CAPSTONE PROJECT
PHASE- 2 - REVIEW-2

TEAM 74

Heart Disease Prediction using Machine Learning Model

TEAM MEMBERS:

1. 21BCE10225 Archita Gupta
2. 21BCE10406 Sonali Raghuwanshi
3. 21BCE10439 Priyanshi Yadav
4. 21BCE10669 Tina Chelwani
5. 21BCE10708 Abhinav Shrivastava

Supervisor
Dr.J. Manikandan

Reviewer 1
Dr.Sasmita Padhy

Reviewer 2
Dr. Antima Jain

OBJECTIVE

- Heart Disease:
  - Affects heart structure and function, involving blood vessels, rhythm, or muscle issues.
- Leading cause of death globally
- Common Types:
  - Coronary Artery Disease (CAD): Narrowing/blockage of blood vessels.
  - Arrhythmias: Irregular heartbeats.
  - Heart Failure: Ineffective blood pumping.
  - Congenital Heart Defects: Structural abnormalities from birth.
  - Cardiomyopathy: Heart muscle diseases.
  - Heart Valve Diseases: Valve dysfunction.

OBJECTIVE

- Machine Learning for Prediction:
  - Processes medical datasets to detect hidden patterns and risk factors.
  - Enables early, accurate heart disease prediction.
- Advantages of Machine Learning:
  - Non-invasive, scalable, cost-effective, and precise.
  - Empowers healthcare professionals with data-driven decision-making.
  - Supports timely interventions, improving patient outcomes.

OBJECTIVE

- Process patient dataset
- Identify key predictors (cholesterol, BP, smoking)
- Implement Machine Learning Algorithms
- Compare accuracy, F1-score, ROC-AUC
- Develop user-friendly interface
- Enable real-time predictions
- Provide personalized recommendations

PHASE 1 REVIEW

- Research about existing works
- Pre-process 9,776 patient dataset
- Identify important features(cholesterol, BP, smoking factors,etc.)
- Implement XGBoost, AdaBoost models
- Compare accuracy, F1, ROC-AUC metrics

SUGGESTIONS FOR PHASE 2

- Implement more algorithms
- Build user interface
- Enable real-time prediction
- Incorporate patient history
- Add personalized recommendations
- User Authentication

LITERATURE REVIEW

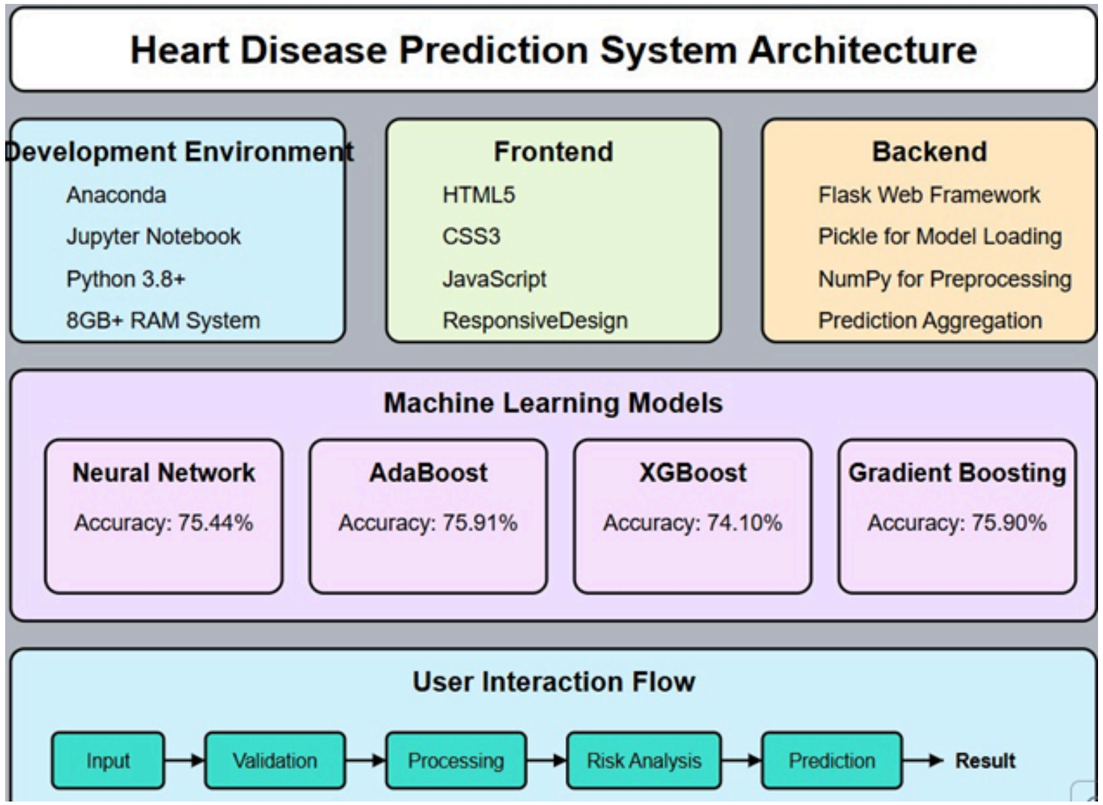
- Yang (2023)
  - Model: XGBoost
  - Key finding: XGBoost with tuned hyperparameters shows impressive performance
- Arshit Jindal et al. (2024)
  - Models: Logistic Regression, KNN, Random Forest
  - Accuracy: 88.52%
  - Key finding: Traditional ML effective but needs optimization
- Chintan M. Bhatt et al. (2023)
  - Models: MLP, XGBoost, Random Forest, Decision Tree
  - Highest accuracy: 87.28% (MLP)
  - Key finding: Deep learning provides better generalization

LITERATURE REVIEW

- Baban Uttamrao Rindhe et al. (2021)
  - Models: ANN, Random Forest
  - Accuracy: 83.5% (ANN)
  - Key finding: ANN handles non-linear relationships well
- R. Fadnavis et al. (2021)
  - Models: Naive Bayes (85.25%), Decision Trees (81.97%)
  - Key finding: Simpler models can be effective and interpretable
- Abhijeet Jagtap et al. (2019)
  - Models: Logistic Regression, Naive Bayes
  - Accuracy: 61.45% (Logistic Regression)
  - Key finding: Complex models needed for better results

Year	Proposed techniques	Tools	Accuracy
2021[1]	logistic regression, Random Forest Classifier and KNN	Jupyter Notebook	87.5%
2019[2]	Support Vector Machine (SVM) Logistic Regression Naïve Bayes Algorithm	Jupyter Notebook, Web Framework	64.4% 61.45% 60%
2021[3]	Support Vector Classifier Neural Network Random Forest Classifier	MS excel, Python	84.0 % 83.5 % 80.0 %
2023[4]	Random forest Decision tree Multilayer perception XGBoost classifier.	Python, Jupyter Notebook	87.05% 86.37% 87.28% 86.87%
2021[5]	Recurrent Neural Network (RNN)	Python 3.7	98.6876%
2018[6]	Recurrent Fuzzy Neural Network (RFNN)	MATLAB	96.63%
2012[7]	Naive Bayes Decision Trees Neural Networks	Jupyter Notebook Python	90.74% 96.66% 99.25%
2021[8]	Naive Bayes Decision Trees	Jupyter Notebook Python	85.25% 81.97%
2024 [9]	Random forest Ada Boost Gradient Boosting Naive Bayes Logistic Regression	Python, Jupyter notebook	98.71% 88% 93% 80% 80%
2024[10]	Bat Algorithm Particle Swarm Optimization Random Forest	Python, Jupyter notebook	96.88 97.53 94.79

SYSTEM ARCHITECTURE

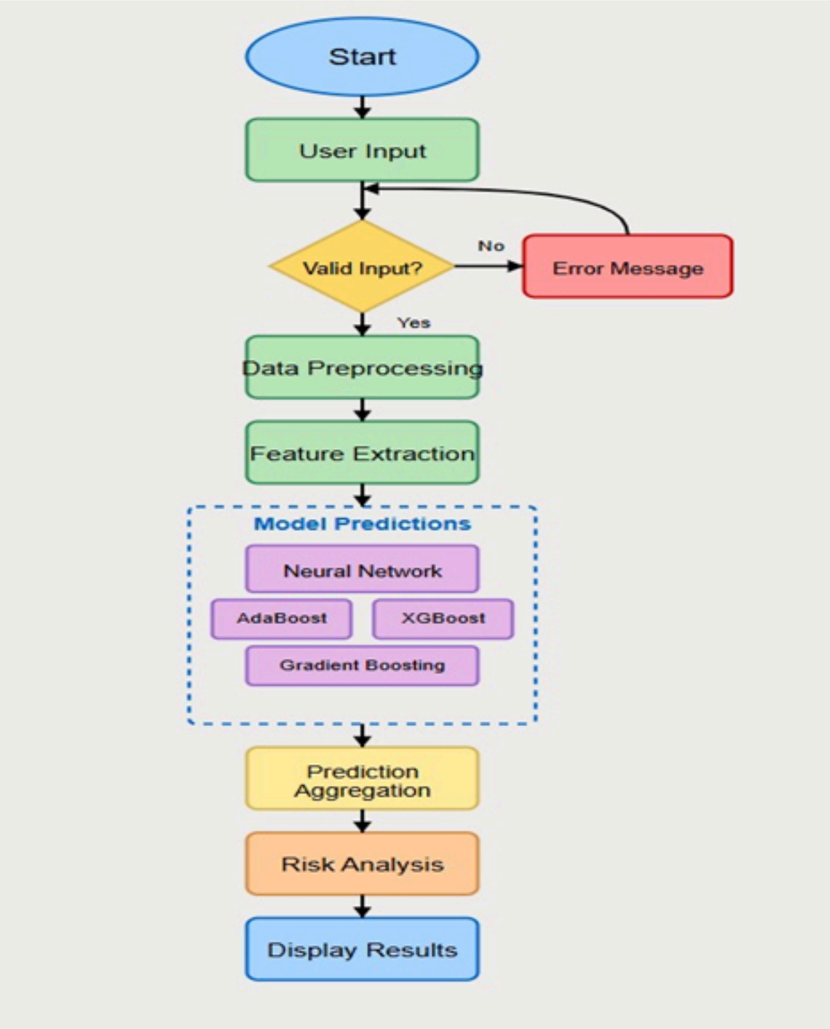


1. Development Environment:
   - Anaconda
   - Jupyter Notebook
   - Python 3.x
   - Scikit-learn/NumPy
2. Frontend:
   - HTML5
   - CSS
   - JavaScript
3. Backend:
   - Flask Web Framework

SYSTEM ARCHITECTURE

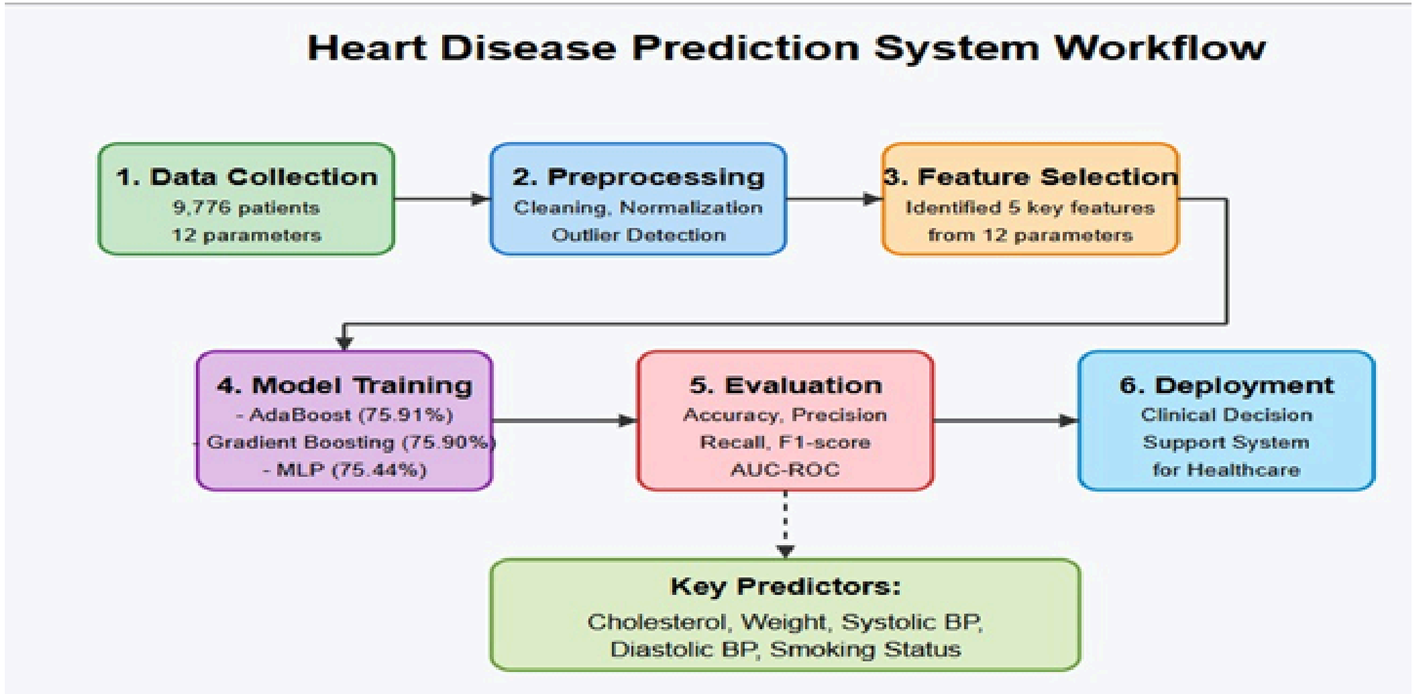
4. Machine Learning Models:
    - Neural Network (Accuracy: 78.45%)
    - AdaBoost (Accuracy: 79.01%)
    - XGBoost (Accuracy: 81.05%)
    - Gradient Boosting (Accuracy: 76.90%)
5. User Interaction Flow:
    - Input: User enters patient health metrics
    - Processing: ML models analyze input data
    - Risk Analysis: System calculates disease probability
    - Display: Results shown via visualizations
    - Result: Prediction with health recommendations

DATA FLOW DIAGRAM



- Start: System initialization
- User Input: Patient health data entry
- Input Validation: Checks data completeness and format
- Data Preprocessing: Cleaning and normalization
- Feature Extraction: Identifies relevant health indicators
- Model Predictions: Runs multiple ML models (Neural Network, AdaBoost, XGBoost, Gradient Boosting)
- Prediction Aggregation: Combines results from all models
- Risk Analysis: Calculates heart disease probability
- Display Results: Shows prediction with visualizations

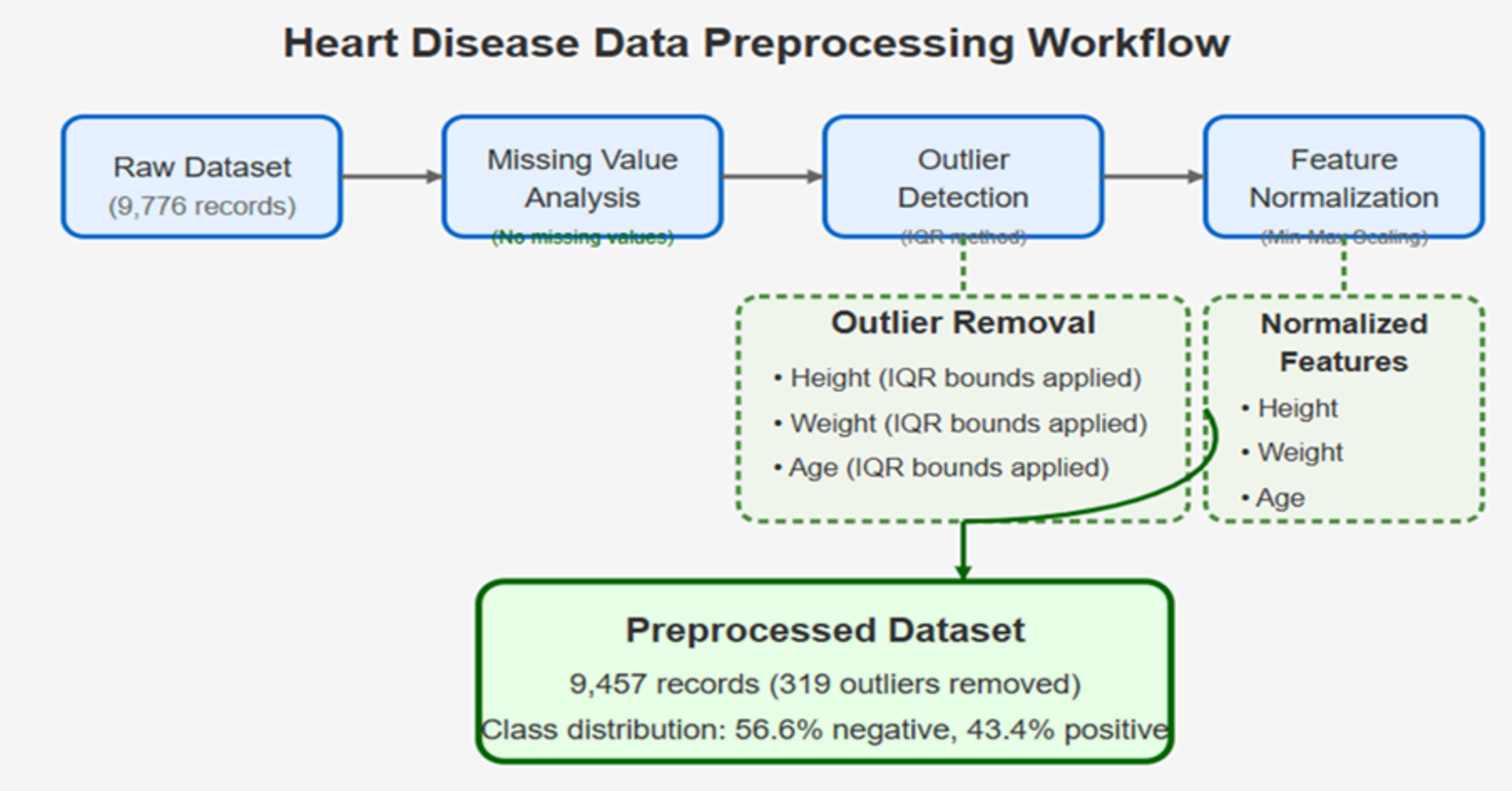
SYSTEM WORKFLOW



- Data Collection: 9,776 patients with 12 health parameters
- Preprocessing: Cleaning, normalization, and outlier detection
- Feature Selection: Identified 5 key features from 12 parameters
- Model Training: AdaBoost (79.91%), Gradient Boosting (75.90%), MLP (78.45%)
- Evaluation: Assessed models using accuracy, precision, recall, F1-score, AUC-ROC
- Deployment: Implemented as clinical decision support system for healthcare

Project Modules

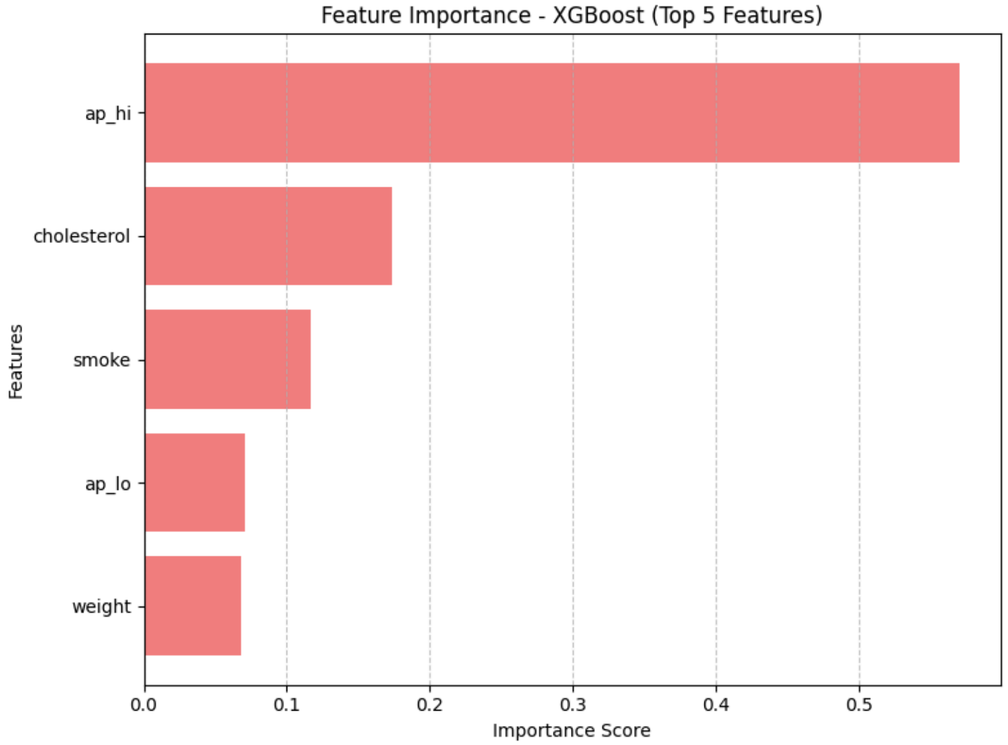
1. DATA PREPROCESSING

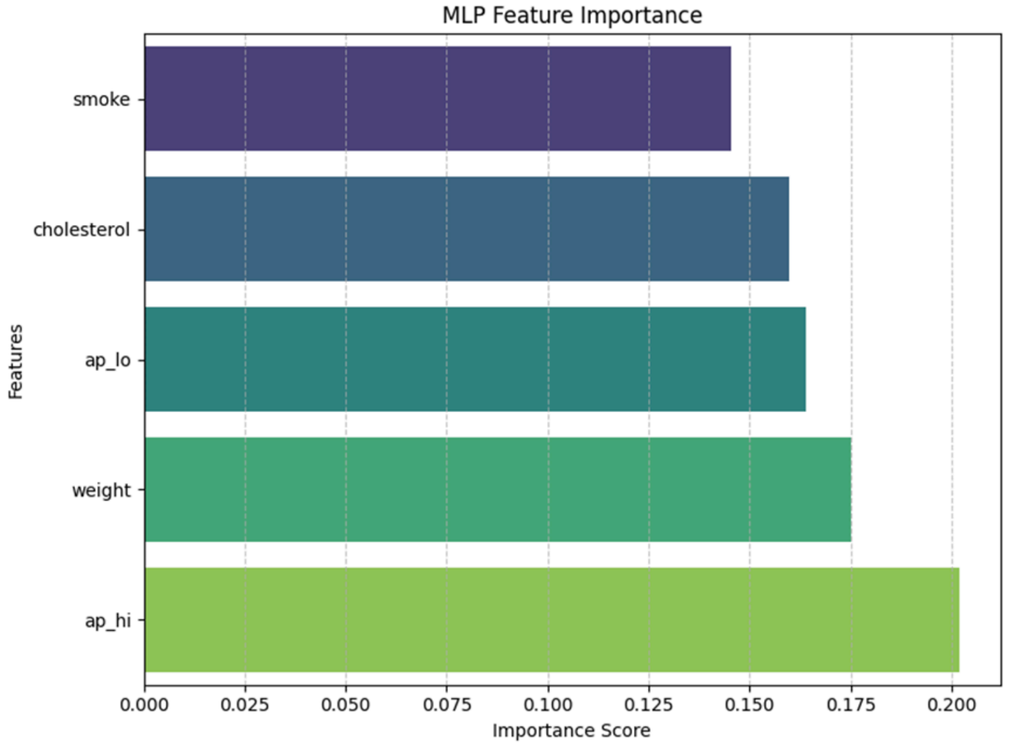


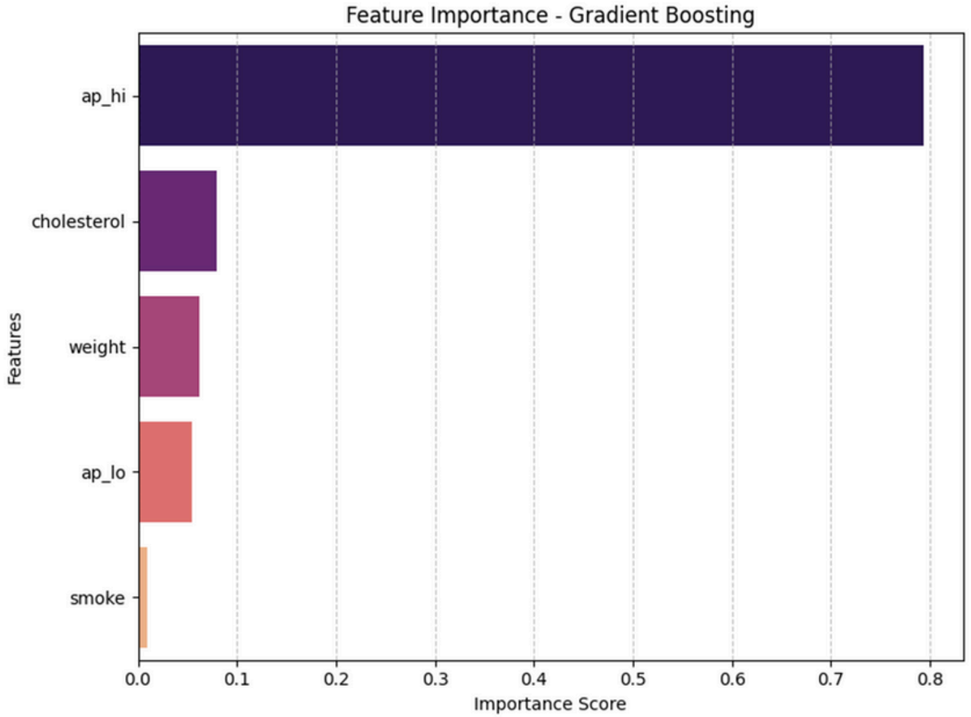
- Raw Dataset: Started with 9,776 patient records
- Missing Value Analysis: Verified dataset was complete with no missing values
- Outlier Detection: Applied IQR (Interquartile Range) method
- Outlier Removal: Removed extreme values for height, weight, and age using IQR bounds
- Feature Normalization: Applied Min-Max scaling to standardize numerical features
- Normalized Features: Standardized height, weight, and age values
- Final Preprocessed Dataset: 9,457 records (319 outliers removed)
- Class Distribution: 56.6% negative (no heart disease), 43.4% positive (heart disease)

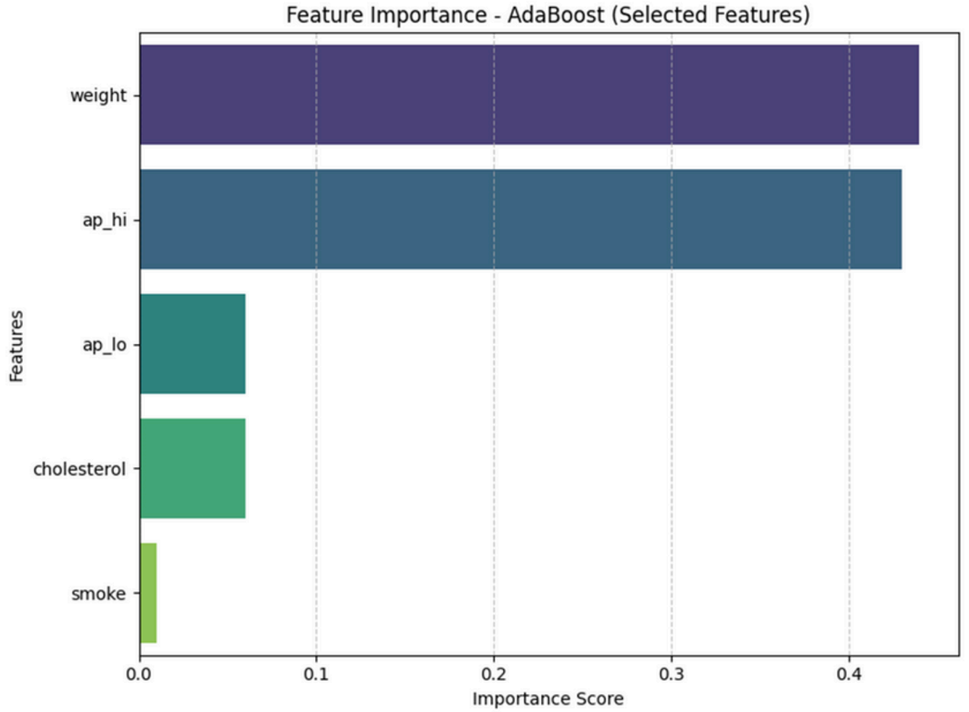
2. FEATURE SELECTION

- Conducted on a dataset of 9,776 patients with 12 health parameters.
- Feature importance analysis identified the top 5 predictors of heart disease:
  - Systolic BP (ap_hi)
  - Diastolic BP (ap_lo)
  - Cholesterol
  - Smoking Status
  - Weight
- These features were selected for their strong correlation with heart disease and to improve model efficiency and interpretability.


Feature Importance - XGBoost (Top 5 Features)
ap_hi
cholesterol
smoke
ap_lo
weight
Features
0.0
0.1
0.2
0.3
0.4
0.5
Importance Score


MLP Feature Importance
smoke
cholesterol
ap_lo
weight
ap_hi
Features
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
Importance Score


Feature Importance - Gradient Boosting
ap_hi
cholesterol
weight
ap_lo
smoke
Features
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Importance Score


Feature Importance - AdaBoost (Selected Features)
weight
ap_hi
ap_lo
cholesterol
smoke
Features
0.0
0.1
0.2
0.3
0.4
Importance Score

3. MODEL SELECTION

- **Existing Model XG Boost:-**

  - XGBoost (Extreme Gradient Boosting) is a powerful ensemble learning algorithm known for its speed and high predictive accuracy.
  - It uses gradient boosting with regularization, helping to reduce overfitting and improve model performance**.**
  - **Advantages:-**
  - High accuracy – Uses advanced boosting and regularization.
  - Fast training – Optimized for speed and performance.
  - Handles missing data – Automatically manages null values.

XG BOOST ALGORITHM

- Initialize the model with a constant value (usually the average of target values for regression).
- Compute residuals (errors) from the current model predictions.
- Fit a decision tree to these residuals (errors).
- Update the prediction by adding the new tree’s output (scaled by a learning rate).
- Repeat steps 2–4 for a fixed number of iterations or until convergence.
- Final prediction is the sum of all weak learner predictions (trees).

Results

We have achieved an overall accuracy of 74.41%, with Class 1 yielding a precision of 0.74, recall of 0.60, and an F1-score of 0.67

Class	precision	recall	f1-score	support
0	0.74	0.84	0.79	1628
1	0.74	0.60	0.67	1210
accuracy			0.74	2838
macro avg	0.74	0.72	0.73	2838
weighted avg	0.74	0.74	0.74	2838

PROPOSED MODEL

1.(Neural Network) MLP

- Feedforward Neural Network – MLP consists of an input layer, one or more hidden layers, and an output layer, where data flows in one direction.
- Uses Backpropagation – It learns by adjusting weights using backpropagation and gradient descent.
- **Advantages:-**
- Handles Non-linear Data – Can model complex, non-linear relationships.
- Flexible Architecture – Can be adapted for classification, regression, and more.
- Works Well with Clean Data – Gives good performance when input data is well-preprocessed and normalized.

MLP ALGORITHM STEPS

- Initialize weights and biases randomly for all neurons in the network.
- Perform forward propagation to calculate the output using activation functions.
- Compute the loss using a suitable loss function (e.g., Cross-Entropy or MSE).
- Apply backpropagation to compute gradients of the loss w.r.t. weights.
- Update weights and biases using gradient descent or an optimizer like Ada

Results

We have achieved an overall accuracy of 75.44%, with Class 1 yielding a precision of 0.76, recall of 0.61, and an F1-score of 0.68

Class	precision	recall	f1-score	support
0	0.75	0.86	0.80	1628
1	0.76	0.61	0.68	1210
accuracy			0.75	2838
macro avg	0.76	0.74	0.74	2838
weighted avg	0.76	0.75	0.75	2838

2.Gradient Boosting

- Optimization Technique – It minimizes a loss function by updating model parameters (weights) in the direction of the negative gradient.
- Iterative Process – Repeats updates over several epochs until convergence or a minimum loss is achieved.
- **Advantages:-**
- Simple and Efficient – Easy to implement and works well for most ML models.
- Scalable – Can handle large datasets using variants like Stochastic Gradient Descent (SGD) and Mini-batch GD.
- Works with Any Differentiable Function – Doesn't require the loss function to be convex or linear.

GRADIENT ALGORITHM STEPS

- Initialize parameters (weights and biases) randomly.
- Select learning rate (α), which controls the update step size.
- Make predictions using the current parameters.
- Compute the loss (e.g., Mean Squared Error).
- Calculate gradients of the loss with respect to each parameter.

Results

We have achieved an overall accuracy of 75.9%, with Class 1 yielding a precision of 0.77, recall of 0.62, and an F1-score of 0.69

Class	precision	recall	f1-score	support
0	0.75	0.86	0.80	1628
1	0.77	0.62	0.69	1210
accuracy			0.76	2838
macro avg	0.76	0.74	0.75	2838
weighted avg	0.76	0.76	0.75	2838

3.ADA BOOST

- Ensemble Learning Method – AdaBoost combines multiple weak learners (usually decision stumps) to form a strong classifier.
- Focuses on Hard Samples – It assigns higher weights to misclassified instances in each round to improve future predictions.
- **Advantages:-**
- Improves Weak Learners – Boosts the performance of simple models effectively.
- Reduces Overfitting – Less prone to overfitting compared to other models.
- Flexible & Versatile – Can be used for classification and regression tasks.

ADA BOOST ALGORITHM STEPS

- Start with equal weights for all training data.
- Train a weak learner (like a decision stump).
- Check which samples are misclassified.
- Increase weights of misclassified samples.
- Train next learner focusing more on those hard samples.
- Combine all weak learners to make a strong final model.

Results

We have achieved an overall accuracy of 75.91%, with Class 1 yielding a precision of 0.74, recall of 0.64, and an F1-score of 0.69

Class	precision	recall	f1-score	support
0	0.76	0.84	0.79	1628
1	0.74	0.64	0.69	1210
accuracy			0.75	2838
macro avg	0.75	0.74	0.74	2838
weighted avg	0.75	0.75	0.75	2838

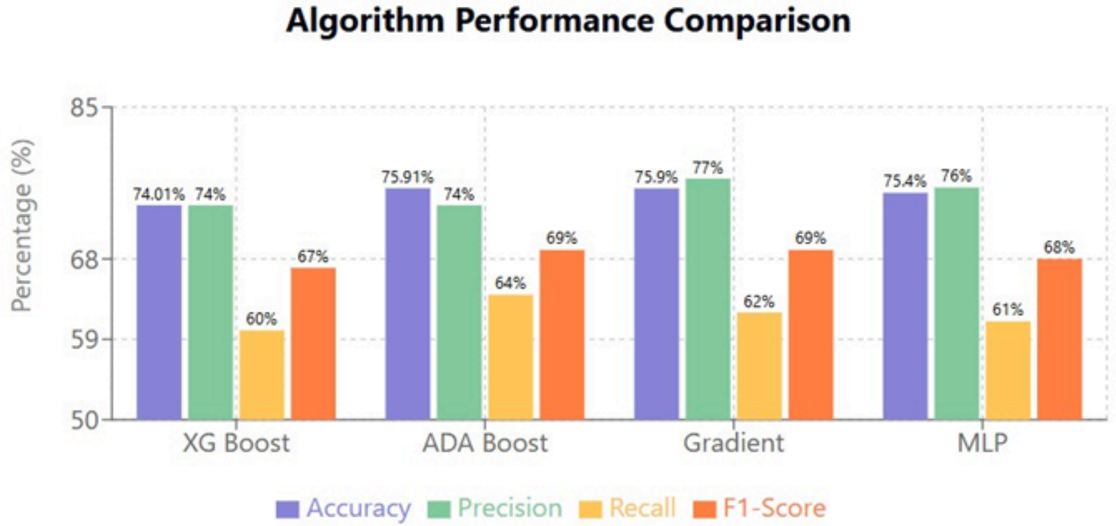
4.MODEL TRAINING

After splitting the dataset, typically in an 80-20 ratio, the model is trained on the training set and evaluated on the testing set to effectively measure its performance on unseen data

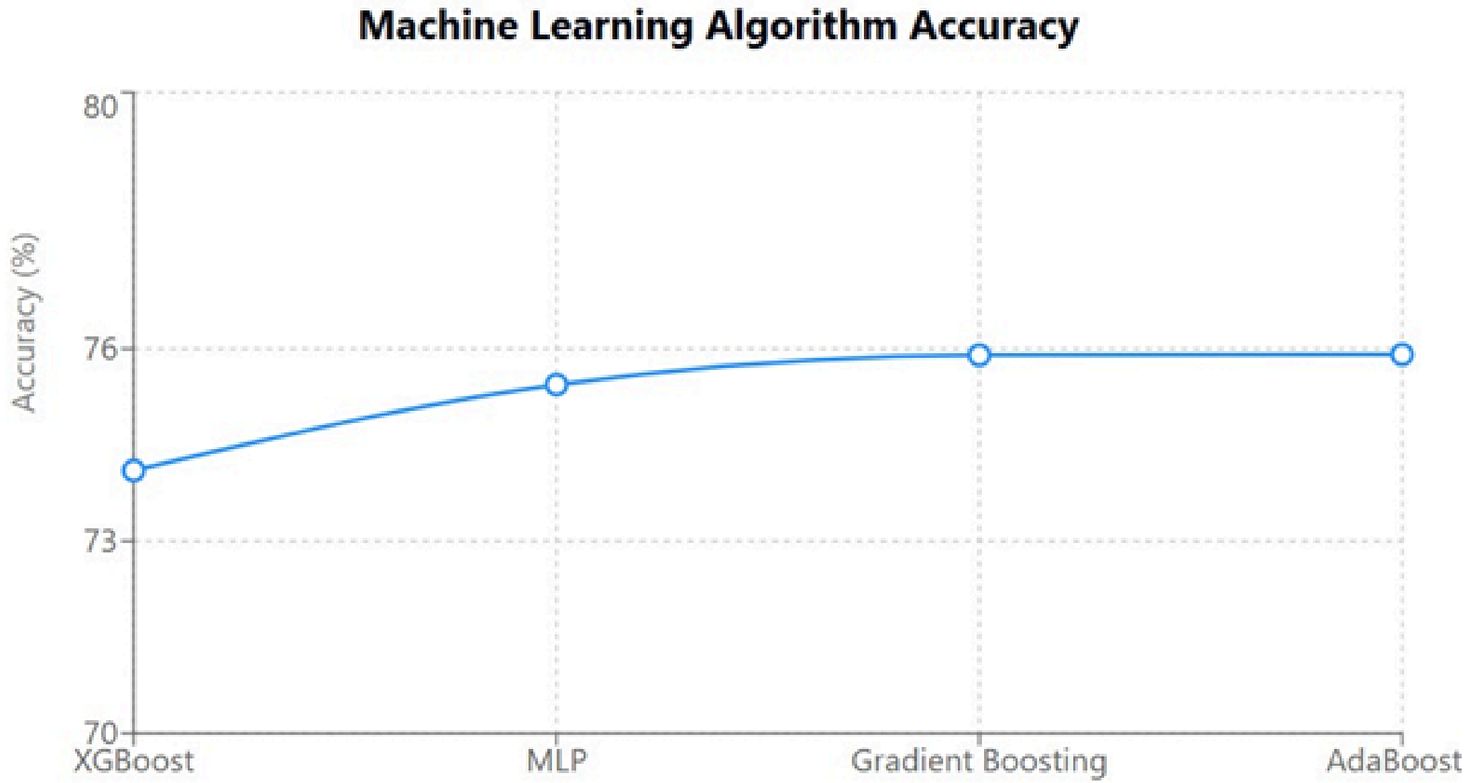
- Train the selected algorithms on the training data, adjusting parameters to optimize performance.

5.MODEL EVALUATION

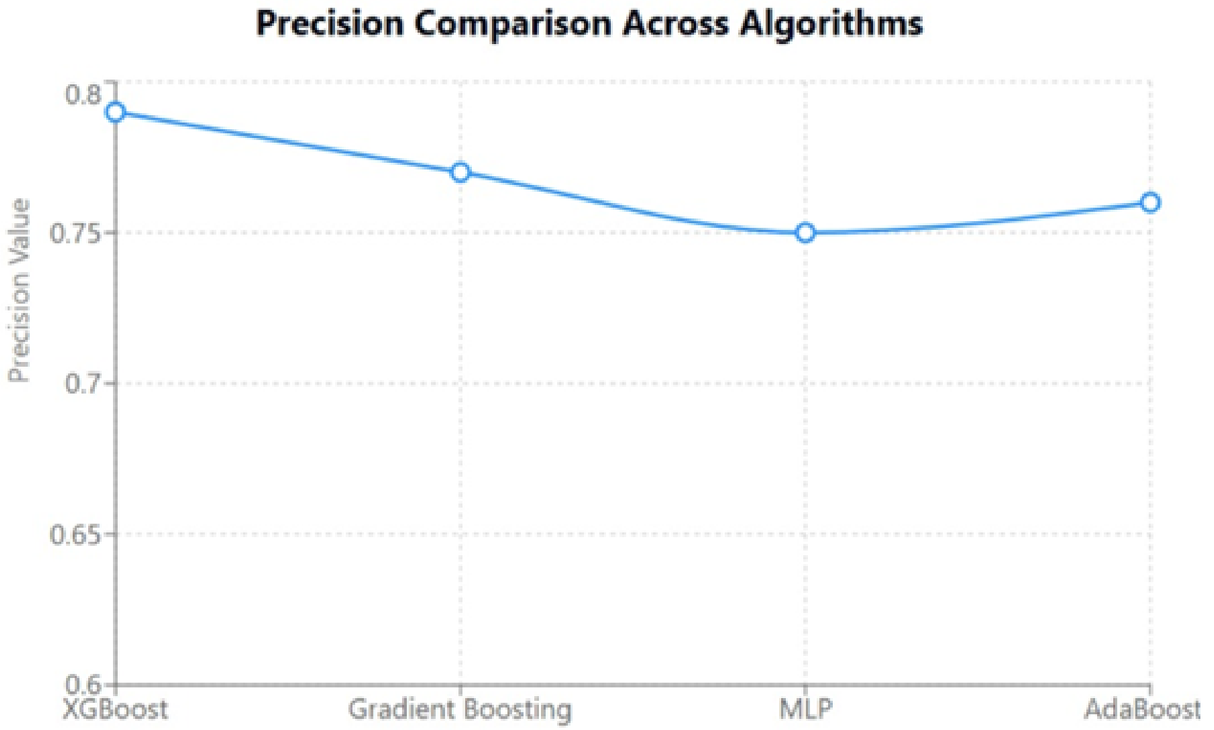
To assess the effectiveness of our models, we employed key performance metrics including accuracy, precision, recall, F1-score, and the Area Under the ROC Curve (AUC). Additionally, we implemented k-fold cross-validation to ensure robustness and generalizability to unseen data. Among the three algorithms evaluated, AdaBoost demonstrated the highest accuracy of 75.91%, making it the most effective model based on our evaluation criteria.



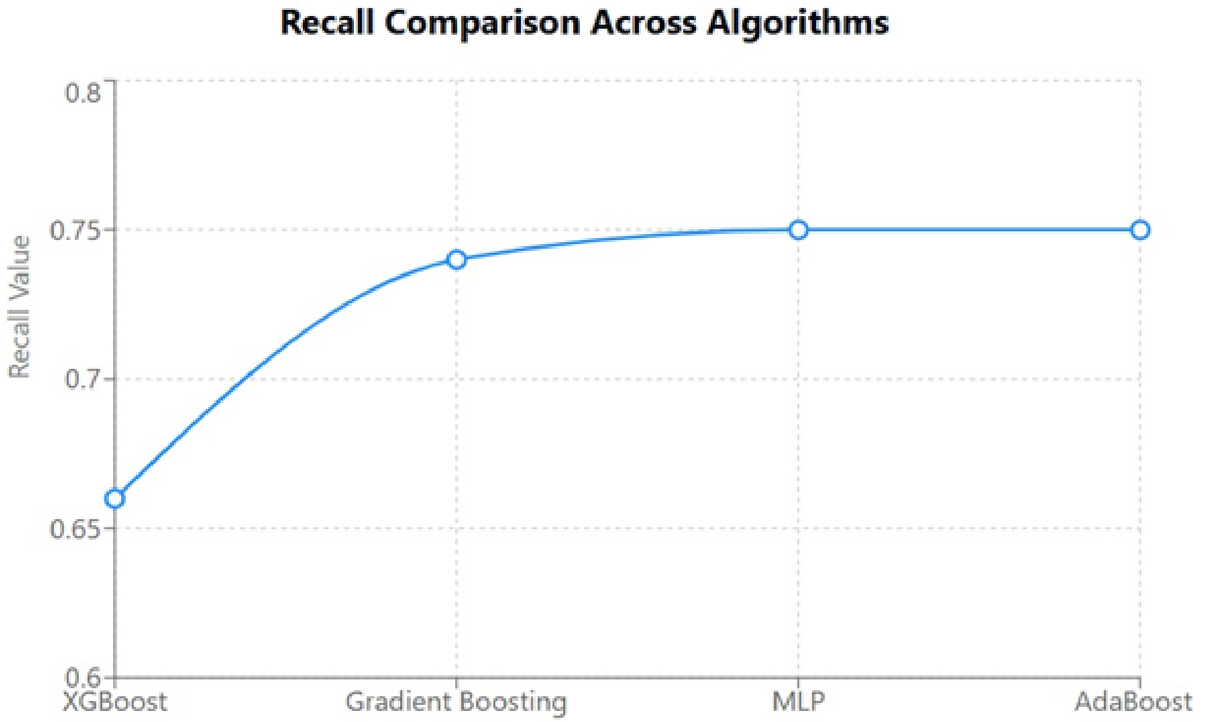
ACCURACY COMPARISON



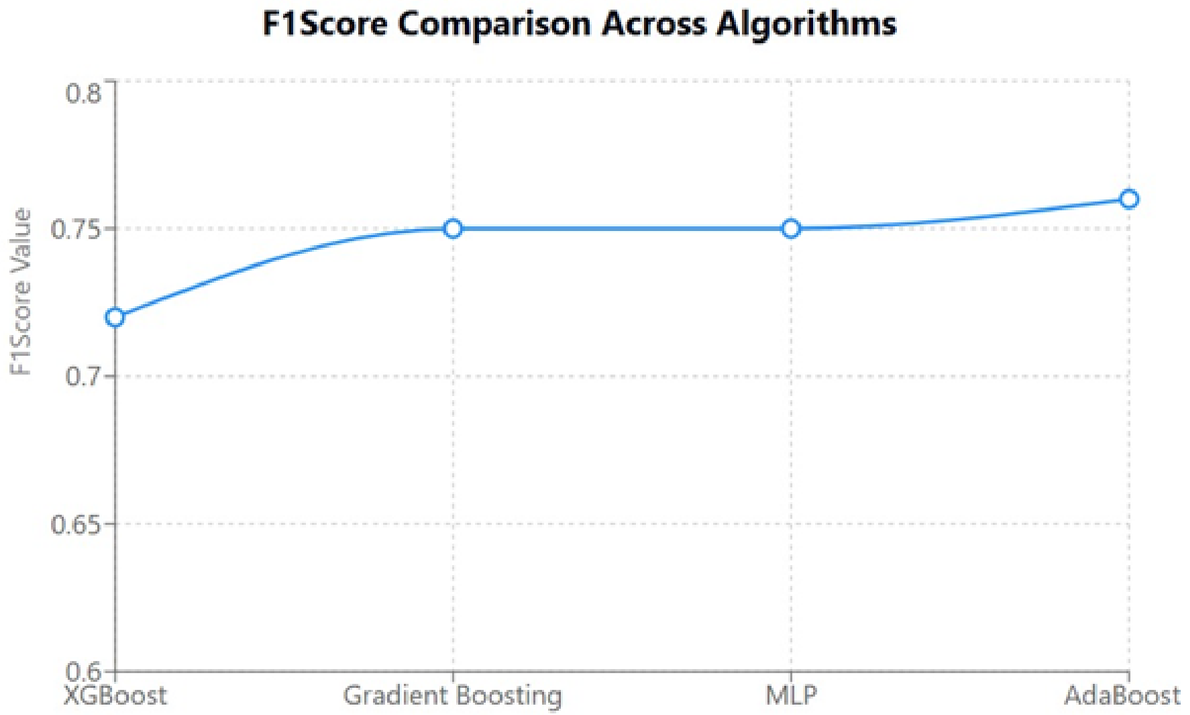
PRECISION COMPARISON



RECALL COMPARISON



F1 SCORE COMPARISON



6. Deployment:

- Integration: Implement the trained model into a user-friendly application or interface, enabling healthcare professionals to input patient data and receive predictions.

- Monitoring: Continuously monitor the model's performance in a real-world setting, updating it as necessary to maintain accuracy and relevance.

7. Frontend

Frontend Overview

Technologies Used:

- HTML, CSS, JavaScript

Design Principles:

- Modular, responsive, and user-friendly
- Medical-themed aesthetic
- Cross-device compatibility

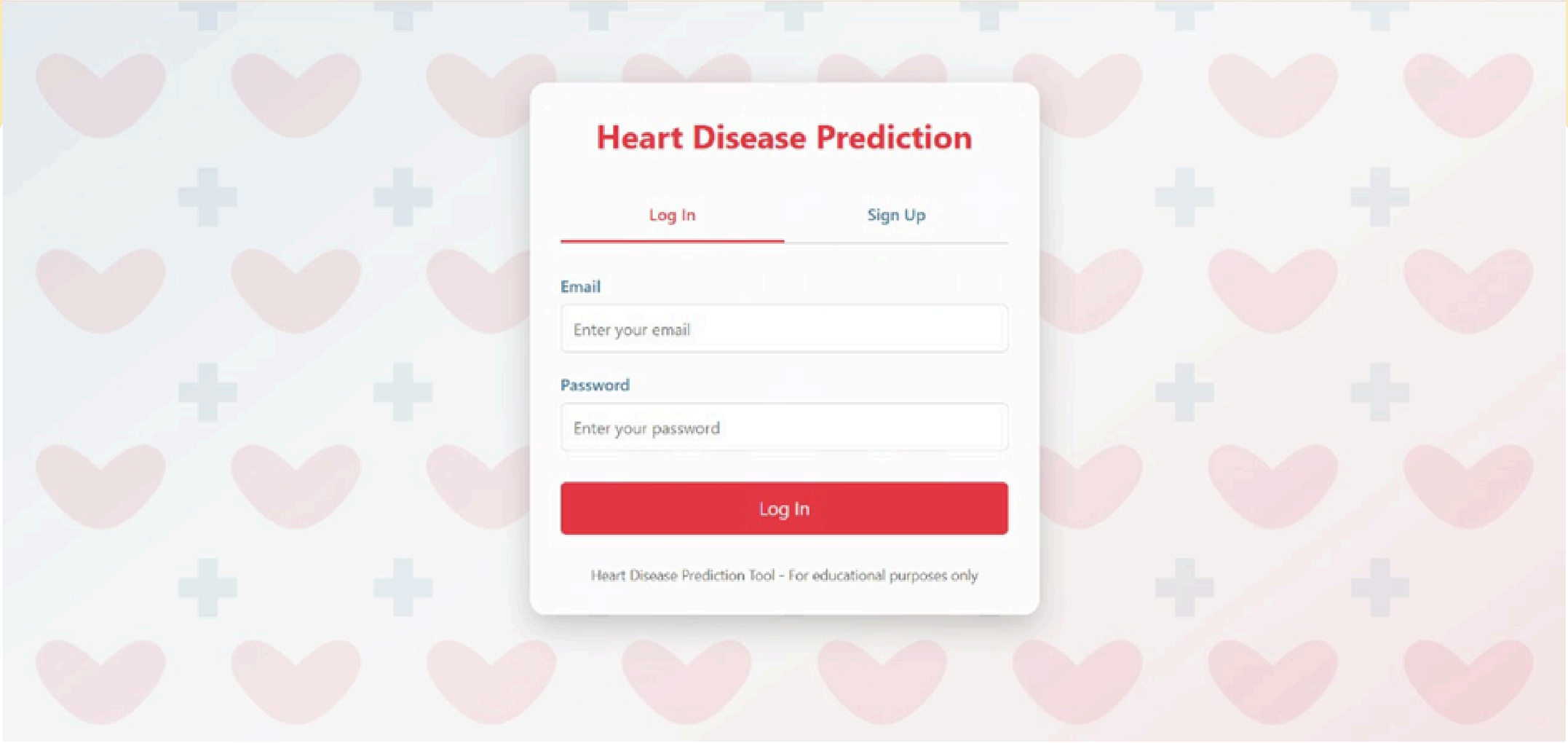
Login Page (login.html)

- Tab-based login/signup interface
- Email/password validation
- Password strength indicator
- Flash messaging system
- Responsive mobile-friendly design

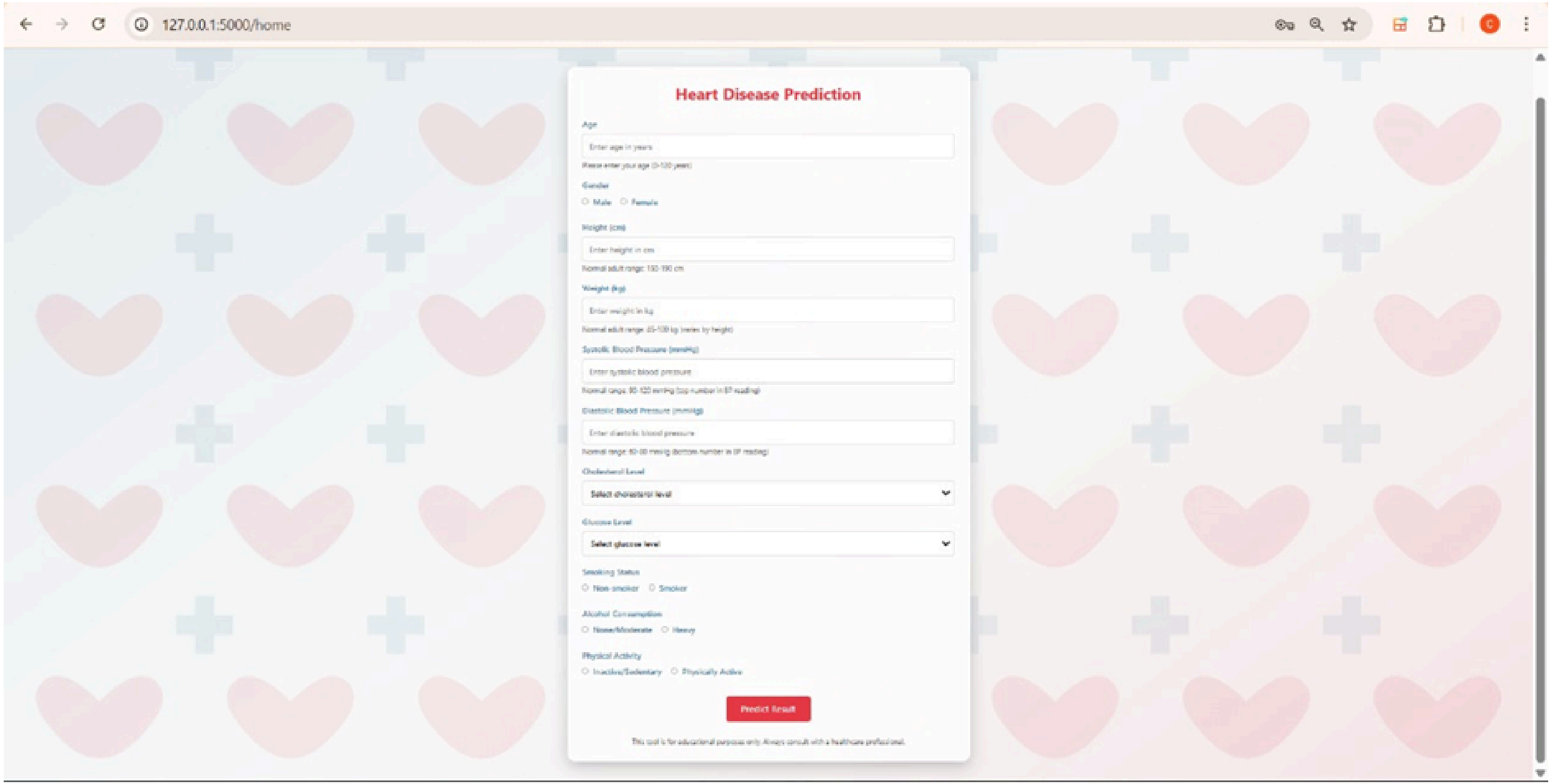
Error Page (error.html)

- User-friendly error messages
- Return-to-home navigation
- Consistent app styling

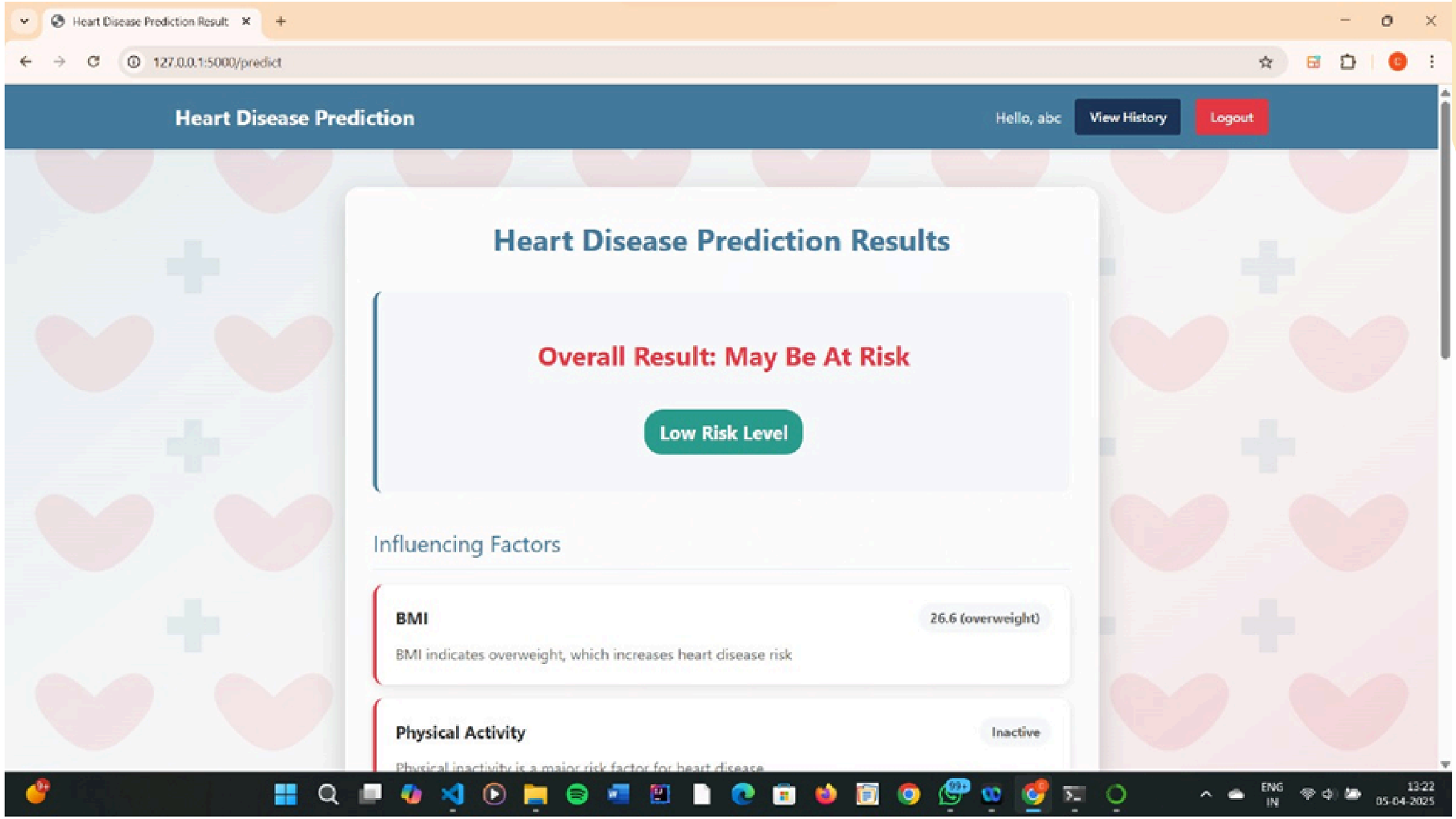
- **Home Page (home.html)**
- Form inputs: age, gender, height, weight
- Blood pressure, cholesterol, glucose inputs
- Lifestyle factors: smoking, alcohol, activity
- Input validation with min/max limits
- Normal range hints & tooltips
- Navigation header with greeting & logout
- **Results Page (result.html)**
- Prediction outcome (at risk/not at risk)
- Risk level: low, medium, high
- Model-wise confidence levels
- Personalized risk factor breakdown
- Health recommendations (BP, cholesterol, etc.)
- Color-coded indicators for clarity
- Navigation to history/retry options



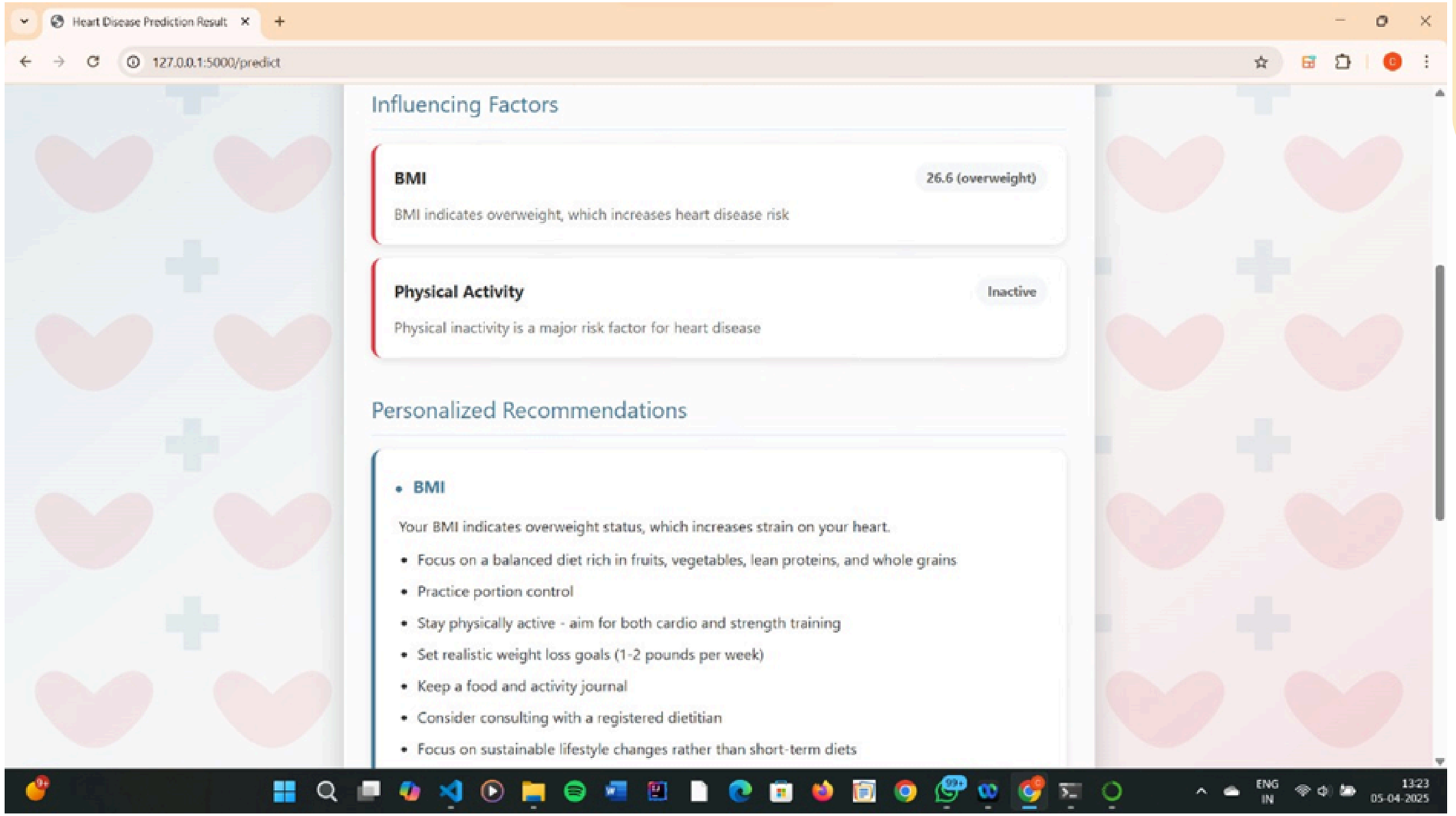
Screenshot of Login Page

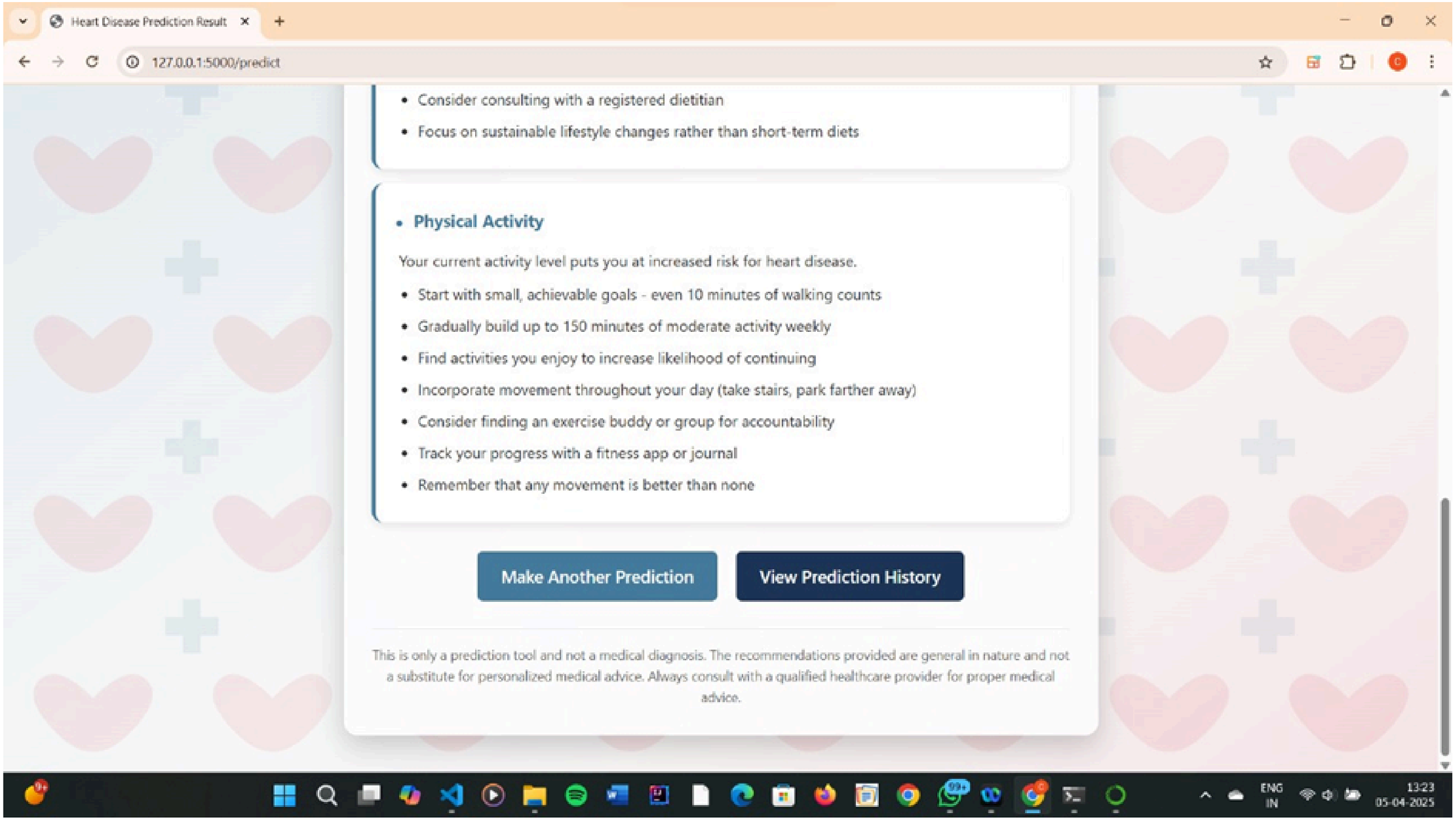


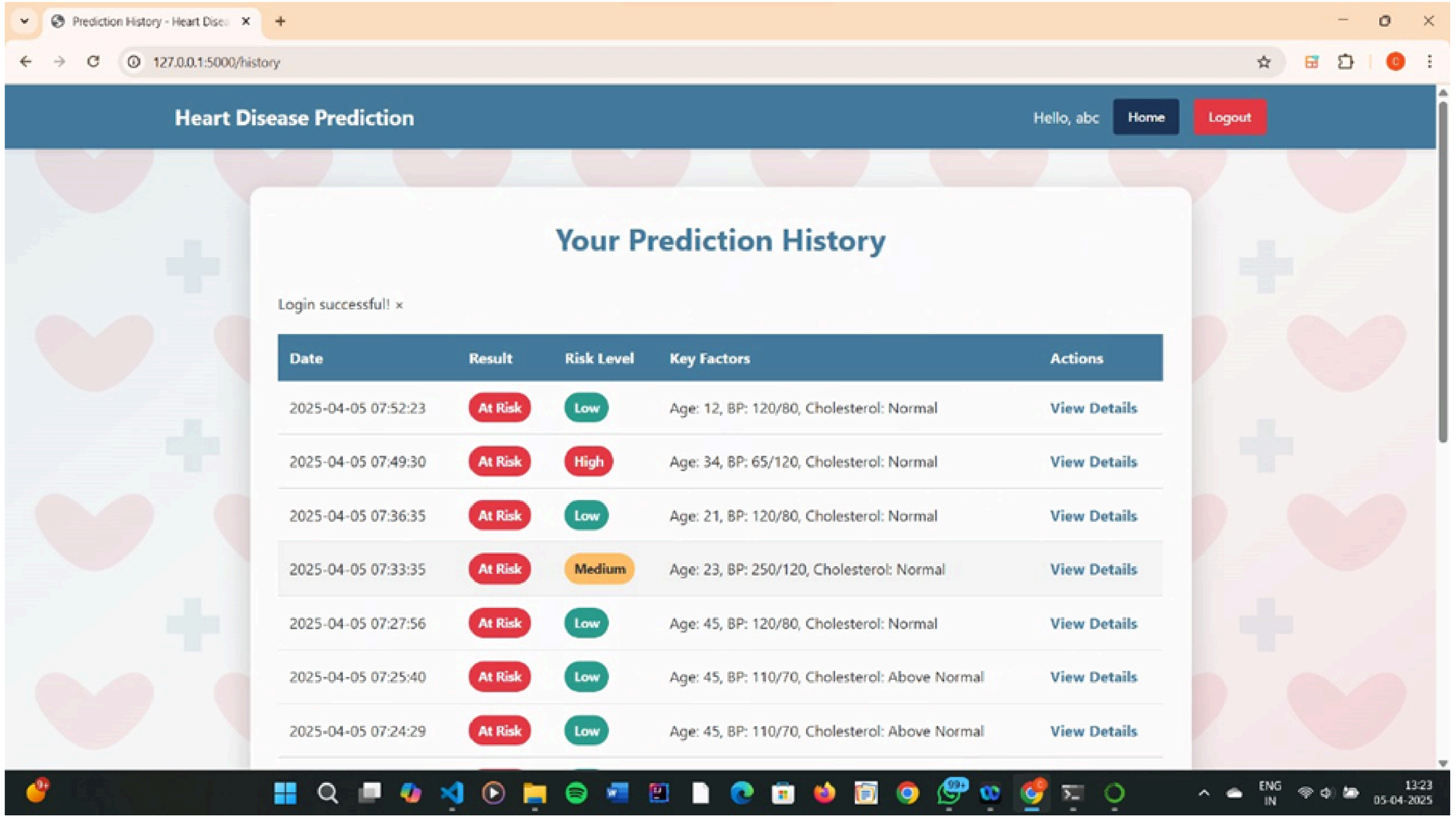
Screenshot of Prediction Page



Screenshot of Results Page


Heart Disease Prediction Result
127.0.0.1:5000/predict
Influencing Factors
BMI
26.6 (overweight)
BMI indicates overweight, which increases heart disease risk
Physical Activity
Inactive
Physical inactivity is a major risk factor for heart disease
Personalized Recommendations
BMI
Your BMI indicates overweight status, which increases strain on your heart.
Focus on a balanced diet rich in fruits, vegetables, lean proteins, and whole grains
Practice portion control
Stay physically active - aim for both cardio and strength training
Set realistic weight loss goals (1-2 pounds per week)
Keep a food and activity journal
Consider consulting with a registered dietitian
Focus on sustainable lifestyle changes rather than short-term diets
ENG
IN
13:23
05-04-2025


Heart Disease Prediction Result
127.0.0.1:5000/predict
Consider consulting with a registered dietitian
Focus on sustainable lifestyle changes rather than short-term diets
Physical Activity
Your current activity level puts you at increased risk for heart disease.
Start with small, achievable goals - even 10 minutes of walking counts
Gradually build up to 150 minutes of moderate activity weekly
Find activities you enjoy to increase likelihood of continuing
Incorporate movement throughout your day (take stairs, park farther away)
Consider finding an exercise buddy or group for accountability
Track your progress with a fitness app or journal
Remember that any movement is better than none
Make Another Prediction
View Prediction History
This is only a prediction tool and not a medical diagnosis. The recommendations provided are general in nature and not a substitute for personalized medical advice. Always consult with a qualified healthcare provider for proper medical advice.
ENG
IN
13:23
05-04-2025



Screenshot of Prediction History Page

5. Backend Integration

Backend Integration

- Framework Used: Flask (Python)
- Architecture: Full-stack, modular design

Model Handling:

- Load ML models at startup
- Generate predictions from multiple models
- Final result via consensus logic

Prediction Pipeline

- Input processing & validation
- Real-time risk score calculation
- Risk level categorization (Low/Medium/High)
- Identify risk factors for suggestions

User Authentication

- User signup with password hashing
- Login verification with session management
- Secure logout functionality

OUTCOME

- Model Development
  - Successfully built heart disease prediction system with multiple ML algorithms. AdaBoost: 75.91%, Gradient Boosting: 75.90%, Neural Network (MLP): 78.45%, XGBoost: 81.05%
  - AdaBoost achieved highest accuracy (75.91%)
- Top predictors: cholesterol, weight, systolic BP, diastolic BP, smoking status
- Frontend Interface
  - User-friendly design for easy patient data input
  - Real-time risk prediction with intuitive visualization

- Risk Analysis
  - Personalized risk assessment for each patient
  - Explains contribution of each feature to predicted risk
  - Provides actionable health recommendations based on risk factors
- Patient Management
  - Maintains assessment history for tracking risk changes
  - Supports long-term monitoring and treatment planning

INDIVIDUAL CONTRIBUTION

ABHINAV SHRIVASTAVA - DATA COLLECTION & PREPROCESSING

- Acquired 9,776-patient dataset with 13 features
- Conducted missing value & outlier analysis (IQR method)
- Removed 319 outliers (9,457 final records)
- Applied Min-Max scaling for normalization
- Ensured balanced class distribution (43.4% positive)

SONALI RAGHUWANSHI - FEATURE SELECTION & MODELS

- Identified 5 key predictors (BP, cholesterol, smoking, weight)
- Implemented 4 ML algorithms:
- Neural Network: 75.44%
- AdaBoost: 75.91% (highest)
- XGBoost: 74.10%
- Gradient Boosting: 75.90%

TINA CHELWANI - LOGIN & INPUT INTERFACES

- Designed responsive authentication system
- Created health data collection form with validation
- Implemented medical-themed UI with tooltips
- Developed user session management

PRIYANSHI YADAV - RESULTS & HISTORY PAGES

- Created risk visualization dashboard
- Designed personalized recommendations section
- Developed prediction history tracking interface
- Implemented error handling pages

ARCHITA GUPTA - BACKEND DEVELOPMENT

- Built complete Flask application framework
- Created model loading & prediction pipeline
- Developed risk assessment algorithm
- Implemented user authentication system
- Designed SQLite database for user data & history

THANK YOU