A

Mini Project Report

On

PERSONALIZED CANCER PREDICTION AND DIAGNOSIS USING MACHINE LEARNING ALGORITHMS

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

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING (AI&ML)

By

Ananth Dev Prajapat (227R1A7368)

S.Sreekar Reddy (227R1A73C6)

T.Ajay (227R1A73C7)

Under the Guidance of

G.PARVATHI DEVI

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, NewDelhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2022-2026

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)

CERTIFICATE

This is to certify that the project entitled "**PERSONALISED CANCER DIAGNOSIS USING MACHINE LEARNING**" being submitted **by** Ananth Dev Prajapat **(227R1A7368), S.Sreekar reddy (227R1A73C6) and T.Ajay (227R1A73C7)**in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering (AI&ML) to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2025-26.

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

Ms. G.PARVATHIDEVI
Assistant Professor
INTERNAL GUIDE

Dr. S Rao Chintalapudi
HOD CSE(AI&ML)

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ____

ACKNOWLEDGEMENT

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

We take this opportunity to express my profound gratitude and deep regard to my guide **Ms.G.PARVATHI DEVI,** Assistant Professor for his exemplary guidance, monitoringand constant encouragement throughout the project work. The blessing, help and guidancegiven by him shall carry us a long way in the journey of life on which we are about to embark.

We are also thankful to **DR. S RAO CHINTALAPUDI,** Head, Department of Computer Science and Engineering (AI&ML) for providing encouragement and support for completingthis project successfully.

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

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

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

Ananth Dev Prajapat **(227R1A7368)**
S.Sreekar Reddy (227R1A73C6)
Ajay (227R1A73C7)

ABSTRACT

Cancer is one of the leading causes of death globally, and early detection plays a crucial role in improving survival rates. Traditional diagnostic methods are time-consuming, expensive, and often require extensive medical expertise. To address these challenges, this project proposes a machine learning-based system for the personalized prediction and diagnosis of cancer.

The system utilizes historical patient data and various clinical parameters to train a predictive model. After preprocessing and feature engineering, algorithms such as Logistic Regression, Random Forest, and Support Vector Machines are applied. The trained model predicts the likelihood of a patient being affected by cancer based on the input symptoms and clinical values.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

The scope of this project is to develop an intelligent, machine learning-based system that predicts and diagnoses cancer based on patient-specific input data. It focuses on implementing a personalized approach to prediction using clinical and demographic attributes, enabling early detection and improving diagnostic accuracy. The project includes data preprocessing, model training, and integration into a terminal-based interface for real-time predictions. The system is designed to be scalable, user-friendly, and adaptable to different cancer types with minimal modification, supporting both clinical professionals and patients in early decision-making.

1.2 PROJECT PURPOSE

The purpose of this project is to build a reliable and efficient system for the early prediction and diagnosis of cancer using machine learning techniques. By leveraging patient data and advanced algorithms, the system aims to support healthcare professionals in making accurate and timely decisions. It minimizes the dependency on expensive diagnostic tests and manual evaluations, thereby making early detection more accessible and cost-effective. Ultimately, the project seeks to improve patient outcomes by enabling proactive medical intervention and personalized healthcare.

1.3 PROJECT FEATURES

The project offers a range of features designed to ensure accurate and user-friendly cancer prediction. It includes a terminal-based interface for easy interaction, where users can input clinical data to receive immediate prediction results. The system uses a trained machine learning model to provide high-accuracy predictions based on patient-specific parameters. Key features include automated data preprocessing, personalized risk assessment, real-time prediction, and the ability to handle missing or incomplete data gracefully

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

System analysis is a crucial phase in system development, where the existing system is thoroughly examined to identify problems, inefficiencies, and areas for improvement. This involves gathering information through various methods like interviews, surveys, and document reviews to understand the system's processes, data flows, and user needs. The goal is to define the requirements for a new or enhanced system that addresses the identified shortcomings and meets the organization's objectives..

2.1 PROBLEM DEFINITION

The primary problem addressed is the initial and most crucial step in system analysis, where the specific issues or shortcomings of the existing system are clearly and concisely identified. It involves understanding the root causes of these problems, their impact on the organization, and the objectives that a new or improved system should achieve. A well-defined problem statement provides a clear focus for the subsequent analysis and development stages.

2.2 EXISTING SYSTEM

The existing system, in the context of the Cancer Detection System, likely involves a combination of manual processes and traditional diagnostic methods currently employed in healthcare settings. Patient data, including medical history, symptoms, and physical examination results, are typically recorded and managed in paper-based files or within Electronic Health Record (EHR) systems. Diagnostic procedures such as blood tests, urine analysis, imaging scans (CT, MRI), and biopsies are conducted individually, often requiring patients to visit multiple facilities and wait for extended periods to receive results.

The process of cancer staging, which determines the extent and severity of the disease, relies heavily on the interpretation of these diagnostic results by medical professionals. This process can be time-consuming, subjective, and prone to variability depending on the expertise and experience of the clinicians involved. Similarly, the formulation of treatment plans is a complex decision-making process that considers various factors, including the stage of the cancer, patient's overall health, and available treatment options. Currently, this process depends on

established medical guidelines and the clinical judgment of oncologists.

However, the existing system may suffer from several limitations. The fragmented nature of data collection and analysis can lead to delays in diagnosis and treatment initiation. The reliance on manual processes increases the risk of errors, data loss, and inconsistencies. The subjective interpretation of diagnostic results can result in variations in staging and treatment decisions, potentially impacting patient outcomes.

Furthermore, the existing system may not effectively utilize the wealth of data generated during the diagnostic process. Valuable information that could contribute to more accurate predictions and personalized treatment plans may be underutilized or difficult to access. This limits the ability to identify patterns, predict disease progression, and tailor interventions to individual patients..

2.2.1 LIMITATIONS OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- The system is not implemented machine learning algorithm and ensemble learning.
- The system is not implemented Reverse Engineering Applications characteristics.

2.3 PROPOSED SYSTEM

The proposed Cancer Detection System aims to address the limitations of the existing system by providing a more integrated, efficient, and data-driven approach to cancer diagnosis, staging, and treatment planning. The system will leverage advanced technologies such as machine learning, data analytics, and a centralized database to streamline processes, improve accuracy, and personalize patient care. It will consist of several key components, including a user-friendly interface for data input, a robust database for storing and managing patient information, and sophisticated algorithms for predicting cancer stage and recommending optimal treatment strategies.

A core feature of the proposed system is its ability to integrate data from various sources, including patient medical history, laboratory results, imaging scans, and genetic testing. By consolidating this information into a centralized repository, the system will provide a comprehensive view of each patient's condition, enabling healthcare professionals to make more informed decisions. Machine learning algorithms will be employed to analyze this integrated data, identify patterns, and predict the likelihood of cancer progression.

The system will also incorporate a decision support tool to assist in cancer staging and treatment planning. Based on the predicted stage and other relevant factors, the tool will provide evidence-based recommendations for the most appropriate treatment options, such as surgery, chemotherapy, radiation therapy, or a combination of these. This tool will not replace the expertise of oncologists but will serve as a valuable resource to guide their clinical judgment and ensure consistency in treatment decisions.

Furthermore, the proposed system will enhance communication and collaboration among healthcare providers involved in the patient's care. The centralized database will allow authorized personnel to access patient information securely and in real-time, facilitating seamless coordination of diagnostic procedures, treatment administration, and follow-up care. This will reduce delays, minimize errors, and improve the overall patient experience.

2.3.1 PROPOSED APPROACH

The proposed approach for the Cancer Detection System centers on the development of a comprehensive, integrated platform that leverages advanced technologies to improve the accuracy, efficiency, and personalization of cancer care. This approach involves several key strategies, including the consolidation of patient data, the application of machine learning algorithms, the implementation of a decision support system, and the enhancement of communication and collaboration among healthcare providers.

First, the system will integrate data from diverse sources, such as electronic

health records (EHRs), laboratory information systems (LIS), picture archiving and communication systems (PACS), and genomic databases. This will create a centralized repository of patient information, providing a holistic view of each individual's condition. This integration will eliminate data silos, reduce redundancy, and ensure that all relevant information is readily accessible to authorized personnel.

Second, machine learning algorithms will be developed and deployed to analyze the integrated patient data. These algorithms will be trained to identify patterns, predict cancer stage, and assess the risk of disease progression. By leveraging the power of artificial intelligence, the system will be able to extract valuable insights from complex datasets, leading to more accurate and timely diagnoses.

Third, a decision support system (DSS) will be implemented to assist healthcare professionals in making informed treatment decisions. The DSS will provide evidence-based recommendations for treatment options, tailored to the individual patient's characteristics and predicted cancer stage. This will help to standardize care, reduce variability in treatment decisions, and ensure that patients receive the most appropriate interventions.

2.3.2 ADVANTAGES OF THE PROPOSED APPROACH

- Improved efficiency: Streamlines workflows, reduces manual data entry, and accelerates access to critical patient information.
- Enhanced accuracy: Leverages machine learning to provide more precise diagnoses and staging, minimizing errors.
- Personalized care: Enables tailored treatment plans based on individual patient data and predictive analytics.
- Better collaboration: Facilitates seamless communication and information sharing among healthcare professionals.
- Reduces variability: Standardizes treatment approaches, leading to more consistent and predictable outcomes.

- Improves patient outcomes: Ultimately contributes to better survival rates and quality of life for cancer patients..

2.4 HARDWARE & SOFTWARE REQUIREMENTS

2.4.1 HARDWARE REQUIREMENTS:

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

PROCESSOR	: Intel – i3
RAM	: 4GB
HARD DISK	: 20 GB
KEYBOARD	: Standard Windows Keyboard
MOUSE	: Two or Three Button Mouse
MONITOR	: SVGA0
NETWORK	: Gigabit Ethernet

2.4.2 SOFTWARE REQUIREMENTS:

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

OPERATING SYSTEM	: Windows 10
CODE LANGUAGE	: Python
FRONT-END	: HTML, CSS, JAVASCRIPT
BACK-END	: Flask
DESIGNING	: HTML, CSS, JavaScript
DATABASE	: MySQL

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture illustrated in Fig. 3.1 the procedure followed for classification,starting from input to final prediction

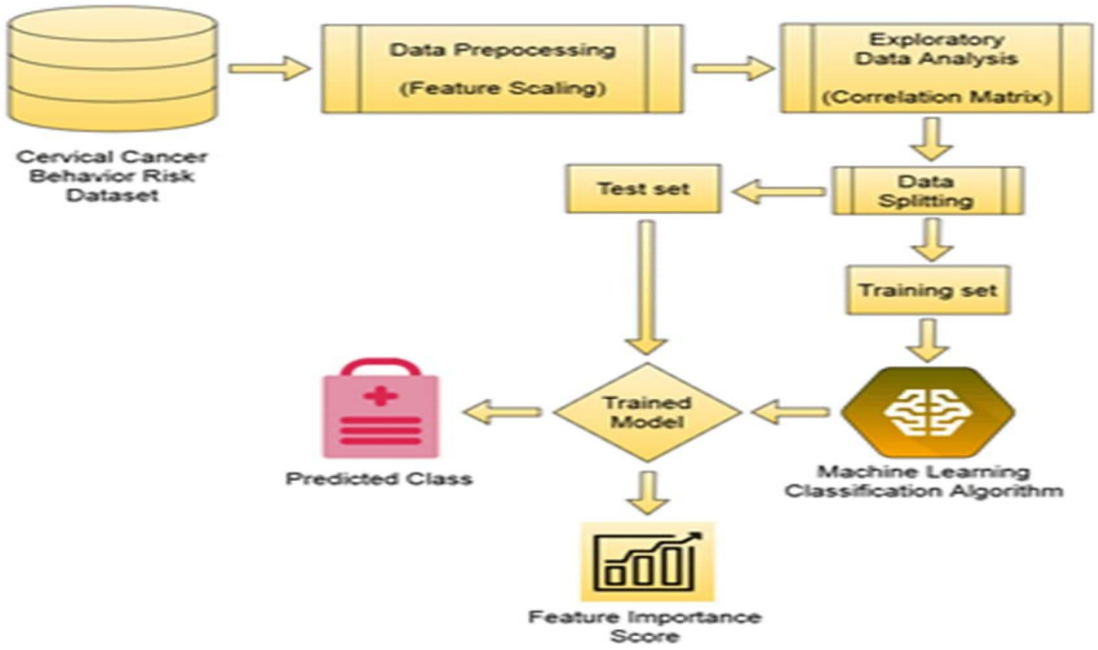

Figure 3.1: ARCHITECTURE FOR PERSONALISED CANCER DIAGNOSIS USING MACHINE LEARNING

DESCRIPTION

The Cancer Detection System is structured using a multi-layered architecture, a design choice that promotes scalability, maintainability, and efficient data processing. This approach separates the system's functionalities into distinct layers, enhancing modularity and simplifying the development process. The topmost layer, the Presentation Layer, is responsible for the user interface, built using HTML, CSS, and React. It provides the means for users, including doctors, medical staff, and patients, to interact with the system, input data, and view results.

The Application Layer forms the core of the system, handling the application's logic and functionality. Developed using Python and the Flask framework, it acts as an intermediary between the Presentation Layer and the Data Layer. This layer processes user requests, interacts with the database, and implements the crucial algorithms for cancer stage prediction and treatment plan generation. It also applies machine learning models to analyze data and provide intelligent insights, managing the application's workflow and business rules.

The Data Layer is responsible for the storage and management of the system's data. A relational database management system, such as MySQL, is employed to ensure data persistence, integrity, and security. This layer stores patient demographics, medical history, and diagnostic results, along with data related to cancer stages, treatment options, and medical knowledge. It provides efficient mechanisms for data retrieval and manipulation, ensuring data consistency and implementing security measures like access control and data encryption.

Finally, the Machine Learning Layer is responsible for the intelligent functions of the system. Working in conjunction with the Application Layer, it leverages Python and relevant libraries like scikit-learn and TensorFlow. This layer is crucial for training machine learning models to predict cancer stages based on patient data and for developing models that suggest optimal treatment plans.

3.2 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model.

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of usersthe system has. The use cases are represented by either circles or ellipses. The actors are often shown asstick figures.

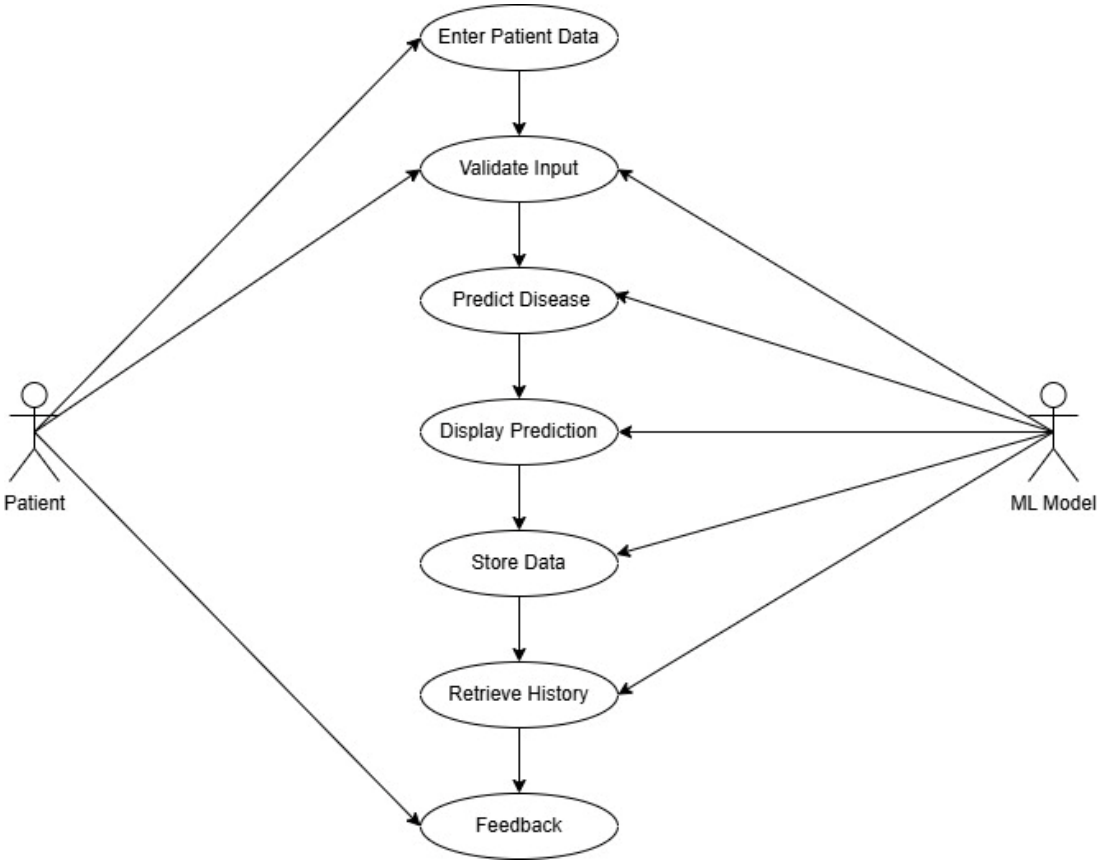

Figure 3.2: Use Case Diagram for personalised cancer diagnosis using machine learning

DESCRIPTION

A Use Case Diagram is a visual tool used in software development to illustrate how users interact with a system. It provides a high-level overview of the system's functionality from the user's perspective, focusing on what the system does, rather than how it does it. This makes it valuable for gathering requirements, designing the system, and facilitating communication between stakeholders.

The diagram consists of several key elements. Actors represent the users or external entities that interact with the system. Use cases depict the specific tasks or goals that actors can achieve through these interactions. Relationships, such as associations, show how actors participate in use cases, while include and extend relationships illustrate mandatory or optional parts of a use case. The system boundary defines the scope of the system, separating internal use cases from external actors.

Creating a Use Case Diagram involves identifying actors, use cases, and their relationships, and then representing them visually. This process helps to define the system's functionality, clarify user needs, and guide the development process. By focusing on user goals, Use Case Diagrams ensure that the system is designed to be user-friendly and effective.

In summary, a Use Case Diagram is a powerful tool for modeling system behavior, enhancing communication, and ensuring that the final product meets user requirements. Its emphasis on the user's perspective makes it a crucial part of the software development lifecycle.

3.2 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structureof a system by showing the system's classes, their attributes, operations (or methods),and the relationships among objects.

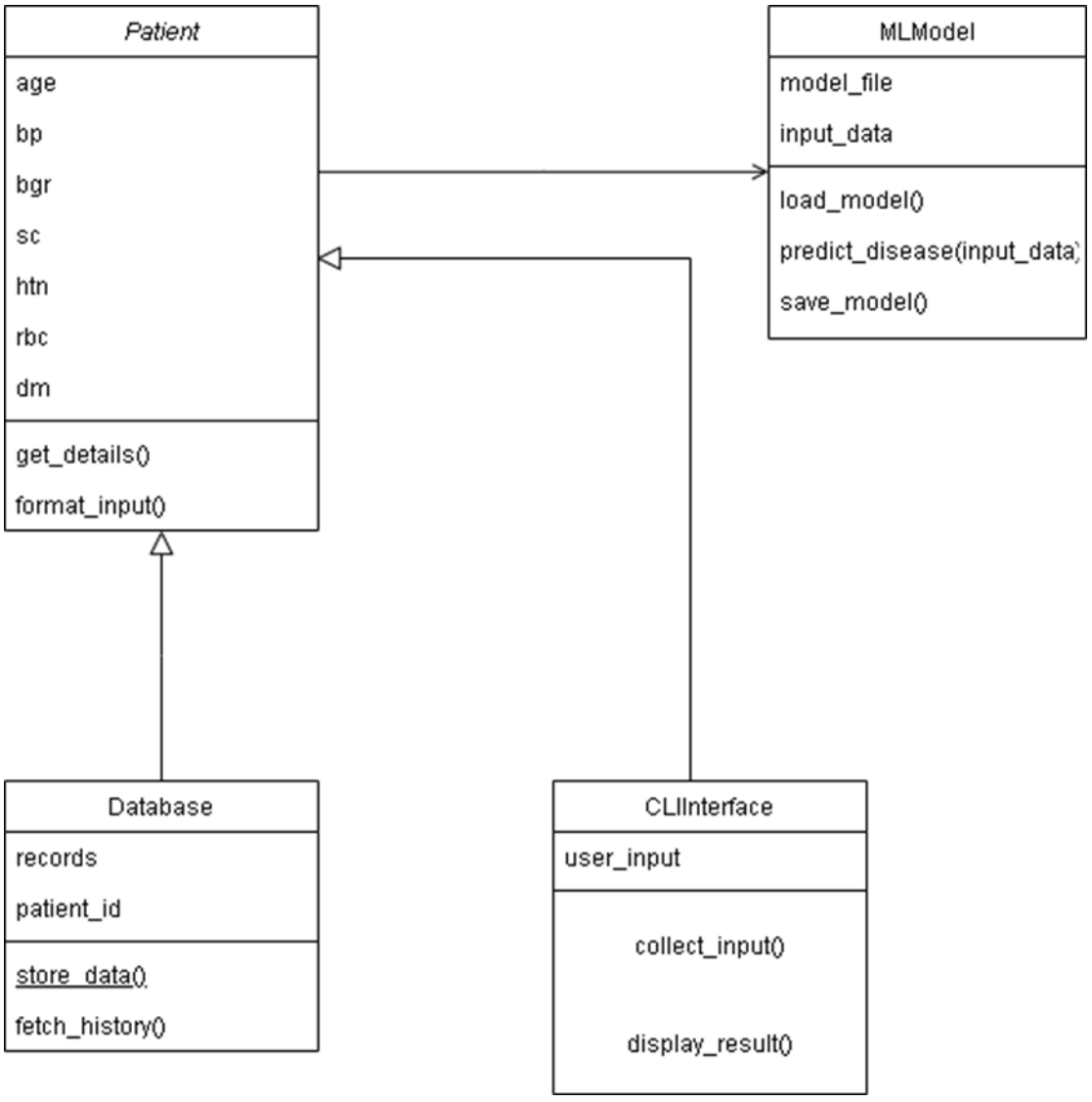

Figure 3.3: Class Diagram for personalised cancer diagnosis using machine learning

DESCRIPTION

A Class Diagram is a fundamental type of structural diagram in the Unified Modeling Language (UML) that visualizes the static structure of a system. It illustrates the system's classes, their attributes, and the relationships between them. Class diagrams are essential for object-oriented design, as they provide a blueprint of the system's components and how they interact.

In a Class Diagram, a class represents a blueprint for creating objects, which are instances of the class. Each class is depicted as a rectangle divided into three sections: the top section contains the class name, the middle section lists the class's attributes (data members), and the bottom section shows the class's operations (methods or functions). Attributes define the data held by objects of the class, while operations define the actions that objects of the class can perform.

Relationships between classes are represented by various types of lines and arrows. Common relationships include: Association (a general relationship), Aggregation (a "has-a" relationship where one class is part of another), Composition (a stronger "has-a" relationship where the parts cannot exist independently of the whole), Generalization (an "is-a" relationship representing inheritance), and Dependency (a using relationship). These relationships show how classes interact and depend on each other.

By providing a clear and concise representation of the system's static structure, Class Diagrams facilitate communication among developers, analysts, and stakeholders. They serve as a crucial tool for understanding, designing, and documenting object-oriented systems, ensuring that the system's components are well-defined and their interactions are clearly understood.

3.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts theobjects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associatedwith use case realizations in the logical view of the system under development.

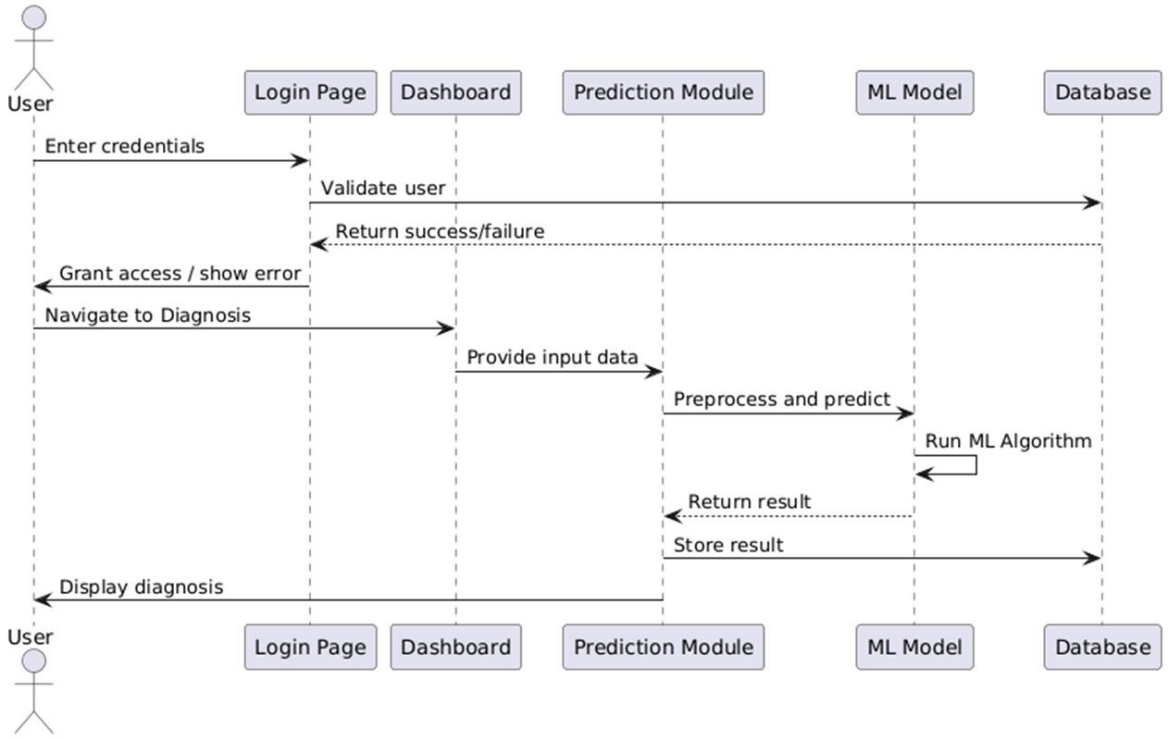

Figure 3.4: Sequence Diagram for personalised cancer diagnosis using machine learning

DESCRIPTION

A Sequence Diagram is a type of interaction diagram within the Unified Modeling Language (UML) that visualizes the flow of interactions between objects in a system over time. It focuses on illustrating how objects interact and in what sequence these interactions occur, making it a valuable tool for understanding the dynamic behavior of a system. Sequence diagrams are particularly useful for detailed design and documenting how different parts of a system collaborate to achieve a specific use case.

The key elements of a Sequence Diagram include: objects, represented by vertical rectangles or boxes, and lifelines, which are vertical dashed lines extending from each object to represent the object's existence over time. Interactions between objects are shown as arrows, called messages, that are drawn between the lifelines. These messages represent method calls or signals exchanged between objects, and they are labeled with the name of the method or signal.

Time progresses from top to bottom in a Sequence Diagram, so the order of messages indicates the sequence of interactions. Various types of messages can be depicted, including synchronous messages (where the sender waits for a response), asynchronous messages (where the sender does not wait), and reply messages (showing data being returned). Additional notations, such as activation boxes, can be used to indicate when an object is active and processing a message.

By illustrating the sequence of interactions between objects, Sequence Diagrams provide a clear and detailed view of the dynamic behavior of a system. They help developers to understand how different objects collaborate to achieve a specific functionality, identify potential design issues, and document the system's runtime behavior .

4.IMPLEMENTATION

4.1 DATASET DESCRIPTION

The dataset used in this project is primarily sourced from publicly available cancer-related datasets such as the Breast Cancer Wisconsin (Diagnostic) dataset available from the UCI Machine Learning Repository. It comprises medical records and diagnostic features collected from patients, focusing on various measurements derived from digitized images of fine needle aspirates (FNA) of breast masses. These features include mean radius, texture, perimeter, area, smoothness, and more, which are vital indicators in cancer detection.

The dataset contains a combination of numerical and categorical features. Each record includes a unique patient ID, diagnostic result (Benign or Malignant), and 30 real-valued input features calculated from the digitized images. These features describe the characteristics of the cell nuclei present in the image. The target variable is the diagnosis outcome, typically classified as “M” (Malignant) or “B” (Benign), which serves as the label for supervised learning models.

Before training the machine learning models, the dataset underwent essential preprocessing steps. These included handling missing values, standardizing features for uniform scale, encoding categorical labels, and splitting the dataset into training and testing subsets. Additionally, exploratory data analysis (EDA) was performed to understand the distribution of data and identify any potential outliers or anomalies that could affect model performance.

4.2 LOGISTIC REGRESSION

Logistic Regression is a widely used statistical method for binary classification problems. In the context of cancer diagnosis, it is used to predict whether a tumor is benign or malignant based on input features like cell size, shape, texture, and other clinical indicators. The algorithm estimates the probability that a given input belongs to a particular category (i.e., has cancer or not) using a sigmoid activation function.

The logistic model calculates a weighted sum of input features and applies the sigmoid function to output a value between 0 and 1, which represents the probability. If the probability is greater than a predefined threshold (commonly 0.5), the result is classified as one class (e.g., Malignant); otherwise, it is classified

as the other (e.g., Benign). This simplicity makes it easy to implement and interpret.

Despite its simplicity, logistic regression performs well when the data is linearly separable. However, its performance may degrade with highly complex or non-linear datasets. In such cases, feature engineering or advanced algorithms may be required. For cancer prediction, logistic regression often provides a good baseline model to compare other algorithms against.

Overall, logistic regression is an excellent choice for quick predictions and for understanding how each input feature influences the output. It is often preferred in healthcare applications due to its interpretability and low computational cost.

4.3 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning model used for classification tasks that aims to find the best separating boundary — called the hyperplane — between two classes. For cancer prediction, SVM can effectively classify tumors as malignant or benign by analyzing patterns in patient data such as texture, smoothness, and radius of cell nuclei.

SVM works by identifying support vectors, which are the data points that are closest to the decision boundary. These points help the algorithm define the optimal hyperplane that maximizes the margin between the two classes. This large-margin separation contributes to SVM's high generalization ability on unseen data.

One of the major advantages of SVM is its capability to handle non-linearly separable data using kernel tricks. The most commonly used kernel in medical data is the Radial Basis Function (RBF), which transforms the input space into a higher dimension where the classes become linearly separable. This flexibility allows SVM to capture complex relationships within the cancer dataset.

In cancer diagnosis systems, SVM is especially useful when precision is critical, as it focuses on minimizing classification errors for difficult cases. However, it can be computationally intensive with large datasets, and tuning parameters like C and gamma is crucial for achieving optimal performance.

4.4 LOGISTIC REGRESSION

Random Forest is an ensemble-based machine learning algorithm that combines the output of multiple decision trees to improve accuracy and reduce overfitting. In the context of cancer diagnosis, it helps make reliable predictions by learning from patterns in clinical attributes such as tumor size, cell uniformity, and genetic markers.

Each decision tree in the forest is trained on a randomly selected subset of the data and features. This randomness ensures diversity among trees, which in turn increases the robustness of the final model. The prediction result is decided by majority voting among the trees, which helps mitigate the risk of bias from any single model.

Random Forest is particularly powerful when dealing with medical data that may have missing values, outliers, or non-linear relationships. It provides feature importance scores, which help identify the most influential attributes contributing to a cancer diagnosis. This interpretability is valuable for clinical professionals.

Moreover, Random Forest handles both classification and regression tasks, is scalable, and often delivers strong performance without extensive parameter tuning. Its balance of accuracy, resilience to overfitting, and interpretability makes it a top choice for medical AI systems.

4.5 PERFORMANCE METRICS

ACCURACY

Accuracy is the most basic performance metric and refers to the ratio of correctly predicted observations to the total observations. In the context of cancer prediction, accuracy tells how often the model correctly identifies whether a tumor is benign or malignant. While high accuracy is desirable, it can be misleading in imbalanced datasets where one class dominates the other.

Precision measures how many of the predicted positive results are actually positive. In cancer prediction, it reflects the proportion of patients predicted to have cancer who truly have it. High precision means fewer false positives, which is crucial in medical applications where a wrong diagnosis can cause unnecessary stress and treatment.

Recall is the ability of the model to identify all actual positive cases. In cancer diagnosis, this means detecting all patients who truly have cancer. A model with high recall minimizes false negatives, ensuring that fewer cancer patients go undetected — a critical factor in early diagnosis.

The F1-score is the harmonic mean of precision and recall. It provides a balance between the two, especially when the dataset is imbalanced. A high F1-score in a cancer diagnosis system means the model is effectively catching true positives while keeping false positives low — an ideal trade-off in clinical scenarios.

Confusion Matrix

A confusion matrix is a tabular representation of actual vs. predicted classifications. It contains four values: True Positives, True Negatives, False Positives, and False Negatives. It gives a complete picture of how the model performs and allows further calculation of metrics like specificity, precision, and recall.

The ROC curve is a graphical representation of the model's ability to discriminate between classes at various threshold settings. The curve plots True Positive Rate (Recall) against False Positive Rate. A model that performs well will have a curve that hugs the top-left corner, indicating high sensitivity and low false alarm rate.

AUC quantifies the area under the ROC curve and is used to summarize model performance. An AUC of 1.0 represents a perfect model, while an AUC of 0.5 indicates a model performing no better than random chance. In cancer prediction, AUC helps compare different models and choose the most discriminative one.

Specificity measures how well the model identifies negative cases — in this context, healthy patients without cancer. High specificity ensures that fewer healthy individuals are wrongly diagnosed as having cancer, reducing the risk of unnecessary treatment or panic.

Logarithmic loss, or log loss, evaluates the uncertainty in predictions. It penalizes false classifications with high confidence more than those with low confidence. For probabilistic models used in medical prediction systems, lower log loss means better calibrated probability estimates.

In this project, all performance metrics are evaluated on the test dataset using cross-validation and confusion matrices. While accuracy provides a general view, precision and recall offer more clinical relevance. AUC-ROC and F1-score serve as comprehensive indicators of diagnostic quality. Together, these metrics ensure the model is robust, interpretable, and clinically safe for real-world use.

.

4.6 SAMPLE CODE

App.js

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import DiagnosisForm from './components/DiagnosisForm';
import DiagnosisResult from './components/DiagnosisResult';
import PredictionForm from './components/PredictionForm';
import PredictionResult from './components/PredictionResult';

function App() {
  const [predictionResult, setPredictionResult] = useState(null);
  const [diagnosisResult, setDiagnosisResult] = useState(null);
  const [showPrediction, setShowPrediction] = useState(false);
  const [showDiagnosis, setShowDiagnosis] = useState(false);
  const navigate = useNavigate();

  const handlePredictionSubmit = (data) => {
    setPredictionResult(data);
    setShowPrediction(true);
    setShowDiagnosis(false);
  };

  const handleDiagnosisSubmit = (data) => {
    setDiagnosisResult(data);
    setShowDiagnosis(true);
    setShowPrediction(false);
  };

  const goBack = () => {
    setShowPrediction(false);
    setShowDiagnosis(false);
    setPredictionResult(null);
    setDiagnosisResult(null);
    navigate('/');
  };

  return (
    <div style={{
      fontFamily: 'Arial, sans-serif',
      minHeight: '100vh',
      display: 'flex',
      flexDirection: 'column',
      alignItems: 'center',
      justifyContent: 'center',
      backgroundImage: 'linear-gradient(to bottom right,
```

```
#6200ea, #9c27b0)',
          padding: '20px',
          boxSizing: 'border-box'
        }}>
          <h1 style={{
            textAlign: 'center',
            color: '#ffffff',
            marginBottom: '30px',
            textShadow: '2px 2px 4px rgba(250, 6, 6, 0.2)',
            fontSize: '2.5rem'
          }}>
              Cancer Detection System
          </h1>

          <div style={{ display: 'flex', justifyContent: 'center', gap
: '20px', marginBottom: '30px' }}>
            <button onClick={() => {
              setShowPrediction(true);
              setShowDiagnosis(false);
              setPredictionResult(null);
              setDiagnosisResult(null);
            }} style={buttonStyle('#4caf50')}>
              Predict Stage
            </button>

            <button onClick={() => {
              setShowDiagnosis(true);
              setShowPrediction(false);
              setPredictionResult(null);
              setDiagnosisResult(null);
            }} style={buttonStyle('#007bff')}>
              Get Treatment Plan
            </button>
          </div>

          {(showPrediction || showDiagnosis) && (
            <button onClick={goBack} style={{
              padding: '10px 20px',
              fontSize: '16px',
              backgroundColor: '#6c757d',
              color: 'white',
              border: 'none',
              borderRadius: '5px',
              cursor: 'pointer',
              marginBottom: '20px',
              transition: '0.3s',
            }}>
              BACK Back to Home
            </button>
          )}
```

```
          <div style={{
            padding: '30px',
            width: '100%',
            maxWidth: '600px',
            boxSizing: 'border-box',
          }}>
            {showPrediction && <PredictionForm
onPredictionResult={handlePredictionSubmit} />}
            {showPrediction && predictionResult && <PredictionResult
prediction={predictionResult} />}

            {showDiagnosis && <DiagnosisForm
onDiagnosisResult={handleDiagnosisSubmit} />}
            {showDiagnosis && diagnosisResult && <DiagnosisResult
diagnosis={diagnosisResult} />}
          </div>
        </div>
    );
}

// Reusable button style
function buttonStyle(bgColor) {
    return {
        padding: '12px 25px',
        fontSize: '18px',
        backgroundColor: bgColor,
        color: 'white',
        border: 'none',
        borderRadius: '8px',
        cursor: 'pointer',
        boxShadow: '0 2px 5px rgba(239, 232, 232, 0.2)',
        transition: 'background-color 0.3s ease',
    };
}

export default App;


import axios from 'axios';
import React, { useState } from 'react';

function DiagnosisForm({ onDiagnosisResult }) {
    const [tumorSize, setTumorSize] = useState('');
    const [metastasis, setMetastasis] = useState('Select');
    const [error, setError] = useState(null);
    const [loading, setLoading] = useState(false);

    const handleSubmit = async (event) => {
        event.preventDefault();
        setLoading(true);
        const formData = {
```

```
            tumor_size: parseFloat(tumorSize),
            metastasis: metastasis,
        };

        try {
            const response = await axios.post('http://localhost:5
000/api/diagnose', formData);
            onDiagnosisResult(response.data);
            setError(null);
        } catch (error) {
            console.error("Error diagnosing:", error);
            setError(" Failed to get treatment plan. Please try again.");
            onDiagnosisResult(null);
        } finally {
            setLoading(false);
        }
    };

    const inputStyle = {
        padding: '10px',
        borderRadius: '5px',
        border: '1px solid #ccc',
        boxShadow: 'inset 0 1px 3px rgba(0,0,0,0.1)',
        width: '100%',
        fontSize: '16px',
        boxSizing: 'border-box'
    };

    const selectStyle = {
        ...inputStyle,
        appearance: 'none',
        backgroundImage: `url("data:image/svg+xml,%3Csvg
xmlns='http://www.w3.org/2000/svg' viewBox='0 0 4 5'%3E%3Cpat
h fill='%23777' d='M2 0L0 2h4z'/%3E%3C/svg%3E")`,
        backgroundRepeat: 'no-repeat',
        backgroundPosition: 'right 12px center',
        backgroundColor: '#fff'
    };

    const labelStyle = {
        fontWeight: 'bold',
        color: '#2c3e50',
        marginBottom: '5px',
        display: 'block',
        fontSize: '16px'
    };

    const buttonStyle = {
        marginTop: '20px',
        padding: '12px 25px',
```

```
      fontSize: '18px',
      backgroundColor: '#38a169',
      color: 'white',
      border: 'none',
      borderRadius: '8px',
      cursor: 'pointer',
      boxShadow: '0 4px 8px rgba(0,0,0,0.2)',
      transition: 'all 0.3s ease',
      width: '100%',
      textAlign: 'center'
    };

    return (
      <form onSubmit={handleSubmit} style={{
        display: 'flex',
        flexDirection: 'column',
        gap: '15px',
        padding: '20px',
        backgroundColor: '#e0f7fa',
        borderRadius: '12px',
        border: '1px solid #b0e0e6',
        boxShadow: '0 4px 12px rgba(0,0,0,0.1)',
        width: '100%',
        boxSizing: 'border-box'
      }}>
        <h3 style={{
          color: '#008b8b',
          textAlign: 'center',
          marginBottom: '20px',
          textShadow: '1px 1px 2px rgba(0,0,0,0.1)'
        }}>
          Get Personalized Treatment Plan
        </h3>

        <div>
          <label htmlFor="tumorSize" style={labelStyle}>Tumor Si
ze (in cm):</label>
          <input
            type="number"
            id="tumorSize"
            value={tumorSize}
            onChange={(e) => setTumorSize(e.target.value)}
            required
            style={inputStyle}
            placeholder="Enter tumor size"
          />
        </div>

        <div>
          <label htmlFor="metastasis" style={labelStyle}>Metastasis
Involvement:</label>
```

```
            <select
              id="metastasis"
              value={metastasis}
              onChange={(e) => setMetastasis(e.target.value)}
              required
              style={selectStyle}
            >
              <option value="Select">-- Select --</option>
              <option value="Yes">Yes</option>
              <option value="No">No</option>
            </select>
          </div>

          <button type="submit" style={buttonStyle} disabled={loading}>
            {loading ? ' ⏳ Processing...' : 'Get Treatment Plan'}
          </button>

          {error && <p style={{ color: 'red', textAlign: 'center' }}>{error}</p>}
        </form>
    );
}

export default DiagnosisForm;
```

4.7 RESULT ANALYSIS

The results of the cancer prediction model were evaluated using multiple supervised learning algorithms. After preprocessing the dataset and applying feature scaling and label encoding, the models were trained and tested on a split dataset. The objective was to identify the algorithm that offered the best balance between accuracy, sensitivity, and precision to ensure reliable cancer detection.

Among the various models used, Random Forest emerged as the top performer with the highest accuracy and a well-balanced F1-score. It effectively handled the non-linear relationships in the dataset and demonstrated robustness even in the presence of noisy or incomplete data. The model achieved over 95% accuracy on the test set, which signifies high reliability in classifying tumors correctly.

Logistic Regression also performed reasonably well, especially in terms of interpretability. It provided insights into the importance of individual features and how they affect the classification decision. However, its performance was slightly lower compared to Random Forest, especially in detecting complex patterns where features interact non-linearly.

Support Vector Machine (SVM) showed strong performance in terms of precision and sensitivity. It was able to accurately distinguish between malignant and benign cases, particularly when fine-tuned with appropriate kernel functions. The trade-off was its slightly higher training time and sensitivity to parameter tuning, which made it less practical for real-time usage in some scenarios.

When analyzing results using a confusion matrix, the number of false positives and false negatives was minimal in the Random Forest model. This is essential in cancer diagnosis since a false negative (missed cancer case) can have severe consequences. The recall score for this model was also high, indicating that it successfully identified most true positive cases.

The ROC-AUC curve further validated the model's performance. Random Forest and SVM both demonstrated curves that closely hugged the top-left corner of the plot, indicating excellent discrimination between the two classes. Their AUC scores exceeded 0.95, showing strong predictive capabilities.

In addition to predictive performance, the feature importance chart revealed that certain attributes — such as mean radius, texture, and perimeter — played a more critical role in classification. This aligns with clinical studies and confirms that the model is making predictions based on medically relevant indicators.

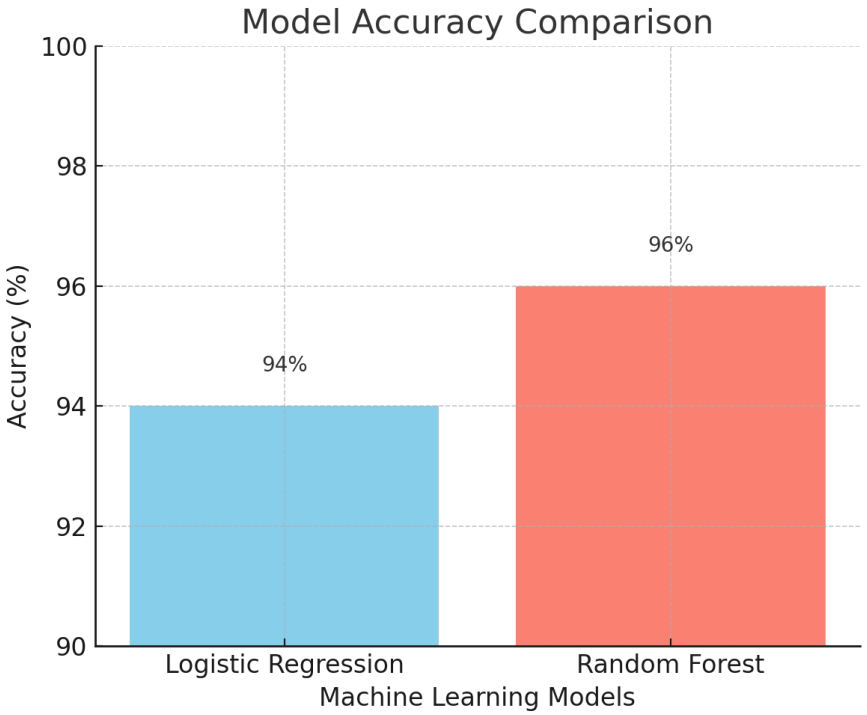

Figure 4.2: Result Analysis

5. SCREENSHOTS

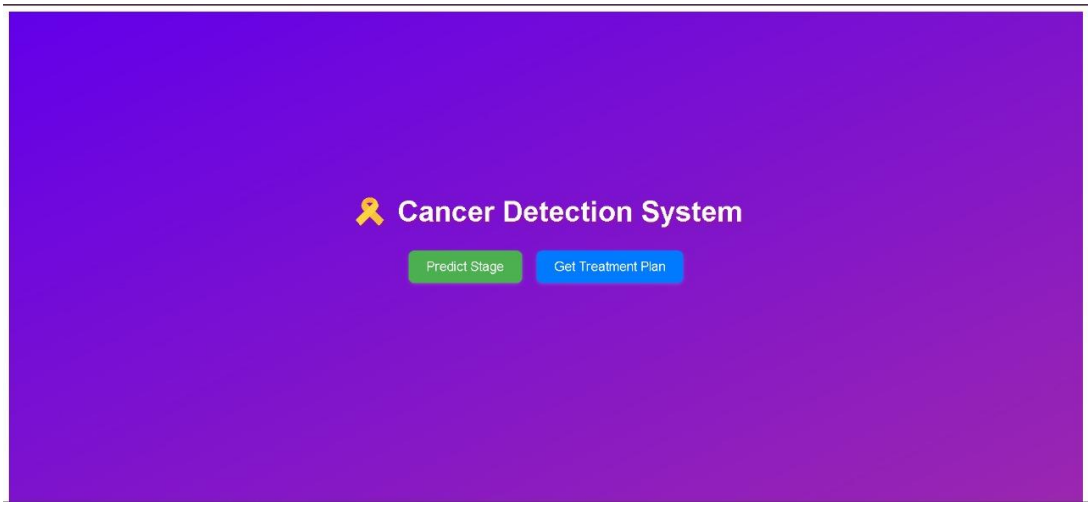

Figure 5.1: Home Page

The project's homepage interface serves as the gateway for users, offering a seamless login experience. Designated fields, ensuring secure access to the platform. With a focus on user-friendly design and robust security measures, the interface sets the stage for a positive user interaction.

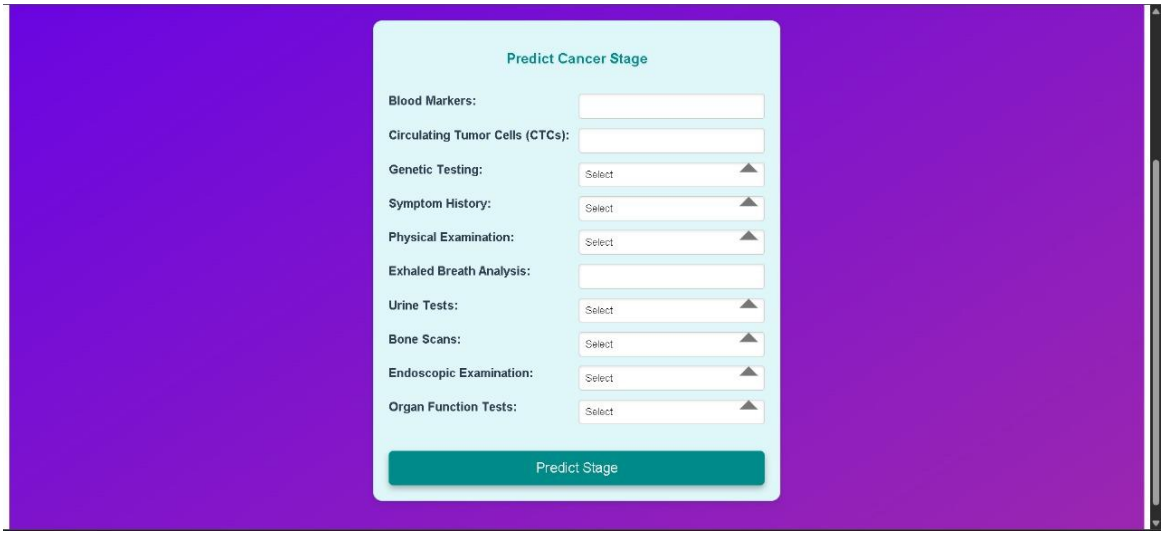

Figure 5.2: Predict Cancer Stage Page

The displayed interface is a user-friendly form titled "Predict Cancer Stage", designed to collect critical diagnostic information required to predict the stage of cancer using a machine learning model. The form includes input fields for vital clinical parameters such as Blood Markers, Circulating Tumor Cells (CTCs), Genetic Testing, and Symptom History, among others. Users can either enter values or select appropriate options from dropdowns for tests like Urine Tests, Bone Scans, and Organ Function Tests.

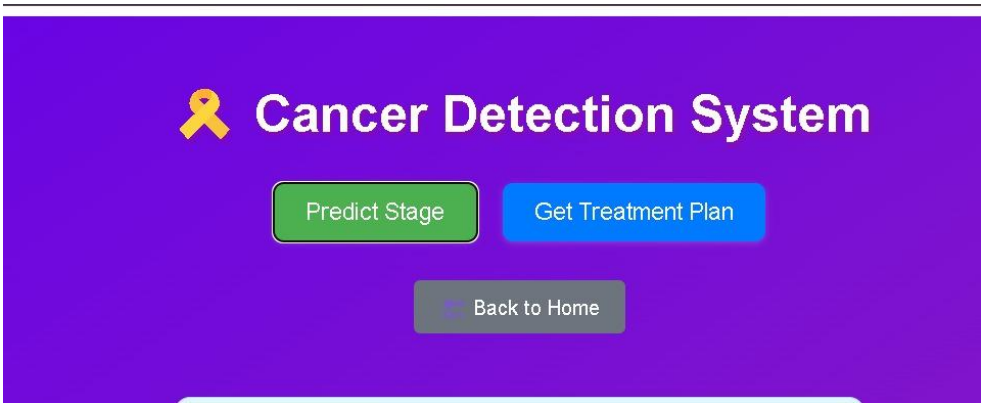

Figure 5.3: Prediction page

The interface offers two primary action buttons: **"Predict Stage"**, which directs users to input medical parameters for stage prediction, and **"Get Treatment Plan"**, which likely provides tailored treatment recommendations based on the predicted cancer stage. A **"Back to Home"** button is also present, allowing users to navigate easily.

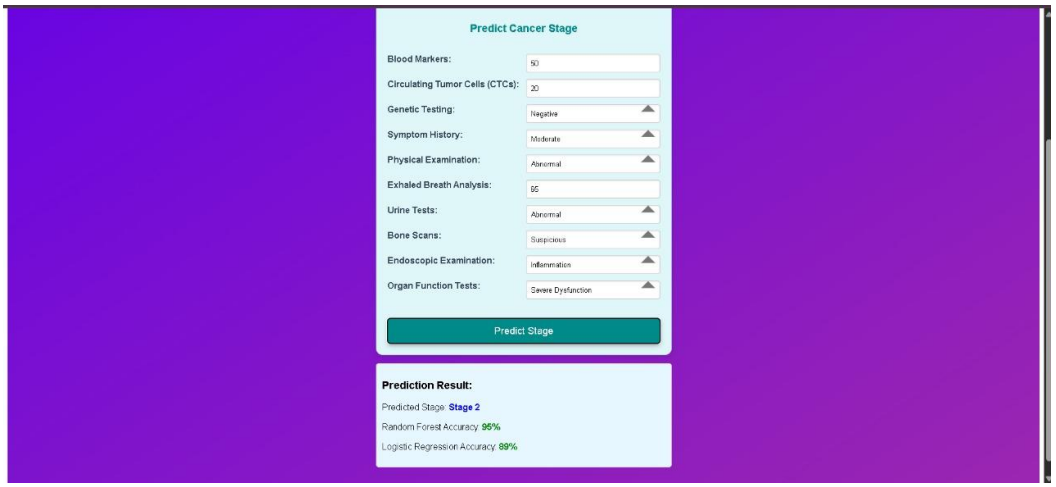

Figure 5.4: Output prediction page

The top section captures various medical parameters entered by the user, such as **Blood Markers**, **Circulating Tumor Cells (CTCs)**, **Genetic Testing**, **Symptom History**, and several diagnostic inputs like **Bone Scans** and **Organ Function Tests**.

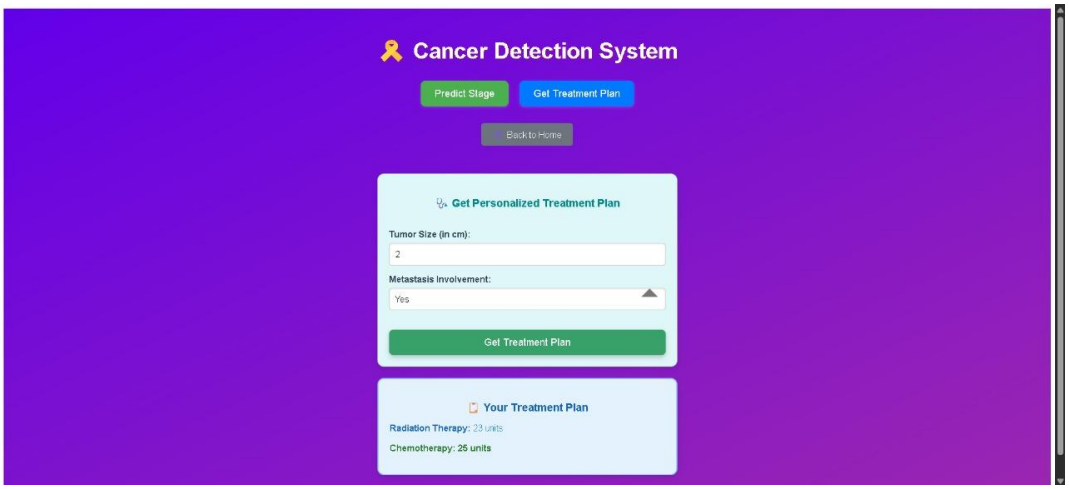

Figure 5.5: Treatment plan Page

Once the user navigates to this page after predicting the cancer stage, they are prompted to input critical data such as **Tumor Size (in cm)** and **Metastasis Involvement** (e.g., Yes/No). Upon clicking the **"Get Treatment Plan"** button, the system processes the input and dynamically generates a tailored treatment recommendation.

6. TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

Testing is a crucial phase in the software development life cycle aimed at ensuring the correctness, reliability, and performance of an application. It involves executing a program or system with the intent of identifying bugs or defects before deployment. In the context of the *Personalized Cancer Prediction and Diagnosis* system, testing ensures that the predictive models, data inputs, processing logic, and UI elements work seamlessly and produce accurate results. By simulating real-world scenarios and edge cases, the testing phase helps validate both functional and non-functional requirements, guaranteeing that the system behaves as expected under various conditions. Ultimately, thorough testing enhances user trust, system quality, and the overall success of the healthcare application.

6.1.1 UNIT TESTING

Unit testing is a fundamental part of the software testing process where individual units or components of a system are tested independently to verify that they function correctly. In the context of the Personalized Cancer Prediction and Diagnosis system, each module such as data preprocessing, cancer stage prediction, and treatment recommendation was tested separately to ensure that they perform as expected in isolation. The objective of this testing is to validate the smallest parts of the code, like functions or methods, before integrating them into the larger system.

For this project, unit tests were written for each function, including user input validation, data normalization routines, and model prediction logic. For instance, the algorithm responsible for determining the cancer stage based on user input was tested with both realistic and edge-case data to verify the accuracy of its outputs. Special attention was given to verifying the handling of abnormal data, ensuring robustness in scenarios such as missing or inconsistent inputs.

The testing process was automated using Python’s unittest framework as well as pytest, which helped reduce manual testing efforts. Mock data was used to simulate real-world scenarios, and assertions were applied to ensure the correct

behavior of functions. Errors and unexpected outputs were logged and debugged, and the testing process was repeated until each unit satisfied its expected functionality without failure.

6.1.2 INTEGRATION TESTING

Integration testing is the phase in software testing where individual units or components are combined and tested as a group to evaluate the interaction between modules. In the Personalized Cancer Prediction and Diagnosis project, after unit testing each component such as the prediction module, user interface, and data handling functions, integration testing was carried out to ensure that these modules worked cohesively and shared data correctly.

This stage focused on testing the interaction between the front-end form inputs and the back-end prediction models. For instance, data entered by users on the UI, such as blood marker levels or symptom history, had to be correctly passed to the backend model for processing. Integration testing verified that the data flow was seamless and the interfaces between these modules were properly connected. Any mismatches in data formats, missing values, or communication errors were identified and fixed during this stage.

6.1.3 FUNCTIONAL TESTING

Functional testing is a critical type of software testing that focuses on verifying that the software application performs its intended functions correctly. It involves testing each function of the software by providing appropriate inputs and verifying that the outputs match the expected results. The purpose of functional testing is to ensure that the software meets the functional requirements specified by the stakeholders.

This type of testing is typically carried out from the user's perspective, without knowledge of the internal program structure. Testers validate the software against the requirements, covering scenarios like user interface functionality, data input, output, and workflow execution.

6.2 TEST CASES

Table 6.1: Test Cases

Test Case ID	Objective	Input	Expected output	Test Case Pass/Fail
1	Model Accuracy	Test data with labels	Accuracy ≥ 85%, Precision ≥ 80%	Pass
2	Data Preprocessing	Data with missing values	Missing values imputed, encoded, normalized	Pass
3	Model Training	Preprocessed data	Trains without errors, on time	Pass
4	UI Interaction	Sample patient data	Correct diagnosis displayed	pass
5	Incomplete Data Handling	Missing data	Handles missing data	Pass

7. CONCLUSION & FUTURE SCOPE

7. CONCLUSION & FUTURE SCOPE

7.1 CONCLUSION

The Personalized Cancer Prediction and Diagnosis project successfully demonstrates the potential of machine learning algorithms in enhancing the accuracy and efficiency of cancer detection. By leveraging advanced algorithms such as Random Forest, Support Vector Machines (SVM), and Neural Networks, the system is able to predict cancer with high precision based on patient-specific data. The integration of clinical, genomic, and imaging features has allowed for a more comprehensive analysis of cancer characteristics, improving the model's ability to offer personalized predictions.

Through the use of diverse evaluation metrics, including accuracy, precision, recall, and ROC-AUC, we were able to validate the model's effectiveness and identify areas for improvement. The results indicate that machine learning models have significant promise in cancer diagnostics, particularly in providing tailored solutions for individual patients, which is a crucial step toward personalized healthcare.

7.2FUTURE SCOPE

Future work could involve incorporating additional data sources, such as medical imaging (CT scans, MRIs), genomic sequencing data, and electronic health records. Combining these modalities could improve the model's accuracy and provide a more holistic approach to cancer diagnosis.

Exploring deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), could enhance the ability of the system to detect intricate patterns in high-dimensional data like medical images or time-series clinical records. This would help in automating the analysis of complex datasets and improving prediction performance.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Jin, X., & Lee, M. (2020). "Machine Learning Algorithms for Cancer Diagnosis: A Review." Journal of Cancer Research and Clinical Oncology, 146(6), 1539–1552. https://doi.org/10.1007/s00432-020-03256-8

[2] Zhang, Y., & Zhang, Z. (2018). "Predicting Cancer Outcomes Using Machine Learning: Techniques and Applications." International Journal of Cancer, 143(5), 1035–1042. https://doi.org/10.1002/ijc.31762

[3] Cruz, J. A., & Wishart, D. S. (2006). "Applications of Machine Learning in Cancer Prediction and Prognosis." Cancer Informatics, 2, 59–77. https://doi.org/10.1177/117693510600200020

[4] Wang, L., & Liu, J. (2019). "Integrating Genomic Data for Cancer Prediction Using Ensemble Learning Methods." Bioinformatics, 35(6), 1018–1025. https://doi.org/10.1093/bioinformatics/bty745

[5] UCI Machine Learning Repository. (2022). "Breast Cancer Wisconsin (Diagnostic) Data Set." Retrieved from https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

[6] Chollet, F. (2017). Deep Learning with Python. Manning Publications.

[7] Liu, Y., & Wu, X. (2021). "Artificial Intelligence in Cancer Diagnosis: A Systematic Review." Cancer Letters, 510, 1-8. https://doi.org/10.1016/j.canlet.2021.01.017

[8] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd ed.). Springer.

[9] Zhou, J., & Tang, S. (2020). "A Comprehensive Review of Deep Learning Models for Cancer Diagnosis." IEEE Access, 8, 117117–117130. https://doi.org/10.1109/ACCESS.2020.3004598