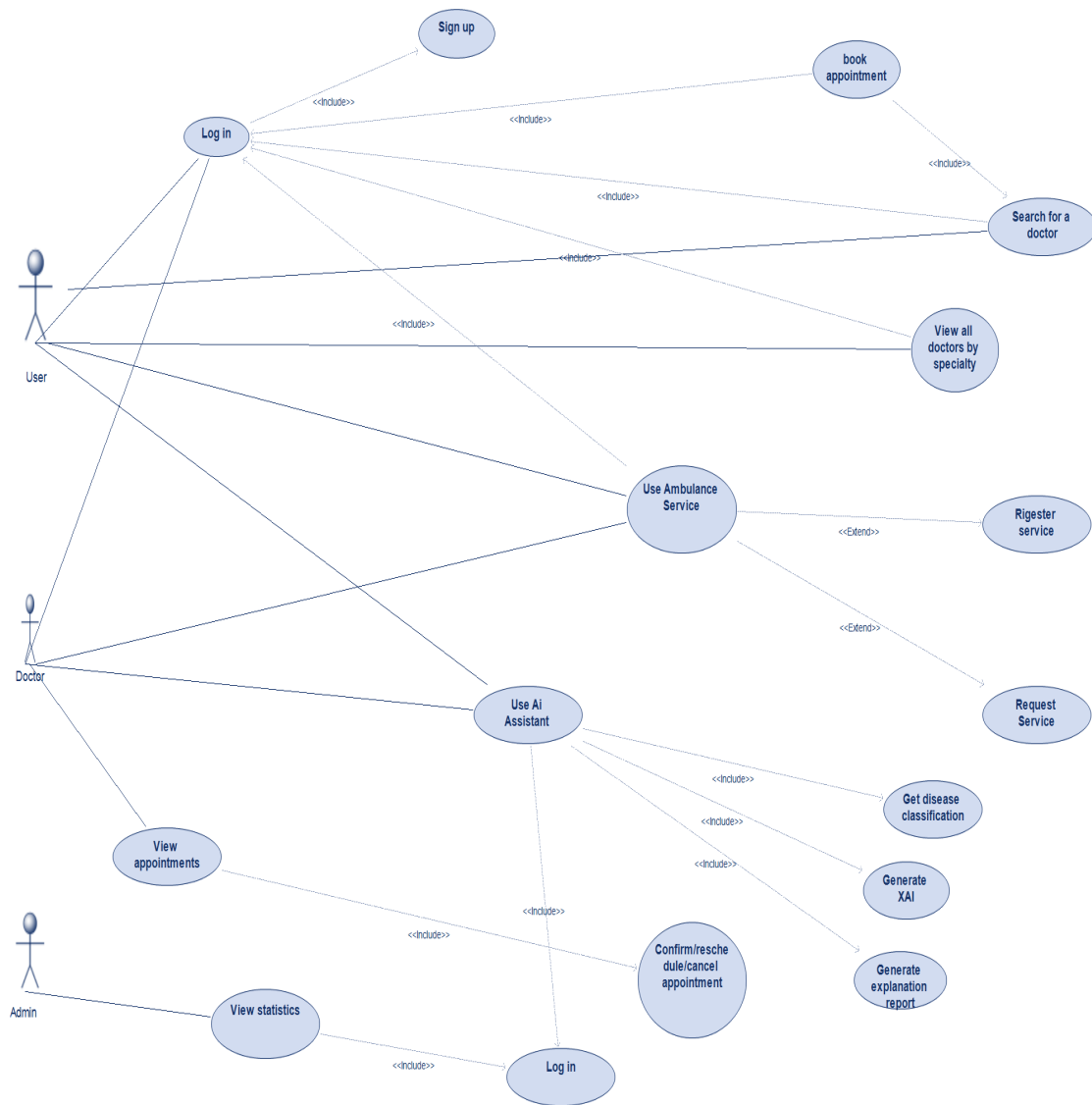


Figure 3.1 – Use Case Diagram of the Web Application

The main characteristics of the healthcare system are illustrated in Figure 3.1 through the interactions between administrators, doctors, and users. It demonstrates how important features—like signing in, making appointments, utilizing AI assistance, and managing emergencies—are related via include and extend relationships.

### 3.4.2 Class Diagram

The most widely used UML diagrams for building software applications are class diagrams. A class diagram is a type of static diagram that is used to represent a system's static perspective. They are used to create executable code for software applications in addition to visualizing, characterizing, and describing various system components [56].

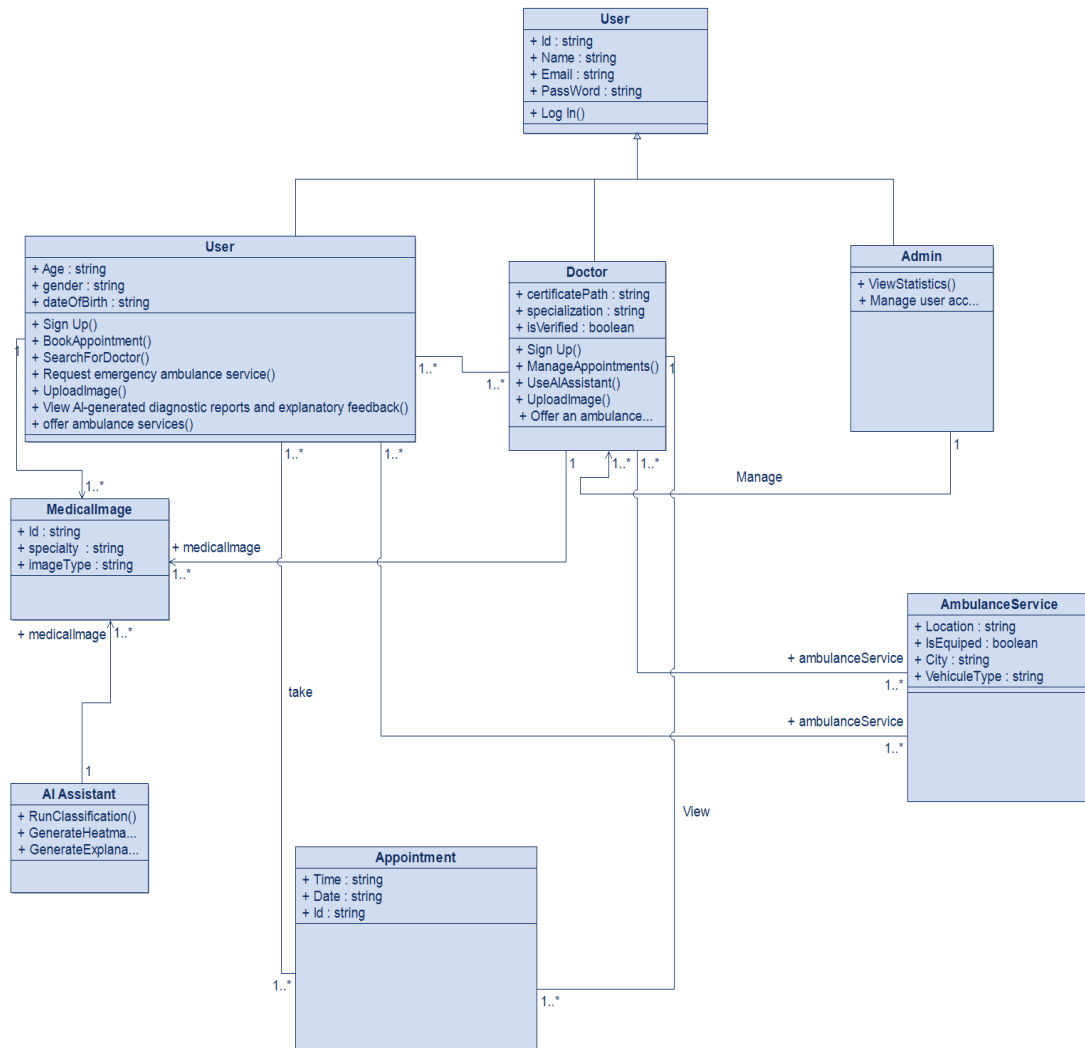


Figure 3.2 – Class Diagram of the Web Application

The healthcare system's class structure is depicted in Figure 3.2, which also shows the connections between important entities including users, doctors, administrators, ambulance services, and medical pictures. It shows interactions like scheduling appointments, uploading images, utilizing AI tools, and monitoring statistics, and it specifies the fundamental properties and methods for every class. The connections between the system's components and the data flow between them are made clear by associations and multiplicities.

### 3.4.3 Sequence Diagram

The UML sequence diagram is the second most common UML diagram that represents how objects interact and exchange messages over time. Sequence diagrams show how events or activities in a use case are mapped into operations of object classes in the class diagram [57].

#### Log-In Sequence Diagram

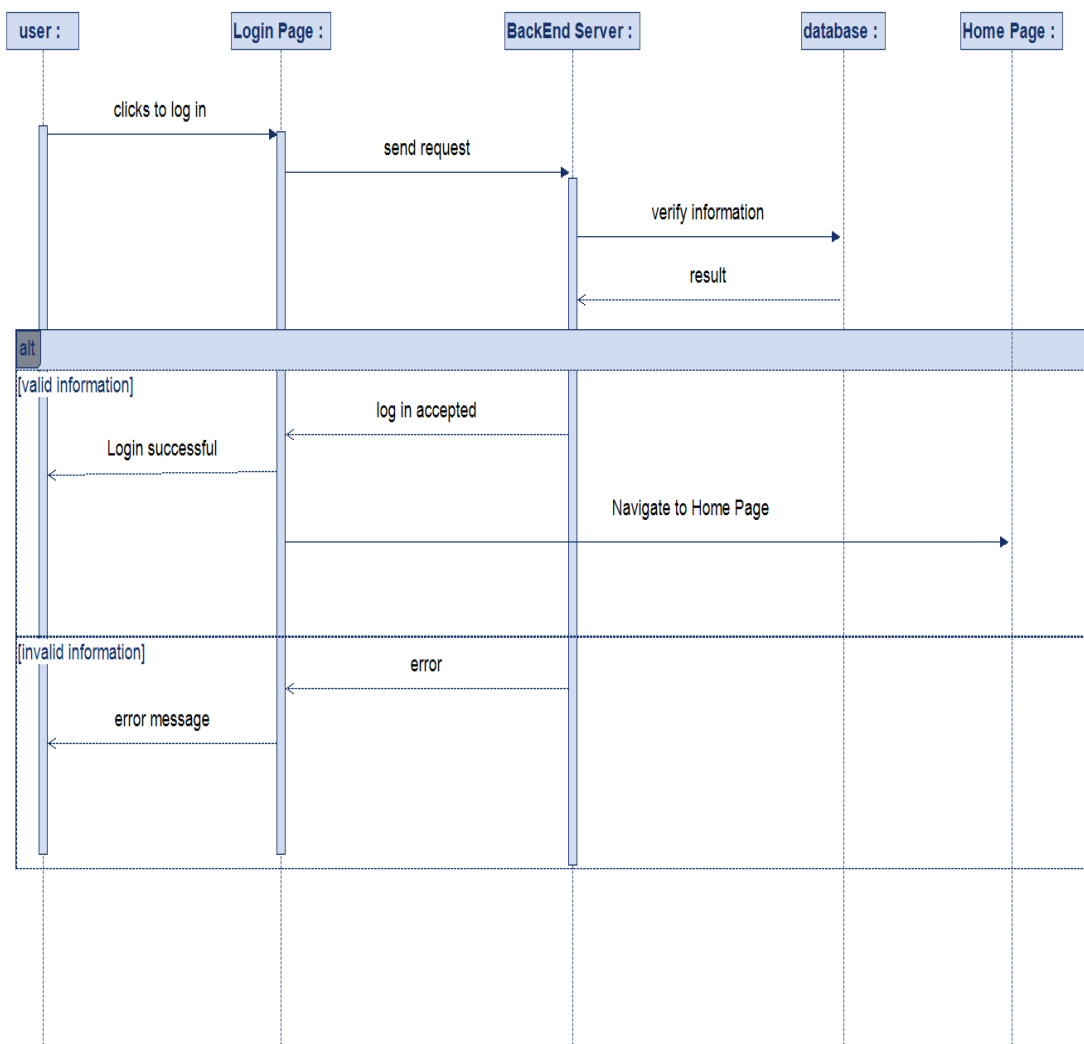


Figure 3.3 – Log-In Sequence Diagram of the Web Application

The login process is depicted in Figure 3.3, where the actor inputs credentials via the login interface. The back-end service receives these credentials and compares them with the user database. Access is allowed if it is valid; if not, an error notice is displayed.

## Ai Assistant Sequence Diagram

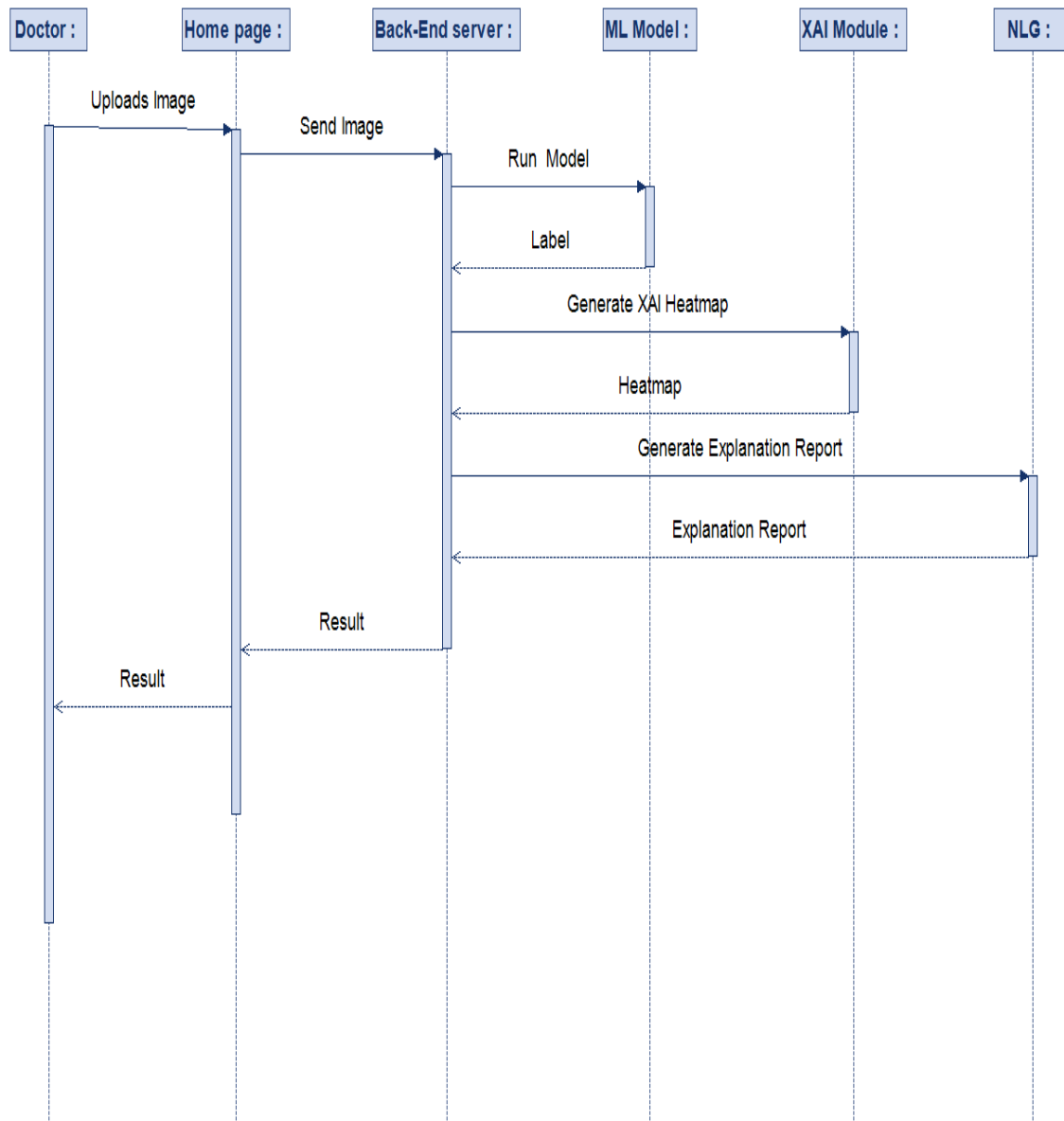


Figure 3.4 – Ai Assistant Sequence Diagram of the Web Application

The flow of communication between the user, the AI assistant interface, and the backend services is depicted in Figure 3.4. The AI assistant receives a medical image from the user and processes it. The assistant evaluates the request, creates a response, and sends it back to the actor. The main processes in the AI assistant’s workflow from taking input to producing intelligent output—are shown in this diagram.



### Doctor's Appointment List Viewing Sequence Diagram

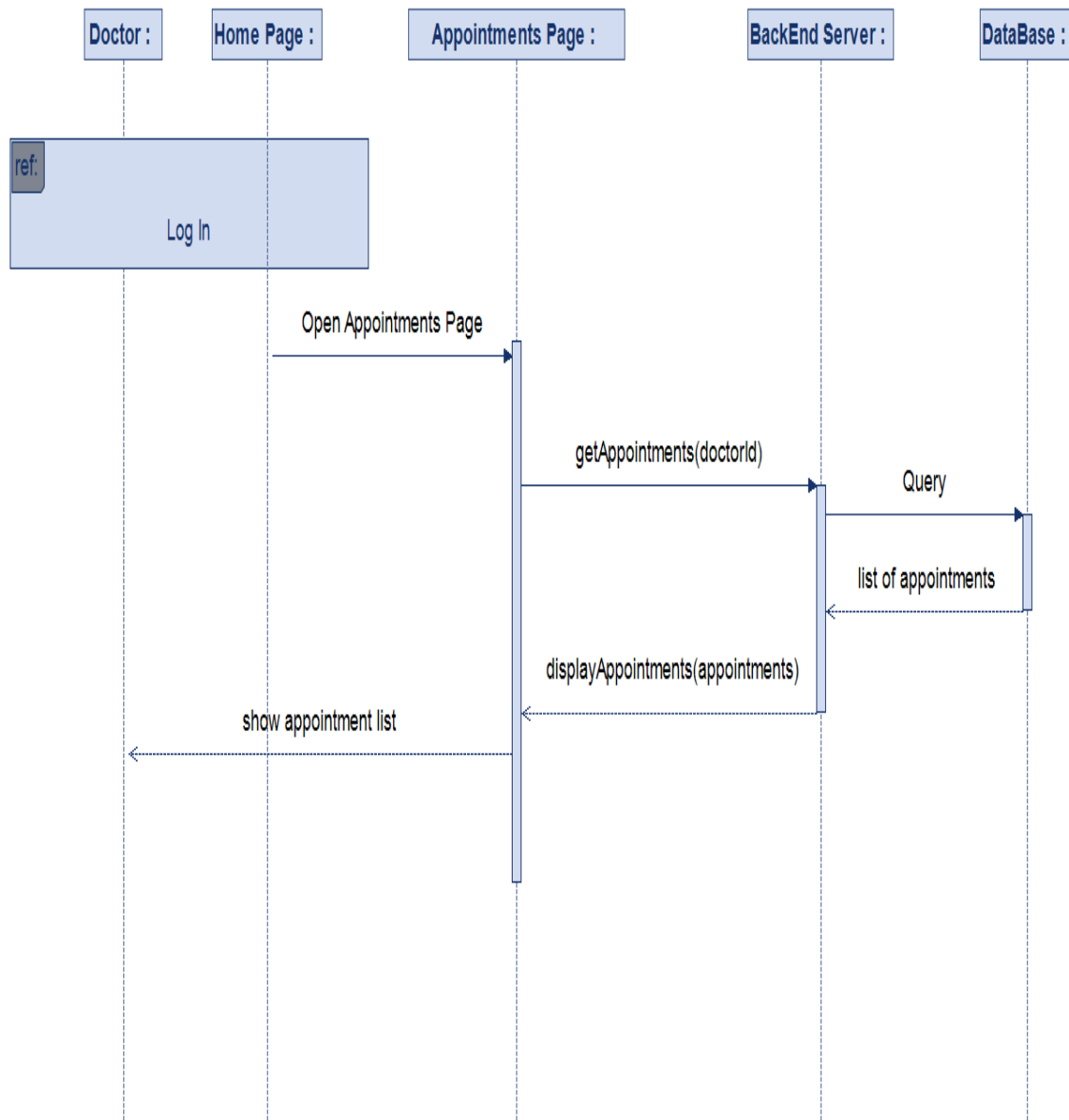


Figure 3.5 – Doctor's Appointment List Viewing Sequence Diagram of the Web Application

The sequence diagram for viewing a doctor's appointment list is displayed in Figure 3.5. The system retrieves the appointment data from the database and sends it back to the interface, which then shows the list to the actor when the doctor requests to view appointments through the interface. This sequence demonstrates how to retrieve and display appointment data.

## User Appointment Booking Sequence Diagram

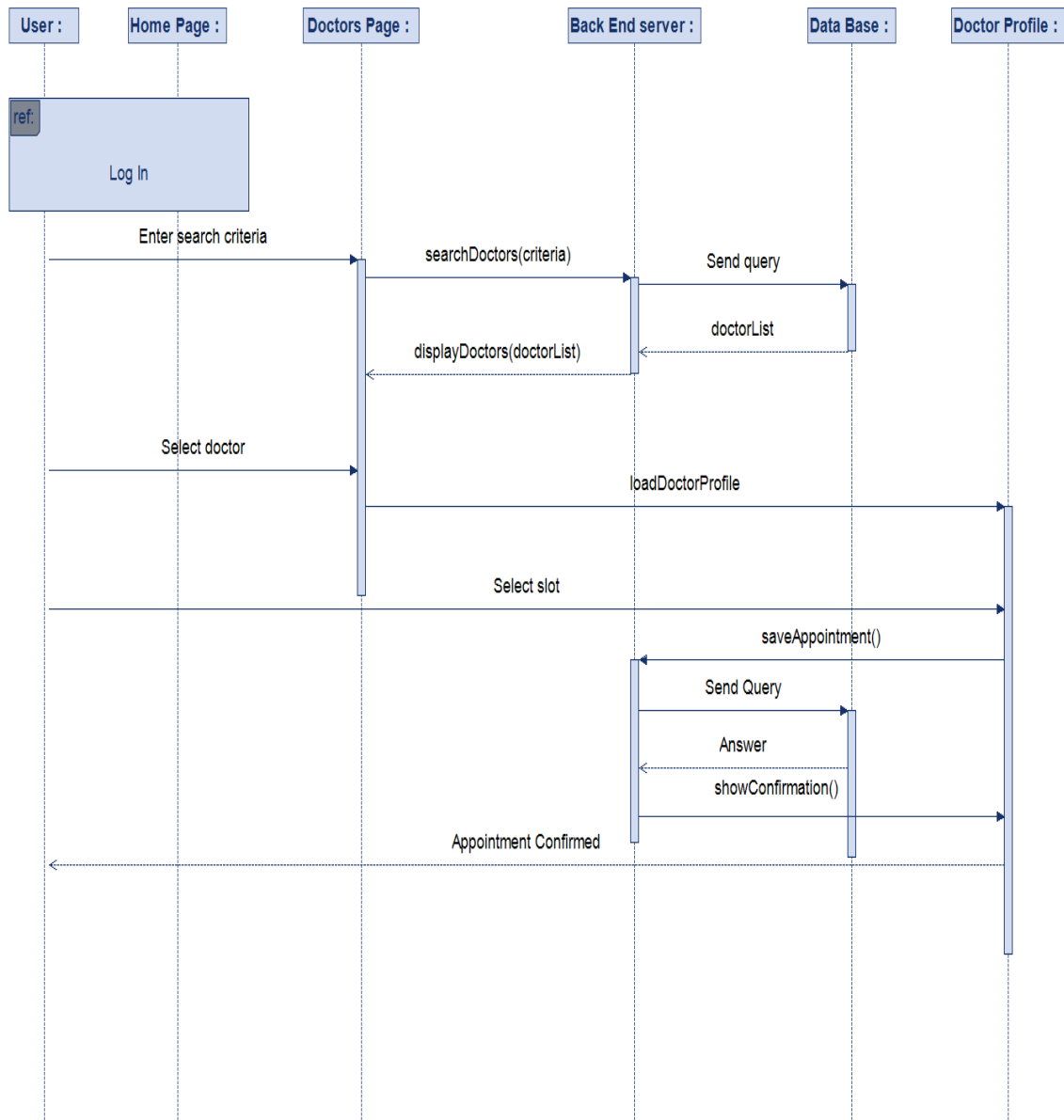


Figure 3.6 – User Appointment Booking Sequence Diagram of the Web Application

The sequence diagram for a user to make an appointment is displayed in Figure 3.6. Through the interface, the user enters the appointment information for a time slot that is open. After processing the request, the system verifies the reservation. The main procedures for setting up and confirming an appointment are described in this sequence.

## Patient Searches for An Ambulance Sequence Diagram

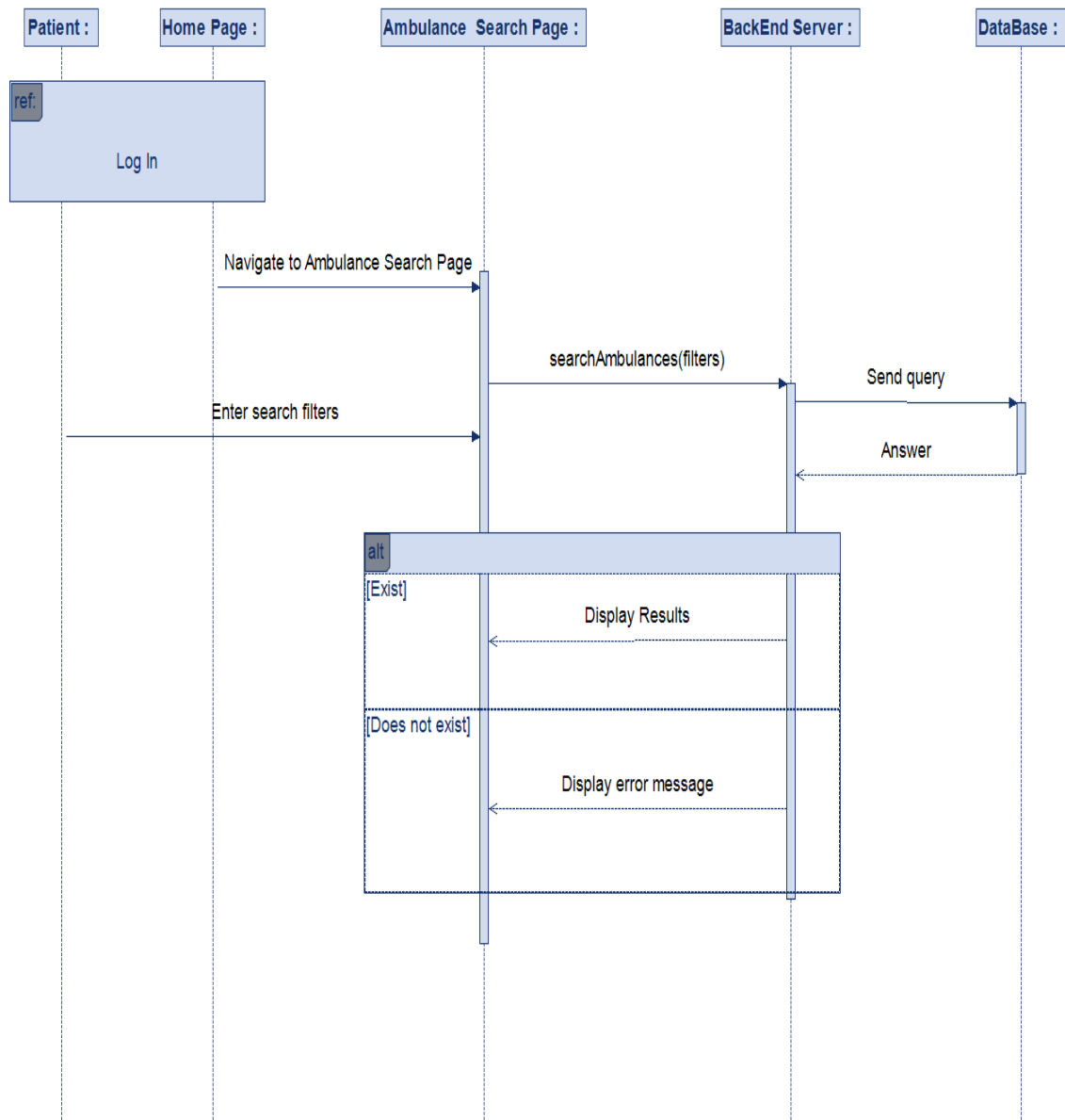


Figure 3.7 – Patient Searches for An Ambulance Sequence Diagram of the Web Application

The sequence diagram for a patient looking for an ambulance can be found in Figure 3.7. The patient uses the interface to submit a request, utilizing geographical filters. After that, the system looks up available ambulances, makes a selection, and returns the outcome to the interface. This scene demonstrates how to find and display available ambulances.

## Conclusion

The conceptual design of our web application has been presented in this chapter. The foundation of our system's specifications is comprised of both functional and non-functional needs, which we first defined along with the main actors. Using UML as our modeling language, we displayed a number of diagrams that show the structure and behavior of the application, such as use case, class, and sequence diagrams. The way the system should function and communicate with its users is clearly and coherently envisioned by these models. The conceptualization phase is essential since it converts preliminary concepts and requirements into a codified and structured plan. The implementation phase will be guided by this foundation, which will ensure that development is in line with system goals and user expectations.

# Chapter 4

## Implementation and GUI of The Web Application

### 4.1 Introduction

This chapter begins by describing the development environment in which the program was created, including both software and hardware components. We describe the development tools and platforms used, as well as the requirements for the computers and devices used during the implementation phase. Following that, the chapter discusses the graphical user interface (GUI), including the layout structure and operation of each piece of interface. Screenshots are supplied to demonstrate the developed features and give a visual picture of the user experience. This section will provide a complete explanation of how the program was created and how users interact with its many components. This section describes the work environment that was utilized for the duration of the project, including the software and material environments that aided in development.

#### 4.1.1 The Material Environment:

In this project, we used the hardware configuration shown in [Figure 4.1](#):

Table 4.1 – Hardware environment used during development

Unit	Characteristics
Processor	AMD Ryzen 5 5625U with Radeon Graphics, 2.30 GHz
Installed RAM	16.0 GB (15.3 GB usable)
System Type	64-bit Operating System, x64-based processor
Touch and Pen Support	Support for pen and touch with 10 touch points
Screen	15.6 inch full HD display
Hard Disk	512 GB SSD

### 4.1.2 The Software Environment :

The software environment used to develop the project is described in this section. The main code editor for front-end and back-end implementation was Visual Studio Code. A cloud-based platform for machine learning model testing and training was made available by Google Colab. The web application employing PHP and MySQL was also tested and deployed well thanks to the use of XAMPP to mimic a local server environment. When used in tandem, these techniques guaranteed a productive and well-structured development process.

#### Visual Studio Code

VS Code is a source-code editor made by Microsoft for Windows, linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. It is a source-code editor that can be used with a variety of programming languages [58].

#### Google Colab

Colab is a hosted Jupyter notebook service that does not require setup and provides free access to computing resources, including GPUs and TPUs. It is particularly well-suited for machine learning, data science, and education. We used it for the development of our classification model, the implementation of explainable AI (XAI) techniques, and to generate reports using the LaMa tool [59].

## XAMPP

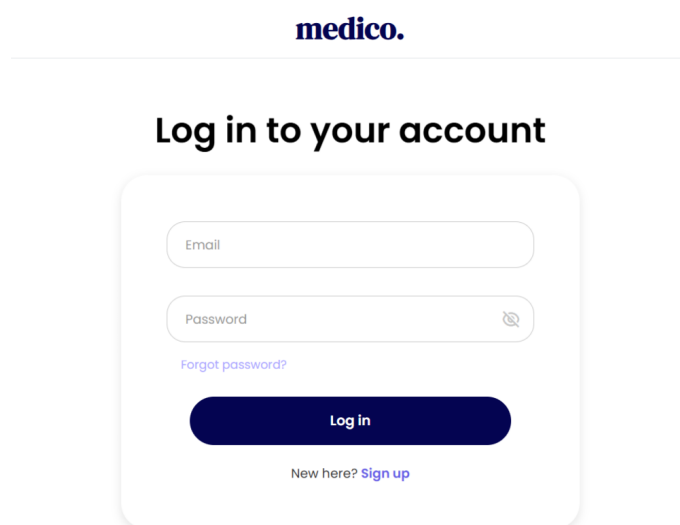
XAMPP is an open-source web development environment that allows programmers to create dynamic websites and apps without paying an upfront license fee. This bundle contains everything you need for web, database, and PHP programming, including the Apache web server, MySQL database, PHP programming language, and a slew of other tools and utilities [60].

## 4.2 Graphical User Interface (GUI)

In this section, we will be displaying a series of screenshots from our application to demonstrate the graphical user interface (GUI) of its various components. These images provide a graphical representation of the user experience while highlighting the interface's design and structure. Each screenshot is accompanied by a brief explanation of the functionality it represents, helping the reader understand how users interact with the system.

### 4.2.1 Login Interface

A straightforward form for user authentication is provided via the login interface, as seen in Figure 4.1. It requires an email address and password, it helps protect system integrity by limiting access to only confirmed users.



The screenshot displays the login interface for 'medico.'. At the top, the brand name 'medico.' is shown in a dark blue font. Below it, the heading 'Log in to your account' is centered in a bold, black font. The login form is contained within a white rounded rectangle with a subtle drop shadow. It features two input fields: 'Email' and 'Password'. The 'Password' field includes a toggle icon for visibility. A link for 'Forgot password?' is positioned below the password field. A prominent dark blue 'Log in' button is centered at the bottom of the form. Below the button, a link for 'New here? Sign up' is displayed.

Figure 4.1 – Login Interface.

## 4.2.2 Sign-Up Interface

New users may create an account by entering personal information through the sign-up window as shown in Figure 4.2. This enables users to safely and individually use the platform's services. Users usually enter their name, email address, password, and other details like their phone number and birthdate, etc., while registering. Their informations are safely kept and utilized to verify users when they log in, allowing for experiences that are customized according to their profile.

The figure displays four wireframes of a 'Create account' sign-up interface, arranged in a 2x2 grid. Each wireframe represents a step in a 4-step process: Contact (1), Name (2), Birth (3), and Submit (4). The steps are indicated by a horizontal line with numbered circles above the input fields.

- Top-left wireframe:** Shows the 'Contact' step (1) with input fields for 'Email', 'Password', and 'Confirm Password'. A 'Next' button is at the bottom.
- Top-right wireframe:** Shows the 'Name' step (2) with input fields for 'First name', 'Last Name', and 'Phone Number'. 'Back' and 'Next' buttons are at the bottom.
- Bottom-left wireframe:** Shows the 'Birth' step (3) with input fields for 'City', 'Gender' (with 'Male' selected), and 'Date of birth' (with 'DD', 'MM', and 'YYYY' sub-fields). 'Back' and 'Next' buttons are at the bottom.
- Bottom-right wireframe:** Shows the 'Submit' step (4) with input fields for 'Speciality' (with 'Médecin généraliste' selected), 'Clinic', and 'Clinic Address'. 'Back' and 'Submit' buttons are at the bottom.

Each wireframe has a 'Get Started' link in the top left and a 'Help' link in the top right.

Figure 4.2 – Sign-Up Interface.



### 4.2.3 Landing page Interface

This first interface presents the platform as shown in Figure 4.3. Its simple and easy-to-use design guarantees a friendly and convenient beginning to the user experience.



Figure 4.3 – Landing Page Interface.

#### 4.2.4 Doctor Profile Settings Interface

As illustrated in Figure 4.6, doctors can access and manage their personal information through this interface. They can edit their contact details, specialization, and other profile data, allowing them to maintain an accurate and professional presence on the platform that remains reliable for patients and colleagues.

The screenshot shows the 'medico.' app interface for a doctor's profile settings. The top navigation bar includes 'Home', 'MyAI', 'Activity', 'Appointment', and 'Me'. The main content area features a sidebar with options: 'General' (selected), 'Information', 'Social links', 'Notifications', and 'Change password'. The 'General' section displays the doctor's profile for 'Dr. Sarah Ferkaoui' with an email 'SarahFerkaoui@example.com'. Below the profile, there are input fields for 'First Name' (Sarah), 'Last Name' (Ferkaoui), 'E-mail' (SarahFerkaoui@example.com), and 'Username' (Sarah). A message states 'Your email is not confirmed. Please check your inbox.' with a 'Resend confirmation' link. At the bottom right, there are 'Update' and 'Cancel' buttons.

Figure 4.4 – Doctor Profile Settings Interface.

#### 4.2.5 Doctor Notification Interface

As shown in Figure 4.7, doctors can customize the types of alerts they wish to receive, such as notifications for rescheduled appointments. This feature enhances their ability to stay informed and manage their schedules efficiently.

The screenshot shows the 'medico.' app interface for a doctor's notification settings. The top navigation bar includes 'Home', 'MyAI', 'Activity', 'Appointment', and 'Me'. The main content area features a sidebar with options: 'General', 'Information', 'Department', 'Specializations', 'Notifications' (selected), and 'Change password'. The 'Notifications' section is divided into 'Activity' and 'Application' categories. Under 'Activity', there are three notification options: 'Notify me when I receive an appointment request.' (checked), 'Notify me when an ambulance is requested for my patient.' (checked), and 'Notify me before my subscription ends.' (unchecked). Under 'Application', there are three notification options: 'News and announcement.' (checked), 'Monthly health report summary.' (unchecked), and 'health news.' (checked). At the bottom right, there are 'Update' and 'Cancel' buttons.

Figure 4.5 – Doctor Notification Interface.

### 4.2.6 User Profile Settings Interface

Through a specific interface, users can view and manage their personal information, as shown in Figure 4.6. With the help of this feature, they may keep their profiles current and accurate, maintaining an official appearance on the platform.

The screenshot displays the 'medico.' app's user profile settings. On the left, a sidebar lists options: General (selected), Information, Social links, Notifications, and Change password. The main area shows the user's profile for 'Salma Achour' with an email 'salmaachour@gmail.com'. Below the profile, there are input fields for 'First Name' (Salma), 'Last Name' (Achour), 'E-mail' (salmaachour@gmail.com), and 'Username' (Salma). A message indicates the email is not confirmed, with a 'Resend confirmation' link. At the bottom right, there are 'Update' and 'Cancel' buttons.

Figure 4.6 – User Profile Settings Interface.

### 4.2.7 User Notification Interface

Users can choose the kinds of alerts they want to receive, like updates on rescheduled appointments, to customize their notification options, as seen in Figure 4.7. This feature guarantees that they remain informed based on their own requirements.

The screenshot shows the 'medico.' app's notification settings. The left sidebar has 'General', 'Information', 'Social links', 'Notifications' (selected), and 'Change password'. The main area is divided into 'Activity' and 'Application' sections. Under 'Activity', there are three notification options: 'Notify me when an appointment is booked.' (checked), 'Notify me when an ambulance is booked.' (checked), and 'Notify me when a new doctor joins the app.' (unchecked). Under 'Application', there are three options: 'News and announcement.' (checked), 'Monthly health report summary.' (unchecked), and 'Health news.' (checked). At the bottom right, there are 'Update' and 'Cancel' buttons.

Figure 4.7 – User Notification Interface.

## 4.2.8 Appointment Scheduling Interface

As shown in Figure 4.8, doctors can specify their using the appointment's duration using the setting interface. This helps doctors effectively manage their schedules.

The interface is for a doctor named Salma Yahia, Gastro-entérolog. The left sidebar contains navigation links: Calender, Appointments, Analytics, and Payment. The main content area is divided into several sections:

- Working Hours Range:** A slider showing a range from 08:00 AM to 05:00 PM.
- Appointment Duration:** Radio buttons for 10 min, 15 min (selected), 30 min, 1 hr, and 2 hrs.
- Break Time:** A toggle switch for "Include Lunch Break" which is turned on. Below it, a time range from 12:00 PM to 12:15 PM is shown.
- Buffer Time Between Appointments:** A dropdown menu set to 5 mins.
- Working Days Calendar:** A calendar for June 2025. Working days (Mon-Fri) are highlighted in dark blue, and day-offs (Sun, Sat) are in light blue. A "Save Schedule" button is at the bottom right.
- Schedule Summary:** A list of scheduled days with their respective times and settings.
 

Day	Time Range	Settings
Monday, June 2	09:00 AM - 05:00 PM	A/D: 15 min Buffer: 10 mins Break: 12:00 PM - 01:00 PM
Tuesday, June 3	08:00 AM - 04:00 PM	A/D: 30 min Buffer: 5 mins Break: 11:00 AM - 12:00 PM
Wednesday, June 4	10:00 AM - 05:00 PM	A/D: 15 min Buffer: No Buffer Break: 01:00 PM - 02:00 PM

Figure 4.8 – Appointment Scheduling Interface.

### 4.2.9 Doctor Appointment List Interface

Doctors can examine their booked consultations together with important information such as patient names, age, and appointment time, as demonstrated in Figure 4.17. This arrangement guarantees effective patient management and simplifies daily planning.

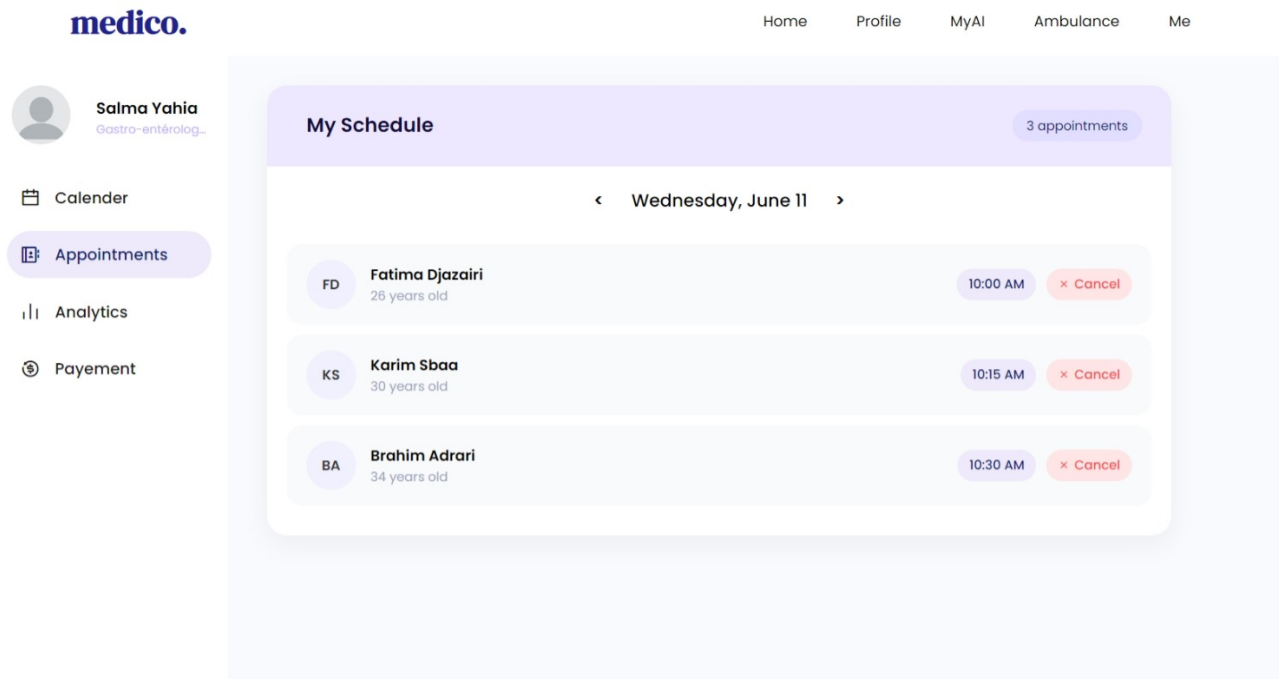


Figure 4.9 – Doctor Appointment List Interface.

### 4.2.10 Doctor Analytics Interface

The doctor analytics interface displays important indicators that doctors can monitor, like the number of appointments, new patients, and overall activity as shown in 4.10. This function helps with data-driven decision-making and gives a clear picture of their performance.

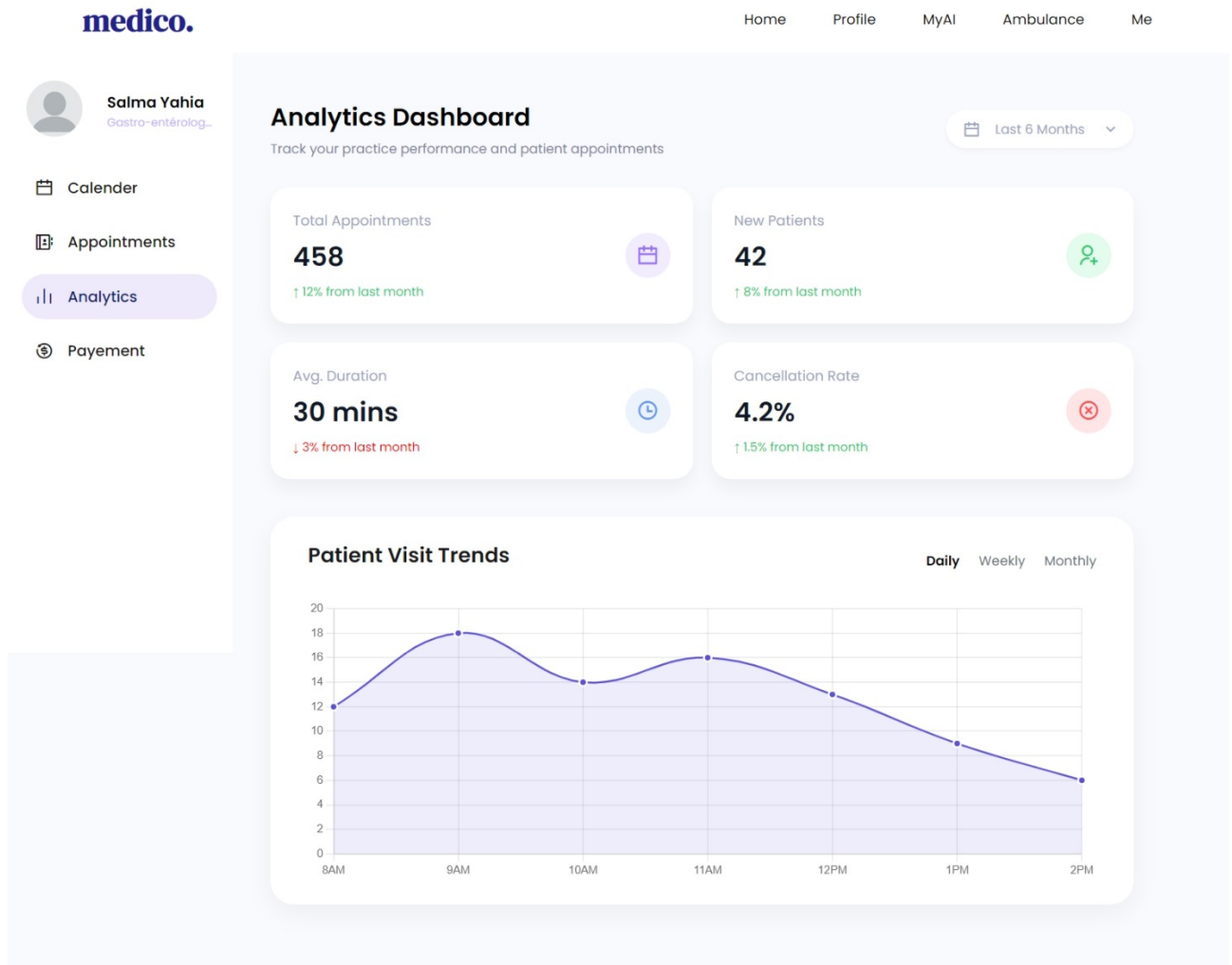


Figure 4.10 – Doctor Analytics Interface.

#### 4.2.11 Doctor Payment Interface

Doctors can manage their subscriptions and pay to unlock premium features or access other platform functionality , as demonstrated in Figure 4.11.

**medico.**

Home Profile MyAI Ambulance Me

**Salma Yahia**  
Gastro-entérolog...

Calendar

Appointments

Analytics

**Payment**

### Subscription Details

**Professional Plan** Current  
Monthly billing

- ✓ Full analytics dashboard
- ✓ Unlimited appointments
- ✓ Advanced reporting

Subscription	500.00DA
Tax	19%
<b>Total</b>	<b>1450.00DA</b>

Need help? [medicoPlus@doctordash.com](mailto:medicoPlus@doctordash.com)

### Payment Method

GOLD Algerian Gold Card

**Cardholder Name**  
As shown on card

**Card Number**  
XXXX XXXX XXXX XXXX

**Expiry Date**  
MM/YY

**CVV**  
XXX

**Billing Address**

**Address**

**Phone Number**

**Postal Code**  
XXXXX

**City**  
Select City

☐ I agree to the Terms of Service and Privacy Policy

**Important Notice**  
If payment is not received by the end of your current subscription period, your dashboard access will be automatically deactivated.

**Save Payment Info**

Figure 4.11 – Doctor Payment Interface.

### 4.2.12 User Home Page Interface

The following interface gives users quick access to necessary services including booking appointments, accessing Ai Assistant, and ambulance service , as demonstrated in Figure 4.12. Users can navigate and manage their healthcare needs with ease thanks to its user-friendly design.

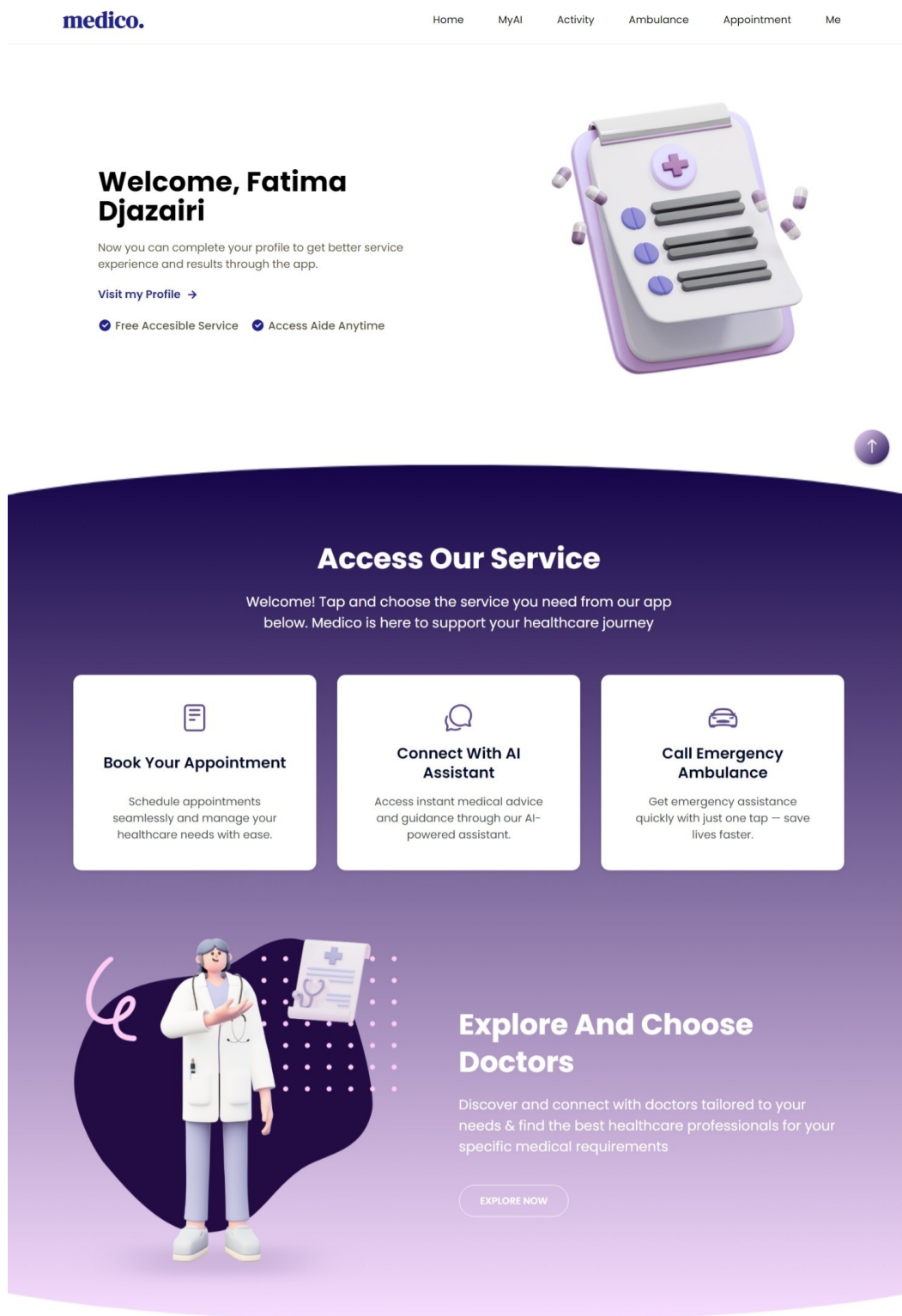


Figure 4.12 – User Home Page Interface.



### 4.2.13 User Search Doctor Interface

Users can use filters like specialty and city to find doctors, as demonstrated in Figure 4.13. Users may easily find pertinent doctors who fit their unique requirements and preferences due to this functionality.

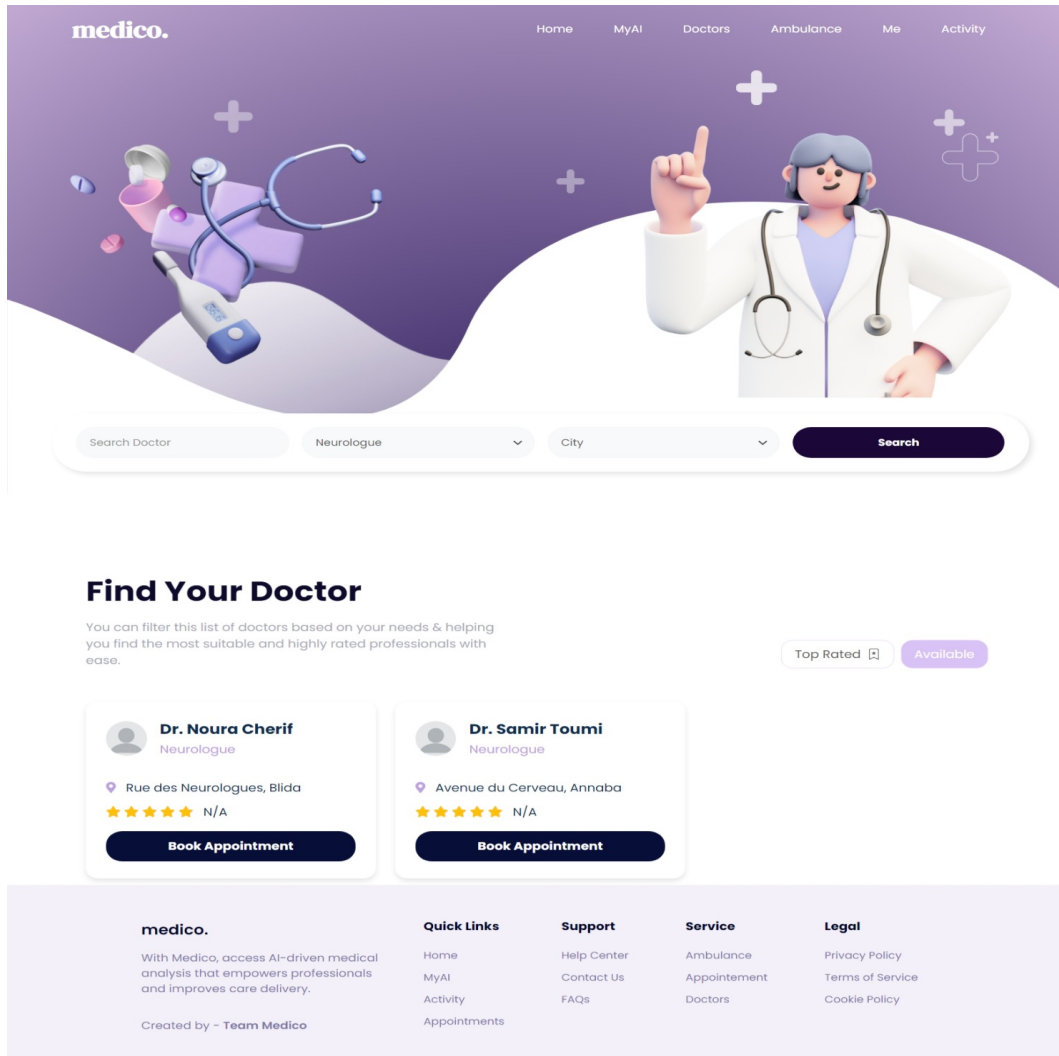


Figure 4.13 – User Search Doctor Interface.

### 4.2.14 Doctor Profile: Search Result Interface

Users can quickly and easily obtain important information like location, expertise, and patient reviews from the doctor profile search results as shown in Figures 4.14 and 4.15. By providing a clear and effective means of comparing medical experts according to pertinent criteria.

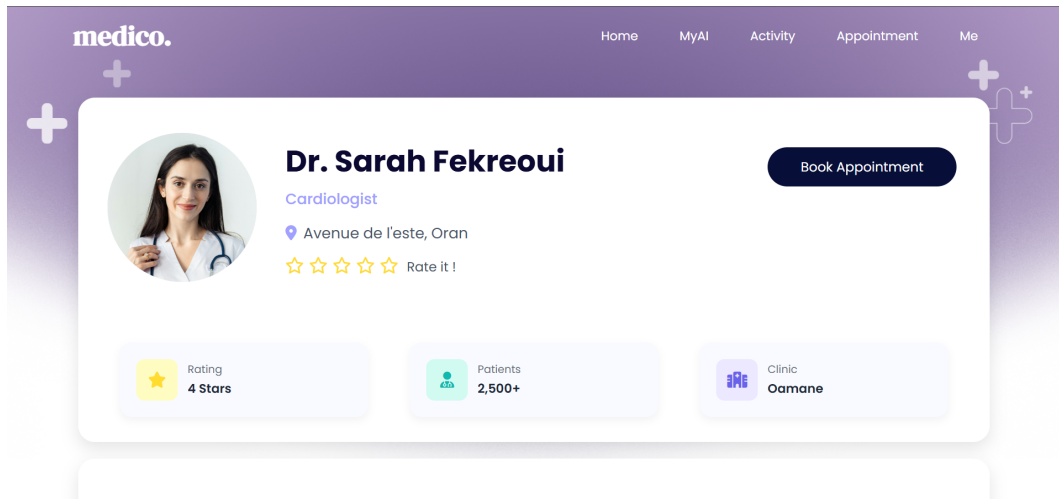


Figure 4.14 – Search Result Interface.

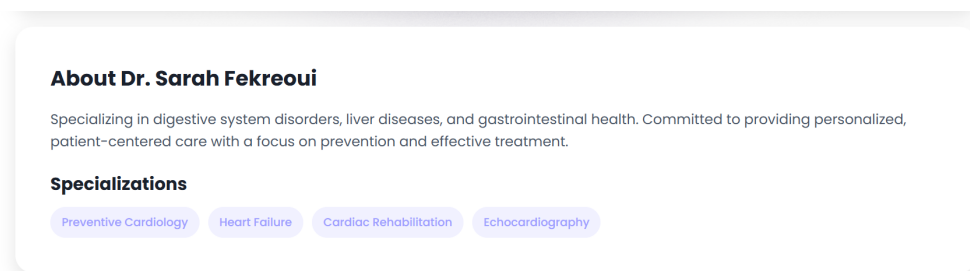


Figure 4.15 – Search Result Interface.

#### 4.2.15 User Book Appointment Interface

Users can schedule consultations by choosing a time slot and confirming the appointment, as shown in Figure 4.16. Access to healthcare services is enhanced and the appointment process is made simpler by this feature. In order to prevent bookings, the system automatically deactivates non-working days and allows doctors to choose their working days in advance. To ensure that only legitimate, future appointments can be set, any dates earlier than the present day are also disabled.

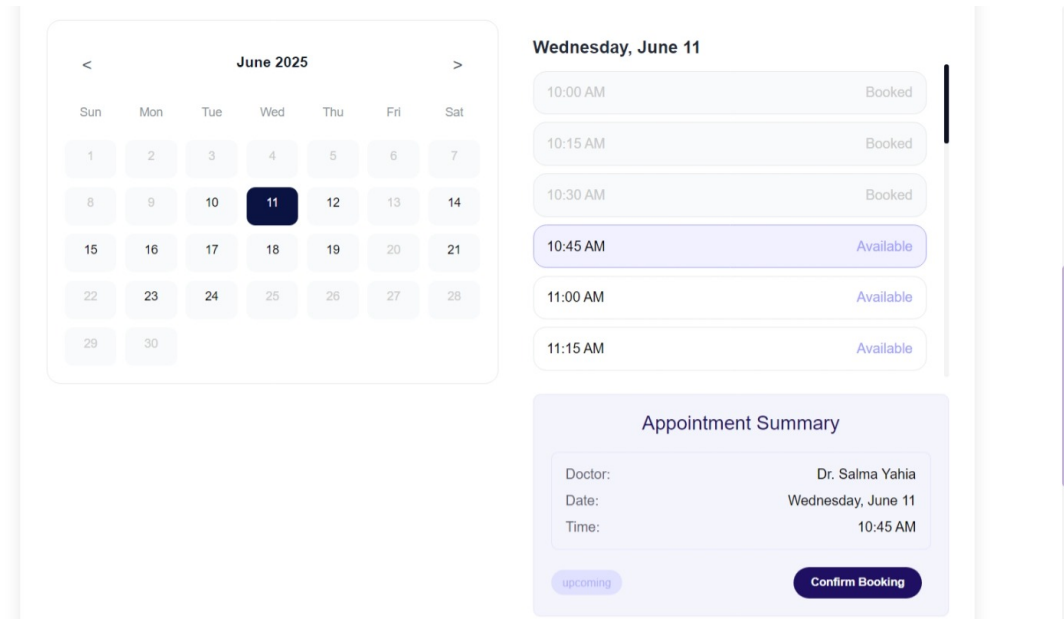


Figure 4.16 – User Book Appointment Interface.

### 4.3 User Appointment List Interface

An appointment is added to a specific list available within the app as soon as the user confirms it. All of the user's booked consultations are included as shown in Figure 4.17, along with important information like the name of the doctor, the date, the time, and their area of expertise. Users may effectively manage their medical appointments and keep track of forthcoming consultations with the aid of this well-organized view.

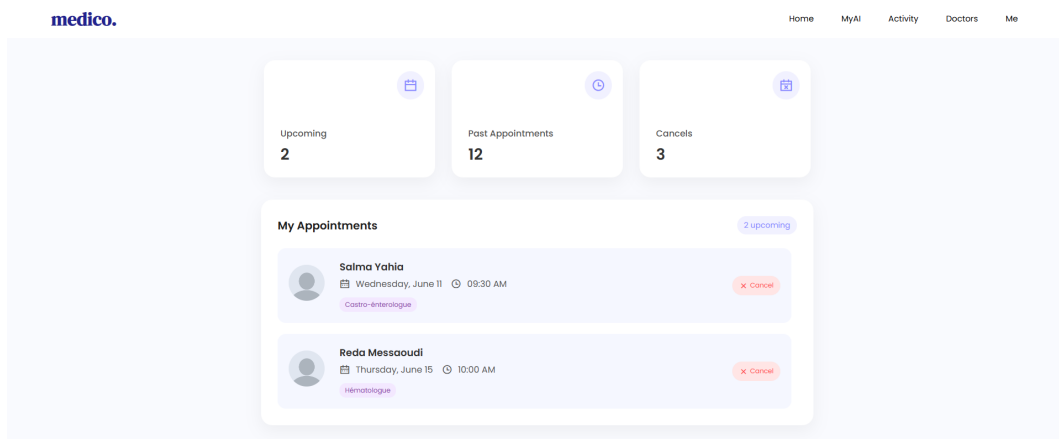


Figure 4.17 – User Appointment List Interface.

### 4.3.1 Available Ambulance List Interface

Users can view a list of ambulances that are currently available with additional information such as location and vehicle type as demonstrated in Figure 4.18.

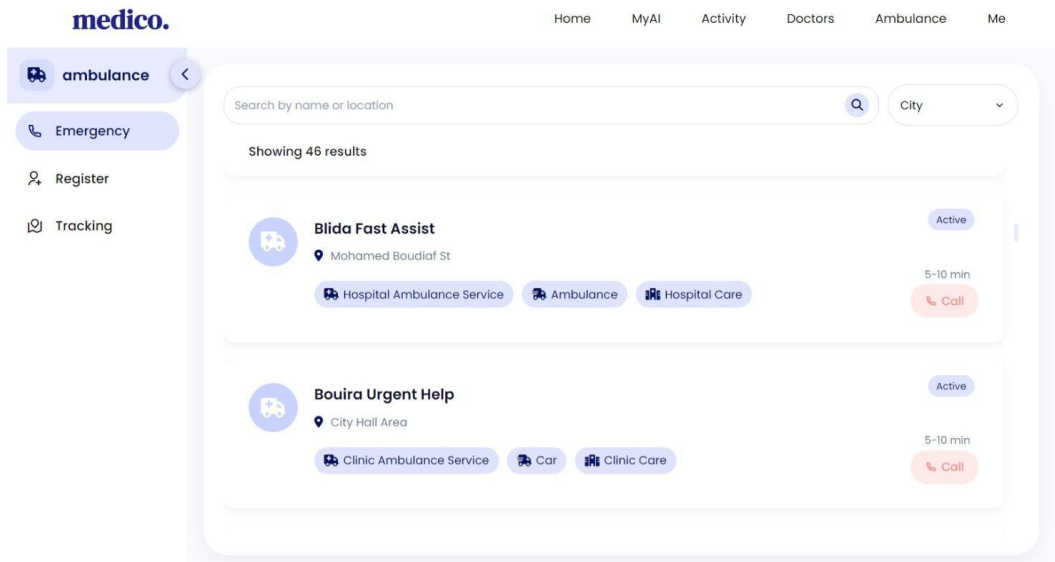


Figure 4.18 – User Available Ambulance List Interface.

### 4.3.2 Register Ambulance Service Interface

As can be seen in Figure 4.19, service providers are able to enter and submit information about their ambulance, such as the city, driver contact information, and vehicle details.

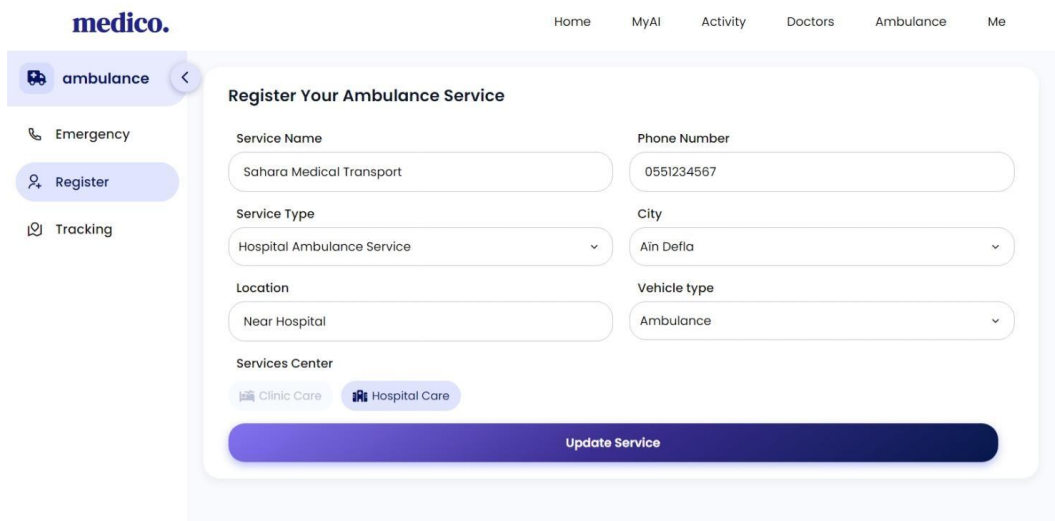


Figure 4.19 – Register Ambulance Service Interface.

### 4.3.3 Ambulance Tracking Interface

Users can trace the location of ambulances while they are traveling thanks to the web application's real-time ambulance tracking capability. By enhancing patient-provider communication, it guarantees quicker cooperation during emergencies.

The Figure 4.20 shows key details and a live map. It displays distance, destination, and includes a button to contact the driver.

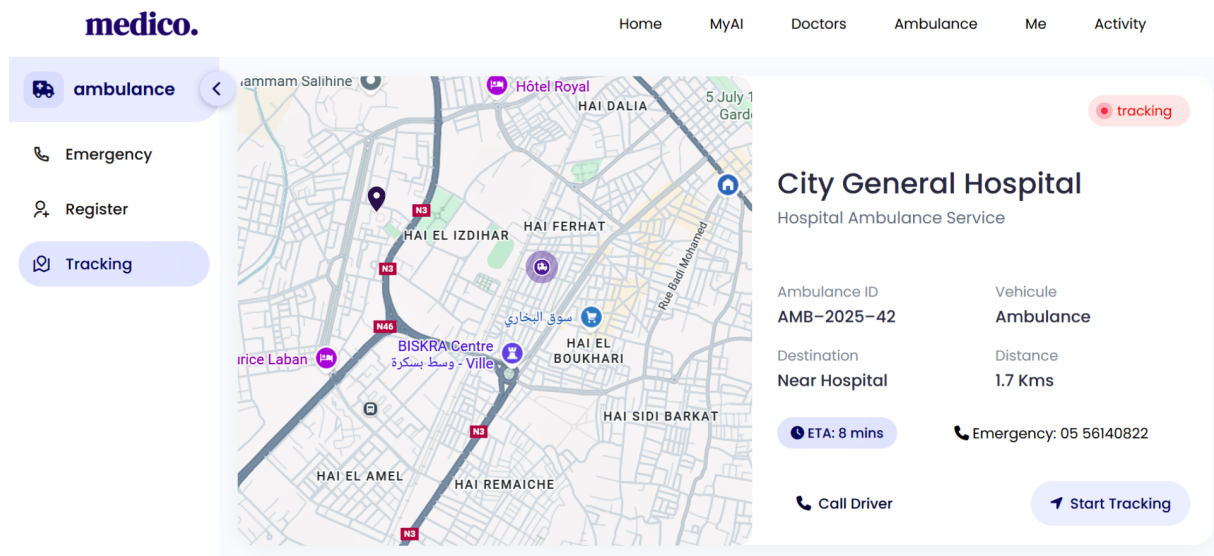


Figure 4.20 – Ambulance Tracking Interface.

### 4.3.4 AI Assistant Interface

As demonstrated in Figures 4.21, 4.22, users can obtain an AI-based diagnosis by uploading medical images, such as MRIs or X-rays. To aid users in understanding the issue, the assistant creates a comprehensive medical report and emphasizes the afflicted area for visual explanation as shown in 4.23 and 4.24.



Figure 4.21 – AI Assistant Interface -Part 01.

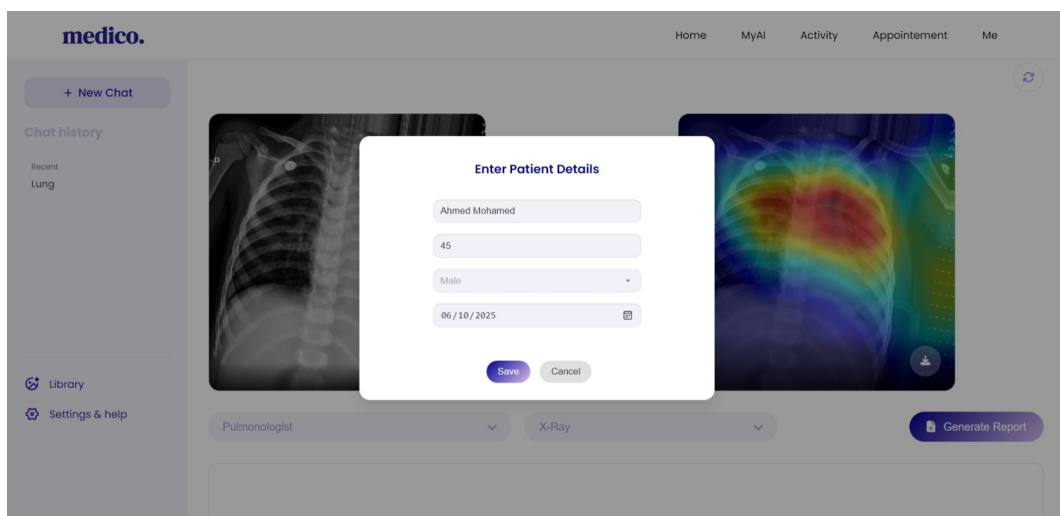


Figure 4.22 – AI Assistant Interface -Part 02.



2. Coughing up mucus or phlegm
3. Shortness of breath
4. Chest pain or pressure
5. Difficulty breathing
6. Fatigue, weakness, and loss of appetite
7. Nausea or vomiting
8. Sore throat
9. Headache
10. Feverish sweats

Available treatment options:

1. Antibiotics (for acute pneumonia)
2. Ventilation (for chronic pneumonia)
3. Oxygen therapy (for severe cases of acute pneumonia)
4. Supportive care (such as fluids, nutrition, and pain management)
5. Home care (for short-term treatment)
6. Immunization (if appropriate)
7. Follow-up with a healthcare provider

Management strategies:

1. Early diagnosis and treatment are crucial for improving outcomes.
2. Antibiotics should be prescribed for acute pneumonia, especially in cases of severe disease or when there is evidence of bacterial infection.
3. Ventilation is indicated for patients with severe pneumonia who cannot tolerate antibiotics or have underlying medical conditions that make them more susceptible to complications.
4. Supportive care such as fluids, nutrition, and pain management can be beneficial for short-term treatment.
5. Immunization (if appropriate) can help prevent the spread of pneumonia in high-risk populations.
6. Follow-up with a healthcare provider is recommended to monitor for complications and adjust treatment as needed.

Figure 4.24 – Medical Imaging Diagnostic Report -Part 02 .



## 4.4 Conclusion

The implementation and GUI of The Web Application thoroughly outlines the practical execution of the "Medico Plus" platform. It begins by detailing the technical work environment, encompassing both the hardware and software specifications critical to the project's development. then transitions into a comprehensive presentation of the graphical user interface (GUI), showcasing the various interactive modules that constitute the web application. Key functionalities highlighted through the GUI include the ambulance emergency booking, the streamlined appointment scheduling system, and the AI Assistant module, which is central to the platform's diagnostic capabilities. In essence, this chapter effectively demonstrates how the theoretical design principles were translated into a tangible, functional, and user-friendly digital healthcare platform.

# General Conclusion

This project, directly addresses critical deficiencies within Algeria’s healthcare system, particularly in patient oversight, appointment logistics, emergency response, and diagnostic support. Our objective was to develop a sophisticated, web-based healthcare management tool that delivers efficient and accessible care through integrated digital solutions.

Medico Plus’s sophisticated artificial intelligence integration, which is intended to revolutionize diagnostic healthcare, lies at its heart. To accurately identify diseases, we have integrated AI models, concentrating on examining medical pictures like CT, MRI, and X-rays. Utilizing deep learning methods—in particular, convolutional neural networks, or CNNs—which are excellent at gleaning intricate features from visual data—this is accomplished. Grad-CAM is one of the Explainable AI (XAI) techniques used to graphically emphasize parts of medical pictures that affected the AI’s predictions in order to increase transparency and confidence in diagnostic decisions. The gap between clinical comprehension and machine interpretation is also filled by the employment of Natural Language Generation (NLG) tools, which translate diagnostic outputs into understandable, human-readable reports.

The platform’s architectural design involved a robust web application framework, detailing distinct layers for front-end, back-end, and full-stack development. The entire system was meticulously conceptualized using Unified Modeling Language (UML) diagrams, including use case diagram , class diagram, and cequence diagrams, which effectively modeled system requirements and illustrated the interactions among key stakeholders.

Medico Plus is ultimately a user-centric solution, offering essential services through an intuitive graphical user interface. Key functionalities include AI-assisted diagnostics via image uploads, streamlined appointment scheduling, and immediate access to emergency ambulance services. Specialized interfaces were also developed for doctors to manage their appointments and for ambulance service providers to register within the system. This

comprehensive approach ensures that all critical aspects of patient care are integrated into a single, accessible platform.

In essence, "Medico Plus" represents a pragmatic and impactful step toward modernizing Algeria's healthcare infrastructure. By strategically combining AI capabilities with a user-friendly web interface, this project aims to refine clinical decision-making, alleviate operational burdens, and significantly improve the speed and efficiency of healthcare delivery. While acknowledging the ongoing challenges in AI interpretability and data privacy within medical imaging, our work prioritizes addressing these concerns to foster greater trust and adoption within the clinical community.

# Bibliography

- [1] E Glover. *What Is Artificial Intelligence (AI)?* 2024, April 2. URL: <https://www.coursera.org/articles/what-is-artificial-intelligence>.
- [2] A. Azizi, M. Azizi, and M. Nasri. “Artificial Intelligence Techniques in Medical Imaging: A Systematic Review”. In: *International Journal of Online and Biomedical Engineering (iJOE)* 19.17 (2023), pp. 66–97. DOI: [10.3991/ijoe.v19i17.42431](https://doi.org/10.3991/ijoe.v19i17.42431).
- [3] Luís Pinto-Coelho. “How Artificial Intelligence Is Shaping Medical Imaging Technology: A Survey of Innovations and Applications”. In: *Bioengineering (Basel)* 10.12 (Dec. 2023), p. 1435. DOI: [10.3390/bioengineering10121435](https://doi.org/10.3390/bioengineering10121435).
- [4] N. Ghaffar Nia, E. Kaplanoglu, and A. Nasab. “Evaluation of Artificial Intelligence Techniques in Disease Diagnosis and Prediction”. In: *Discov. Artif. Intell.* 3 (2023), p. 5. DOI: [10.1007/s44163-023-00049-5](https://doi.org/10.1007/s44163-023-00049-5).
- [5] Y. Kumar et al. “Artificial Intelligence in Disease Diagnosis: A Systematic Literature Review, Synthesizing Framework and Future Research Agenda”. In: *J. Ambient. Intell. Humaniz. Comput.* 14 (2023), pp. 8459–8486. DOI: [10.1007/s12652-021-03612-z](https://doi.org/10.1007/s12652-021-03612-z).
- [6] A. Hosny et al. “Artificial Intelligence in Radiology”. In: *Nat. Rev. Cancer* 18 (2018), pp. 500–510. DOI: [10.1038/s41568-018-0016-5](https://doi.org/10.1038/s41568-018-0016-5).
- [7] S.M. Waldstein et al. “Unbiased Identification of Novel Subclinical Imaging Biomarkers Using Unsupervised Deep Learning”. In: *Sci. Rep.* 10 (2020), p. 12954. DOI: [10.1038/s41598-020-69814-1](https://doi.org/10.1038/s41598-020-69814-1).
- [8] J. Plested and T. Gedeon. “Deep Transfer Learning for Image Classification: A Survey”. In: *arXiv* (2022). eprint: [2205.09904](https://arxiv.org/abs/2205.09904).

- [9] S.A. Alowais et al. “Revolutionizing Healthcare: The Role of Artificial Intelligence in Clinical Practice”. In: *BMC Med. Educ.* 23 (2023), p. 689. DOI: [10.1186/s12909-023-04698-z](https://doi.org/10.1186/s12909-023-04698-z).
- [10] Michele Avanzo et al. “The Evolution of Artificial Intelligence in Medical Imaging: From Computer Science to Machine and Deep Learning”. In: *Cancers (Basel)* 16.21 (Nov. 2024), p. 3702. DOI: [10.3390/cancers16213702](https://doi.org/10.3390/cancers16213702).
- [11] Heang-Ping Chan et al. “Deep Learning in Medical Image Analysis”. In: *Adv. Exp. Med. Biol.* 1213 (2020), pp. 3–21. DOI: [10.1007/978-3-030-33128-3\\_1](https://doi.org/10.1007/978-3-030-33128-3_1).
- [12] S. Kevin Zhou et al. “A Review of Deep Learning in Medical Imaging: Imaging Traits, Technology Trends, Case Studies with Progress Highlights, and Future Promises”. In: *Proc. IEEE Inst. Electr. Electron. Eng.* 109.5 (Feb. 2021), pp. 820–838. DOI: [10.1109/JPROC.2021.3054390](https://doi.org/10.1109/JPROC.2021.3054390).
- [13] John Stephenson. “AI in Medical Imaging Advancements Applications and Challenges”. In: 15.4 (July 2023), pp. 84–86. DOI: [10.37532/1755-5191.2023.15\(4\).84-86](https://doi.org/10.37532/1755-5191.2023.15(4).84-86).
- [14] Scott M McKinney et al. “International evaluation of an AI system for breast cancer screening”. In: *Nature* 577.7788 (2020), pp. 89–94. DOI: [10.1038/s41586-019-1799-6](https://doi.org/10.1038/s41586-019-1799-6).
- [15] Hugo JWL Aerts et al. “Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach”. In: *Nature communications* 5 (2014), p. 4006. DOI: [10.1038/ncomms5006](https://doi.org/10.1038/ncomms5006).
- [16] Mohammad Hosein Hesamian et al. “Deep learning techniques for medical image segmentation: Achievements and challenges”. In: *Journal of digital imaging* 32.4 (2019), pp. 582–596. DOI: [10.1007/s10278-019-00227-x](https://doi.org/10.1007/s10278-019-00227-x).
- [17] Abdelhakim Mekkadem et al. “Security of medical images in e-health applications: a review”. In: *Multimedia Tools and Applications* 80 (2021), pp. 18407–18440. DOI: [10.1007/s11042-020-10067-1](https://doi.org/10.1007/s11042-020-10067-1).
- [18] Nikolaos Sourlos et al. “Validation and standardization in artificial intelligence medical imaging”. In: *Radiology: Artificial Intelligence* 6.1 (2024), e220322. DOI: [10.1148/ryai.220322](https://doi.org/10.1148/ryai.220322).

- [19] Definitive Healthcare. *4 Trends in Medical Imaging Changing Healthcare*. 2024. URL: <https://www.definitivehc.com/blog/future-trends-in-medical-imaging>.
- [20] R. Yamashita, M. Nishio, R.K.G. Do, et al. “Convolutional Neural Networks: An Overview and Application in Radiology”. In: *Insights Imaging* 9 (2018), pp. 611–629. DOI: [10.1007/s13244-018-0639-9](https://doi.org/10.1007/s13244-018-0639-9).
- [21] Yan Li, Tao Zhang, and Yanjie Liu. “A survey of convolutional neural networks: analysis, applications, and prospects”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.12 (2020), pp. 5749–5779. DOI: [10.1109/TNNLS.2020.2990648](https://doi.org/10.1109/TNNLS.2020.2990648).
- [22] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. Version v5. In: *arXiv preprint* (2019). arXiv: [1905.11946](https://arxiv.org/abs/1905.11946). URL: <https://doi.org/10.48550/arXiv.1905.11946>.
- [23] Mingxing Tan and Quoc V Le. “EfficientNet: Rethinking model scaling for convolutional neural networks”. In: *Proceedings of the 37th International Conference on Machine Learning (ICML)*. Vol. 119. PMLR. 2020, pp. 6105–6114.
- [24] ResearchGate. *Architecture of EfficientNet-B0 with MBConv as Basic Building Blocks*. 2025. URL: [https://www.researchgate.net/figure/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks\\_fig4\\_344410350](https://www.researchgate.net/figure/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks_fig4_344410350).
- [25] IBM. *Explainable AI*. URL: <https://www.ibm.com/think/topics/explainable-ai>.
- [26] Temesgen A Mersha, Al-Sakib Khan Pathan Altaher, Syed Mahmud, et al. “Explainable Artificial Intelligence (XAI): A comprehensive review”. In: *Artificial Intelligence Review* 57 (2024), pp. 345–399. DOI: [10.1007/s10462-023-10547-3](https://doi.org/10.1007/s10462-023-10547-3).
- [27] Melanie. *What is the Grad CAM Method?* 2024. URL: <https://datascientist.com/en/what-is-the-grad-cam-method>.
- [28] Ramprasaath R Selvaraju et al. “Grad-CAM: Visual explanations from deep networks via gradient-based localization”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 618–626. DOI: [10.1109/ICCV.2017.74](https://doi.org/10.1109/ICCV.2017.74).
- [29] Diwakar R. Tripathi and Abha Tamrakar. “Natural Language Generation: Algorithms and Applications”. In: *Turkish Journal of Computer and Mathematics Education* 9.3 (2018), pp. 1394–1399.

- [30] Hongwei Dong et al. “A survey of natural language generation: Core tasks, applications, and evaluation”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 13.1 (2022), pp. 1–45. DOI: [10.1145/3470745](https://doi.org/10.1145/3470745).
- [31] Sebastian Vargas, Catalina Hallett, and Chris Mellish. “SemScribe: A real-time NLG system to generate medical reports for physicians”. In: *Proceedings of the 13th European Workshop on Natural Language Generation*. Association for Computational Linguistics. 2012, pp. 146–149.
- [32] Peiyuan Zhang et al. *TinyLlama: An Open-Source Small Language Model*. 2024. URL: <https://arxiv.org/abs/2401.02385>.
- [33] Joe Johnston. *How to Build a Web App: Beginner’s Guide (2025)*. 2024. URL: <https://budibase.com/blog/how-to-make-a-web-app/>.
- [34] Nazar Kvartalnyi. *The Difference Between a Web App and a Website*. Inoxoft. Mar. 9, 2023. URL: <https://inoxoft.com/blog/the-difference-between-a-web-app-and-a-website/>.
- [35] Shweta Bapna et al. “Analyzing Front-End Web Development Technologies Based on their Interactive Nature, Optimization & Characteristics”. In: *International Journal of Research Publication and Reviews* 5.3 (Mar. 2024), pp. 1706–1709.
- [36] Shreshtha Bhatt et al. “A Review of Current Front-End Development Technologies”. In: *International Journal of Research Publication and Reviews* 5.4 (Apr. 2024), pp. 641–644.
- [37] Amazon Web Services. *What is JavaScript?* 2024. URL: [https://aws.amazon.com/what-is/javascript/?nc1=h\\_ls](https://aws.amazon.com/what-is/javascript/?nc1=h_ls).
- [38] Vishesh S et al. “Back-End Web-Application Development and the Role of an Admin”. In: *International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE)* 6.9 (Sept. 2017). ISSN Online: 2278-1021, Print: 2319-5940. DOI: 10.17148/IJARCCE.2017.6911, p. 60. URL: <https://ijarcce.com/wp-content/uploads/2017/09/IJARCCE.2017.6911.pdf>.
- [39] Teguh Rijanandi and Faisal Dharma Adhinata. “Choosing the Right Programming Language in Making a Website Backend Using the Waterfall Method”. In: *International Journal of Engineering Pedagogy (iJEP)* 10.2 (2022). DOI: [10.3991/ijes.v10i02.30845](https://doi.org/10.3991/ijes.v10i02.30845). URL: <https://doi.org/10.3991/ijes.v10i02.30845>.

- [40] PHP Documentation Team. *What is PHP and what can it do?* 2025. URL: <https://www.php.net/manual/en/introduction.php>.
- [41] Filip Andonov. “Comparative Analysis of PYTHON with Other Programming Languages”. In: *Yearbook Telecommunications* 6 (Sept. 2019), pp. 1–15. DOI: [10.33919/YTelecomm.19.6.1](https://ojs.nbu.bg/index.php/YT/article/view/336). URL: <https://ojs.nbu.bg/index.php/YT/article/view/336>.
- [42] Md Tohidul Islam et al. “Comparative Analysis of Programming Language Preferences Among Computer Science and Non-Computer Science Students”. In: *European Journal of Theoretical and Applied Sciences* 2.3 (May 2024), pp. 900–912. DOI: [10.59324/ejtas.2024.2\(3\).70](https://ejtas.com/index.php/journal/article/view/969). URL: <https://ejtas.com/index.php/journal/article/view/969>.
- [43] MongoDB Inc. *Le développement full-stack expliqué*. 2025. URL: <https://www.mongodb.com/fr-fr/resources/basics/full-stack-development> (visited on 04/23/2025).
- [44] Gurjeet Singh, Madiha Javed, and Balwinder Kaur Dhaliwal. “Full Stack Web Development: Vision, Challenges and Future Scope”. In: *International Research Journal of Engineering and Technology (IRJET)* 9.4 (Apr. 2022). Lovely Professional University, pp. 3083–3087. ISSN: 2395-0056. URL: <https://www.irjet.net/archives/V9/i4/IRJET-V9I4398.pdf>.
- [45] A. Sumalatha, Ramu Vookanti, and Srujan Vannala. “Study on Applications of SQL and Not only SQL Databases used for Big Data Analytics”. In: *International Journal for Research & Development in Technology* 15.3 (Apr. 2021). Chaitanya Deemed to be University - Department of Computer Science and Engineering, pp. 127–130. DOI: [10.2139/ssrn.3817635](https://ssrn.com/abstract=3817635). URL: <https://ssrn.com/abstract=3817635>.
- [46] Geoffrey Mwamba Nyabuto, Victor Mony, and Samuel Mbugua. “Architectural Review of Client-Server Models”. In: *International Journal of Scientific Research & Engineering Trends* 10.1 (Jan. 2024). Kibabii University, Information Technology Department, Bungoma, Kenya, pp. 139–145. ISSN: 2395-566X. URL: [https://ijsret.com/wp-content/uploads/2024/01/IJSRET\\_V10\\_issue1\\_125.pdf](https://ijsret.com/wp-content/uploads/2024/01/IJSRET_V10_issue1_125.pdf).



- [47] Haroon Shakirat Oluwatosin. “Client-Server Model”. In: *IOSR Journal of Computer Engineering (IOSR-JCE)* 16.1 (Feb. 2014). School of Computing, Universiti Utara Malaysia, Kedah, Malaysia, pp. 67–71. ISSN: 2278-0661. URL: <https://www.iosrjournals.org/iosr-jce/papers/Vol16-issue1/Version-9/J016195771.pdf>.
- [48] Christopher S. Guynes and David Windsor. *Information Systems: A Management Approach*. Upper Saddle River, NJ: Prentice Hall, 2011.
- [49] H. Sharanagowda. “Understanding Client-Server Architecture”. In: *International Journal of Computer Science* 14.3 (2022), pp. 45–52.
- [50] Geoffrey Mwamba Nyabuto, Victor Mony, and Samuel Mbugua. “Architectural Review of Client-Server Models”. In: *International Journal of Scientific Research & Engineering Trends* 10.1 (Jan. 2024). Kibabii University, Information Technology, Bungoma, pp. 139–145. ISSN: 2395-566X. URL: [https://www.researchgate.net/publication/376512127\\_Architectural\\_Review\\_of\\_Client-Server\\_Models](https://www.researchgate.net/publication/376512127_Architectural_Review_of_Client-Server_Models).
- [51] uml-diagrams.org. *UML Actors*. 2025. URL: <https://www.uml-diagrams.org/use-case-actor.html>.
- [52] GeeksforGeeks. *Functional vs Non-Functional Requirements*. 2023. URL: <https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/>.
- [53] AltexSoft Editorial Team. *Nonfunctional Requirements*. 2023. URL: <https://www.altexsoft.com/blog/non-functional-requirements/>.
- [54] Object Management Group. *What is UML*. 2025. URL: <https://www.uml.org/what-is-uml.htm>.
- [55] uml-diagrams.org. *UML Use Case Diagrams*. 2025. URL: <https://www.uml-diagrams.org/use-case-diagrams.html>.
- [56] Yashwant Waykar. “Significance of Class Diagram in Software Development”. In: *International Journal of Advanced Research in Computer Science and Software Engineering* 4.2 (2014), pp. 425–429. URL: [https://www.researchgate.net/publication/322991881\\_Significance\\_of\\_class\\_diagram\\_in\\_software\\_development](https://www.researchgate.net/publication/322991881_Significance_of_class_diagram_in_software_development).
- [57] PlantUML Team. *Sequence Diagram - PlantUML*. 2025. URL: <https://plantuml.com/sequence-diagram>.

- [58] Microsoft. *Visual Studio Code*. 2025. URL: <https://code.visualstudio.com/>.
- [59] Google. *Questions fréquentes - Colaboratory*. 2025. URL: <https://research.google.com/colaboratory/faq.html?hl=fr>.
- [60] Apache Friends. *À propos d'Apache Friends et XAMPP*. 2025. URL: <https://www.apachefriends.org/fr/about.html>.