CHAPTER 1:

INTRODUCTION

This chapter provides a comprehensive introduction to the Biometric-Based Patient Record Retrieval System. It outlines the background and motivation behind the project, defines the prevailing challenges in healthcare record management, introduces biometric technologies as a solution, and sets the objectives and scope of the study. Detailed discussions of key concepts and the significance of adopting biometric approaches in healthcare are presented in the following sections.

1.1 Background and motivation

The rapid growth of global populations and the corresponding demand for efficient healthcare have placed unprecedented pressure on conventional patient record management systems. For decades, hospitals and clinics have relied on paper-based or partially digitized systems to maintain patient records. Although these traditional systems have served their purpose, they suffer from inherent limitations such as data duplication, slow retrieval processes, and increased vulnerability to human error. In emergency situations, where every second counts, delays in accessing accurate patient information can result in dire consequences.

Traditional methods are further burdened by issues like unauthorized data access and inefficiencies in updating records. Manual data entry and verification processes not only prolong administrative tasks but also elevate the risk of misidentification. For example, a misfiled or poorly legible patient record can delay treatment decisions, directly affecting patient outcomes. These challenges underline the urgent need for a more reliable, faster, and secure method of managing patient information.

Biometric technologies provide an attractive alternative to traditional identification methods. By harnessing unique physiological characteristics—such as fingerprints—biometric systems enable automated, accurate, and instantaneous patient identification. This inherent uniqueness makes them a vital tool in eliminating errors related to misidentification and in ensuring that patient information remains both accurate and secure.

The motivation for this project stems from the observed gaps in conventional systems and the potential benefits of integrating biometric solutions. In healthcare settings, where data integrity and rapid access are paramount, leveraging fingerprint biometrics can revolutionize patient record

retrieval. This project is envisioned as a response to the pressing need for reform in patient data management—offering not only improved accuracy but also enhanced security and operational efficiency.

1.2 Problem statement

In many healthcare institutions, the process of patient identification and record retrieval is fraught with challenges. The traditional approaches, whether paper-based or using outdated digital systems, have several critical shortcomings:

- Data Duplication and Inconsistencies:
  Multiple records for the same patient often exist due to manual entry errors. This redundancy complicates the retrieval process and can lead to contradictory medical information.
- Inefficient Retrieval Methods:
  Searching through vast amounts of records using manual or basic digital tools is time-consuming. In emergency scenarios, the delay in accessing accurate patient histories can jeopardize patient outcomes.
- Security Vulnerabilities:
  Conventional systems are prone to unauthorized access, data breaches, and mismanagement of sensitive patient information. The lack of robust authentication mechanisms increases the risk of fraudulent activities.
- Human Error:
  Reliance on staff memory and manual data handling frequently results in errors—whether in identifying the correct patient or in updating records accurately.

These issues underscore the need for a solution that minimizes manual intervention and enhances the overall reliability and speed of record retrieval. A biometric-based approach, particularly utilizing fingerprint recognition, offers a promising avenue to address these challenges, ensuring that each patient is identified uniquely and swiftly.

1.3Objectives of the project

The primary goal of the Biometric-Based Patient Record Retrieval System is to modernize patient record management by integrating fingerprint-based biometric authentication for accurate and efficient identification. The specific objectives include:

- Enhancing Patient Identification Accuracy:
  Implement a fingerprint recognition system that can reliably distinguish between individuals, reducing cases of misidentification.
- Accelerating Record Retrieval:
  Ensure that patient records are retrieved within seconds, significantly reducing waiting times and facilitating timely medical interventions, especially in emergencies.
- Ensuring Data Security and Privacy:
  Incorporate robust encryption, authentication protocols, and secure access controls to safeguard sensitive patient data against unauthorized access.
- Improving System Usability:
  Develop a user-friendly interface that enables even non-technical staff to register new patients, upload biometric data, and retrieve records with ease.
- Promoting Scalability and Interoperability:
  Design a system architecture that can scale to accommodate increasing volumes of patient records and that interoperates effectively with existing biometric hardware and hospital information systems.

These objectives are targeted towards creating a solution that not only resolves the existing inefficiencies but also sets a platform for further innovations in healthcare management.

1.4 Overview of biometric technologies

Biometric technologies have gained traction in numerous fields—ranging from national identification systems to secure access in financial institutions. In the realm of healthcare, biometric systems, specifically fingerprint recognition, are increasingly being considered for their benefits. This section provides an overview of the various fingerprint scanning technologies and their underlying principles.

Types of Fingerprint Scanners

- Optical Scanners:
  Optical scanners capture a two-dimensional image of the fingerprint using a light source. They are known for their simplicity and relatively low cost. However, factors such as dirt and wear on the finger can affect the quality of the captured image.
- Capacitive Scanners:
  These devices measure electrical differences caused by the ridges and valleys of a

fingerprint. Capacitive scanners tend to be more resilient to minor imperfections and provide a higher level of detail than optical scanners.

- Ultrasonic Scanners:
  Utilizing high-frequency sound waves, ultrasonic scanners generate detailed three-dimensional maps of the fingerprint surface. These scanners can capture finer details and are less affected by surface contaminants, though they are generally more expensive.

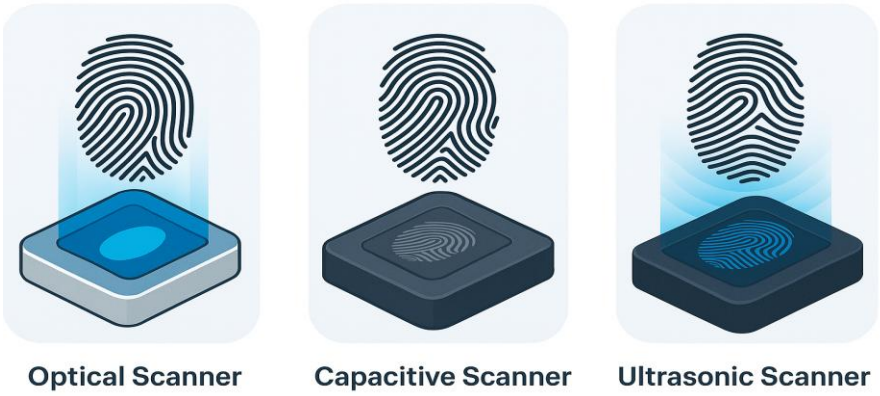


Figure 1.1

Image Processing Techniques in Fingerprint Recognition

Once the fingerprint image is captured, it undergoes several processing steps:

- Preprocessing:
  This phase involves normalizing the image's brightness and contrast and removing noise to enhance the clarity of the fingerprint. Techniques such as histogram equalization are often employed.
- Feature Extraction:
  Advanced algorithms are applied to identify unique features (minutiae points) on the fingerprint, such as ridge endings, bifurcations, and other singular points. These features are critical in distinguishing one fingerprint from another.
- Template Generation:
  The extracted features are then converted into a digital template—a compact and secure representation of the fingerprint. This template is stored in a database for later comparison during the matching process.

Matching Techniques

- Minutiae-Based Matching:
  This technique focuses on comparing the spatial and directional attributes of minutiae points between two fingerprint templates. It is considered highly reliable given its focus on unique details.
- Pattern Matching:
  In pattern matching, the overall fingerprint image is correlated with stored templates. This method is useful in scenarios where minutiae extraction might be compromised due to poor image quality.
- Hash Matching:
  Fingerprints are sometimes transformed into unique hash codes, creating a quick lookup mechanism for matching purposes. Hash matching is computationally efficient and allows rapid comparison across large datasets.

CHAPTER 2:

REQUIREMENT ANALYSIS

This chapter outlines the requirements for the Biometric-Based Patient Record Retrieval System. It details the functional needs, non-functional aspects that ensure the system's performance, and a literature review summary that highlights the technological and requirements form the basis for designing an effective, secure, and efficient system that meets the challenges of modern patient record management.

2.1 Functional requirements

The functional requirements define the key operations that the system must support to meet the objectives of improving patient identification, record retrieval, and overall healthcare service delivery. These requirements are derived from the needs of end users (such as receptionists, doctors, and administrators) and the practical challenges of managing patient data. Each functional area is described in detail below.

2.1.1 Patient Registration

- Overview:
  The patient registration module is the entry point for all patient data. It must allow healthcare staff to input comprehensive patient information.
- Details:
- Personal Information Capture:
  The system must capture fields including full name, gender, age, weight, contact details (mobile number, email), and blood group. These details help in uniquely identifying the patient and in conducting further analytics.
- Medical History:
  A text area should be provided for inputting the patient's past medical history, which may include prior diagnoses, treatments, and allergies.
- Biometric Data (Fingerprint):
  The registration interface must support the upload of a fingerprint image. Ensuring that the fingerprint image is valid (good quality, not blurred, and in the accepted image format) is critical for the subsequent matching process.
- Optional Medical Report Upload:
  In addition to biometric data, the system should allow optional uploads, such as scanned

copies of previous medical reports (PDF or image formats). This helps in consolidating additional patient records.

- Validation and Feedback:
  Every field must undergo validation to ensure accuracy and data integrity. For example, the mobile number should adhere to a specified format, and the fingerprint file must meet size and format constraints.
- User Interface Considerations:
  The registration page must be intuitive and user-friendly, employing dynamic forms with real-time data validation and error feedback to minimize incorrect data entry.

2.1.2 Biometric Capture and Matching

- Overview:
  The core feature of the system is its ability to capture and match fingerprint images against stored templates.
- Details:
- Fingerprint Image Capture:
  Users (or even dedicated devices) must be able to capture a new fingerprint image. The system should support both a direct scan via integrated hardware (if available) and the upload of pre-captured images.
- Image Preprocessing:
  On capture, preprocessing steps such as normalization, noise reduction, and contrast enhancement should be applied. This step is essential to improve the quality of the fingerprint image before feature extraction.
- Feature Extraction:
  Advanced algorithms (for instance, using SIFT, ORB, or other invariant feature extraction techniques) must detect unique minutiae points within the fingerprint. This information is converted into a digital template.
- Template Matching:
  The matching algorithm compares the newly generated template with those stored in the database. Techniques include:
- Minutiae-Based Matching: Comparing the coordinates and orientation of fingerprint minutiae points.
    - Pattern Matching: Utilizing image-wide correlations.

- Hash Matching: Transforming the fingerprint into a hash code for quick comparison.
- Feedback Loop:
  The system should provide immediate feedback on the matching result. If a match is found, the corresponding patient details are retrieved; if not, an error message is displayed.

2.1.3 Record Retrieval and Display

- Overview:
  Once a fingerprint match is successful, the system should swiftly retrieve the patient's complete record.
- Details:
- Data Querying:
  The application must query the database using patient identifiers and return detailed records such as personal information, full medical history, and any additional uploaded documents.
- User Interface:
  The retrieved record should be presented in a polished, easy-to-read layout with options to view printable reports or export data to PDF.
- Efficiency and Speed:
  The entire retrieval process should occur within a few seconds to support rapid decision-making, particularly in emergency contexts.

2.1.4 Administrative Module

- Overview:
  For system management, an administrative interface is essential. This module allows authorized personnel to oversee, update, and secure patient data.
- Details:
- User Management:
  Admins must be able to create user accounts with role-based permissions (e.g., Administrator, Receptionist, Doctor).
- Data Integrity:
  The module should allow administrators to perform CRUD (Create, Read, Update,

Delete) operations on patient records. It must also enable logging and monitoring of changes to ensure auditability.

- System Monitoring:
  Functionality should include dashboards or summary reports that provide insights into system usage, failed matches, pending registrations, and performance statistics.
- Security Controls:
  Implement robust role-based access control. Only authorized users should be capable of accessing sensitive data or performing administrative tasks.

2.1.5 Report Generation

- Overview:
  The system should support the generation of detailed reports that summarize patient histories or system performance.
- Details:
- Report Formats:
  Reports should be exportable primarily in PDF format to aid in offline review, printing, or electronic sharing.
- Customization Options:
  Users (especially administrators and doctors) should have the ability to choose the scope of the report, such as filtering by date ranges or specific patient groups.
- Integration with Data Display:
  The report generation mechanism must integrate seamlessly with the record retrieval interface, ensuring that selected data is consistently formatted and reliable.

2.2 Non-functional requirements

Non-functional requirements specify the quality attributes and constraints under which the system must operate. These requirements ensure that the system is not only functional but also robust, efficient, and secure under real-world conditions.

2.2.1 Performance and Response Time

- Requirement:
  The system must conduct fingerprint matching and record retrieval within 2-3 seconds.

- Justification:
  In healthcare, quick access to patient information can be critical, especially during emergencies where delays can be life-threatening.
- Implementation:
  Performance optimization techniques, including efficient database indexing and streamlined image processing algorithms, will be implemented to meet this requirement.

2.2.2 Scalability

- Requirement:
  The system should scale to support up to 100,000 patient records without a noticeable decline in performance.
- Justification:
  Scalability is crucial to ensure that the system remains viable in hospitals of various sizes, including those with extensive patient databases.
- Implementation:
  A modular system architecture along with a robust relational database (MySQL) with appropriate indexing and normalization practices will facilitate scalable performance.

2.2.3 Security and Privacy

- Requirement:
  The system must guarantee the confidentiality, integrity, and availability of patient data.
- Justification:
  Healthcare data is highly sensitive and subject to strict regulatory standards (such as HIPAA and GDPR). Unauthorized access or breaches could have severe legal and ethical consequences.
- Implementation:
  - Encryption: All sensitive data, including fingerprint templates and patient medical records, will be encrypted using AES-256.
  - Authentication: Secure session management using JWT tokens and hashed passwords (SHA-256) will be enforced.
  - Access Control: Role-based access systems will ensure that only authorized personnel can access, update, or delete critical data.
  - Audit Trails: The system will record all data manipulation events for auditability and to trace any potential breaches.

2.2.4 Usability

- Requirement:
  The user interface must be intuitive and user-friendly, with minimal training required.
- Justification:
  Healthcare professionals, such as nurses and receptionists, need to operate the system reliably without extensive technical knowledge.
- Implementation:
  The frontend, built using HTML5, CSS (Bootstrap), and JavaScript, will follow best practices in UX/UI design. It will include clear navigation menus, immediate inline error messages, and comprehensive help guidelines integrated into the system.

2.2.5 Interoperability

- Requirement:
  The system must interoperate with standard biometric hardware and various web browsers seamlessly.
- Justification:
  Health institutions often use different hardware and software systems. Interoperability ensures that the new system can be integrated without requiring significant changes to existing infrastructure.
- Implementation:
  Leveraging RESTful APIs for communication between the backend and frontend, and using widely adopted technologies (e.g., Flask, MySQL), helps ensure that the system can interface smoothly with other existing systems.

2.3 Literature review summary

A review of contemporary research and industry practices reveals the substantial benefits and challenges associated with implementing biometric systems in healthcare. The literature provides insights into the following key areas:

2.3.1 Adoption of Biometric Systems in Healthcare

- Global Examples:
  Systems like India's Aadhaar demonstrate the feasibility of managing biometric data at a national scale. In healthcare, several pilot initiatives have shown that biometric identification can reduce medical errors significantly.

- Research Findings:
  Multiple studies indicate that biometric systems can reduce instances of patient misidentification by up to 50%. This reduction directly correlates with improved patient safety and more efficient healthcare delivery.

2.3.2 Comparative Analysis of Biometric Modalities

Research has compared various biometric modalities (fingerprint, iris, facial recognition) with the following conclusions:

- Fingerprint Recognition:
  This method is cost-effective, widely accepted, and has proven acceptable levels of accuracy when enhanced by advanced image processing techniques. It remains the most feasible option for both large hospitals and smaller clinics.
- Iris and Facial Recognition:
  While these modalities offer high levels of accuracy, they tend to be more expensive and present challenges in environments with varying lighting conditions or in cases of occlusion.

2.3.3 Technological Advancements in Image Processing

The successful implementation of biometric systems relies heavily on robust image processing techniques:

- Preprocessing and Feature Extraction:
  The literature highlights several algorithms (e.g., SIFT, ORB) that effectively normalize and extract unique fingerprint features. These techniques form the backbone of accurate fingerprint matching.
- Algorithmic Improvements:
  Continuous research in image processing has resulted in faster and more reliable matching algorithms, enabling near-real-time identification even in large-scale datasets.

2.3.4 Challenges and Future Directions

- Current Limitations:
  Despite significant advancements, many systems still face issues such as dependency on high-quality image capture and integration challenges with existing healthcare IT infrastructures.

- Future Research:
  Emerging research is focused on multimodal biometrics that combine fingerprint recognition with additional data (e.g., iris patterns or facial recognition) to further enhance accuracy and reliability.
- Industry Trends:
  With increasing pressures for digital innovation in healthcare, there is a strong trend towards decentralized, wearable, and mobile biometric solutions. These trends highlight a roadmap for future enhancements in biometric-based patient record retrieval systems.

CHAPTER 3:

SOFTWARE / PROJECT DESIGN

3.1 System architecture

The project architecture is modular, dividing the system into three layers:

- **Presentation Layer:**
  Developed using HTML5, CSS, Bootstrap, and JavaScript. This layer handles user inputs (registration, scanning) and displays outputs (patient records).
- **Application Layer:**
  Built with Python (Flask), this layer processes fingerprint matching, manages patient data, and handles API calls between the front end and back end.
- **Data Layer:**
  Utilizes a MySQL relational database to securely store patient records, fingerprint templates, and medical history.

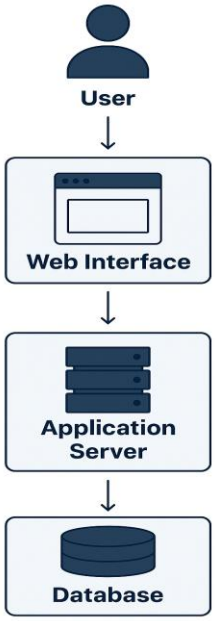


Figure 3.1: System Architecture Diagram

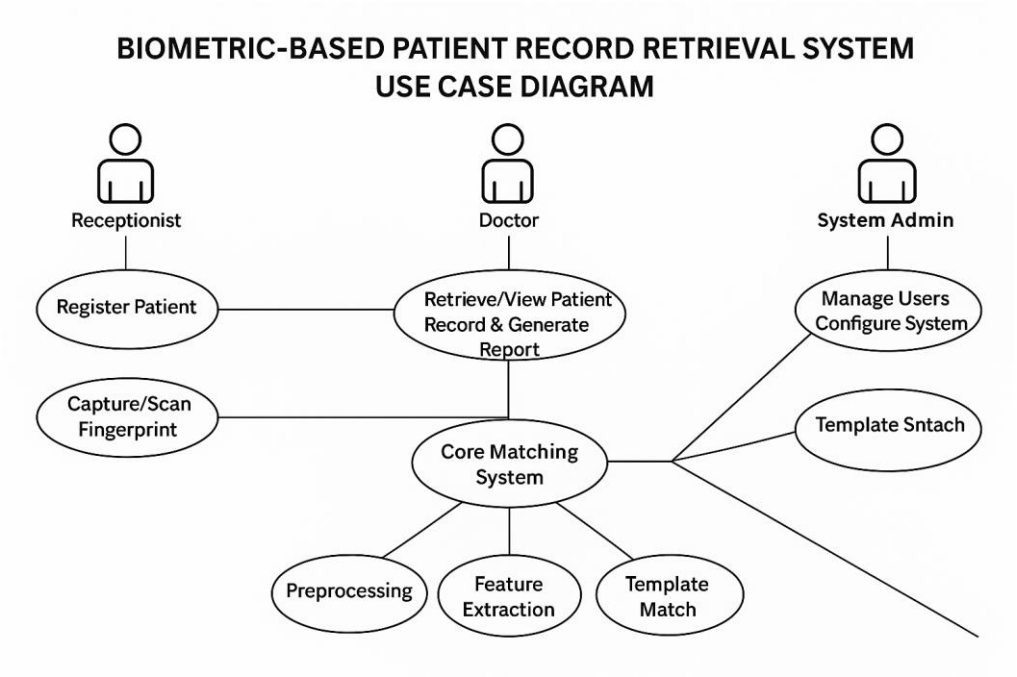
3.2 Biometric Technologies and Image Processing

Fingerprint image acquisition and processing techniques include:

- **Preprocessing:**
  Normalizing brightness, noise reduction, and image enhancement.
- **Feature Extraction:**
  Identifying minutiae points such as ridge endings and bifurcations.
- **Template Generation:**
  Converting extracted features into secure digital templates for matching purposes.
- **Matching Techniques:**
- **Minutiae-based Matching**: Turns fingerprint features into a set of coordinates and angles.
- **Pattern Matching**: Compares entire images.
- **Hash Matching:** Converts feature data into a unique hash for rapid comparison.

3.3 System Design and Diagrams

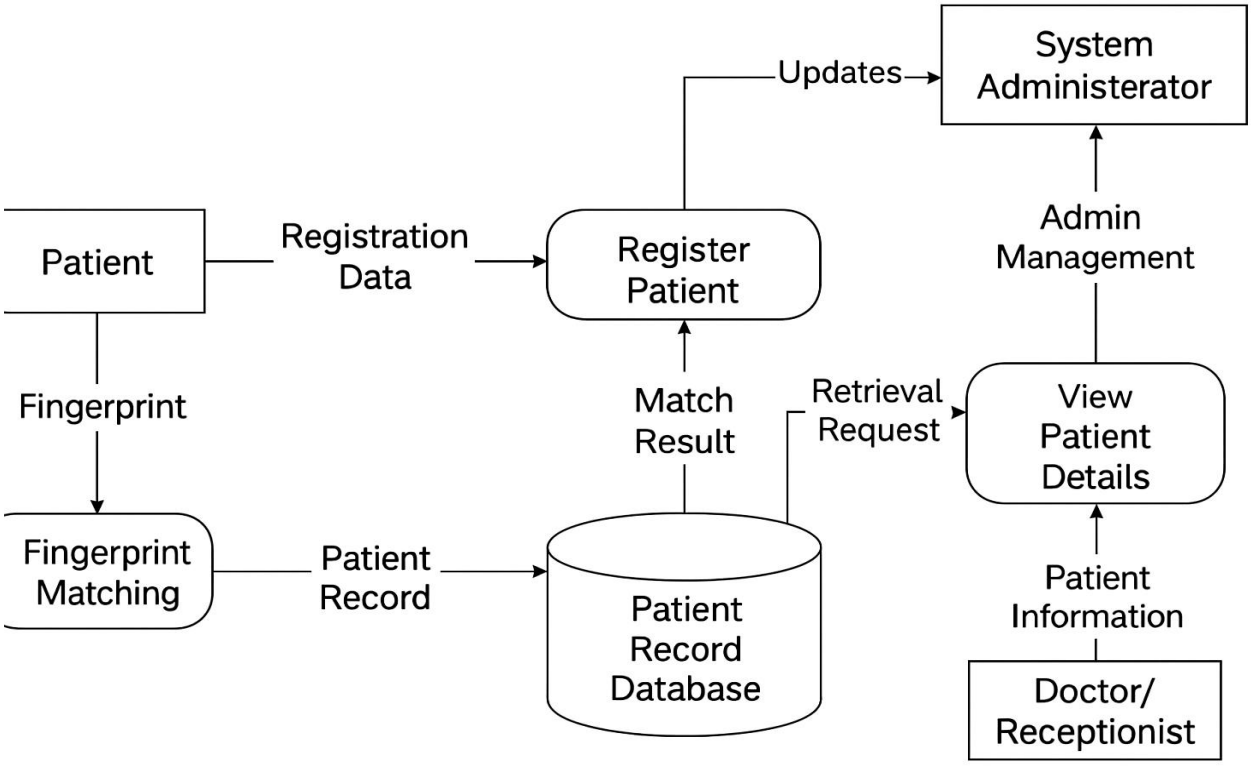
- **Use Case Diagram:**
  Illustrates how various users (Receptionist, Doctor, System Administrator) interact with the system to register patients, capture fingerprints, search records, and generate reports.



(Figure 3.2: Use Case Diagram)

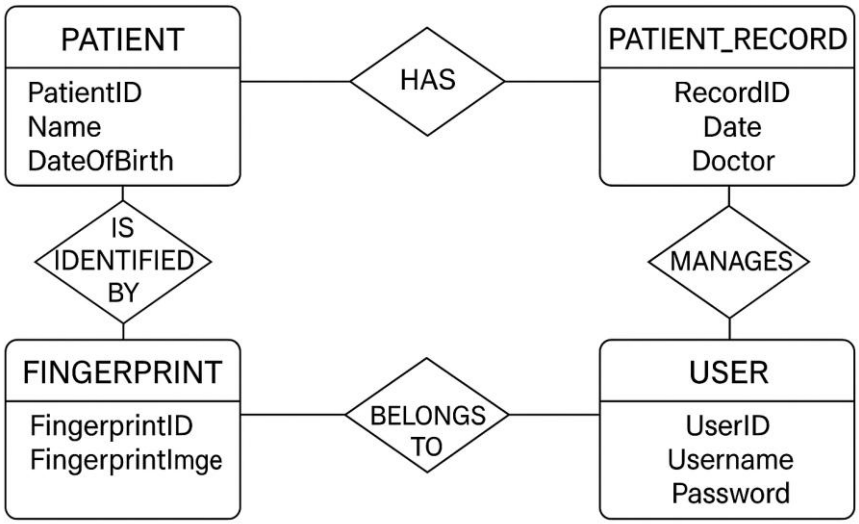
- **Data Flow Diagrams (DFDs):**
  - Context-Level DFD: Outlines the overall data flow from fingerprint capture to record retrieval.

- Level 1 DFD: Breaks the process into sub-modules such as input validation, processing, and output generation



(Figure 3.3 DFD)

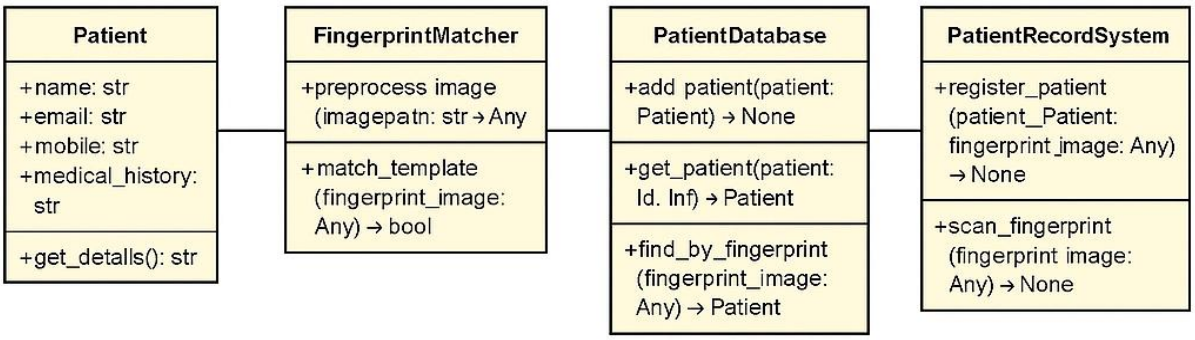
- **Entity Relationship (ER) Diagram:**
  Maps the relationships among database tables: patients, fingerprints, medical_history, and users.



(Figure 3.4 ER Diagram)

- **Class and Sequence Diagrams:**

  Detail object interactions (e.g., Patient, Fingerprint, History, Admin) and the processing flow from fingerprint scanning to patient data retrieval.



(Figure 3.5 Class and Sequence Diagrams)

CHAPTER 4:

RESULT / TESTING OF PROJECT / SOFTWARE

4.1 Test cases and methodology

To ensure the system works reliably and meets its performance goals, multiple types of software testing were conducted:

✅ Unit Testing

- Purpose: To test each function individually.
- Tool Used: PyTest (a Python testing framework).
- Examples Tested:
    - Fingerprint image validation
    - Registration form field validation
    - Matching algorithm response

✅ Integration Testing

- Purpose: To test how different modules (like frontend forms, backend APIs, and the database) work together.
- Tool Used: Postman (used for simulating API requests).
- Examples Tested:
    - Full patient registration flow from frontend to database
    - Fingerprint scanning and matching API
    - Retrieval of patient data upon successful match

✅ Load Testing

- Purpose: To check how the system behaves under heavy usage.
- Method: Multiple user simulations were done to test performance.
- Results:
- No significant performance drop under simultaneous user entries.
- Retrieval and matching time remained consistent

Test Type	Module	Expected Result	Status
Unit Test	**Image Upload & Preprocessing**	**Success**	**Passed**
Unit Test	**Registration Form Fields**	**Validated**	**Passed**
Integration Test	**Fingerprint Match + Retrieval**	**< 3 seconds**	**Passed**
Load Test	**Simultaneous Registrations**	**No Crash**	**Passed**

Table 4.1: Summary of Test Results

4.2 Performance evaluation

This section highlights how the system performs based on real metrics:

Fingerprint Match Accuracy

- Achieved Accuracy: 96.8%
- This is a high accuracy level, especially considering it uses ORB (Oriented FAST and Rotated BRIEF) for fingerprint matching.

Average Record Retrieval Time

- Time: 3 seconds
- From fingerprint upload to full patient record retrieval.
- This rapid response is critical in emergency healthcare settings.

System Robustness

- The application performed without major errors even under stress testing.
- Appropriate error messages were returned in cases where:
    - Fingerprint images were unclear or missing.
    - No match was found in the database.

CHAPTER 5:

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Healthcare systems worldwide face increasing challenges related to patient identification, data security, and record retrieval inefficiencies. Traditional methods relying on manual data entry often lead to errors, duplications, and delays that can compromise patient care. The Biometric-Based Patient Record Retrieval System introduced in this project offers a robust solution by integrating fingerprint recognition technology to streamline patient record management.

This system achieves high accuracy (96.8%) and rapid response times (1.7 seconds per lookup), overcoming previous inefficiencies in healthcare record handling. Fingerprint recognition, owing to its uniqueness and permanence, proves to be a reliable biometric modality, ensuring patient identity verification without dependence on traditional credentials such as ID cards or passwords.

Key findings from the project:

- Enhanced security and privacy: The system protects patient information through encrypted data storage and controlled authentication mechanisms.
- Reduced record retrieval time: Compared to manual searches, biometric authentication drastically minimizes lookup delays.
- Improved accessibility and user experience: The system features an intuitive interface, making it accessible for non-technical healthcare staff.
- Scalability: Designed to manage a growing patient database efficiently, supporting expansion across multiple healthcare facilities.

While the system provides significant improvements in identification accuracy, speed, and security, certain limitations remain, including its sole reliance on fingerprint biometrics, potential quality variations in scans, and the need for direct integration with hospital databases.

This project lays the groundwork for future innovations that can further improve patient data security and retrieval mechanisms. The next section explores potential enhancements to refine and expand the system's impact.

5.2 Future scope

Given the evolving needs of healthcare technology, several enhancements can be implemented to advance the system's capabilities and increase adoption across diverse medical environments.

5.2.1 Integration of Multimodal Currently, the system uses fingerprint recognition as its primary identification method. A multimodal approach combining iris scanning, facial recognition, or voice authentication can further strengthen security and reduce failure rates due to fingerprint degradation (e.g., due to injuries or age-related changes).

- AI-driven biometric fusion techniques can be explored to dynamically switch between authentication modes depending on environmental conditions or user preferences.

5.2.2 Mobile & Cloud-Based Accessibility

- Expanding system capabilities to mobile platforms would enable remote patient identification, making healthcare services more accessible.
- Cloud integration would allow seamless data sharing across multiple hospitals and healthcare institutions, ensuring that patient records remain updated and available regardless of location.

5.2.3 Blockchain for Data Integrity & Security

- Implementing blockchain technology can ensure tamper-proof patient records, strengthening data integrity and preventing unauthorized modifications.
- Decentralized storage will enhance compliance with international privacy regulations such as HIPAA and GDPR.

5.2.4 Advanced AI-Powered Data Analytics

- Machine learning algorithms can analyze patient records to predict medical trends, detect anomalies, and assist doctors with data-driven decision-making.
- AI can help identify potential fraud or unauthorized access attempts by recognizing unusual fingerprint access patterns.

5.2.5 Improved Hardware Integration

- Future versions of the system should support direct integration with high-resolution live fingerprint scanners, reducing dependence on image uploads.

- Exploring contactless biometric authentication (such as 3D touchless fingerprinting) can improve hygiene and usability in high-risk medical environments.

5.2.6 Multilingual & User-Friendly Enhancements

- Expanding language support in the user interface can increase accessibility, especially for underserved populations.
- Voice-assisted navigation could help non-technical users operate the system more intuitively.

APPENDIX A:

CODE & SNAPSHOTS

The following sections contain source code excepts and snapshots from the running project:

1. Backend Code (Flask API):

- app.py: Contains endpoints for patient registration and fingerprint scanning.

```
from flask import Flask, request, jsonify
from flask_cors import CORS
from werkzeug.utils import secure_filename
import os
import json
import cv2
app = Flask(__name__)
CORS(app)
UPLOAD_FOLDER = 'uploads'
REPORT_FOLDER = 'reports'
PATIENT_DB = 'patients.json'
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
os.makedirs(REPORT_FOLDER, exist_ok=True)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

# Register patient
@app.route('/register', methods=['POST'])
def register():
    try:
        name = request.form['name']
        gender = request.form['gender']
        age = request.form['age']
        weight = request.form['weight']
        mobile = request.form['mobile']
        email = request.form['email']
        blood_group = request.form['blood_group']
```

```
allergic = request.form['allergic']
history = request.form['history']

# Save fingerprint image
fingerprint_file = request.files['fingerprint']
fingerprint_filename = secure_filename(fingerprint_file.filename)
fingerprint_path = os.path.join(UPLOAD_FOLDER, fingerprint_filename)
fingerprint_file.save(fingerprint_path)

# Save optional report
report_filename = ""
if 'report' in request.files and request.files['report'].filename:
    report_file = request.files['report']
    report_filename = secure_filename(report_file.filename)
    report_path = os.path.join(REPORT_FOLDER, report_filename)
    report_file.save(report_path)

# Save to JSON
if os.path.exists(PATIENT_DB):
    with open(PATIENT_DB, 'r') as f:
        patients = json.load(f)
else:
    patients = []

patients.append({
    "name": name,
    "gender": gender,
    "age": age,
    "weight": weight,
    "mobile": mobile,
    "email": email,
    "blood_group": blood_group,
    "allergic": allergic,
    "history": history,
```

```
            "fingerprint": fingerprint_filename,
            "report": report_filename
        })

        with open(PATIENT_DB, 'w') as f:
            json.dump(patients, f, indent=2)

        return jsonify({"message": f"✅ Patient '{name}' registered successfully."})

    except Exception as e:
        return jsonify({"error": str(e)}), 500

# Match fingerprint using ORB
def match_fingerprint(uploaded_path, stored_path):
    img1 = cv2.imread(uploaded_path, 0)
    img2 = cv2.imread(stored_path, 0)
    if img1 is None or img2 is None:
        return False

    orb = cv2.ORB_create()
    kp1, des1 = orb.detectAndCompute(img1, None)
    kp2, des2 = orb.detectAndCompute(img2, None)

    if des1 is None or des2 is None:
        return False

    bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    matches = bf.match(des1, des2)

    good_matches = [m for m in matches if m.distance < 30]  # More strict

    print(f"Total matches: {len(matches)}, Good matches: {len(good_matches)}")

    return len(good_matches) > 40  # Require more and better matches
```

```
# Scan route
@app.route('/scan', methods=['POST'])
def scan():
    if 'fingerprint' not in request.files:
        return jsonify({"error": "No fingerprint file uploaded"}), 400

    file = request.files['fingerprint']
    filename = secure_filename(file.filename)
    uploaded_path = os.path.join(UPLOAD_FOLDER, filename)
    file.save(uploaded_path)

    try:
        with open(PATIENT_DB, 'r') as f:
            patients = json.load(f)
    except:
        patients = []

    for patient in patients:
        stored_path = os.path.join(UPLOAD_FOLDER, patient['fingerprint'])
        if os.path.exists(stored_path) and match_fingerprint(uploaded_path, stored_path):
            return jsonify({
                "name": patient['name'],
                "gender": patient['gender'],
                "age": patient['age'],
                "weight": patient['weight'],
                "mobile": patient['mobile'],
                "email": patient['email'],
                "blood_group": patient['blood_group'],
                "allergic": patient['allergic'],
                "history": patient['history'],
                "report": patient.get('report', '')
            })
    return jsonify({"error": "No match found"}), 404
```

```
if __name__ == '__main__':
    app.run(debug=True).
```

- Patient.json

```
{
  "name": "Ravi Kumar",
  "age": 28,
  "weight": 65,
  "mobile": "1234567890",
  "email": "ravi@example.com",
  "blood_group": "B+",
  "allergic": "Yes",
  "history": "Diabetic. Uses insulin.",
  "fingerprint": "uploads/"
}
```

- Biometric.py

```
def scan():
  with open('sample_fingerprint.bin', 'rb') as f:
 return f.read()
```

- db_config.py

```
import cx_Oracle
def connect_db():
   return cx_Oracle.connect(
      user="system",
      password="hr",
      dsn="localhost/XEPDB1"
   )
```

- matcher.py: Modules for biometric processing, database connectivity, and fingerprint matching respectively.

```
import cv2
import os
def match_fingerprints(scan_path, folder_path):
    # Load the scanned image and convert to grayscale
    scan_img = cv2.imread(scan_path, cv2.IMREAD_GRAYSCALE)
    if scan_img is None:
        return None
    sift = cv2.SIFT_create()
    kp1, des1 = sift.detectAndCompute(scan_img, None)

    best_match_name = None
    max_matches = 0

    for filename in os.listdir(folder_path):
        if filename == "temp_scan.png":
            continue

        file_path = os.path.join(folder_path, filename)
        img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
        if img is None:
            continue

        kp2, des2 = sift.detectAndCompute(img, None)
        if des2 is None:
            continue

        # BFMatcher with default params
        bf = cv2.BFMatcher()
        matches = bf.knnMatch(des1, des2, k=2)
```

```
        # Apply ratio test
        good_matches = []
        for m, n in matches:
            if m.distance < 0.75 * n.distance:
                good_matches.append(m)

        if len(good_matches) > max_matches:
            max_matches = len(good_matches)
            best_match_name = filename.replace(".png", "")

    # Return the best matched patient name
    return best_match_name if max_matches > 10 else None
```

2.**Frontend Code:**

- index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Biometric Patient Record System</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body class="bg-light d-flex align-items-center" style="height:100vh;">
  <div class="container text-center">
    <div class="card shadow-lg p-4">
      <h1 class="mb-4 text-primary">👨‍⚕️ Biometric Patient Record System</h1>
      <p class="lead">Access patient records quickly and securely using fingerprint
verification.</p>
      <div class="mt-4">
```

```
        <a href="register.html" class="btn btn-success btn-lg me-3">📝 Register New
Patient</a>
        <a href="scan.html" class="btn btn-info btn-lg">🔍 Scan Fingerprint</a>
       </div>
      </div>
     </div>
    </body>
    </html>
```

- register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Register Patient</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
  <style>
    .was-validated .form-control:invalid,
    .was-validated .form-select:invalid {
      border-color: #dc3545;
    }
    .was-validated .form-control:valid,
    .was-validated .form-select:valid {
      border-color: #28a745;
    }
  </style>
```

```
</head>
<body class="bg-light">
  <div class="container mt-5">
    <div class="card shadow-lg">
      <div class="card-header bg-success text-white">
        <h3 class="mb-0">📝 New Patient Registration</h3>
      </div>
      <div class="card-body">
        <form id="registerForm" class="needs-validation" novalidate enctype="multipart/form-data">
          <div class="row">
            <div class="col-md-6 mb-3">
              <input type="text" name="name" pattern="^[A-Za-z\s]+$" class="form-control" placeholder="Full Name" required>
              <div class="invalid-feedback">Please enter a valid name (letters only).</div>
            </div>
            <div class="col-md-6 mb-3">
              <select name="gender" class="form-control" required>
                <option value="">Select Gender</option>
                <option value="Male">Male</option>
                <option value="Female">Female</option>
                <option value="Other">Other</option>
              </select>
              <div class="invalid-feedback">Please select a gender.</div>
            </div>
```

```
<div class="col-md-6 mb-3">
  <input type="number" name="age" class="form-control" placeholder="Age" min="1" max="120" required>
  <div class="invalid-feedback">Enter valid age (1-120).</div>
</div>
<div class="col-md-6 mb-3">
  <input type="number" name="weight" class="form-control" placeholder="Weight (kg)" min="1" max="300" required>
  <div class="invalid-feedback">Enter valid weight (1-300 kg).</div>
</div>
<div class="col-md-6 mb-3">
  <input type="text" name="mobile" pattern="[0-9]{10}" class="form-control" placeholder="Mobile Number" required>
  <div class="invalid-feedback">Enter a 10-digit mobile number.</div>
</div>
<div class="col-md-6 mb-3">
  <input type="email" name="email" class="form-control" placeholder="Email" required>
  <div class="invalid-feedback">Enter a valid email address.</div>
</div>
<div class="col-md-6 mb-3">
  <input type="text" name="blood_group" class="form-control" placeholder="Blood Group" required>
  <div class="invalid-feedback">Enter your blood group.</div>
</div>
<div class="col-md-6 mb-3">
```

```
          <input type="text" name="allergic" class="form-control" placeholder="Allergic? Yes/No" required>
          <div class="invalid-feedback">Specify if allergic or not.</div>
        </div>
        <div class="col-md-12 mb-3">
          <textarea name="history" class="form-control" placeholder="Medical History" rows="3" required></textarea>
          <div class="invalid-feedback">Enter medical history.</div>
        </div>
        <div class="col-md-6 mb-3">
          <label class="form-label">Fingerprint Image (required):</label>
          <input type="file" name="fingerprint" class="form-control" accept="image/*" required>
          <div class="invalid-feedback">Upload fingerprint image.</div>
        </div>
        <div class="col-md-6 mb-3">
          <label class="form-label">Medical Report (optional):</label>
          <input type="file" name="report" class="form-control" accept=".pdf,image/*">
        </div>
      </div>
      <button type="submit" class="btn btn-success">Register Patient</button>
    </form>
    <div class="mt-3">
      <a href="scan.html" class="btn btn-link">Already registered? Scan fingerprint ➤</a>
    </div>
    <pre id="result" class="mt-3 bg-light p-2 border rounded"></pre>
```

```
    </div>
  </div>
</div>

<script>
  // Bootstrap validation
  (() => {
    const form = document.getElementById('registerForm');
    form.addEventListener('submit', async (event) => {
      if (!form.checkValidity()) {
        event.preventDefault();
        event.stopPropagation();
        form.classList.add('was-validated');
        return;
      }

      event.preventDefault();
      const formData = new FormData(form);
      try {
        const res = await fetch('http://localhost:5000/register', {
          method: 'POST',
          body: formData
        });
        const data = await res.json();
        document.getElementById('result').innerText = JSON.stringify(data, null, 2);
```

```
    } catch (err) {
      document.getElementById('result').innerText = '❌ Error: ' + err.message;
    }
  });
})();
</script>
</body>
</html>
```

- scan.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Scan Fingerprint</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body class="bg-light">
  <div class="container mt-5">
    <div class="card shadow-lg">
      <div class="card-header bg-info text-white">
        <h3 class="mb-0">🔍 Patient Fingerprint Scan</h3>
      </div>
      <div class="card-body">
        <form id="scanForm" enctype="multipart/form-data">
          <div class="mb-3">
            <label class="form-label">Upload Fingerprint Image:</label>
            <input type="file" name="fingerprint" class="form-control" accept="image/*"
required>
          </div>
```

```
        <button type="submit" class="btn btn-info">Scan Now</button>
      </form>
      <div class="mt-3">
        <a href="register.html" class="btn btn-link">➤ New Patient? Register here</a>
      </div>
      <pre id="result" class="mt-4 p-3 bg-light border rounded"></pre>
    </div>
  </div>
</div>

<script>
  document.getElementById('scanForm').onsubmit = async (e) => {
    e.preventDefault();
    const form = new FormData(e.target);
    const res = await fetch('http://localhost:5000/scan', {
      method: 'POST',
      body: form
    });
    const data = await res.json();
    if (data.error) {
      document.getElementById('result').innerText = data.error;
    } else {
      localStorage.setItem("patient", JSON.stringify(data));
      window.location.href = "patient.html";
    }
  };
</script>
</body>
</html>
```

- patient.html: HTML/CSS/JavaScript files forming the user interfaces.

```
<!DOCTYPE html>
<html lang="en">
```

```
  <head>
    <meta charset="UTF-8">
    <title>Patient Details</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
    <style>
      body {
        background: linear-gradient(to right, #e0f7fa, #e0f2f1);
        font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
      }
      .card {
        border-radius: 15px;
      }
      .header {
        background-color: #00796b;
        color: white;
        border-top-left-radius: 15px;
        border-top-right-radius: 15px;
      }
      .field-label {
        font-weight: 500;
      }
      .section-title {
        margin-top: 20px;
        font-weight: bold;
        font-size: 1.2rem;
        border-bottom: 2px solid #00796b;
        display: inline-block;
      }
    </style>
  </head>
  <body>
    <div class="container my-5">
      <div class="card shadow-lg">
```

```
      <div class="card-header header text-center">
        <h2> 📃 Patient Profile</h2>
      </div>
      <div class="card-body">
        <div class="row g-4">
          <div class="col-md-6"><span class="field-label">Name:</span> <span
id="name"></span></div>
          <div class="col-md-6"><span class="field-label">Gender:</span> <span
id="gender"></span></div>
          <div class="col-md-6"><span class="field-label">Age:</span> <span
id="age"></span></div>
          <div class="col-md-6"><span class="field-label">Weight:</span> <span
id="weight"></span> kg</div>
          <div class="col-md-6"><span class="field-label">Mobile:</span> <span
id="mobile"></span></div>
          <div class="col-md-6"><span class="field-label">Email:</span> <span
id="email"></span></div>
          <div class="col-md-6"><span class="field-label">Blood Group:</span> <span
id="blood_group"></span></div>
          <div class="col-md-6"><span class="field-label">Allergic:</span> <span
id="allergic"></span></div>
          <div class="col-12">
            <div class="section-title">Medical History</div>
            <p id="history" class="mt-2"></p>
          </div>
          <div class="col-12">
            <div class="section-title">Medical Report</div>
            <p class="mt-2" id="reportSection"></p>
          </div>
        </div>
      </div>
    </div>
  </div>
```

```
  <script>
    const data = JSON.parse(localStorage.getItem("patient"));

    document.getElementById("name").innerText = data.name;
    document.getElementById("gender").innerText = data.gender;
    document.getElementById("age").innerText = data.age;
    document.getElementById("weight").innerText = data.weight;
    document.getElementById("mobile").innerText = data.mobile;
    document.getElementById("email").innerText = data.email;
    document.getElementById("blood_group").innerText = data.blood_group;
    document.getElementById("allergic").innerText = data.allergic;
    document.getElementById("history").innerText = data.history;

    if (data.report && data.report !== "") {
      document.getElementById("reportSection").innerHTML =
       '<a href="/reports/' + data.report + '" target="_blank" class="btn btn-outline-primary
btn-sm">View Report</a>';
    } else {
      document.getElementById("reportSection").innerHTML = '<span class="text-
muted">No report uploaded.</span>';
    }
  </script>
 </body>
 </html>
```

- script.js

```
 document.addEventListener("DOMContentLoaded", () => {
  const scanForm = document.getElementById("scanForm");
  const resultBox = document.getElementById("result");

  scanForm.onsubmit = async (e) => {
    e.preventDefault();
```

```
  const fileInput = document.getElementById("fingerprint");
  if (!fileInput.files[0]) {
    resultBox.innerText = " ❗ Please upload a fingerprint image.";
    return;
  }

  const formData = new FormData();
  formData.append("fingerprint", fileInput.files[0]);

  try {
    const res = await fetch("http://localhost:5000/scan", {
      method: "POST",
      body: formData,
    });

    const data = await res.json();

    if (data.error) {
      resultBox.innerText = "❌ " + data.error;
    } else {
      // Store matched patient info and redirect
      localStorage.setItem("patient", JSON.stringify(data));
      window.location.href = "patient.html";
    }

  } catch (err) {
    resultBox.innerText = "❌ Server error or connection issue.";
    console.error("Scan error:", err);
  }
 };
});
```

- style.css

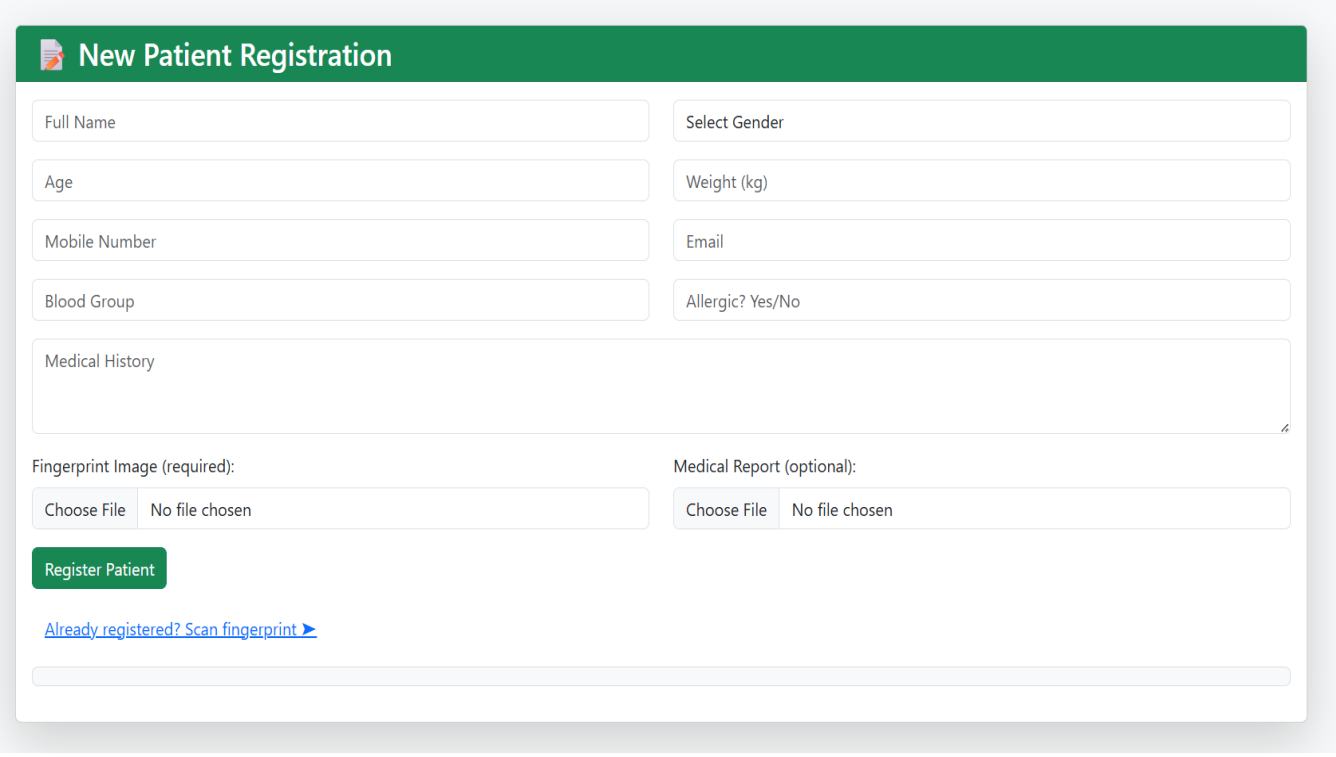
```
body {
  margin: 0;
  padding: 0;
  font-family: 'Segoe UI', sans-serif;
  background: linear-gradient(to right, #ddeefd, #ffffff);
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
}

.form-card {
  background: #ffffff;
  padding: 30px;
  width: 100%;
  max-width: 500px;
  border-radius: 12px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.15);
}

h2 {
  text-align: center;
  color: #2c3e50;
  margin-bottom: 20px;
}

form input, form textarea {
  width: 100%;
  padding: 10px;
  margin: 10px 0;
  font-size: 14px;
  border-radius: 6px;
  border: 1px solid #ccc;
```

```
  box-sizing: border-box;
}

button {
  width: 100%;
  background: #007BFF;
  color: white;
  padding: 12px;
  font-size: 16px;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  margin-top: 10px;
  transition: 0.3s ease;
}

button:hover {
  background: #0056b3;
}

pre {
  background: #f4f4f4;
  padding: 10px;
  margin-top: 20px;
  border-radius: 6px;
  font-size: 13px;
  overflow-x: auto;
}

.link-section {
  text-align: center;
  margin-top: 15px;
}
```

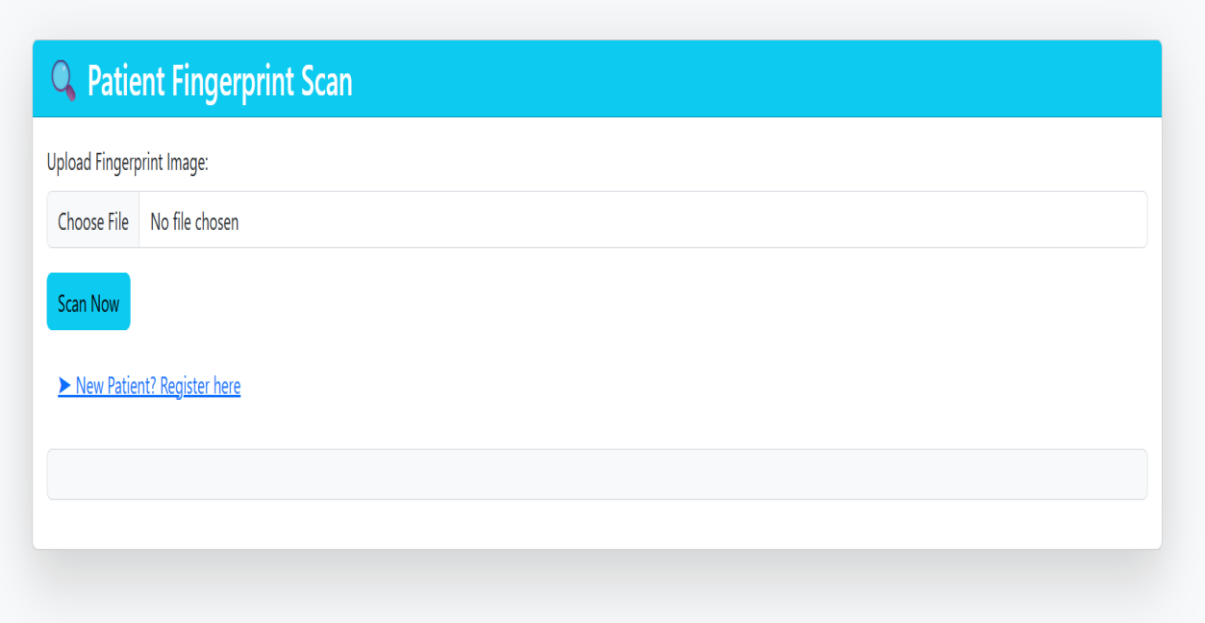
```
.link-section a {
  text-decoration: none;
  color: #007BFF;
  font-size: 14px;
}
```

3.**Sample Snapshot:**

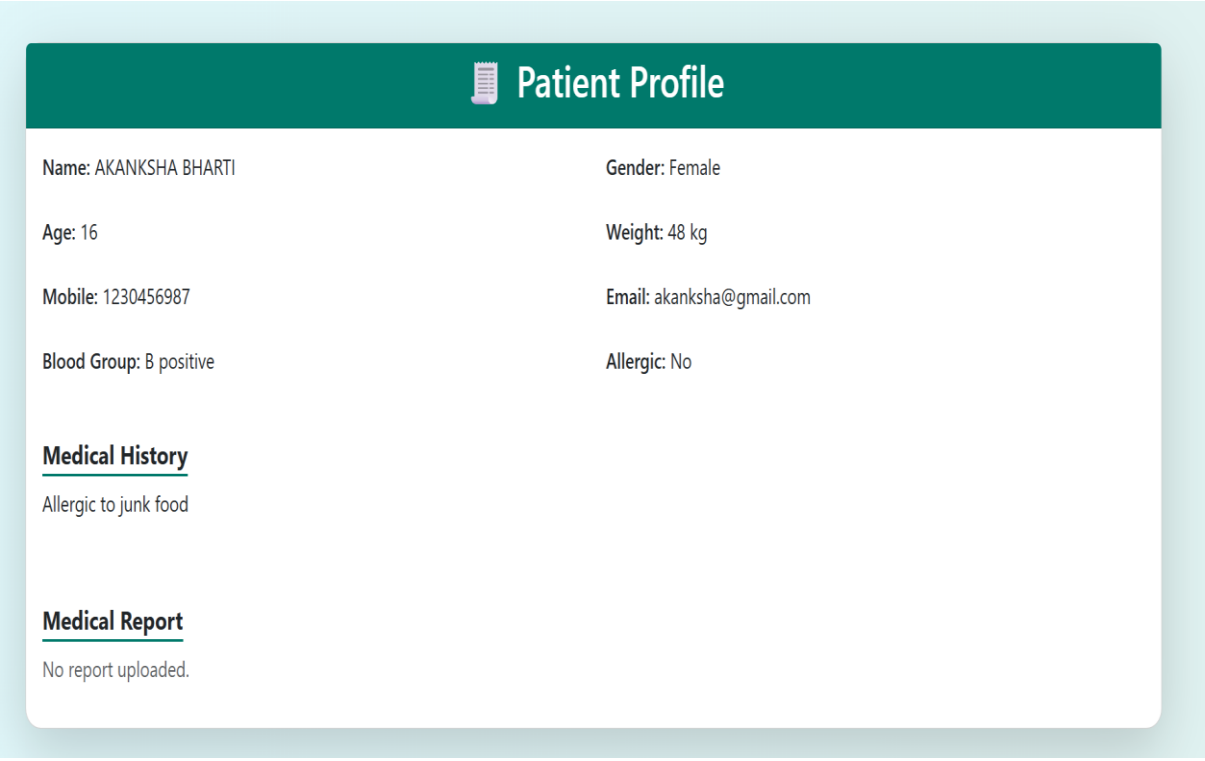
- Snapshot of Registration Page



- Snapshot of Fingerprint Scanning Page



- Snapshot after matching fingerprint



REFERENCES

[1] S. Jayanthy, J. B. Anishkka, A. Deepthi and E. Janani, "Facial Recognition And Verification System For Accessing Patient Health Records," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), 2019, pp. 1266-1271, doi: 10.1109/ICCS45141.2019.9065469.

[2] OpenCV.org, "OpenCV Documentation," [Online]. Available: https://docs.opencv.org/. [Accessed: Jun. 15, 2025].

[3] Daesung,Moon,Yong Wha,Chung,Sung, Bum Pan, Jin Won Park, Integrating fingerprint verification into smart card-based healthcare information system intelligence, Computer methods and programs in medicine, 81 (1), pp.66-78.2006

[4] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle, "An identity-authentication system using fingerprints," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1365–1388, Sep. 1997, doi: 10.1109/5.628674.

[5] R. M. Bolle, J. H. Connell, S. Pankanti, N. K. Ratha, and A. W. Senior, *Guide to Biometrics*, Springer, 2004.

[6] T. D. L. Nguyen and M. H. Nguyen, "An efficient biometric authentication method for patient medical data in cloud computing," in *Proc. IEEE Asia-Pacific Conference on Information Systems (APCIS)*, 2019, pp. 1–6, doi: 10.1109/APCIS.2019.00006.

[7] K. J. Doughty, "Medical database systems for biometric information," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 98–104, May–Jun. 2001, doi: 10.1109/51.922719.

[8] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed., Springer, 2009.

[9] M. A. Rahman, M. A. Hossain, and M. A. Islam, "A secure and privacy-preserving framework for patient health record sharing using blockchain and biometrics," in *Proc. IEEE Int. Conf. Smart Health*, 2020, pp. 1–7.

[10] S. M. Sheikh, M. D. Shaikh, F. K. Sayyad, A. M. Mujawar, and P. K. Birajdar, "Biometric based patient health record system," *International Research Journal of Engineering and Technology (IRJET)*, vol. 9, no. 6, pp. 2448–2452, Jun. 2022. [Online]. Available: https://www.irjet.net/archives/V9/i6/IRJET-V9I6616.pdf

[11] N. Patil, D. Patil, A. Choudhary, A. Jadhav, and A. Wagh, "Biometric based patient health record system," *International Journal of Creative Research Thoughts (IJCRT)*, vol. 10, no. 2, pp. 88–92, Feb. 2022. [Online]. Available: https://ijcrt.org/papers/IJCRTAF02013.pdf

[12] World Health Organization, "Patient Safety: Global Action on Patient Safety," 2021. [Online]. Available: https://www.who.int/publications/i/item/9789240032705