

# TABLE OF CONTENT

S.NO	TOPIC	PAGE NO.
1	Abstract	6
2	List of figures	7
3	List of Abbreviation	8
4	CHAPTER 1: PROBLEM FORMULATION: - 1.1 Introduction about the Company 1.2 Introduction about the Problem 1.3 Present State of the Art 1.4 Need of Computerization 1.5 Proposed Software 1.6 Importance of the work	9
5	CHAPTER 2: SYSTEM ANALYSIS: - 2.1. Feasibility study 2.1.1 Technical feasibility 2.1.2 Economical feasibility 2.1.3 Operational feasibility 2.1.4 Other feasibility dimentions 2.2 Analysis methodology 2.3 Choice of problem 2.3.1 Software used 2.3.2 Hardware used	14
6	CHAPTER 3: SYSTEM DESIGN: - 3.1 Design methodology 3.2 Database design 3.2.1 DFD 3.3 Input design 3.4 Output design	19
7	CHAPTER 4: TESTING AND IMPLEMENTATION	25

	4.1 Test methodology 4.1.1 Unit testing 4.1.2 Module testing 4.1.3 Integration testing 4.1.4 System testing 4.1.5 White box testing 4.1.6 Black box testing 4.2 Future testing for heart disease prediction	
8	CHAPTER 5: CONCLUSION AND REFERENCES: 5.1. Conclusion: 5.2. System specifications: 5.2.1. Hardware Requirements 5.2.2. Software Requirements	29
9	CHAPTER 6: ANNEXURES: A-11 Coding A-10 Sample Output	33

# **ABSTRACT**

## **Heart Disease Prediction through Exploratory Data Analysis**

Heart disease remains one of the leading causes of mortality worldwide, making its early prediction a critical aspect of preventive healthcare. This study employs Exploratory Data Analysis (EDA) to uncover patterns, trends, and relationships in clinical datasets to enhance the prediction of heart disease. By analyzing key features such as age, cholesterol levels, blood pressure, and lifestyle habits, EDA offers a comprehensive understanding of the underlying data and its impact on heart disease diagnosis.

The research begins with detailed data preprocessing, including handling missing values, outlier detection, and encoding categorical variables. Visualization techniques such as histograms, box plots, and correlation heatmaps are used to identify significant features and relationships. Statistical methods, including hypothesis testing and correlation analysis, further validate these findings. Feature engineering, such as grouping age ranges and normalizing continuous variables, enhances the dataset's quality and predictive capability.

The results of EDA highlight critical predictors of heart disease, including cholesterol levels, age, and resting blood pressure, and reveal patterns such as gender-based risk disparities. This analysis provides actionable insights for healthcare practitioners, facilitating early interventions and risk stratification. Moreover, the study establishes a robust foundation for predictive modeling by selecting the most impactful features.

In conclusion, EDA proves to be a powerful tool in the initial stages of heart disease prediction, enabling data-driven decision-making and supporting the development of effective machine-learning models. This approach emphasizes the importance of understanding data before embarking on predictive analytics in healthcare.

## **LIST OF FIGURES**

<b>Figure No.</b>	<b>Description</b>
3.1	DFD L-0
3.2	DFD L-1
3.3	DFD L-2

### **LIST OF ABBREVIATIONS**

<b>Abbreviation</b>	<b>Description</b>
IBM	International Business Machines Corporation
CTR	Computing-Tabulating-Recording Company
AI	Artificial intelligence
UPC	Universal Product Code
CVDs	Cardiovascular diseases
WHO	World Health Organization
SDGs	Sustainable development goals
EDA	Exploratory Data Analysis
EHRs	Electronic health records
DFD	Data flow diagram
API	Application programming interface
UI	User interface

## **CHAPTER 1: PROBLEM FORMULATION**

### **1.1. Introduction about the company:**

**IBM**, or International Business Machines Corporation, is a globally recognized technology and consulting company with headquarters in Armonk, New York. Established in 1911 as the Computing-Tabulating-Recording Company (CTR) and rebranded as IBM in 1924, the company has been a pioneer in technological innovation for over a century. Its contributions have transformed industries and set new standards in computing and information technology.

IBM specializes in various core domains, including cloud computing, artificial intelligence (AI), quantum computing, software development, IT infrastructure, and consulting services. IBM Watson, the company's flagship AI platform, has revolutionized sectors such as healthcare and finance by providing advanced data analytics and machine learning solutions. IBM Cloud offers secure, scalable, and efficient resources, catering to modern enterprises' needs. Additionally, IBM Quantum leads the way in quantum computing, offering groundbreaking technology for research and development.

A hallmark of IBM's legacy is its unwavering commitment to innovation. It consistently ranks among the world's top patent holders, with significant advancements in AI, blockchain, and semiconductor technology. The company has been instrumental in key technological milestones, including the development of the first hard disk drive, relational databases, and the Universal Product Code (UPC). These innovations underscore IBM's role as a global leader in technological progress.

Operating in over 170 countries, IBM serves a wide range of industries, including healthcare, finance, retail, and government. Its global reach and expertise enable the company to deliver tailored solutions that address localized challenges while maintaining a universal focus on innovation and excellence. Guided by its mission to foster progress and empower businesses worldwide, IBM continues to shape the future of technology while addressing critical global challenges.

### **1.2. Introduction about the problem:**

Heart disease is a significant global health issue that impacts millions of lives annually and places a considerable burden on healthcare systems worldwide. Despite advancements in medical science and public health initiatives, cardiovascular diseases (CVDs) remain the leading cause of death globally. According to the World Health Organization (WHO), heart disease accounts for approximately 32% of all deaths globally, with an estimated 17.9 million lives lost each year. This issue highlights the critical need for targeted interventions and improved healthcare strategies to combat its prevalence.

The burden of heart disease arises from various factors across multiple levels, including genetic predisposition, lifestyle choices, healthcare disparities, and socio-economic conditions. In low- and middle-income countries, limited access to preventive healthcare, diagnostic facilities, and treatment options exacerbates the impact of heart disease. Conversely, in high-income countries, lifestyle factors such as unhealthy diets, sedentary behavior, smoking, and stress contribute significantly to the prevalence of cardiovascular conditions. Additionally, risk factors like hypertension, diabetes, and high cholesterol amplify the likelihood of developing heart disease.

Addressing heart disease is crucial for achieving sustainable development goals (SDGs), particularly SDG 3 (Good Health and Well-being). By leveraging data-driven approaches and focusing on prevention, early detection, and better management of heart disease, we can reduce mortality rates, alleviate healthcare burdens, and improve overall public health outcomes. These efforts also align with the broader goals of promoting equitable access to healthcare and fostering a healthier global population.

### 1.3. Present state of the art:

Addressing human heart disease remains a priority for researchers, healthcare professionals, and policymakers worldwide. Exploratory Data Analysis (EDA) plays a crucial role in understanding patterns, trends, and key risk factors associated with cardiovascular diseases (CVDs). The present state of the art in EDA and related technologies for heart disease includes:

1. **Big Data and Electronic Health Records (EHRs):** Large-scale patient datasets, often derived from EHRs, provide comprehensive insights into cardiovascular risk factors, treatment outcomes, and population health trends. These datasets enable researchers to identify correlations and patterns at a granular level.
2. **Machine Learning for Pattern Discovery:** Advanced machine learning algorithms are employed to explore associations between variables, detect anomalies in data, and uncover hidden risk factors for heart disease. Techniques such as clustering, classification, and feature selection are commonly used in the EDA process.
3. **AI-Powered Predictive Models:** Artificial intelligence helps identify high-risk individuals by analyzing EDA findings and building models based on historical patient data. These models are used to predict the likelihood of heart disease and prioritize early interventions.
4. **Visualization Tools:** Tools like Tableau, Matplotlib, and Seaborn are integral to EDA, providing interactive and static visualizations of data distributions, correlations, and trends. Visualization aids in interpreting complex datasets and presenting findings to healthcare stakeholders.
5. **Wearable Devices and IoT Sensors:** Wearables like smartwatches and fitness trackers collect real-time data on heart rate, blood pressure, and physical activity. Analyzing this data helps detect early warning signs of heart disease and provides insights into lifestyle impacts on cardiovascular health.

Despite these advancements, challenges persist. The diversity and complexity of heart disease data, coupled with privacy concerns and ethical considerations, make data collection and analysis challenging. Moreover, integrating EDA insights into clinical practice requires robust collaboration between data scientists, healthcare providers, and policymakers. However, ongoing innovation in EDA methods and tools promises to enhance our understanding of heart disease and improve patient outcomes.

### 1.4. Need of computerization

In today's healthcare landscape, computerization is essential for effectively managing and addressing the complex challenges associated with human heart disease. The need for computerization in the context of exploratory data analysis (EDA) on heart disease is driven by several critical factors:

1. **Efficient Data Management:** Cardiovascular research generates vast amounts of data from electronic health records (EHRs), wearable devices, imaging studies, and genetic databases. Computerized systems are necessary to store, organize, and process this data for efficient analysis.
2. **Real-Time Monitoring:** Advanced technologies, including IoT-enabled devices and sensors, enable real-time monitoring of vital signs such as heart rate, blood pressure, and ECG readings. This real-time data helps detect anomalies and provides actionable insights to prevent adverse events.
3. **Predictive Analytics:** Computerized models utilize historical and real-time patient data to predict heart disease risk, allowing for proactive management of high-risk individuals and early intervention strategies to mitigate progression.
4. **Enhanced Diagnostics:** Machine learning algorithms and image processing tools improve the accuracy and speed of diagnosing cardiovascular conditions, such as detecting blockages or anomalies in heart function from medical imaging.
5. **Personalized Medicine:** Computerization supports data-driven decision-making by analyzing patient-specific data to develop personalized treatment plans. Insights from EDA help tailor interventions to individual risk profiles and medical histories.

### 1.5. Proposed project:

Project Title: Exploratory Data Analysis on Human Heart Disease (SDG 3:Good Health and Well-being)

The proposed project aims to address the global challenge of heart disease by leveraging data analytics to identify patterns, predict cardiovascular risks, and suggest actionable solutions for prevention and improved healthcare outcomes. This initiative aligns with global health goals to reduce mortality and morbidity associated with cardiovascular diseases.

#### Key Features

1. **Data Collection and Analysis:**
  - Collect comprehensive datasets on patient demographics, lifestyle factors, clinical measurements, and medical histories from sources like hospitals, public health records, and wearable devices.
  - Analyze trends in cardiovascular risk factors across different demographics, regions, and time periods to uncover patterns in heart disease prevalence.
2. **Predictive Models:**
  - Utilize machine learning algorithms such as logistic regression, decision trees, and neural networks to predict the likelihood of heart disease based on risk factors like age, cholesterol levels, and blood pressure.
  - Build predictive models to forecast the progression of heart disease and identify individuals at high risk for targeted interventions.
3. **Visualization:**
  - Develop interactive dashboards to represent cardiovascular health metrics, trends, and predictive insights.
  - Use visualizations such as heatmaps, scatter plots, and correlation matrices to highlight relationships between risk factors and heart disease outcomes.
4. **Optimization Algorithms:**
  - Create algorithms to optimize resource allocation in healthcare, ensuring early detection and treatment of heart disease.



- Design models to optimize patient care pathways and minimize delays in diagnosis and treatment.
- 5. Public Awareness and Education:**
- Develop an interactive platform to educate the public on heart disease prevention, healthy lifestyle choices, and the importance of regular health check-ups.
  - Offer resources on managing risk factors like hypertension, diabetes, and smoking cessation through personalized recommendations.

## Components

- 1. Data Sources:**
  - Medical databases, electronic health records (EHRs), and clinical research datasets.
  - Data from wearable devices and remote health monitoring systems tracking vitals like heart rate and activity levels.
- 2. Data Processing:**
  - Perform data cleaning and preprocessing to handle missing values, outliers, and inconsistencies in health records.
  - Extract and normalize features such as cholesterol levels, BMI, and blood pressure for use in machine learning models.
- 3. Model Building:**
  - Logistic regression and random forest models for classifying individuals into risk categories.
  - Deep learning models to detect complex patterns in ECG data or medical imaging for early disease diagnosis.
- 4. Deployment:**
  - Develop a user-friendly web or mobile application using frameworks like Flask or Django for patients and healthcare providers to access insights and recommendations.
  - Integrate predictive tools into hospital systems or public health platforms for real-time use.
- 5. Impact Assessment:**
  - Evaluate the project's success through key performance indicators (KPIs), such as reduced cardiovascular mortality rates, increased early diagnosis rates, and improved patient adherence to preventive measures.

By combining exploratory data analysis with actionable insights, this project aims to enhance the understanding of heart disease, drive preventive healthcare initiatives, and contribute to global health goals, including reduced cardiovascular mortality and improved quality of life.

## 1.6. Importance of the work:

- 1. Early Detection and Prevention of Heart Disease:**  
This project directly contributes to reducing the global burden of cardiovascular diseases by enabling early detection of heart disease risk factors. Early interventions can prevent disease progression, improving patient outcomes and reducing the prevalence of heart-related complications.
- 2. Improving Healthcare Decision-Making:**  
By leveraging advanced data analytics and machine learning, the project empowers

healthcare providers with actionable insights. This supports more accurate diagnosis, personalized treatment plans, and better resource allocation within healthcare systems.

3. **Enhancing Public Awareness and Health Education:**  
The project raises awareness about the importance of heart health by educating individuals on managing risk factors such as smoking, poor diet, and physical inactivity. Improved public understanding encourages proactive lifestyle changes, reducing the likelihood of heart disease.
4. **Reducing Healthcare Costs:**  
Heart disease is a leading cause of healthcare expenditure worldwide. Predictive models that focus on early diagnosis and prevention can significantly lower treatment costs by reducing the need for expensive procedures, hospitalizations, and long-term care.
5. **Promoting Data-Driven Healthcare Practices:**  
The project supports the transition to data-driven healthcare by integrating predictive analytics into clinical workflows. This fosters innovation and enhances the ability of medical professionals to address complex cardiovascular conditions.

Through these contributions, the project addresses critical aspects of heart disease management, improving individual health outcomes, advancing healthcare systems, and supporting global health priorities.

## **CHAPTER 2: SYSTEM ANALYSIS**

### **2.1. Feasibility Study:**

The feasibility study indicates that the project is technically viable using advanced data analytics and statistical techniques, economically beneficial by potentially reducing healthcare costs, and operationally scalable with partnerships in the medical and research community. Socially, it aligns with increasing awareness of cardiovascular health, and legal and ethical factors, along with environmental considerations, support its implementation. Overall, the project is practical and holds significant potential to improve public health outcomes.

#### **2.1.1 Technical Feasibility**

Objective: Assess whether the project can be implemented from a technical perspective.

- **Data Availability:** Historical and current datasets on heart disease, including clinical trials, hospital records, and open data repositories (e.g., UCI Heart Disease dataset, Framingham Heart Study), are available.
- **Technology Stack:** The required infrastructure is accessible, including data processing tools (e.g., Python, R) and visualization platforms (e.g., Tableau, Power BI).
- **Expertise:** The project requires skills in data science, statistical analysis, and domain knowledge in cardiology.

#### **2.1.2 Economical Feasibility**

- **Cost of Data Acquisition:** Free datasets are accessible through public repositories; additional data can be acquired via collaborations with hospitals and research organizations.
- **Technology and Infrastructure Costs:** Open-source libraries (e.g., TensorFlow, Pandas) and free tools like Google Colab minimize software and computational expenses.
- **Operational Costs:** Analytical tasks can be carried out using an internal team of data analysts and medical consultants, supplemented by external collaborations if needed.
- **Return on Investment (ROI):** Insights from EDA can reduce healthcare costs by early identification of high-risk groups and preventing complications.
- **Scalability:** The analysis framework can scale to accommodate larger datasets and integrate with predictive modeling tools with minimal additional costs.

#### **2.1.3 Operational Feasibility**

Objective: Determine whether the project can be smoothly integrated into existing operations.

- **Considerations:**
  - **Data Integration:** Evaluate the feasibility of incorporating heart disease datasets into existing healthcare IT systems or research tools.
  - **Scalability:** Assess whether the tools and models used for EDA can handle growing data volumes and complexity as the project expands.
  - **User Acceptance:** Understand the readiness of stakeholders, such as healthcare providers and public health officials, to use insights generated by the EDA in decision-making processes.

### 2.1.4 Other Feasibility Dimensions

- Legal and Ethical Feasibility: Compliance with healthcare data privacy regulations such as HIPAA and GDPR is essential. Implement measures to anonymize patient data and secure sensitive information during storage and analysis.
- Environmental Feasibility: Minimize the computational footprint by optimizing algorithms and leveraging energy-efficient data centers for cloud-based computations.

## 2.2. Analysis methodology:

This methodology allows for efficient analysis and understanding of human heart disease, providing actionable insights for risk assessment, prevention, and treatment strategies.

### 1. Importing the Necessary libraries and loading the Dataset:

```
from google.colab import files
uploaded = files.upload()
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv('heart.csv')
print("DATA\n",df)
```

### 2. Data cleaning:

```
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)
```

```
print("different set of age groups\n",df['age'].unique(),'\n')
print("types of genders\n",df['sex'].unique(),'\n')
print("chest pain types\n",df['cp'].unique(),'\n')
print("range of resting blood pressure\n",df['trestbps'].unique(),'\n')
print("number of vessels\n",df['ca'].unique(),'\n')
print("is the disease curable\n",df['thal'].unique())
```

### 3. Descriptive Statistical Analysis:

```
print('\n shape of data\t',df.shape)
print('\n info of data\t',df.info())
print('\n describe data\t',df.describe())
```

```
targets = df['target'].value_counts()
targets
```

#### 4. Exploratory Data Analysis:

```
targets.plot(
    kind='bar',
    color=['salmon', 'lightblue'],
    figsize=(10,6)
)
plt.xticks(rotation=0)
plt.show()
```

```
import matplotlib.pyplot as plt

# Histogram of Age Distribution
plt.hist(df['age'], bins=20, color='pink', edgecolor='purple')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Age Distribution')
plt.grid(True)
plt.show()
```

```
sns.histplot(df.thalach, bins=20, kde=True, color='blue', edgecolor='black')
# The Best number of bits is chosen based on Rice Criterion
plt.show()
```

```
df['target'].value_counts().plot(kind='bar', color=['green', 'grey'])
plt.xlabel('Heart Disease')
plt.ylabel('Count')
plt.title('Presence of Heart Disease (0 = No, 1 = Yes)')
plt.xticks(rotation=0)
plt.show()
```

```
plt.figure(figsize=(10, 6))
sns.countplot(x='sex', data=df)
plt.title('Distribution of Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

#### 5. Hypothesis

```
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='age', hue='target', multiple='stack', palette='viridis')
plt.title('Distribution of Age with Heart Disease')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.legend(title='Heart Disease', labels=['No', 'Yes'])
plt.show()
```

```

df['sex'] = df['sex'].map({0: 'Female', 1: 'Male'})
# Distribution of Heart Disease by Gender
plt.figure(figsize=(10, 6))
sns.countplot(x='sex', hue='target', data=df)
plt.title('Distribution of Heart Disease by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.legend(title='Heart Disease', loc='upper right', labels=['No', 'Yes'])
plt.show()

gender_heart_disease = df.groupby('sex')['target'].mean().reset_index()

# Heart Disease Prevalence by Gender
plt.figure(figsize=(10, 6))
sns.barplot(x='sex', y='target', data=gender_heart_disease, palette='viridis')
plt.title('Heart Disease Prevalence by Gender')
plt.xlabel('Gender')
plt.ylabel('Prevalence (Proportion)')
plt.show()

```

### 2.3. Choice of the platforms:

Given below combination of platforms, software, and hardware enables the efficient execution of exploratory data analysis on human heart disease, allowing for insights into risk factors and disease patterns.

#### ☐ Frameworks:

- Python: Widely used for data analysis and machine learning, with libraries like Pandas, Scikit-learn, TensorFlow, and Keras for preprocessing, modeling, and prediction tasks.
- Jupyter/Google Colab: Ideal for interactive coding, rapid prototyping, and collaboration, with free access to GPUs for model training and exploratory analysis.

#### ☐ Data Visualization Tools:

- Matplotlib/Seaborn: For static and interactive visualizations, like histograms, heatmaps, and scatter plots, to explore the relationship between various features and heart disease.
- Plotly: For creating interactive and visually appealing charts and graphs, such as correlation matrices and distribution plots.

#### ☐ Data Manipulation Libraries:

- Pandas: Essential for data cleaning, preprocessing, and manipulation of structured data.
- NumPy: For handling numerical data, performing calculations, and matrix operations necessary in data exploration and feature engineering.

#### ☐ Data Storage:

- **Cloud Storage** (e.g., AWS S3, Google Cloud): For storing large datasets, including historical health records, medical images, or patient demographics related to heart disease.

### 2.3.1. Software Used:

#### □ **Data Pre-processing and Analysis:**

- **Jupyter Notebook:** Provides an interactive environment for data analysis in Python, ideal for cleaning and visualizing data, and documenting insights.
- **Pandas:** For cleaning, filtering, and transforming the dataset to prepare it for analysis.
- **NumPy:** For performing numerical operations such as calculating correlations, means, or performing matrix-based operations for feature engineering.

#### □ **Database Management:**

- **SQL:** If dealing with structured datasets stored in relational databases, such as patient health records or clinical data.
- **NoSQL:** For unstructured data, such as patient notes or data from wearable health devices (e.g., MongoDB or Cassandra).

### 2.3.2. Hardware Used:

□ **Development Machine:** A computer with sufficient processing power to handle data analysis and modeling tasks. The specific hardware requirements depend on the dataset size and complexity of the machine learning models being used.

□ **Cloud-Based Resources:** For large-scale data processing or training deep learning models, cloud-based resources such as virtual machines or GPU instances provided by platforms like AWS, Google Cloud, or Microsoft Azure may be necessary.

□ **Distributed Computing:** If working with very large datasets or complex analyses (e.g., real-time patient monitoring data), distributed computing tools like Hadoop or Apache Spark may be utilized to process the data efficiently.

## **CHAPTER 3: SYSTEM DESIGN**

### **3.1 Design Methodology:**

#### **1. Problem Definition**

- **Objective:** Predict the presence or absence of heart disease using EDA and machine learning.
- **Outcome:** Understand the dataset through detailed exploratory analysis and prepare features for predictive modeling.

#### **2. Data Collection**

- **Source:** Obtain a suitable heart disease dataset, such as the publicly available UCI Heart Disease Dataset.
- **Content:** Ensure the dataset contains relevant features such as age, cholesterol, resting heart rate, blood pressure, and other clinical variables.

#### **3. Data Preprocessing**

- **Steps:**
  1. **Load Data:** Import the dataset into a Python environment using pandas.
  2. **Data Types:** Identify data types (categorical, ordinal, numerical).
  3. **Handle Missing Values:** Impute missing values using appropriate techniques:
    - Numerical: Mean, median, or KNN imputation.
    - Categorical: Mode imputation.
  4. **Outlier Treatment:** Use IQR or z-scores to detect and cap/remove outliers.
  5. **Encode Categorical Variables:**
    - Use one-hot encoding for non-binary categories.
    - Label encoding for ordinal data.

#### **4. Exploratory Data Analysis (EDA)**

##### **A. Univariate Analysis**

- Analyze each feature individually to understand its distribution.
- **Numerical Features:** Histograms, density plots, boxplots.
- **Categorical Features:** Count plots, bar charts.

##### **B. Bivariate Analysis**

- Study the relationship between features and the target variable:
  - **Numerical vs Target:** Boxplots, violin plots.
  - **Categorical vs Target:** Bar plots, cross-tabulations.
- Analyze interactions between features, e.g., age vs cholesterol.

##### **C. Multivariate Analysis**

- Compute a **correlation matrix** and visualize it with a heatmap.
- Use pair plots to observe patterns among features.



- Identify multicollinearity and drop redundant features if necessary.

#### **D. Statistical Testing**

- Perform hypothesis testing to determine significant relationships:
  - Chi-square test for categorical features.
  - ANOVA or t-tests for numerical features.

#### **5. Feature Engineering**

- Create derived features based on domain knowledge:
  - Example: Combine cholesterol and age into a risk index.
- Discretize numerical features into bins (e.g., age groups).
- Perform feature scaling (standardization or normalization) for numerical features.

#### **6. Data Visualization**

- Use visualizations to communicate insights:
  - Scatter plots for trends and clusters.
  - Heatmaps for correlations.
  - Stacked bar charts for categorical comparisons.

#### **7. Insights and Actionable Outcomes**

- Identify key features that influence heart disease risk, such as:
  - Age, cholesterol, and blood pressure.
  - Smoking status or physical activity levels.
- Document findings and relationships revealed through EDA.

#### **8. Dataset Preparation for Modeling**

- Split the dataset into training and testing sets.
- Normalize/standardize features as required by the algorithm.
- Balance the target classes using SMOTE or undersampling if necessary.

#### **9. Predictive Modeling**

Build machine learning models to predict heart disease:

- Logistic Regression
  - Random Forest
  - XGBoost or CatBoost
  - Neural Networks (for complex datasets)
- Evaluate models using metrics such as:
  - Accuracy
  - Precision, Recall, F1-score
  - ROC-AUC curve

#### **10. Conclusion and Reporting**

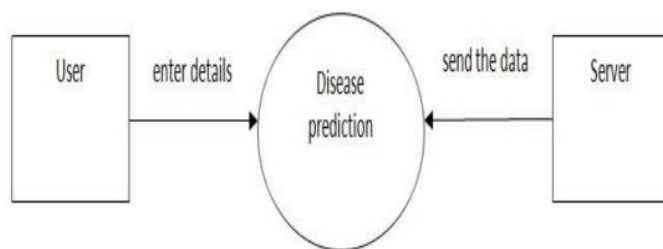
- Summarize findings from EDA and their implications for modeling.
- Report insights with visualizations, ensuring clarity for stakeholders.
- Provide actionable recommendations based on the analysis.

### Tools and Libraries

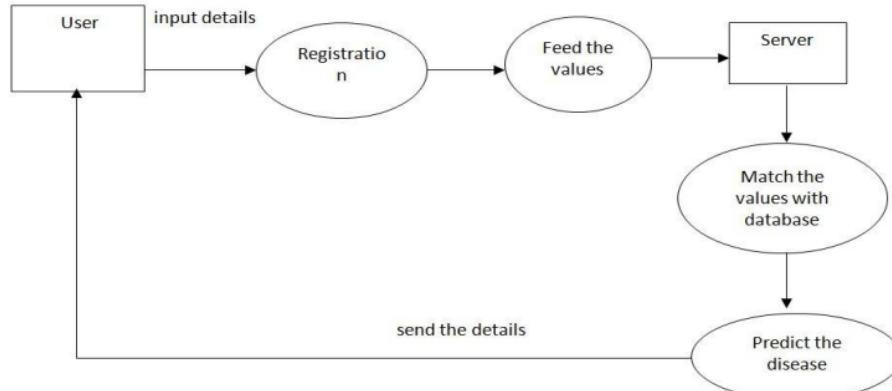
- **Python:** pandas, numpy, matplotlib, seaborn, scipy, statsmodels
- **Machine Learning :** scikit-learn, xgboost

### 3.2 Database Design :

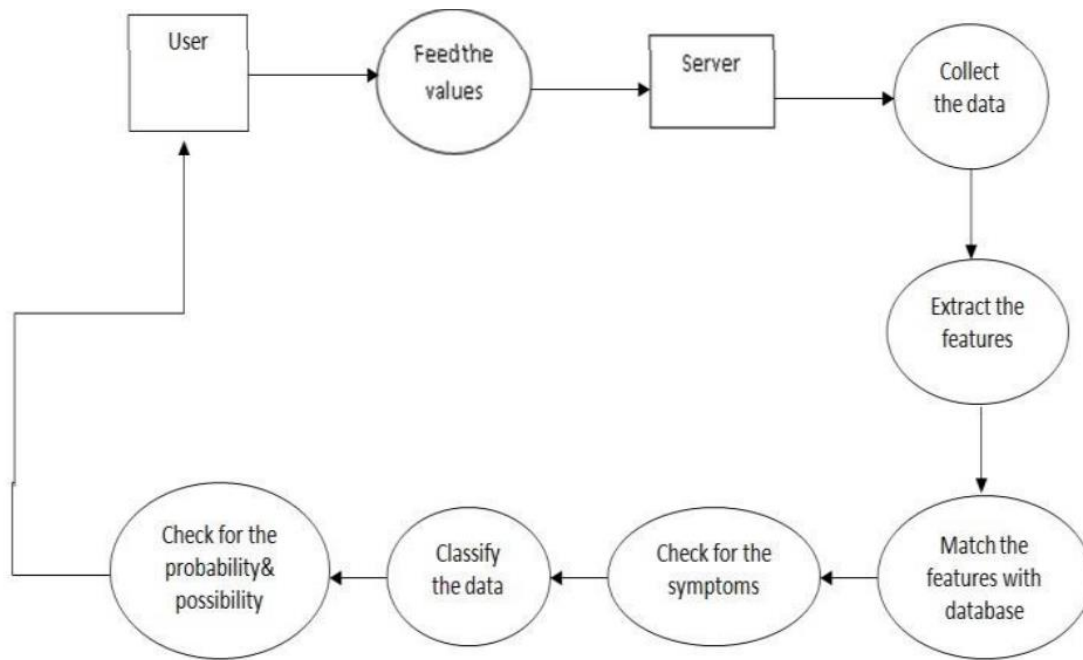
#### 3.2.2. DFD:



**Fig 3.1 DFD L-0**



**Fig 3.2 DFD L-1**



**Fig 3.3: DFD L-2**

### 3.3. Input design:

Input design ensures the dataset and user inputs are properly structured and formatted for analysis and prediction.

#### 1. Data Sources

- **Primary Data:** Clinical datasets like UCI Heart Disease Dataset or Kaggle datasets.
- **Manual Entry:** Input from users through forms or web applications for real-time prediction.

#### 2. Input Fields

Key features for prediction, generally based on medical and clinical parameters:

1. **Demographics:**
  - Age
  - Gender
2. **Medical History:**
  - Hypertension (Yes/No)
  - Diabetes (Yes/No)
  - Family history of heart disease (Yes/No)
3. **Clinical Measurements:**
  - Resting blood pressure (mmHg)
  - Serum cholesterol (mg/dL)
  - Fasting blood sugar (>120 mg/dL, Yes/No)
  - Resting electrocardiographic results (categorical)
4. **Lifestyle Parameters:**
  - Smoking status (Yes/No)
  - Physical activity (Yes/No)
5. **Test Results:**
  - Maximum heart rate achieved

- Exercise-induced angina (Yes/No)
- Oldpeak (ST depression induced by exercise)
- Number of major vessels (0–3)

### 3. Data Validation

- **Validation Rules:**
  - Numerical fields must be within realistic ranges (e.g., age: 1–120, cholesterol: 100–600).
  - Categorical inputs should match predefined options.
- **Error Handling:**
  - Notify users of invalid or missing inputs.
  - Suggest corrections or default values.

### 4. Input Format

- **For EDA:** Structured CSV/Excel files with the above columns.
- **For Deployment:** JSON payloads for APIs or form fields in a UI.

### 3.4 Output Design:

Output design defines how results, insights, and predictions are presented to the user or stakeholder.

#### 1. Outputs from EDA

- **Descriptive Statistics:**
  - Mean, median, mode, and standard deviation of numerical features.
  - Frequency counts for categorical features.
- **Visualizations:**
  - Histograms and density plots for distributions.
  - Correlation heatmaps for relationships.
  - Bar charts and box plots for feature-target relationships.
- **Insights:**
  - Summary of key patterns (e.g., older age and higher cholesterol correlate with heart disease).

#### 2. Outputs for Prediction

- **Prediction Results:**
  - Predicted probability of heart disease (e.g., 0.85 for high likelihood).
  - Binary classification: "Heart Disease Detected" or "No Heart Disease."
- **Detailed Insights:**
  - Feature contributions (e.g., SHAP values or LIME for feature importance).
  - Risk factors for individual predictions.
- **Risk Category:**
  - Categorize results into groups like Low, Medium, and High Risk.

#### 3. Visualization of Results

- **EDA Outputs:**
  - Interactive dashboards with filters for specific variables.
  - Heatmaps for correlations and feature importance rankings.
- **Prediction Outputs:**

- Risk probability displayed as a gauge chart.
- Tabular summary of patient data and corresponding predictions.
- Visual decision boundaries (if applicable).

#### **4. Output Formats**

- **For Analysis:**
  - CSV/Excel for numerical summaries and reports.
  - PDF/PNG for visual reports.
- **For Deployment:**
  - JSON API responses.
  - Interactive dashboards (using tools like Streamlit, Dash, or Tableau).

## **CHAPTER 4: TESTING**

### **4.1 TESTING METHODOLOGY**

#### **4.1.1 Unit Testing: -**

In unit testing, individual components or functions of your reducing food waste are tested in isolation.

- This would involve testing specific algorithms, data processing functions, or any small units of code that make up the model.
- The goal is to ensure that each unit performs as expected and produces accurate results.

#### **4.1.2 Module Testing: -**

Module testing focuses on testing groups of related units or modules that work together.

- This could involve testing different parts of your reducing food waste, such as data preprocessing, feature engineering, and model training as separate modules.
- The aim is to verify that these modules interact correctly and produce the desired outputs.

#### **4.1.3 Integration Testing: -**

Integration testing examines the interactions between different modules or components to ensure they function correctly when combined.

- This would involve testing how data flows between data preprocessing, model training, and prediction modules, ensuring seamless integration.

#### **4.1.4 System Testing: -**

System testing assesses the entire reduce food waste prediction system as a whole.

- This involves testing the end-to-end functionality of your model, from data input to generating rainfall predictions.
- You'll check if the system meets its intended goals and requirements.

#### **4.1.5 White Box Testing: -**

White box testing examines the internal logic and structure of your code.

- For your project, you'd analyze the code of your reducing food waste model to ensure it follows best practices, doesn't have any code smells, and is efficient.
- This can involve code reviews and static analysis tools.

#### **4.1.6 Black Box Testing: -**

Black box testing assesses the functionality of the system without examining its internal code.

- Testers focus on inputs and expected outputs, ensuring that the model produces accurate reduce food waste predictions without needing to know the implementation details.

## **4.2 Future Testing for Heart Disease Prediction**

Once the model and exploratory data analysis (EDA) are complete, various testing strategies ensure the robustness, reliability, and applicability of the system. These can be categorized into **data validation tests**, **model evaluation tests**, and **deployment-related tests**.

### **1. Data Validation Testing**

These tests ensure that the dataset used for training and predictions is clean and reliable.

#### **a. Data Quality Checks**

- **Missing Data Testing:**
  - Test if all missing values are handled properly (imputation or removal).
- **Outlier Detection:**
  - Verify if outliers are capped or treated effectively.
- **Data Distribution Testing:**
  - Compare training and test set distributions using statistical tests (e.g., KS Test).
- **Feature Consistency:**
  - Ensure feature values remain within expected ranges during new data entry or updates.

#### **b. Input Validation**

- Test the input system to verify that:
  - Invalid data types (e.g., strings in numeric fields) are rejected.
  - Invalid category labels are flagged.
  - Default values are assigned when necessary.

### **2. Model Evaluation Testing**

These tests focus on assessing the model's predictive performance and reliability.

#### **a. Performance Metrics**

- Evaluate the model using cross-validation:
  - **Accuracy:** Percentage of correctly classified cases.
  - **Precision, Recall, and F1-Score:** For imbalanced datasets, these are more informative than accuracy.
  - **ROC-AUC Curve:** Measures the model's ability to distinguish between classes.
  - **Confusion Matrix:** For detailed performance evaluation.

#### **b. Sensitivity and Specificity**

- Test the model's:

- **Sensitivity (True Positive Rate):** Ability to correctly predict heart disease cases.
- **Specificity (True Negative Rate):** Ability to correctly predict the absence of heart disease.

### **c. Generalization Testing**

- Evaluate the model's performance on unseen datasets to ensure it generalizes well:
  - Test on separate datasets (from other hospitals or regions).
  - Test with real-world data to check for discrepancies.

## **3. Stress Testing**

Assess the model's performance under extreme or unusual conditions.

### **a. Edge Cases**

- Input extreme values (e.g., very high/low age or cholesterol) and check predictions.
- Test with datasets containing a high percentage of missing or noisy data.

### **b. Data Drift Testing**

- Simulate future datasets where the data distribution changes (e.g., new age demographics) and evaluate if the model still performs well.
- Use statistical drift detection tools.

### **c. Imbalanced Class Testing**

- Simulate datasets with varying class imbalance and observe if the model retains predictive power.

## **4. Explainability Testing**

Ensure the model provides interpretable and actionable predictions.

### **a. Feature Importance Testing**

- Use tools like SHAP or LIME to confirm that the most important features are medically relevant (e.g., cholesterol, age, blood pressure).

### **b. Counterfactual Testing**

- Test if the model's predictions change logically when input variables are altered.
  - Example: Increasing cholesterol should generally increase heart disease risk.

## **5. System Integration Testing**

These tests validate the seamless functioning of the entire system.

### **a. API and UI Testing**



- Test APIs for:
  - Correct request-response cycle.
  - Proper handling of missing or invalid inputs.
- Test UI for:
  - Clear presentation of results.
  - Ease of use for healthcare professionals.

#### **b. Real-Time Prediction Testing**

- Test with live patient data from integrated healthcare systems.
- Ensure predictions are generated within acceptable time limits.

### **6. Deployment and Monitoring Testing**

Focuses on evaluating the system in production and ensuring its continued effectiveness.

#### **a. Monitoring System Performance**

- Continuously monitor:
  - Prediction accuracy using feedback loops.
  - System latency and resource usage.

#### **b. Retraining and Model Update Testing**

- Test periodic retraining workflows to incorporate new data.
- Evaluate the updated model's performance before deployment.

#### **c. Security and Privacy Testing**

- Ensure compliance with data protection laws like GDPR or HIPAA.
- Test for vulnerabilities in data storage and transmission.

## **CHAPTER 5: CONCLUSION AND REFERENCES**

### **5.1. CONCLUSION**

The summer training project on **Exploratory Data Analysis (EDA) of Human Heart Disease** under the framework of **SDG 3: Good Health and Well-being** provided critical insights into the factors influencing heart health and potential avenues for improving public health outcomes.

#### **Key Findings:**

##### **1. Risk Factor Insights:**

- Factors such as high cholesterol, elevated blood pressure, smoking, and diabetes showed a strong correlation with an increased risk of heart disease.
- Lifestyle choices like diet, physical activity, and alcohol consumption also played a significant role in determining heart health.

##### **2. Age and Gender Disparities:**

- Heart disease prevalence was higher among older populations, with males displaying slightly elevated risks compared to females in the early stages.

##### **3. Socioeconomic and Behavioral Impact:**

- Education level, access to healthcare, and early diagnosis were observed to be key determinants of better health outcomes.

#### **Implications for SDG 3:**

The findings emphasize the need for targeted interventions aligned with SDG 3 to reduce premature mortality from heart diseases. These include:

- **Awareness Programs:** Promoting preventive measures and healthy lifestyle habits.
- **Accessible Healthcare:** Ensuring affordable diagnosis and treatment options.
- **Policy Support:** Developing policies to reduce risk factors like tobacco use and unhealthy diets.

#### **Learning Outcomes:**

The project enhanced practical skills in data analysis, visualization, and interpretation, while fostering an understanding of the intersection between data science and public health.

This project underscores the power of data-driven approaches in addressing global health challenges and contributes meaningfully to the achievement of SDG 3.

### **5.2. SYSTEM SPECIFICATIONS**

#### **5.2.1 HARDWARE REQUIREMENTS:**

##### **✓ Device Specifications:**

- Device name: LAPTOP-RO7E0J6R

- Processor: Intel(R) Core (TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz
- Installed RAM: 4.00 GB (3.69 GB usable)
- Device ID: E82099BC-4391-4476-B31E-8B8EDC19D618
- Product ID: 00327-35189-88401-AAOEM
- System type: 64-bit operating system, x64-based processor
- Pen and touch: No pen or touch input is available for this display

✓ **Windows Specifications:**

- Edition : Windows 10 Home Single Language
- Version: 22H2
- Installed on: 15-03-2021
- OS build: 19045.5011
- Experience: Windows Feature Experience Pack 1000.19060.1000.0

### 5.2.2 SYSTEM REQUIREMENTS

✓ **Operating System:**

- Windows 10 or 11 (64-bit), macOS (Big Sur or newer), or Linux (Ubuntu preferred).

✓ **Programming Language:**

- Python (with libraries like pandas, NumPy, matplotlib, seaborn, scikit-learn, etc.).
- R (optional, with tidyverse and ggplot2 libraries).

✓ **IDE/Code Editor:**

- Jupyter Notebook, VS Code, PyCharm, or RStudio.

✓ **Data Visualization Tools:**

- Tableau or Power BI (optional, for interactive dashboards).

### 5.3 LIMITATIONS OF THE SYSTEM

While the system used for the **Exploratory Data Analysis (EDA) on Human Heart Disease** provides valuable insights, it has certain limitations that can impact the depth and scope of the analysis. These include:

#### 1. Data Limitations

- **Incomplete or Missing Data:**
  - Some datasets may contain missing values, leading to potential bias or inaccuracies in the analysis.
- **Lack of Real-Time Data:**

- The analysis is based on historical data, which may not reflect real-time trends or recent advancements in healthcare.
- **Sample Representation:**
  - If the dataset lacks diversity (e.g., in terms of geography, age, or ethnicity), the findings may not generalize well to the entire population.

## 2. Computational Constraints

- **Hardware Limitations:**
  - Systems with lower processing power or limited memory may struggle with large datasets, leading to slow performance or crashes.
- **Storage Issues:**
  - Handling large-scale datasets may require additional storage, especially if the data includes high-resolution imaging or time-series information.

## 3. Methodological Limitations

- **EDA is Descriptive, Not Predictive:**
  - EDA focuses on understanding patterns and relationships within the data but does not make predictions or establish causation.
- **Dependence on Assumptions:**
  - Some visualizations and statistical methods assume normal distribution or linear relationships, which may not always hold true.
- **Human Interpretation Bias:**
  - Insights from visualizations may be subject to misinterpretation or bias from the analyst.

## 4. Software and Tool Constraints

- **Tool Limitations:**
  - Tools like Jupyter or Tableau may have limitations in scalability or functionality compared to advanced machine learning platforms.
- **Integration Challenges:**
  - Difficulty in integrating multiple data sources or software platforms for a seamless workflow.

## 5. Scalability Issues

- **Handling Large Datasets:**
  - Systems may struggle with scaling to process massive datasets or perform highly complex operations, especially without cloud resources.
- **Resource-Intensive Algorithms:**

- Advanced analysis or simulations (e.g., predictive modeling) may require high computational resources, which are unavailable in standard setups.

## 6. Ethical and Privacy Concerns

- **Data Privacy:**
  - Analyzing health-related data raises concerns about data confidentiality and compliance with privacy regulations like GDPR or HIPAA.
- **Bias in Data:**
  - If the dataset is biased (e.g., underrepresentation of specific groups), the findings may inadvertently reinforce systemic inequalities.

## 5.4 FUTURE SCOPE FOR MODIFICATIONS

1. **Expanded Data Sources:** Incorporate diverse, real-time, and global datasets for broader applicability.
2. **Advanced Analytics:** Use machine learning, deep learning, and time-series analysis for predictive and actionable insights.
3. **Interactive Visualizations:** Develop dynamic dashboards and geospatial maps for better stakeholder engagement.
4. **Cloud Integration:** Leverage cloud platforms for scalable data processing and analysis.
5. **Cross-Disciplinary Collaboration:** Work with medical experts, policymakers, and behavioral scientists for impactful solutions.
6. **Ethical Improvements:** Enhance data privacy measures and mitigate biases for equitable outcomes.
7. **Public Health Impact:** Use findings to support awareness campaigns and community outreach programs.


These modifications will strengthen the project's scope and its contribution to SDG 3.

## 5.5 REFERENCES

- [https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)?gad\\_source=1&gclid=CjwKCAjw5Ky1BhAgEiwA5jGujtLiF63Sw-nBY8XBTvIW\\_9I8TFsqksd2\\_Hq7yPcRXmAaB3FXHBYZRoCPuoQAvD\\_BwE](https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)?gad_source=1&gclid=CjwKCAjw5Ky1BhAgEiwA5jGujtLiF63Sw-nBY8XBTvIW_9I8TFsqksd2_Hq7yPcRXmAaB3FXHBYZRoCPuoQAvD_BwE)
- <https://www.ncbi.nlm.nih.gov/books/NBK535419/>
- <https://www.ahajournals.org/journal/jaha>

## CHAPTER 6: ANNEXURES


### A-11 CODING:


 ibmproject.ipynb ☆  
File Edit View Insert Runtime Tools Help [Last edited on August 2](#)

+ Code + Text

▽ Importing the Necessary libraries and loading the Dataset


```
[ ] from google.colab import files
    uploaded = files.upload()
```

 Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable Saving heart (1).csv to heart (1).csv




```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df=pd.read_csv('heart.csv')
print("DATA\n",df)
```

 DATA

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	52	1	0	125	212	0	1	168	0	1.0	
1	53	1	0	140	203	1	0	155	1	3.1	
2	70	1	0	145	174	0	1	125	1	2.6	
3	61	1	0	148	203	0	1	161	0	0.0	
4	62	0	0	138	294	1	1	106	0	1.9	
...	...	...	...	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	
1021	60	1	0	125	258	0	0	141	1	2.8	
1022	47	1	0	110	275	0	0	118	1	1.0	
1023	50	0	0	110	254	0	0	159	0	0.0	
1024	54	1	0	120	188	0	1	113	0	1.4	

 DATA

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	52	1	0	125	212	0	1	168	0	1.0	
1	53	1	0	140	203	1	0	155	1	3.1	
2	70	1	0	145	174	0	1	125	1	2.6	
3	61	1	0	148	203	0	1	161	0	0.0	
4	62	0	0	138	294	1	1	106	0	1.9	
...	...	...	...	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	
1021	60	1	0	125	258	0	0	141	1	2.8	
1022	47	1	0	110	275	0	0	118	1	1.0	
1023	50	0	0	110	254	0	0	159	0	0.0	
1024	54	1	0	120	188	0	1	113	0	1.4	

	slope	ca	thal	target
0	2	2	3	0
1	0	0	3	0
2	0	0	3	0
3	2	1	3	0
4	1	3	2	0
...	...	...	...	...
1020	2	0	2	1
1021	1	1	3	0
1022	1	1	2	0
1023	2	0	2	1
1024	1	1	3	0

[1025 rows x 14 columns]

## ✓ Data cleaning

```
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)
```

```
Missing Values:
age      0
sex      0
cp       0
trestbps 0
chol     0
fbs      0
restecg  0
thalach  0
exang    0
oldpeak  0
slope    0
ca       0
thal     0
target   0
dtype: int64
```

```
[ ] print("different set of age groups\n",df['age'].unique(),'\n')
print("types of genders\n",df['sex'].unique(),'\n')
print("chest pain types\n",df['cp'].unique(),'\n')
print("range of resting blood pressure\n",df['trestbps'].unique(),'\n')
print("number of vessels\n",df['ca'].unique(),'\n')
print("is the disease curable\n",df['thal'].unique())
```

```
different set of age groups
[52 53 70 61 62 58 55 46 54 71 43 34 51 50 60 67 45 63 42 44 56 57 59 64
 65 41 66 38 49 48 29 37 47 68 76 40 39 77 69 35 74]
```

```
[ ] different set of age groups
[52 53 70 61 62 58 55 46 54 71 43 34 51 50 60 67 45 63 42 44 56 57 59 64
 65 41 66 38 49 48 29 37 47 68 76 40 39 77 69 35 74]
```

```
types of genders
[nan]
```

```
chest pain types
[0 1 2 3]
```

```
range of resting blood pressure
[125 140 145 148 138 100 114 160 120 122 112 132 118 128 124 106 104 135
 130 136 180 129 150 178 146 117 152 154 170 134 174 144 108 123 110 142
 126 192 115 94 200 165 102 105 155 172 164 156 101]
```

```
number of vessels
[2 0 1 3 4]
```

```
is the disease curable
[3 2 1 0]
```

```
'''df.drop(columns=['ca','thal'], inplace=True)
```

```
[ ] print(df.columns)
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

## ✓ Descriptive Statistical Analysis

```
print('\n shape of data\t',df.shape)
print('\n info of data\t',df.info())
print('\n describe data\t',df.describe())
```



```
shape of data (1025, 14)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  -
0   age         1025 non-null   int64
1   sex         0 non-null      object
2   cp          1025 non-null   int64
3   trestbps    1025 non-null   int64
4   chol        1025 non-null   int64
5   fbs         1025 non-null   int64
6   restecg     1025 non-null   int64
7   thalach     1025 non-null   int64
8   exang       1025 non-null   int64
9   oldpeak     1025 non-null   float64
10  slope       1025 non-null   int64
11  ca          1025 non-null   int64
12  thal        1025 non-null   int64
13  target      1025 non-null   int64
dtypes: float64(1), int64(12), object(1)
memory usage: 112.2+ KB
```

info of data    None

describe data	age	cp	trestbps	chol	fbs
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000



info of data    None

describe data	age	cp	trestbps	chol	fbs
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.942439	131.611707	246.000000	0.149268
std	9.072290	1.029641	17.516718	51.59251	0.356527
min	29.000000	0.000000	94.000000	126.000000	0.000000
25%	48.000000	0.000000	120.000000	211.000000	0.000000
50%	56.000000	1.000000	130.000000	240.000000	0.000000
75%	61.000000	2.000000	140.000000	275.000000	0.000000
max	77.000000	3.000000	200.000000	564.000000	1.000000

	restecg	thalach	exang	oldpeak	slope
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	0.529756	149.114146	0.336585	1.071512	1.385366
std	0.527878	23.005724	0.472772	1.175053	0.617755
min	0.000000	71.000000	0.000000	0.000000	0.000000
25%	0.000000	132.000000	0.000000	0.000000	1.000000
50%	1.000000	152.000000	0.000000	0.800000	1.000000
75%	1.000000	166.000000	1.000000	1.800000	2.000000
max	2.000000	202.000000	1.000000	6.200000	2.000000

	ca	thal	target
count	1025.000000	1025.000000	1025.000000
mean	0.754146	2.323902	0.513171
std	1.030798	0.620660	0.500070
min	0.000000	0.000000	0.000000
25%	0.000000	2.000000	0.000000
50%	0.000000	2.000000	1.000000
75%	1.000000	3.000000	1.000000
max	4.000000	3.000000	1.000000



```
[ ] df.head()
```



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	NaN	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	NaN	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	NaN	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	NaN	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	NaN	0	138	294	1	1	106	0	1.9	1	3	2	0



```
targets = df['target'].value_counts()  
targets
```



	count
target	
1	526
0	499

dtype: int64



```
df.describe([0.01, 0.25, 0.5, 0.75, 0.99]).T
```



	count	mean	std	min	1%	25%	50%	75%	99%	max
age	1025.0	54.434146	9.072290	29.0	35.0	48.0	56.0	61.0	71.000	77.0
cp	1025.0	0.942439	1.029641	0.0	0.0	0.0	1.0	2.0	3.000	3.0
trestbps	1025.0	131.611707	17.516718	94.0	100.0	120.0	130.0	140.0	180.000	200.0
chol	1025.0	246.000000	51.592510	126.0	149.0	211.0	240.0	275.0	407.000	564.0
fbs	1025.0	0.149268	0.356527	0.0	0.0	0.0	0.0	0.0	1.000	1.0
restecg	1025.0	0.529756	0.527878	0.0	0.0	0.0	1.0	1.0	2.000	2.0
thalach	1025.0	149.114146	23.005724	71.0	95.0	132.0	152.0	166.0	192.000	202.0
exang	1025.0	0.336585	0.472772	0.0	0.0	0.0	0.0	1.0	1.000	1.0
oldpeak	1025.0	1.071512	1.175053	0.0	0.0	0.0	0.8	1.8	4.352	6.2
slope	1025.0	1.385366	0.617755	0.0	0.0	1.0	1.0	2.0	2.000	2.0
ca	1025.0	0.754146	1.030798	0.0	0.0	0.0	0.0	1.0	4.000	4.0
thal	1025.0	2.323902	0.620660	0.0	1.0	2.0	2.0	3.0	3.000	3.0
target	1025.0	0.513171	0.500070	0.0	0.0	0.0	1.0	1.0	1.000	1.0

## ✎ Exploratory Data Analysis



```
targets.plot(  
    kind='bar',  
    color=['salmon', 'lightblue'],  
    figsize=(10,6)  
)  
plt.xticks(rotation=0)  
plt.show()
```



```
import matplotlib.pyplot as plt  
  
# Histogram of Age Distribution  
plt.hist(df['age'], bins=20, color='pink', edgecolor='purple')  
plt.xlabel('Age')  
plt.ylabel('Frequency')  
plt.title('Age Distribution')  
plt.grid(True)  
plt.show()
```

```

▶ sns.histplot(df.thalach,bins=20, kde=True , color='blue', edgecolor='black')
# The Best number of bits is chosen based on Rice Criterion
plt.show()

```

```

▶ df['target'].value_counts().plot(kind='bar', color=['green', 'grey'])
plt.xlabel('Heart Disease')
plt.ylabel('Count')
plt.title('Presence of Heart Disease (0 = No, 1 = Yes)')
plt.xticks(rotation=0)
plt.show()

```

```

▶ plt.figure(figsize=(10, 6))
sns.countplot(x='sex', data=df)
plt.title('Distribution of Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()

```

## Relationship Between Gender

```
[ ] df['sex'].value_counts()
```



```

count
sex
1    713
0    312

```

dtype: int64

```

[ ] def to_cross_tab(heart1, index_name, col_name):
    df = pd.crosstab(heart1[index_name], heart1[col_name])
    df['rate'] = df.iloc[:,1] / (df.iloc[:,0] + df.iloc[:,1])
    return df
sex_target = to_cross_tab(df, 'target', 'sex')
sex_target

```



```

sex    0    1    rate
target
0      86  413  0.827655
1     226  300  0.570342

```

## ✓ Relationship between age and maximum heart rate with and without the disease

```
▶ # people with disease
plt.scatter(df['age'][df['target']==1],
            df['thalach'][df['target']==1],
            c='green'
            )

# people without disease
plt.scatter(df['age'][df['target']==0],
            df['thalach'][df['target']==0],
            c='yellow'
            )

#Title
plt.title('Divide people into two groups according to whether they have heart disease to view age and maximum heart rate')
plt.xlabel('age')
plt.ylabel('maximum heart rate')
plt.legend(['disease', 'without disease'])

plt.show()
```

## ✓ Average age

```
▶ df['age'].hist()
```

## ✓ Analysing chest pain types

### Chest pain types

- Value 0: typical angina
- Value 1: atypical angina
- Value 2: non-anginal pain
- Value 3: asymptomatic

```
▶ sns.countplot(x="cp",data=df,color="skyblue")
plt.show()

count=df["cp"].value_counts()
# print(count)

plt.pie(count,shadow=True,explode=(0.05,0,0,0),startangle=90)
plt.title("Pie Chart for chest pain types")
plt.legend(labels=['Type 0', 'Type 1', 'Type 2', 'Type 3'], loc='upper right')
plt.show()
```

```
▶ pd.crosstab(df['target'],df['cp']).plot(kind='bar')
plt.xticks(rotation =1)
plt.show()
```

## ✓ Hypothesis

```
▶ plt.figure(figsize=(12, 6))
  sns.histplot(data=df, x='age', hue='target', multiple='stack', palette='viridis')
  plt.title('Distribution of Age with Heart Disease')
  plt.xlabel('Age')
  plt.ylabel('Frequency')
  plt.legend(title='Heart Disease', labels=['No', 'Yes'])
  plt.show()
```

```
▶ plt.figure(figsize=(10, 6))
  sns.boxplot(x='target', y='age', data=df)
  plt.title('Age vs Heart Disease')
  plt.xlabel('Heart Disease')
  plt.ylabel('Age')
  plt.show()
```

```
▶ plt.figure(figsize=(10, 6))
  sns.histplot(df['trestbps'], kde='true')
  plt.title('Distribution of Blood Pressure')
  plt.xlabel('Resting Blood Pressure (mm Hg)')
  plt.ylabel('Frequency')
  plt.show()
```

```
▶ plt.figure(figsize=(10, 6))
  sns.boxplot(x='target', y='trestbps', data=df)
  plt.title('Blood Pressure vs Heart Disease')
  plt.xlabel('Heart Disease')
  plt.ylabel('Resting Blood Pressure (mm Hg)')
  plt.show()
```

```
▶ plt.figure(figsize=(10, 6))
  sns.histplot(df['chol'], kde=True)
  plt.title('Distribution of Cholesterol Levels')
  plt.xlabel('Serum Cholesterol (mg/dl)')
  plt.ylabel('Frequency')
  plt.show()
```

```
▶ plt.figure(figsize=(10, 6))
  sns.boxplot(x='target', y='chol', data=df)
  plt.title('Cholesterol vs Heart Disease')
  plt.xlabel('Heart Disease')
  plt.ylabel('Serum Cholesterol (mg/dl)')
  plt.show()
```

```
▶ df['sex'] = df['sex'].map({0: 'Female', 1: 'Male'})
  # Distribution of Heart Disease by Gender
  plt.figure(figsize=(10, 6))
  sns.countplot(x='sex', hue='target', data=df)
  plt.title('Distribution of Heart Disease by Gender')
  plt.xlabel('Gender')
  plt.ylabel('Count')
  plt.legend(title='Heart Disease', loc='upper right', labels=['No', 'Yes'])
  plt.show()
```

```

▶ gender_heart_disease = df.groupby('sex')['target'].mean().reset_index()

# Heart Disease Prevalence by Gender
plt.figure(figsize=(10, 6))
sns.barplot(x='sex', y='target', data=gender_heart_disease, palette='viridis')
plt.title('Heart Disease Prevalence by Gender')
plt.xlabel('Gender')
plt.ylabel('Prevalence (Proportion)')
plt.show()

```

## ▼ Other Numerical Features

```

▶ cats = ["sex", "cp", "fbs", "restecg", "exang", "slope", "ca", "thal", "target"]
fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(10, 8))
index = 0
for i in range(3):
    for j in range(3):
        counts = df[cats[index]].value_counts()
        axes[i][j].pie(counts, labels=counts, autopct="%0.2f%%")
        axes[i][j].legend(counts.index)
        index += 1
plt.tight_layout()
plt.show()

```

```

# Constructing the DataFrame
data_dict = {
    'age': [52, 53, 70, 61, 62],
    'sex': [1, 1, 1, 1, 0],
    'cp': [0, 0, 0, 0, 0],
    'trestbps': [125, 140, 145, 148, 138],
    'chol': [212, 203, 174, 203, 294],
    'fbs': [0, 1, 0, 0, 1],
    'restecg': [1, 0, 1, 1, 1],
    'thalach': [168, 155, 125, 161, 106],
    'exang': [0, 1, 1, 0, 0],
    'oldpeak': [1.0, 3.1, 2.6, 0.0, 1.9],
    'slope': [2, 0, 0, 2, 1],
    'ca': [2, 0, 0, 1, 3],
    'thal': [3, 3, 3, 3, 2],
    'target': [0, 0, 0, 0, 0]
}

data = pd.DataFrame(data_dict)

# Box plot for each numerical feature with respect to the target variable
plt.figure(figsize=(15, 10))
numerical_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

for i, col in enumerate(numerical_columns, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(x='target', y=col, data=data)
    plt.title(f'Box plot of {col}')

plt.tight_layout()
plt.show()

```

```

▶ import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Create the DataFrame
data = pd.DataFrame({
    'age': [52, 53, 70, 61, 62],
    'sex': [1, 1, 1, 1, 0],
    'cp': [0, 0, 0, 0, 0],
    'trestbps': [125, 140, 145, 148, 138],
    'chol': [212, 203, 174, 203, 294],
    'fbs': [0, 1, 0, 0, 1],
    'restecg': [1, 0, 1, 1, 1],
    'thalach': [168, 155, 125, 161, 106],
    'exang': [0, 1, 1, 0, 0],
    'oldpeak': [1.0, 3.1, 2.6, 0.0, 1.9],
    'slope': [2, 0, 0, 2, 1],
    'ca': [2, 0, 0, 1, 3],
    'thal': [3, 3, 3, 3, 2],
    'target': [0, 0, 0, 0, 0]
})

# List of categorical columns to create count plots for
categorical_columns = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']

# Create count plots
plt.figure(figsize=(15, 10))
for i, col in enumerate(categorical_columns):
    plt.subplot(3, 3, i+1)
    sns.countplot(x=col, data=data)
    plt.title(f'Count plot of {col}')
plt.tight_layout()
plt.show()

```

```

▶ cats = ["sex", "cp", "fbs", "restecg", "exang", "slope", "ca", "thal", "target"]
nums = ["age", "trestbps", "chol", "thalach", "oldpeak"]
for i in cats[:-1]:
    fig, axes = plt.subplots(nrows=1, ncols=5, figsize=(12, 5))
    for j in range(5):
        sns.kdeplot(df, x=nums[j], hue=i, ax=axes[j])
    plt.tight_layout()
    plt.show()

```

```

▶ sns.pairplot(df, vars=["target", "thalach"])
plt.show()

```

```

# Plot 1: Age vs Target
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.scatter(df['age'], df['target'], alpha=0.5, c='blue')
plt.xlabel('Age')
plt.ylabel('Target')
plt.title('Age vs Target')

# Plot 2: Sex vs Target
plt.subplot(1, 2, 2)
plt.scatter(df['sex'], df['target'], alpha=0.5, c='red')
plt.xlabel('Sex')
plt.ylabel('Target')
plt.title('Sex vs Target')

plt.tight_layout()
plt.show()

```

## Matrix

```

corr_matrix = df.corr()
corr_matrix

```

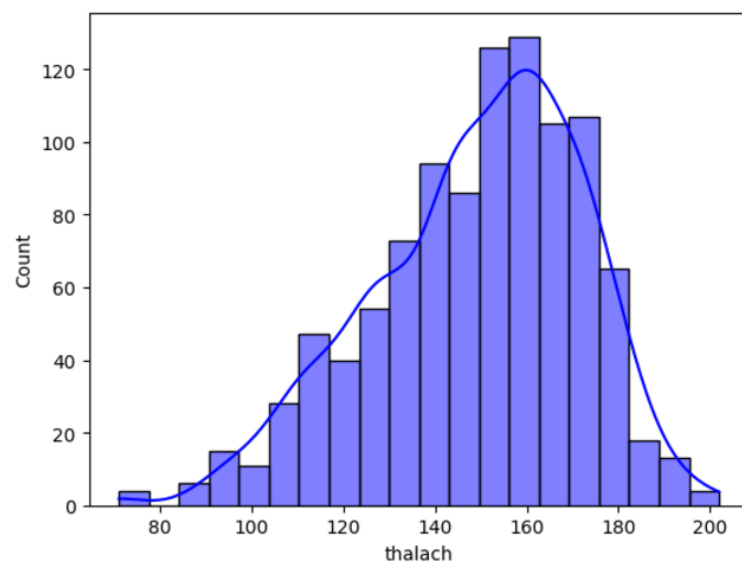
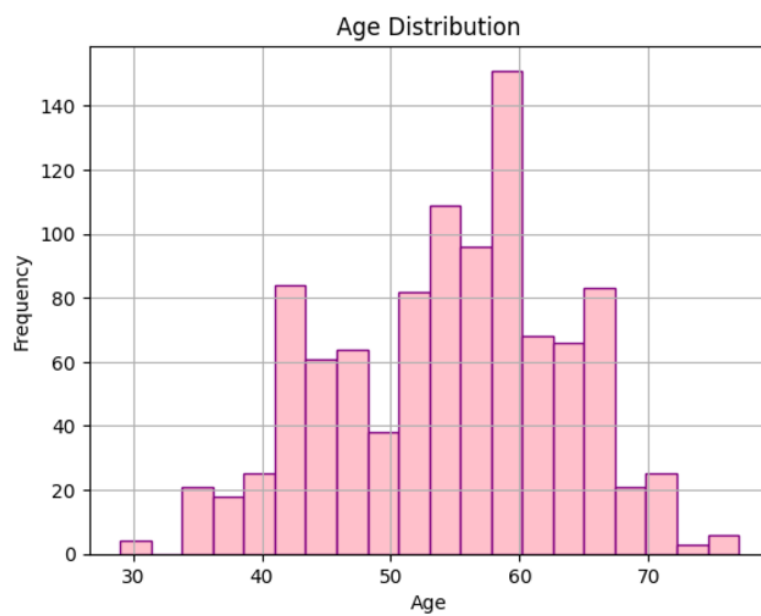
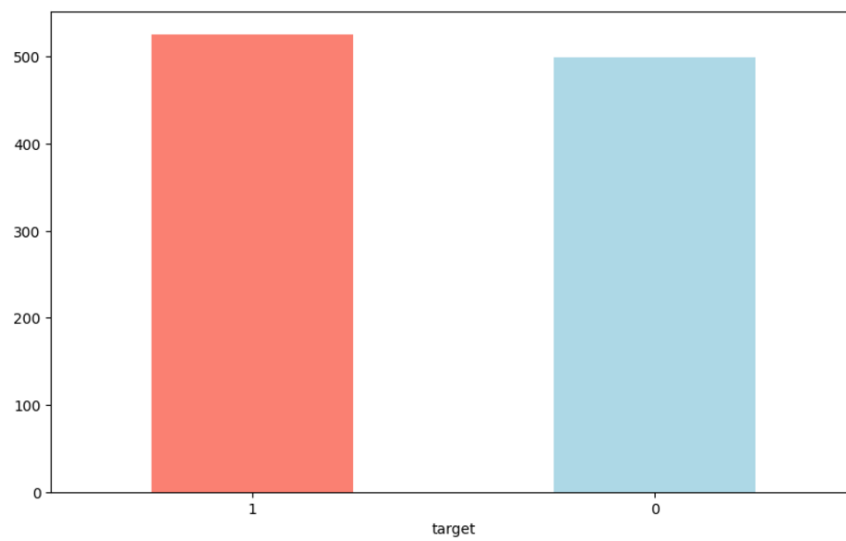
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.103240	-0.071966	0.271121	0.219823	0.121243	-0.132696	-0.390227	0.088163	0.208137	-0.169105	0.271551	0.072297	-0.229324
sex	-0.103240	1.000000	-0.041119	-0.078974	-0.198258	0.027200	-0.055117	-0.049365	0.139157	0.084687	-0.026666	0.111729	0.198424	-0.279501
cp	-0.071966	-0.041119	1.000000	0.038177	-0.081641	0.079294	0.043581	0.306839	-0.401513	-0.174733	0.131633	-0.176206	-0.163341	0.434854
trestbps	0.271121	-0.078974	0.038177	1.000000	0.127977	0.181767	-0.123794	-0.039264	0.061197	0.187434	-0.120445	0.104554	0.059276	-0.138772
chol	0.219823	-0.198258	-0.081641	0.127977	1.000000	0.026917	-0.147410	-0.021772	0.067382	0.064880	-0.014248	0.074259	0.100244	-0.099966
fbs	0.121243	0.027200	0.079294	0.181767	0.026917	1.000000	-0.104051	-0.008866	0.049261	0.010859	-0.061902	0.137156	-0.042177	-0.041164
restecg	-0.132696	-0.055117	0.043581	-0.123794	-0.147410	-0.104051	1.000000	0.048411	-0.065606	-0.050114	0.086086	-0.078072	-0.020504	0.134468
thalach	-0.390227	-0.049365	0.306839	-0.039264	-0.021772	-0.008866	0.048411	1.000000	-0.380281	-0.349796	0.395308	-0.207888	-0.098068	0.422895
exang	0.088163	0.139157	-0.401513	0.061197	0.067382	0.049261	-0.065606	-0.380281	1.000000	0.310844	-0.267335	0.107849	0.197201	-0.438029
oldpeak	0.208137	0.084687	-0.174733	0.187434	0.064880	0.010859	-0.050114	-0.349796	0.310844	1.000000	-0.575189	0.221816	0.202672	-0.438441
slope	-0.169105	-0.026666	0.131633	-0.120445	-0.014248	-0.061902	0.086086	0.395308	-0.267335	-0.575189	1.000000	-0.073440	-0.094090	0.345512
ca	0.271551	0.111729	-0.176206	0.104554	0.074259	0.137156	-0.078072	-0.207888	0.107849	0.221816	-0.073440	1.000000	0.149014	-0.382085
thal	0.072297	0.198424	-0.163341	0.059276	0.100244	-0.042177	-0.020504	-0.098068	0.197201	0.202672	-0.094090	0.149014	1.000000	-0.337838
target	-0.229324	-0.279501	0.434854	-0.138772	-0.099966	-0.041164	0.134468	0.422895	-0.438029	-0.438441	0.345512	-0.382085	-0.337838	1.000000

```

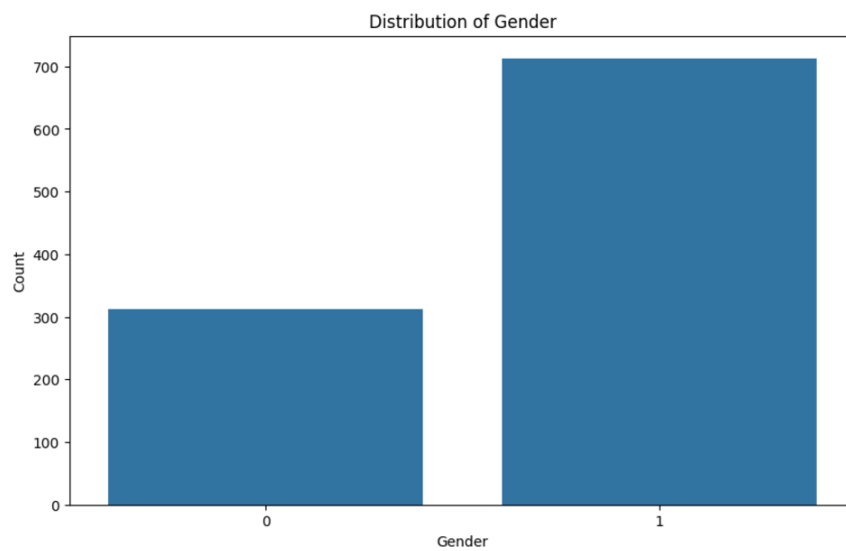
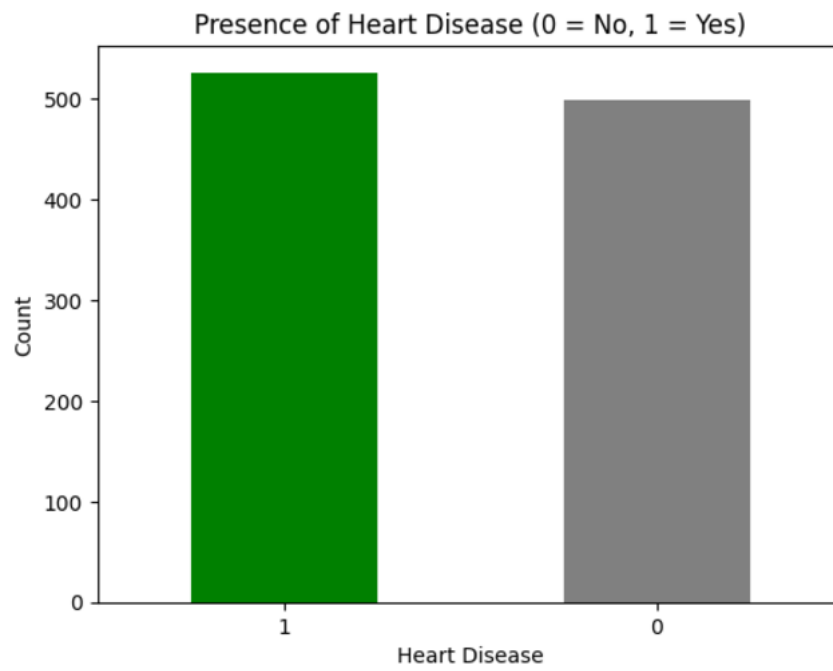
# plot matrix
plt.figure(figsize=(14, 10))
sns.heatmap(
    corr_matrix,
    vmin=-1,
    annot=True,
    linewidth=5,
    fmt='.2f',
    cmap='YlGnBu'
)
plt.show()

```

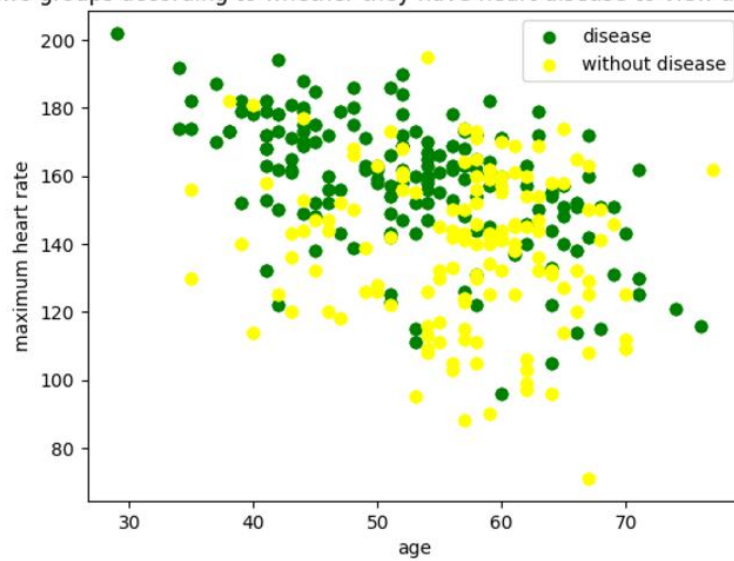
### A-10 SAMPLE OUTPUT:



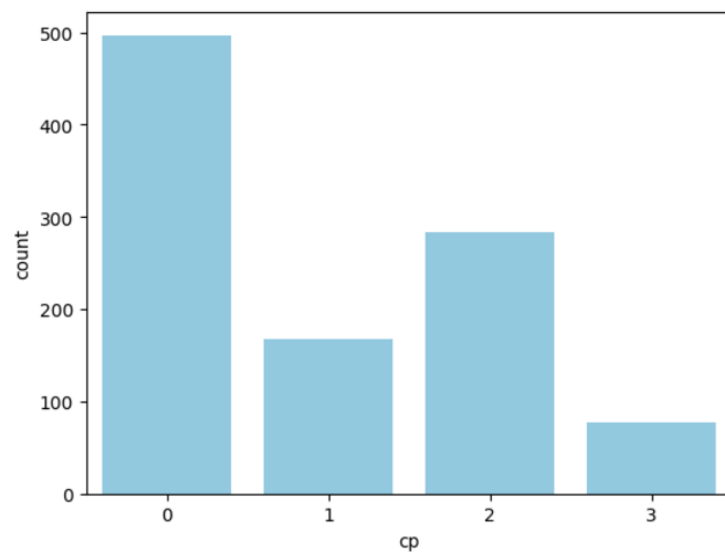
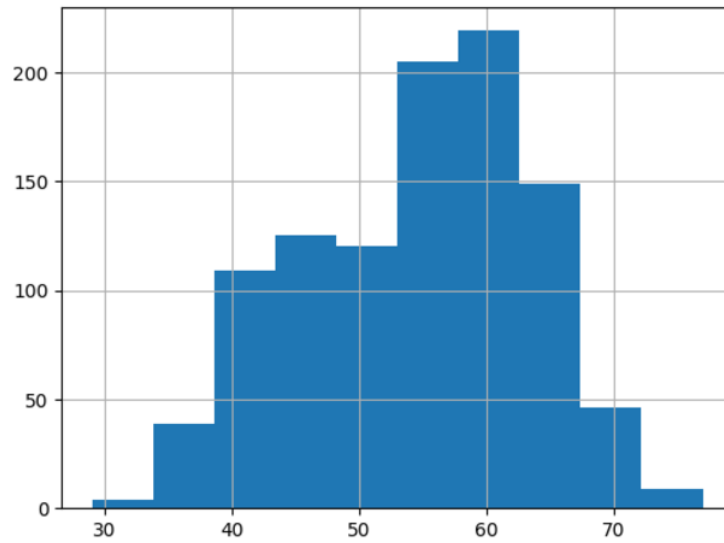




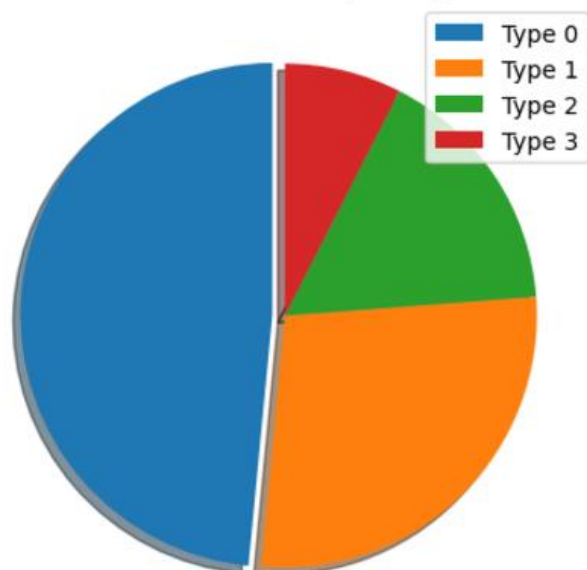
Divide people into two groups according to whether they have heart disease to view age and maximum heart rate

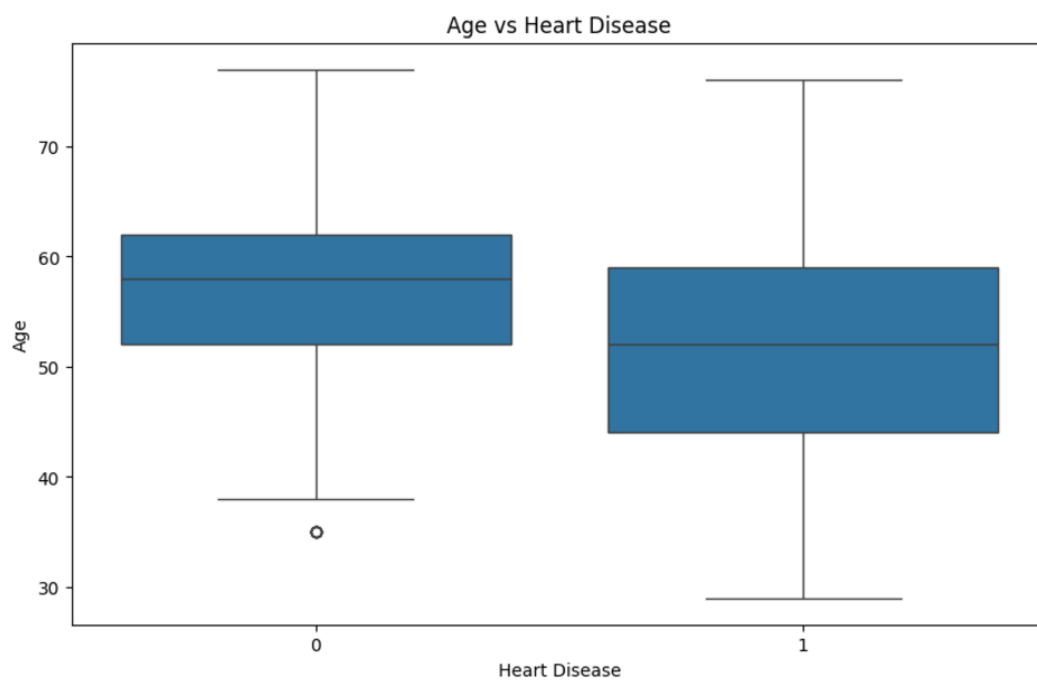
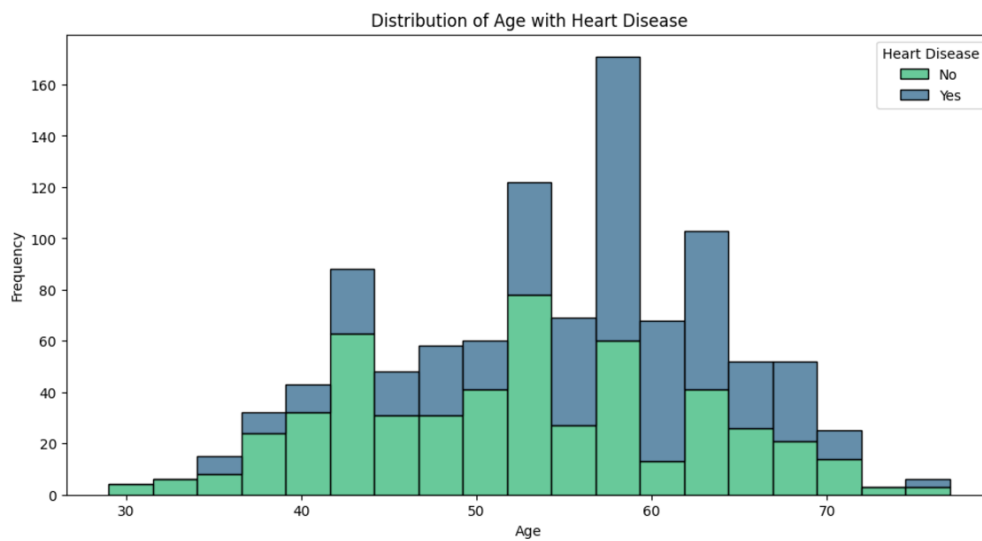
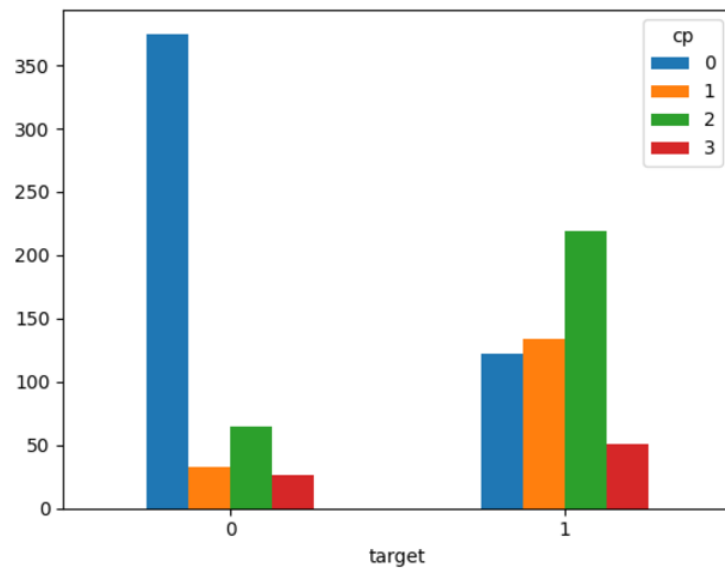


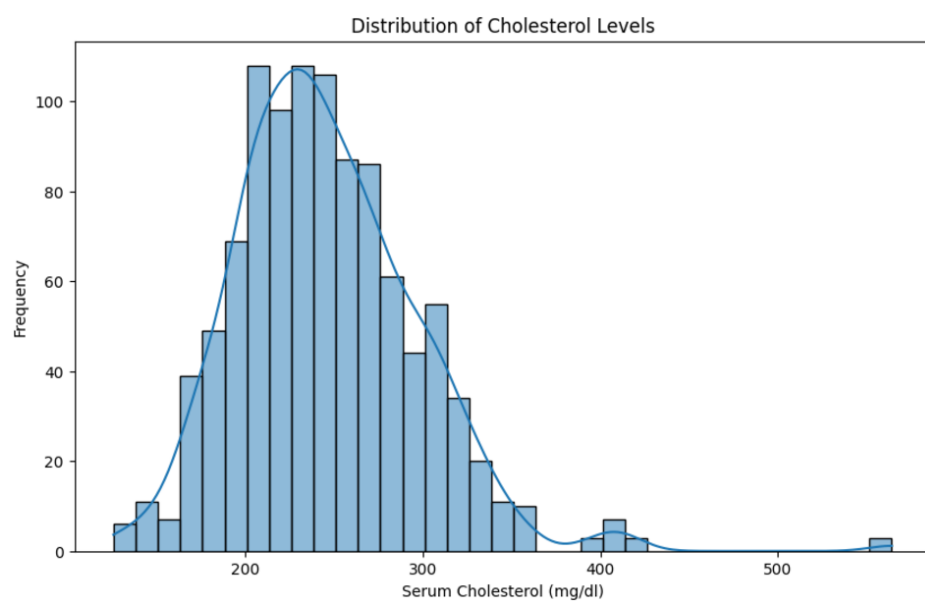
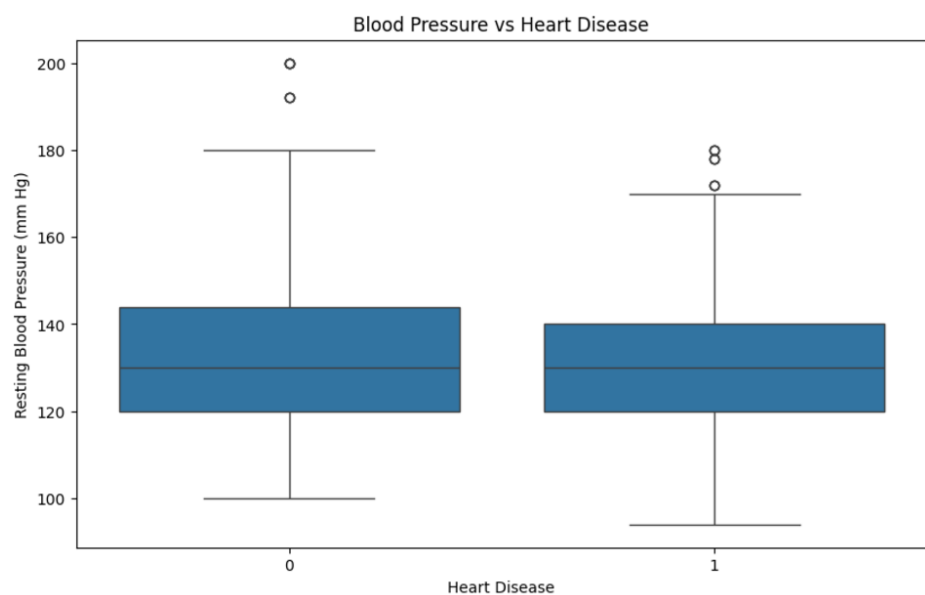
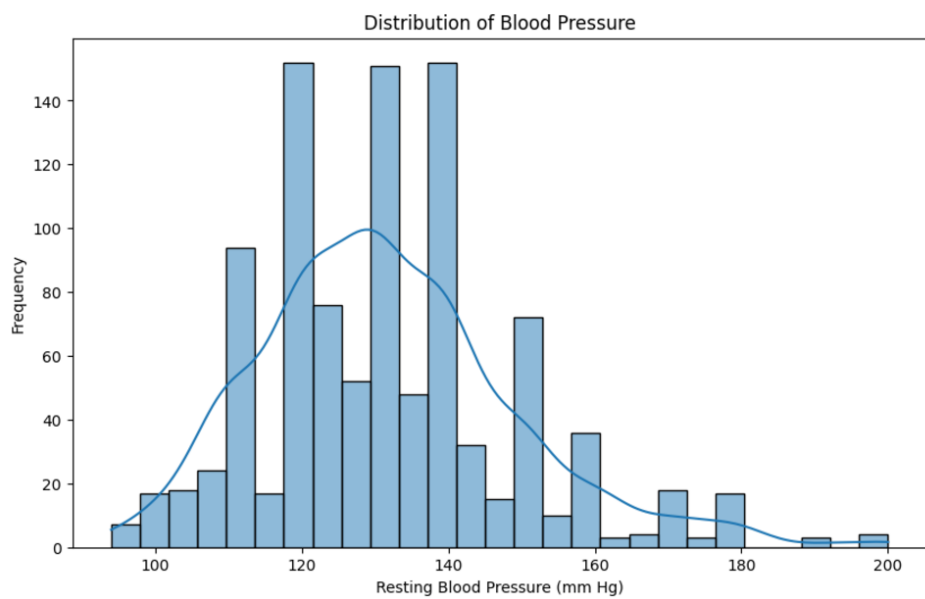
<Axes: >

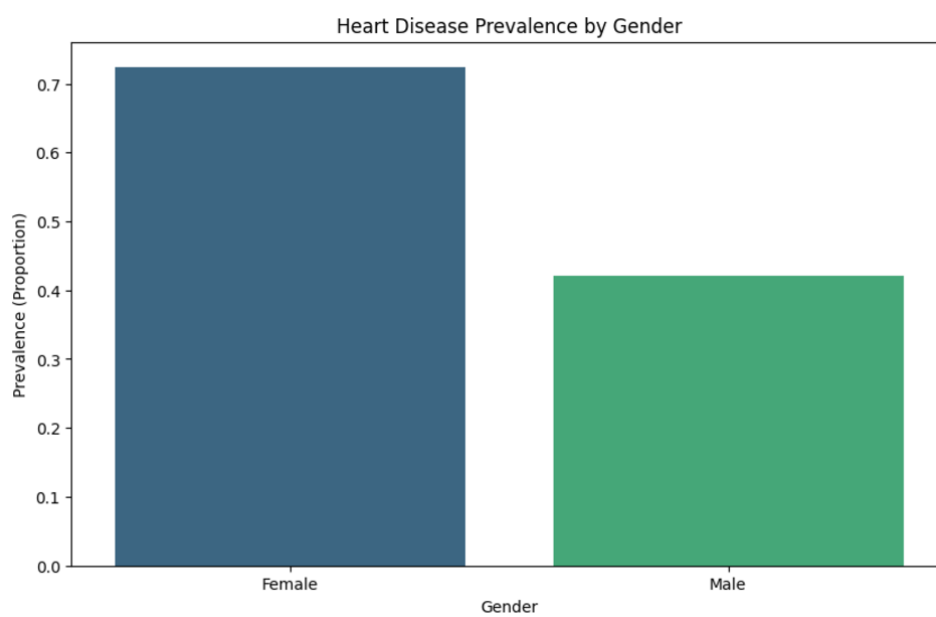
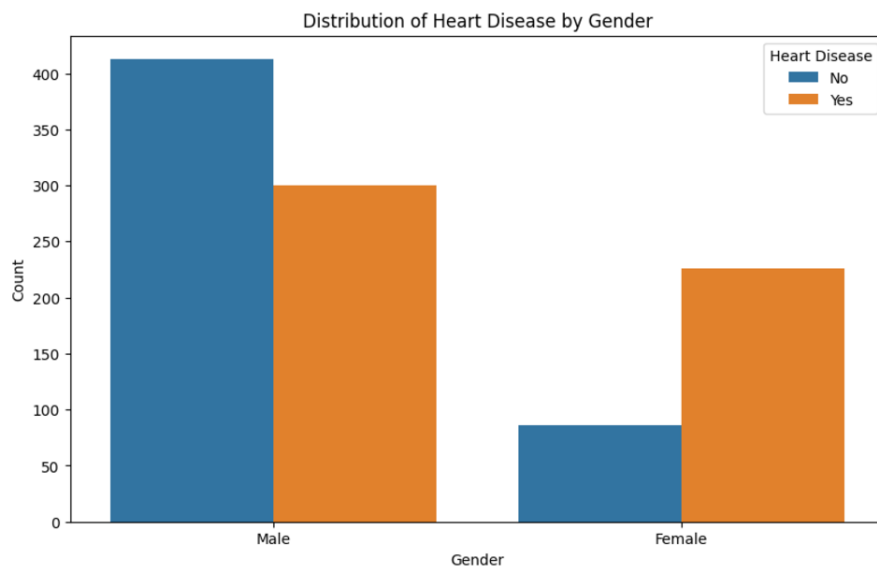
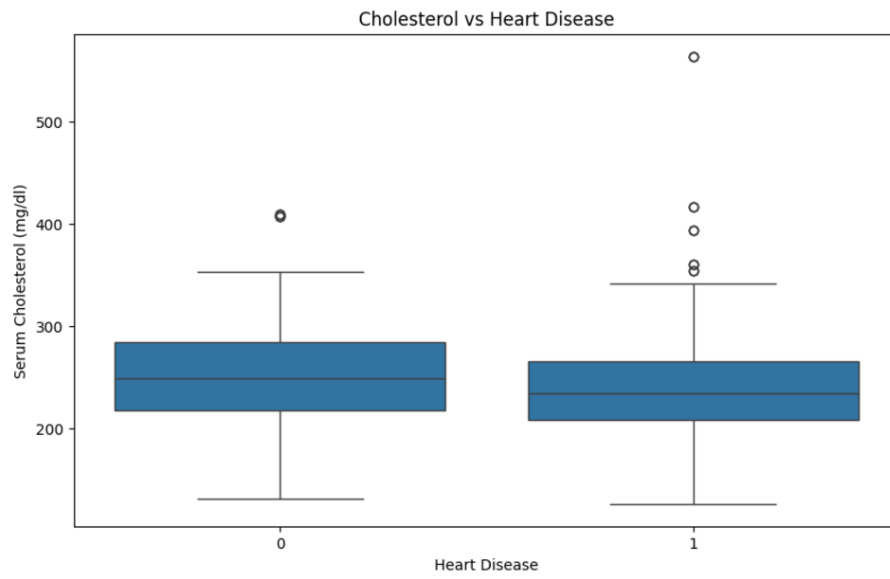


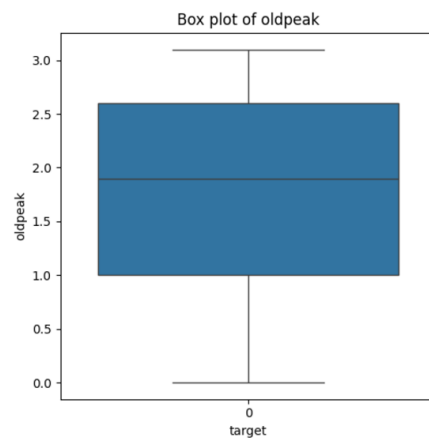
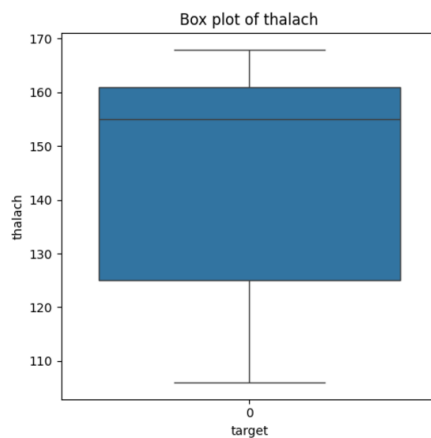
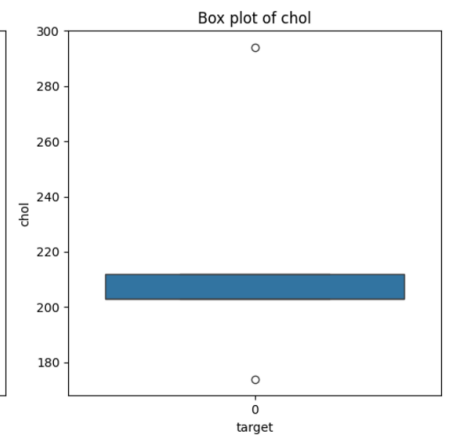
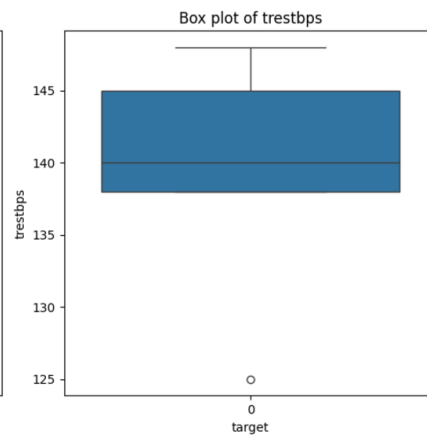
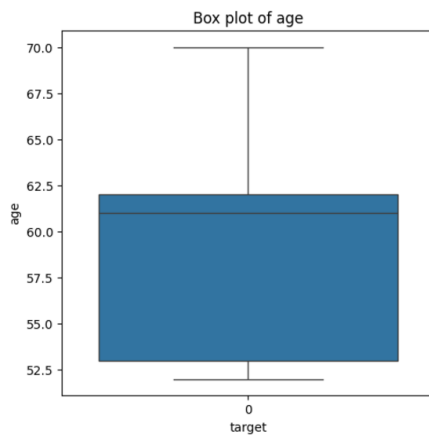
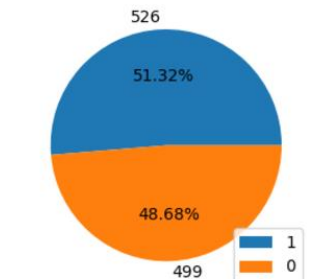
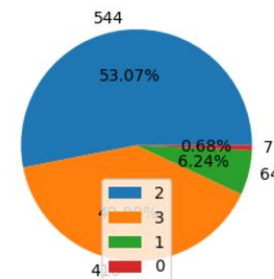
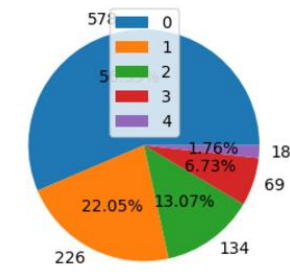
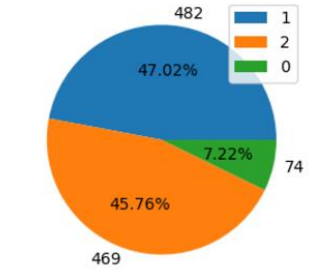
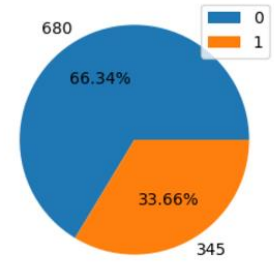
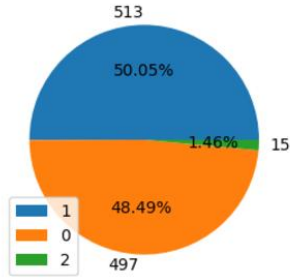
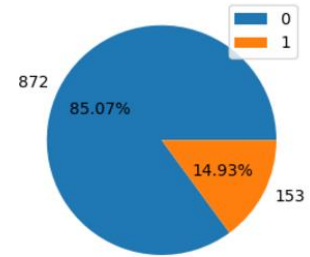
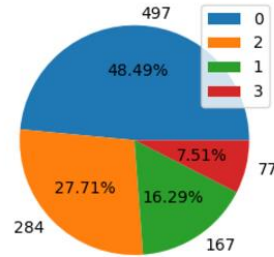
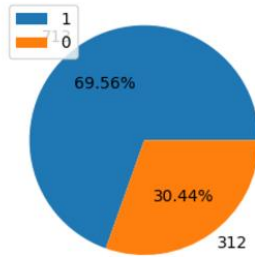
Pie Chart for chest pain types

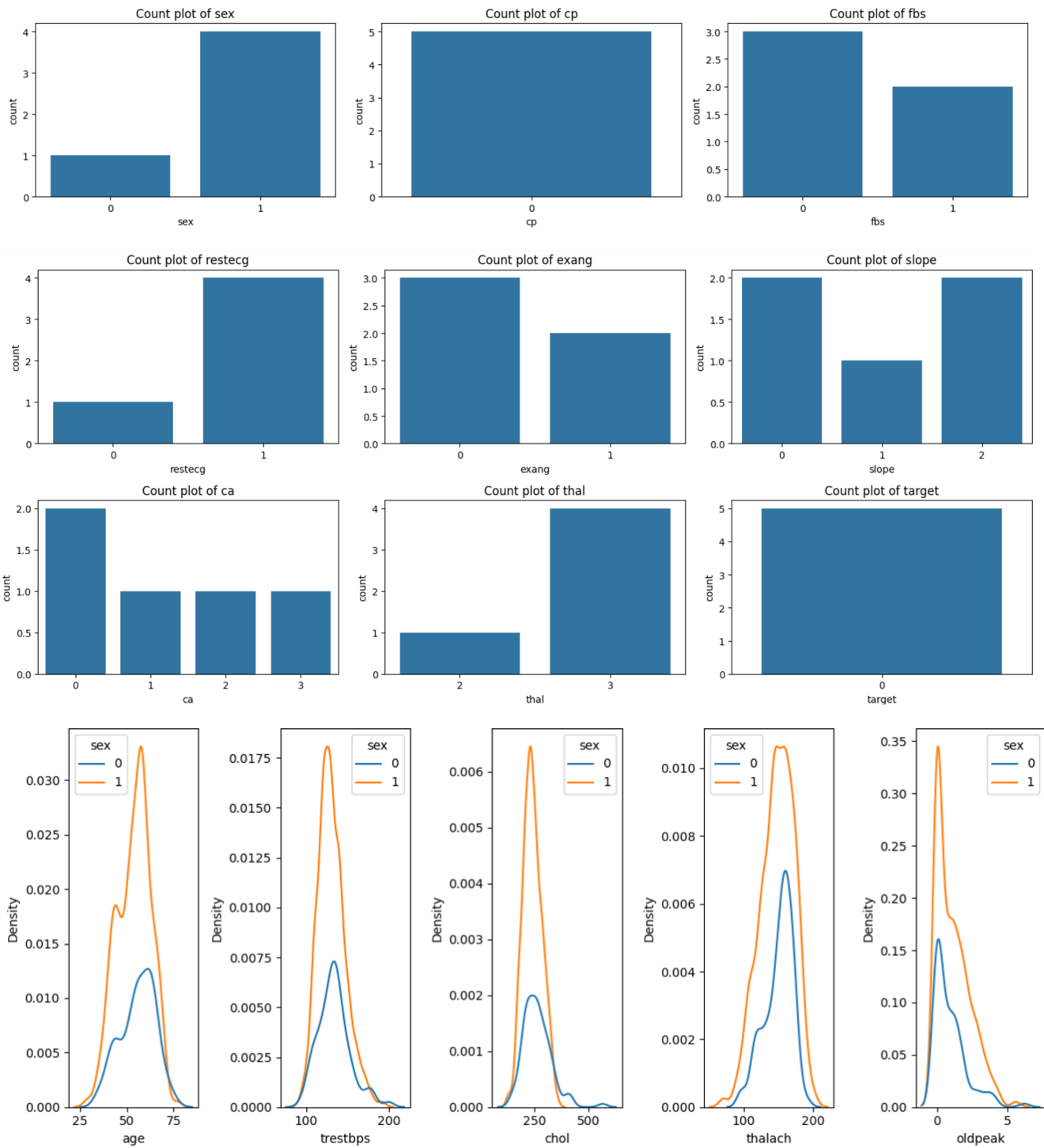


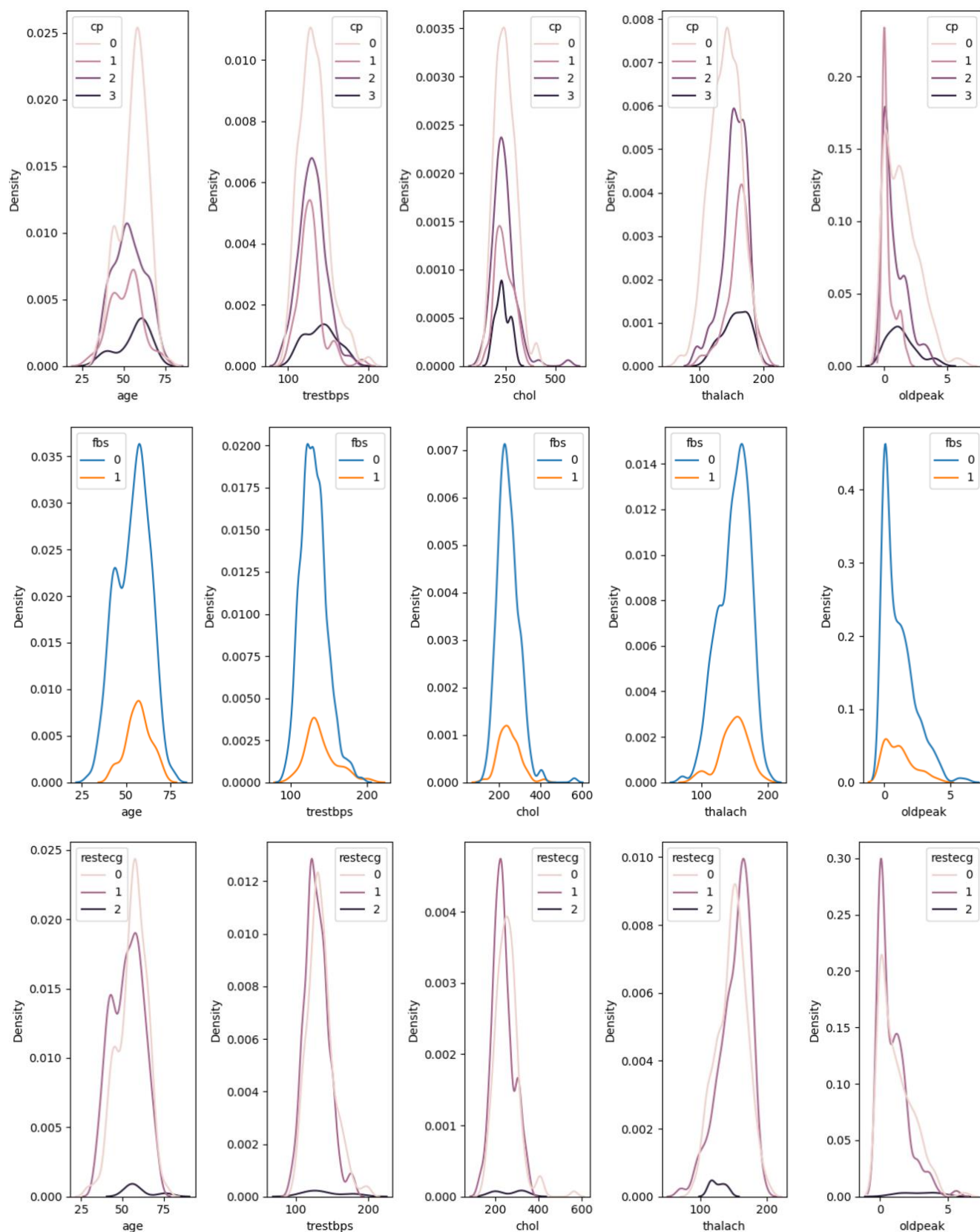




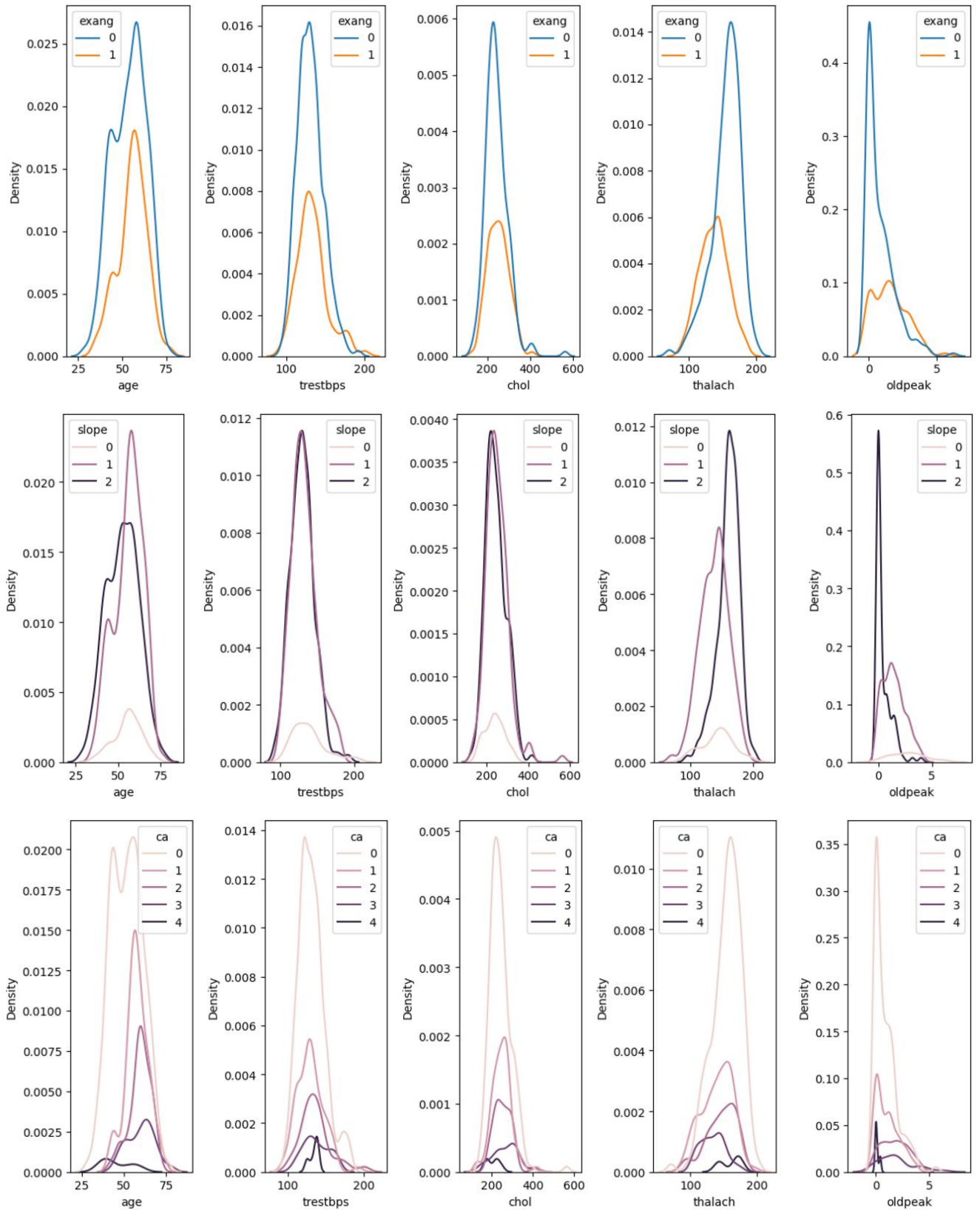












```
<ipython-input-14-aa135af883c3>:6: UserWarning: Dataset has 0 variance; skipping density estimate. Pass `warn_singular=False` to disable this warning.
sns.kdeplot(df, x=nums[j], hue=i, ax=axes[j])
```

