Visualizing And Predicting Heart Diseases with an Interactive Dash Board

Team ID : PNT2022TMID03964

Team Members : **1.Kumar**

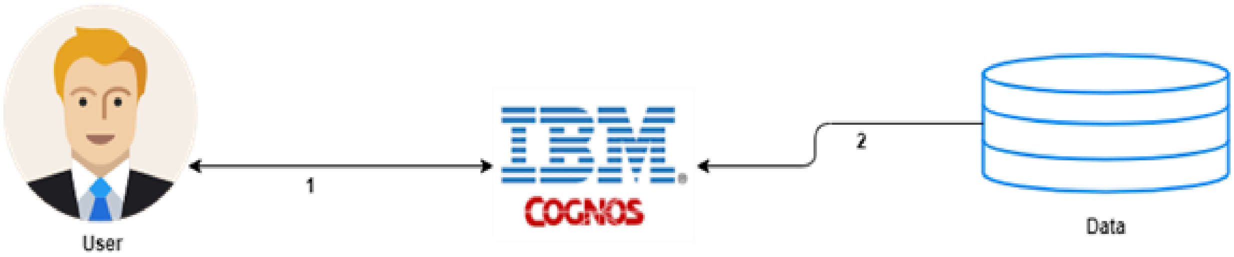
2.Kirankumar

3.Kumuraguru

4.Manigandan

5.Logeshwaran

Technical Architecture :



1.Introduction :

The leading cause of death in the developed world is heart disease. Therefore, there needs to be work done to help prevent the risks of having a heart attack or stroke.Healthcare industries generate enormous amount of data, so called big data that accommodates hidden knowledge or pattern for decision making .

Content: Use this dataset to predict which patients are most likely to suffer from a heart disease in the near future using the features given.

1.1 Project Overview :

Heart disease is perceived as the deadliest disease in the human life across the world. In particular, in this type of disease the heart is not capable in pushing the required quantity of blood to the remaining organs of the human body in order to accomplish the regular functionalities . Some of the symptoms of heart disease include physical body weakness, improper breathing, swollen feet, etc. The techniques are essential to identify the complicated heart diseases which results in high risk in turn affect the human life . Presently, diagnosis and treatment process are highly challenging due to inadequacy of physicians and diagnostic apparatus that affect the treatment of heart patients . Early diagnosis of heart disease is significant to minimize the heart related issues and to protect it from serious risks.

1.2 Purpose :

The goal of our heart disease prediction project is to determine if a patient should be diagnosed with heart disease or not, which is a binary outcome, so:

i. Positive result = 1, the patient **will be** diagnosed with heart disease.

ii. Negative result = 0, the patient **will not be** diagnosed with heart disease.

2 **Literature Survey** :

2.1 Existing Problem :

To predict the heart disease, **K-means clustering algorithm is used along with data analytics and visualization tool**. The paper discusses the pre-processing methods, classifier performances and evaluation metrics. In the result section, the visualized data shows that the prediction is accurate.

2.2 References :

[1] V. Manikantan &S.Latha,”Predicting the Analysis of Heart Disease Symptoms Using Medicinal Data Mining Methods”, International Journal on Advanced Computer Theory and Engineering, Volume-2, Issue-2, pp.5-10, 2013.

[2] Dr.A.V.Senthil Kumar, “Heart Disease Prediction Using Data Mining preprocessing and Hierarchical Clustering”, International Journal of Advanced Trends in Computer Science and Engineering, Volume-4, No.6, pp.07-18, 2015.

[3] Uma.K, M.Hanumathappa, “Heart Disease Prediction Using Classification Techniques with Feature Selection Method”, Adarsh Journal of Information Technology, Volume-5, Issue-2, pp.22-29, 2016 .

[4] Himanshu Sharma, M.A.Rizvi, “Prediction of Heart Disease using Machine Learning Algorithms:ASurvey”,International Journal on Recent and Innovation Trends in Computing and Communication,Volume5,Issue-8,pp.99-104, 2017.

[5] S.Suguna, Sakthi Sakunthala.N ,S.Sanjana, S.S.Sanjhana, “A Survey on Prediction of Heart Disease using Big data Algorithms”, International Journal of Advanced Research in Computer Engineering & Technology,Volume-6,Issue-3,pp.371-378,2017.

2.3 Problem Statement :

Date 23 September 2022

Team ID PNT2022TMID03964

Project Name Project - Visualizing and Predicting Heart Diseases with an Interactive Dash Board

Maximum Marks 2 Marks

Who does the problem affect?

The majority of people who die of coronary heart disease are 65 or older. While heart attacks can strike people of both genders in old age, women are at greater risk of dying.

What are the boundaries of the problem?

Several health conditions, your lifestyle, and your age and family history can increase your risk for heart disease.

What is the issue?

If the person is affected by heart disease, then it produces the side effects like Chest pain, chest tightness, chest pressure, chest discomfort (angina), Shortness of breath, pain in the neck, jaw, throat, upper belly area or back.

When does the issue occur?

Heart disease and the other conditions that lead to it can happen at any age. High rates of obesity and high blood pressure among younger people are putting them at risk for heart disease earlier in their life.

Where is the issue coming?

Coronary artery disease happens when coronary arteries struggle to supply the heart with enough blood, oxygen and nutrients. Cholesterol deposits, or plaques, are almost always to blame. These buildups narrow your arteries, decreasing blood flow to your heart. This can cause chest pain, shortness of breath or even a heart attack.

Why is it important to fix the problem?

Predict if the patient suffers from heart disease. The health professional enters the input values from the patient's health report. The data is fed into model which predicts the probability of having heart disease.

Which solution can be used to address the issue?

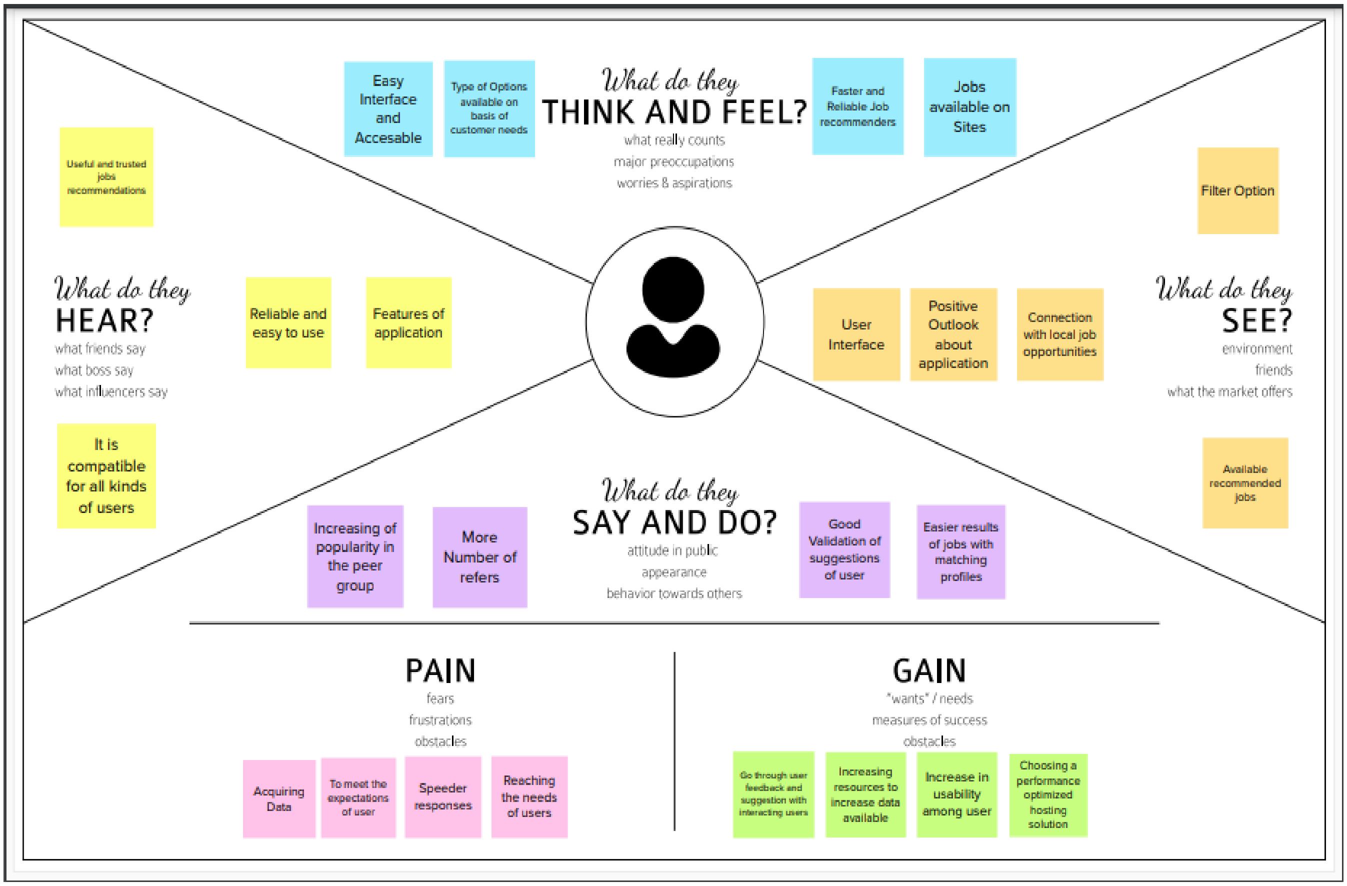
A machine learning powered web application model with the strong building of algorithm that helps to identify and predicts the disease with the identification of symptoms. It processes the breathing signals using a neural network that infers whether the person has heart disease, and if they are identified then it assesses the severity of their disease in accordance with the Movement Disorder Society Unified Heart Disease using ML algorithms.

Which methodologies are used to solve the issue?

Supervised and Un-supervised machine learning, Data mining, Computer vision with OpenCV, Python web application interface - Flask, Jupyter Notebook, IBM Cloud.

3. Ideation & Proposed Solution :

3.1 Empathy Map Canvas :


Easy Interface and Accesable
Type of Options available on basis of customer needs
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Faster and Reliable Job recommenders
Jobs available on Sites
Useful and trusted jobs recommendations
Filter Option
What do they HEAR?
what friends say
what boss say
what influencers say
Reliable and easy to use
Features of application
User Interface
Positive Outlook about application
Connection with local job opportunities
What do they SEE?
environment
friends
what the market offers
It is compatible for all kinds of users
Available recommended jobs
Increasing of popularity in the peer group
More Number of refers
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Good Validation of suggestions of user
Easier results of jobs with matching profiles
PAIN
fears
frustrations
obstacles
Acquiring Data
To meet the expectations of user
Speeder responses
Reaching the needs of users
GAIN
"wants" / needs
measures of success
obstacles
Go through user feedback and suggestion with interacting users
Increasing resources to increase data available
Increase in usability among user
Choosing a performance optimized hosting solution

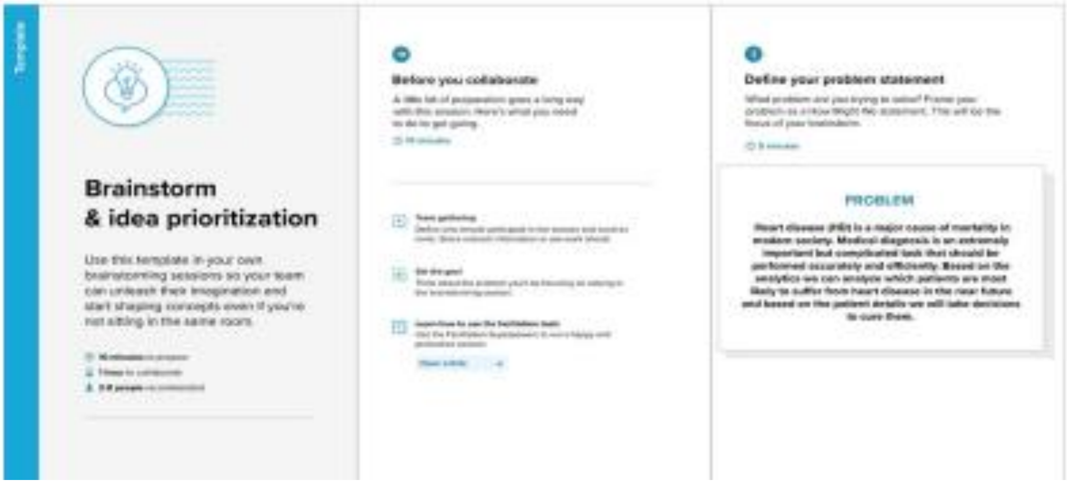
3.2 Ideation &Brainstorming :

Ideation Phase
Brainstorm & Idea Prioritization Template

Date	20 September 2022
Team ID	PNT2022TMID03964
Project Name	Visualizing and Predicting Heart Diseases with an Interactive Dash Board
Maximum Marks	4 Marks

Brainstorm & Idea Prioritization:

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

Heart disease (HD) is a major cause of mortality in modern society. Medical diagnosis is an extremely important but complicated task that should be performed accurately and efficiently. Based on the analytics we can analyze which patients are most likely to suffer from heart disease in the near future and based on the patient details we will take decisions to cure them.

3.3 Proposed Solution :

Date 23 September 2022

Team ID PNT2022TMID03964

Project Name Visualizing and Predicting Heart Diseases with an Interactive Dash Board.

Maximum Marks 2 Marks

1. Problem Statement (Problem to be solved) - The user needs a way to identify whether he/she is affected by Heart disease, improve diagnosis & quality of care, assists in predicting diseases, analysing symptoms, providing appropriate medicines, minimizing cost, extending the life span and reduces the death rate of heart patients.

2. Idea / Solution description - By predicting and visualizing the fundamentals properties that are related to heart disease and visualizing them in a dashboard.

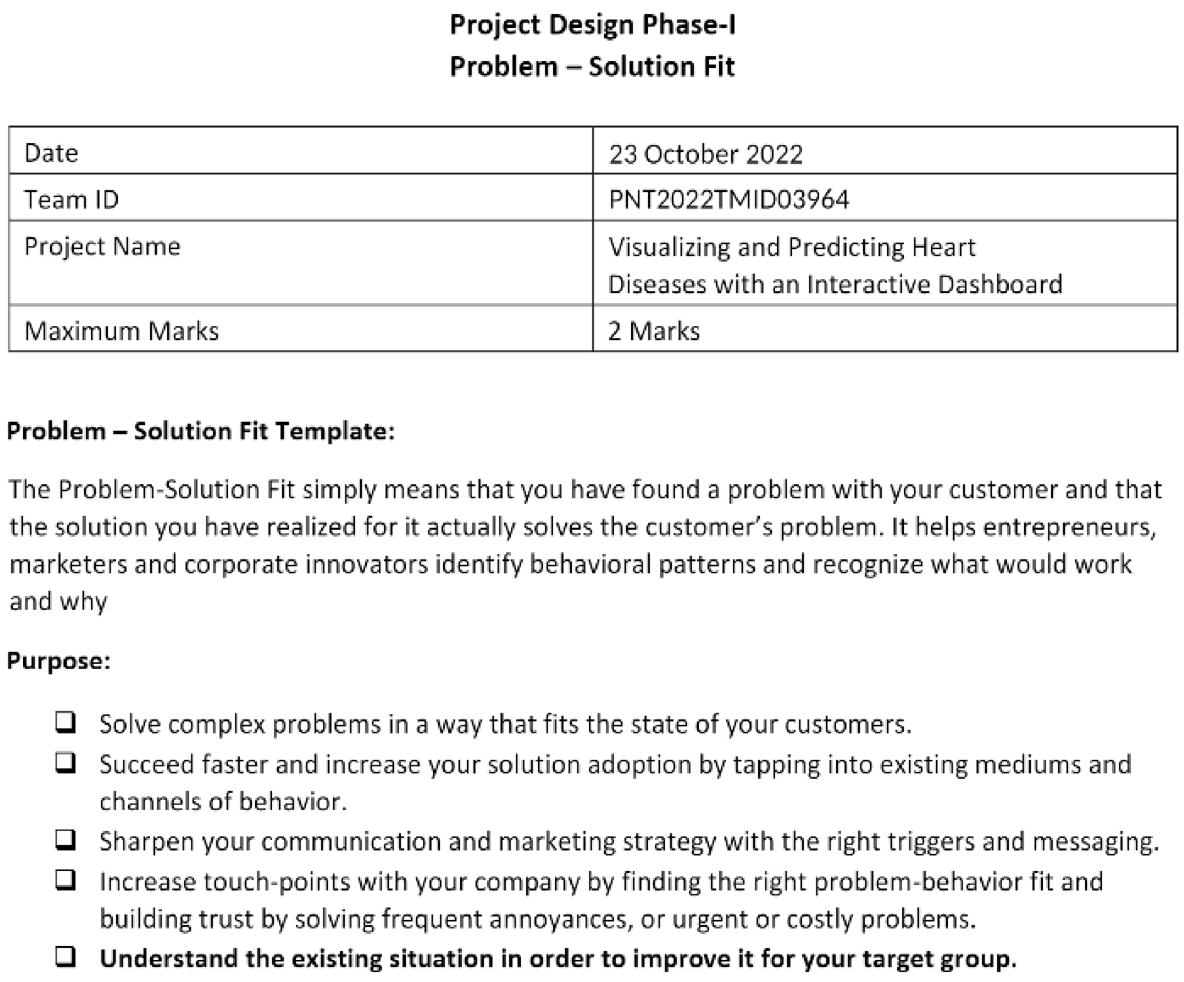
3. Novelty / Uniqueness - The use of analytics in healthcare improves care by facilitating preventive care and visually represented data provide various insights easily. Prediction is non invasive. So it is cost efficient. Earlier prediction is very helpful in reducing mortality rate.

4. Social Impact / Customer Satisfaction - It will reduce the mortality rate due to heart disease. Heart prediction can be done easier and earlier by visual analytics. As it is cost efficient, it is preferred by most of the customers. Most importantly, it is very helpful for doctors to give treatments according to the patients conditions and it's preferred by the doctors as it saves time.

5. Business Model (Revenue Model) - There are 2 ways to generate revenue from this project by creating a product model. By introducing an app for predicting heart disease or it can be integrated with smart watches for producing more efficient models.

6. Scalability of the Solution - The proposed solution will work efficiently in both smaller and larger datasets in a similar manner. In future, it can be changed to predict some other diseases with more accuracy.

3.4 Problem solution :



Project Design Phase-I
Problem – Solution Fit

Date	23 October 2022
Team ID	PNT2022TMID03964
Project Name	Visualizing and Predicting Heart Diseases with an Interactive Dashboard
Maximum Marks	2 Marks

Problem – Solution Fit Template:

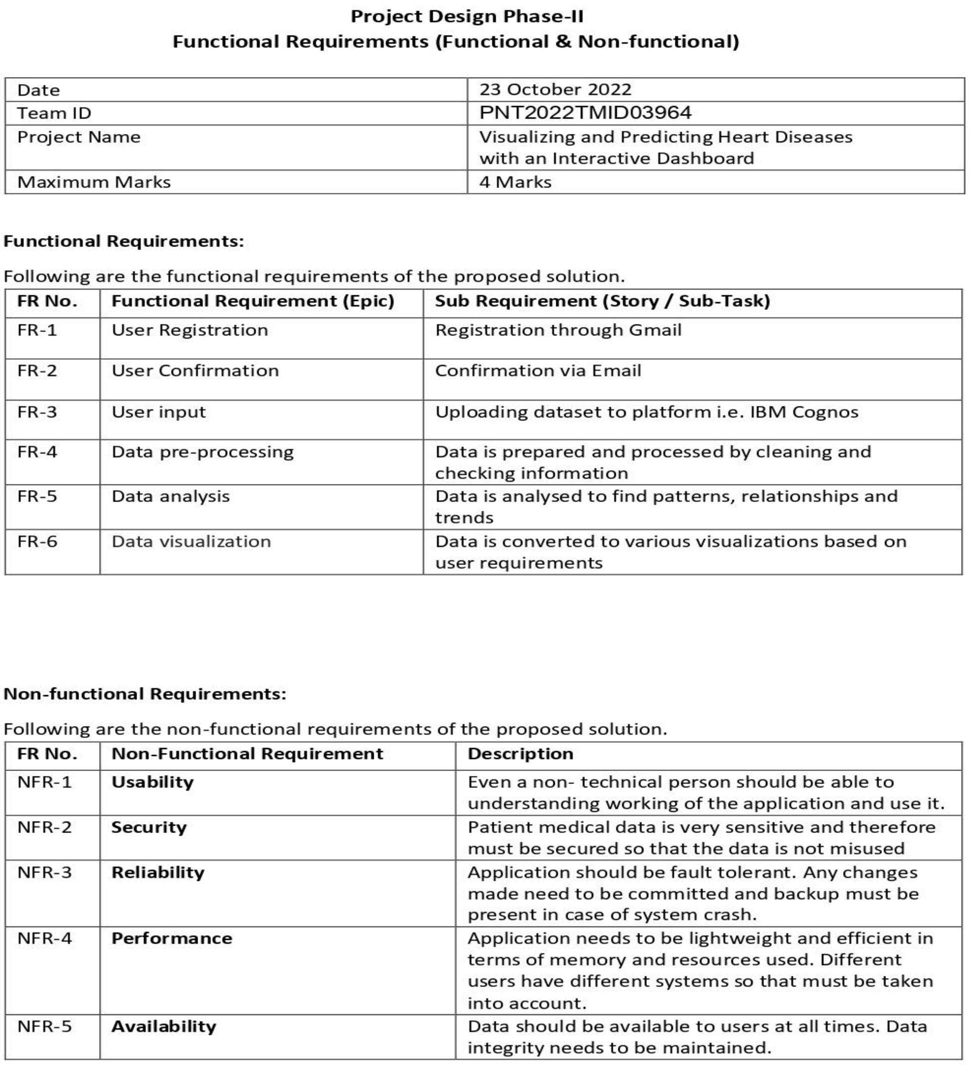
The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

Purpose:

- □ Solve complex problems in a way that fits the state of your customers.
- □ Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
- □ Sharpen your communication and marketing strategy with the right triggers and messaging.
- □ Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
- □ **Understand the existing situation in order to improve it for your target group.**

4.Requirement Analysis :

Functional And Non functional Requirement



Project Design Phase-II
Functional Requirements (Functional & Non-functional)

Date	23 October 2022
Team ID	PNT2022TMID03964
Project Name	Visualizing and Predicting Heart Diseases with an Interactive Dashboard
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	User input	Uploading dataset to platform i.e. IBM Cognos
FR-4	Data pre-processing	Data is prepared and processed by cleaning and checking information
FR-5	Data analysis	Data is analysed to find patterns, relationships and trends
FR-6	Data visualization	Data is converted to various visualizations based on user requirements

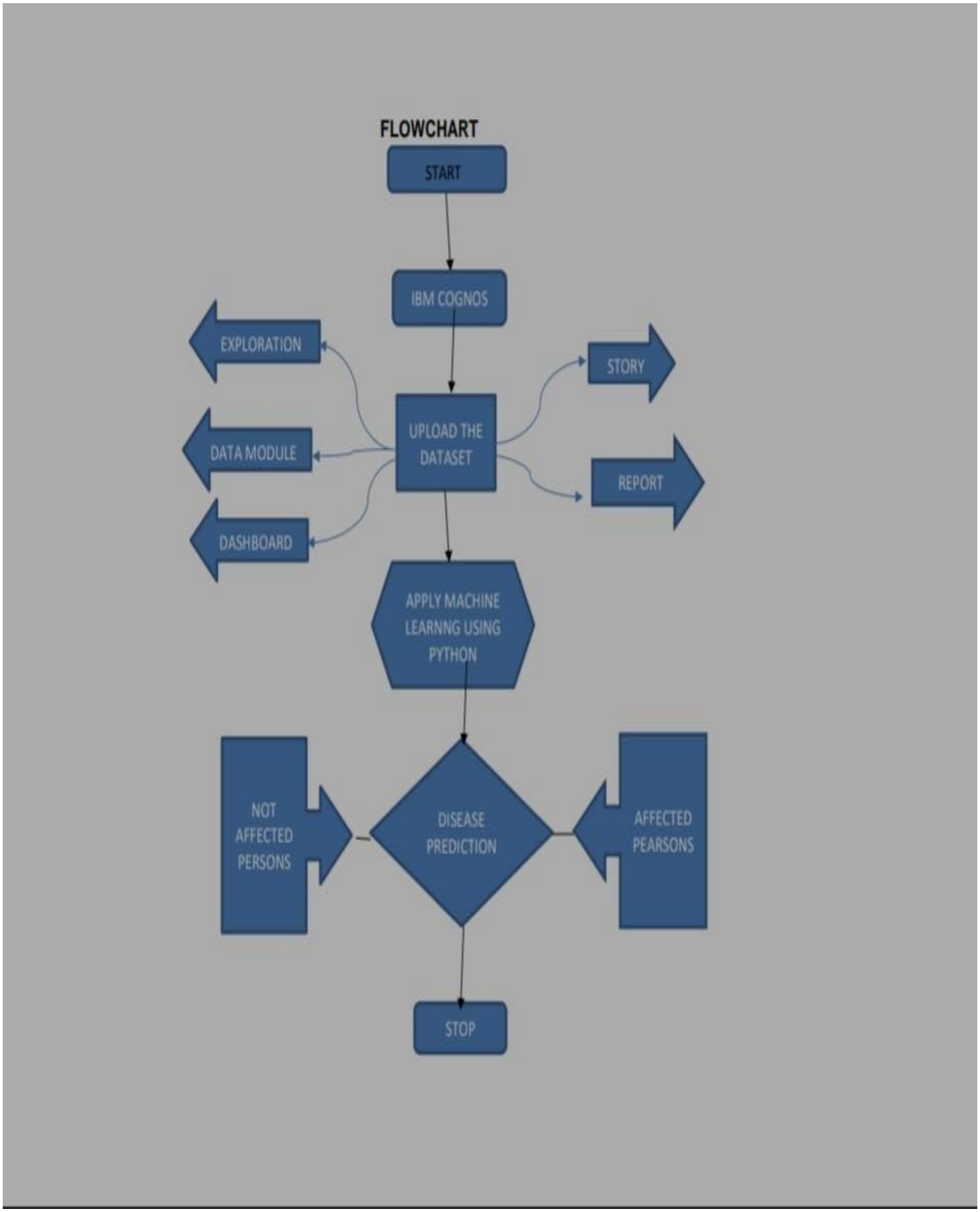
Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

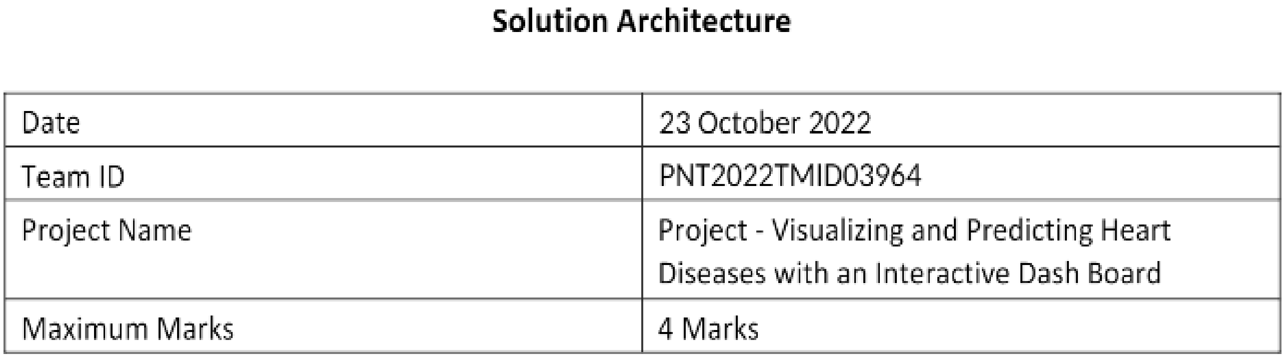
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Even a non- technical person should be able to understanding working of the application and use it.
NFR-2	**Security**	Patient medical data is very sensitive and therefore must be secured so that the data is not misused
NFR-3	**Reliability**	Application should be fault tolerant. Any changes made need to be committed and backup must be present in case of system crash.
NFR-4	**Performance**	Application needs to be lightweight and efficient in terms of memory and resources used. Different users have different systems so that must be taken into account.
NFR-5	**Availability**	Data should be available to users at all times. Data integrity needs to be maintained.

5 Project Design :

5.1 Data Flow Diagram :

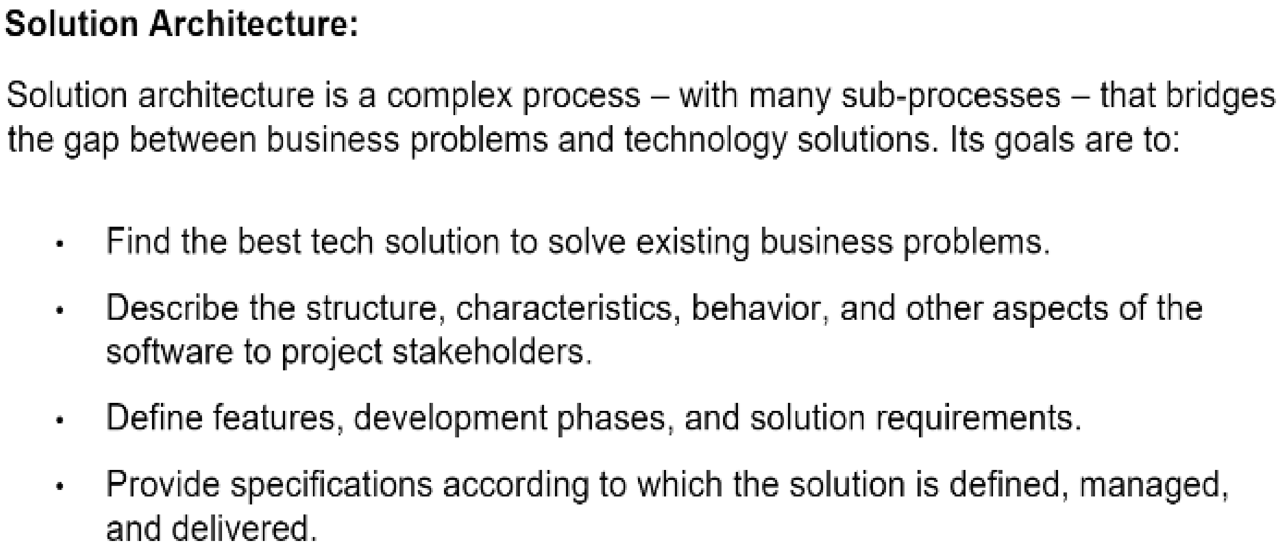


5.2 Solution & Technical Architecture:



Solution Architecture

Date	23 October 2022
Team ID	PNT2022TMID03964
Project Name	Project - Visualizing and Predicting Heart Diseases with an Interactive Dash Board
Maximum Marks	4 Marks

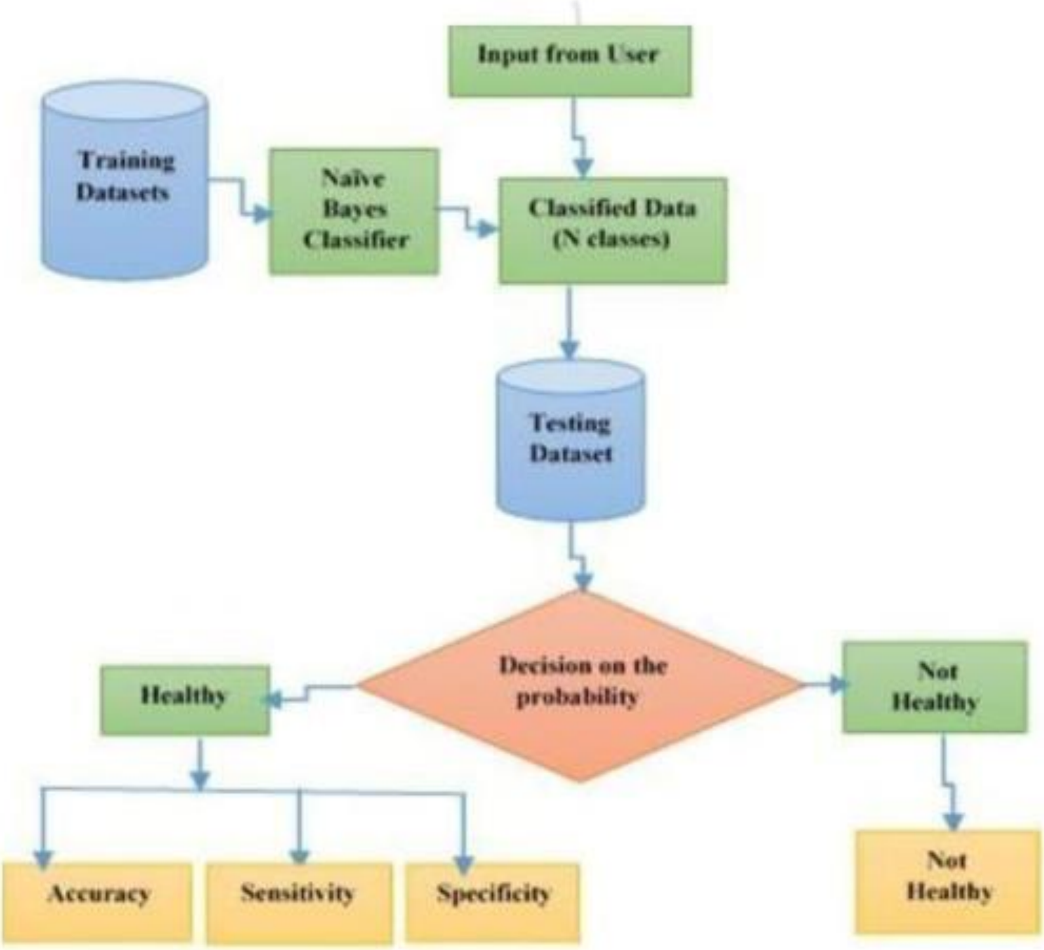


Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

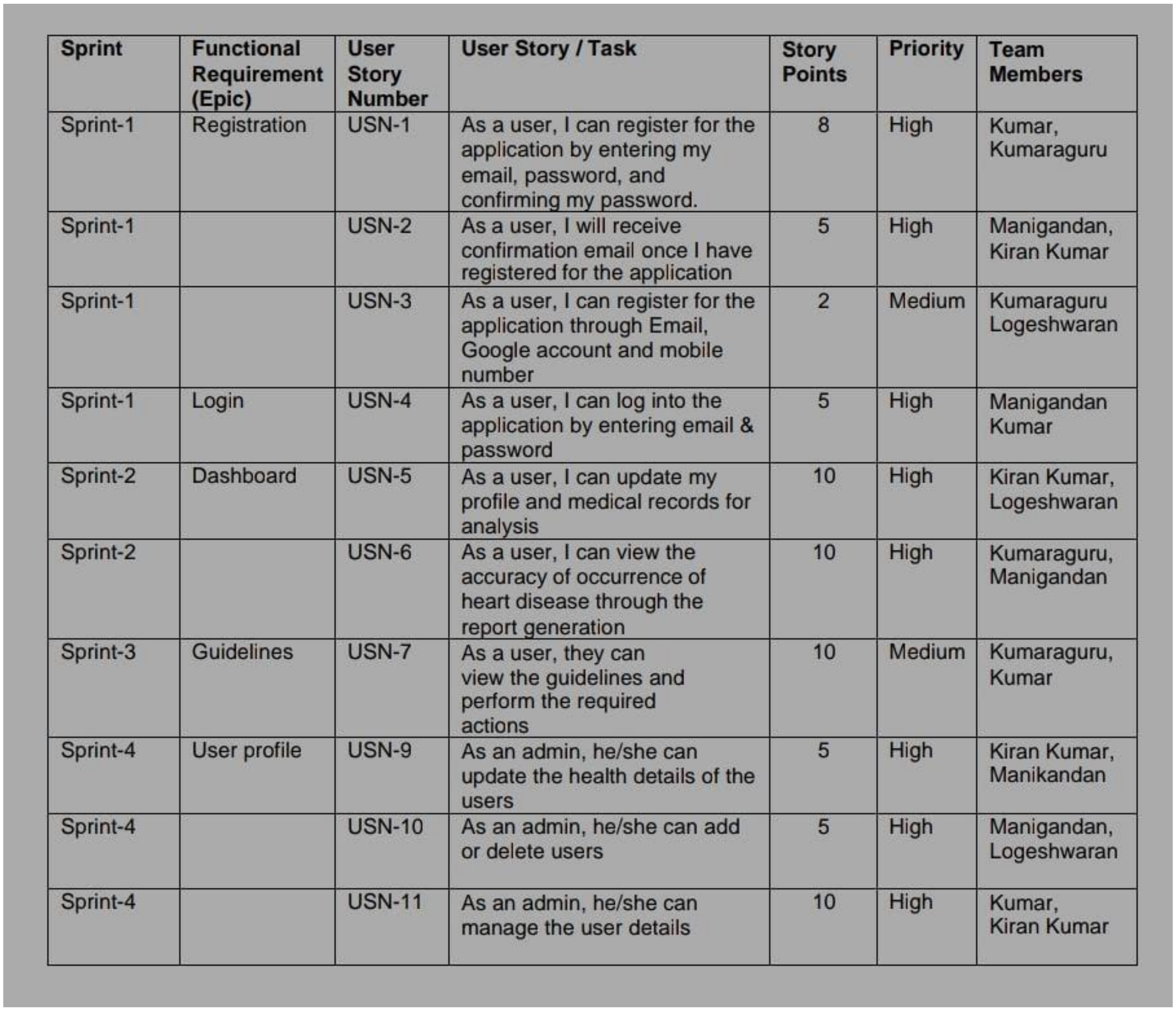
- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:



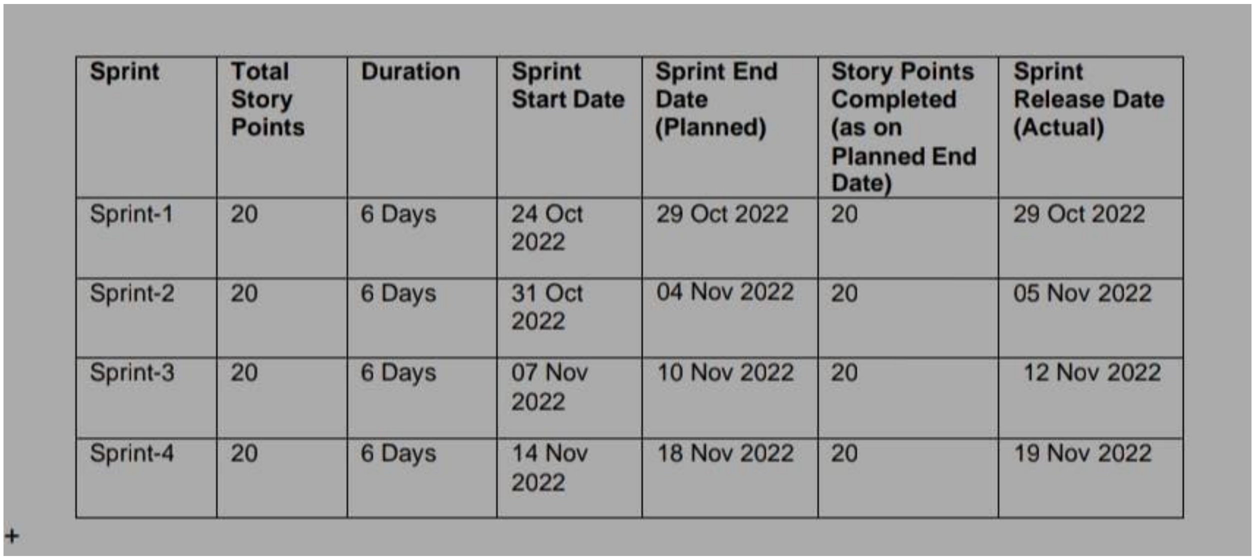
6.Project Planning &Scheduling :

6.1 Sprint Planning &Estimation :



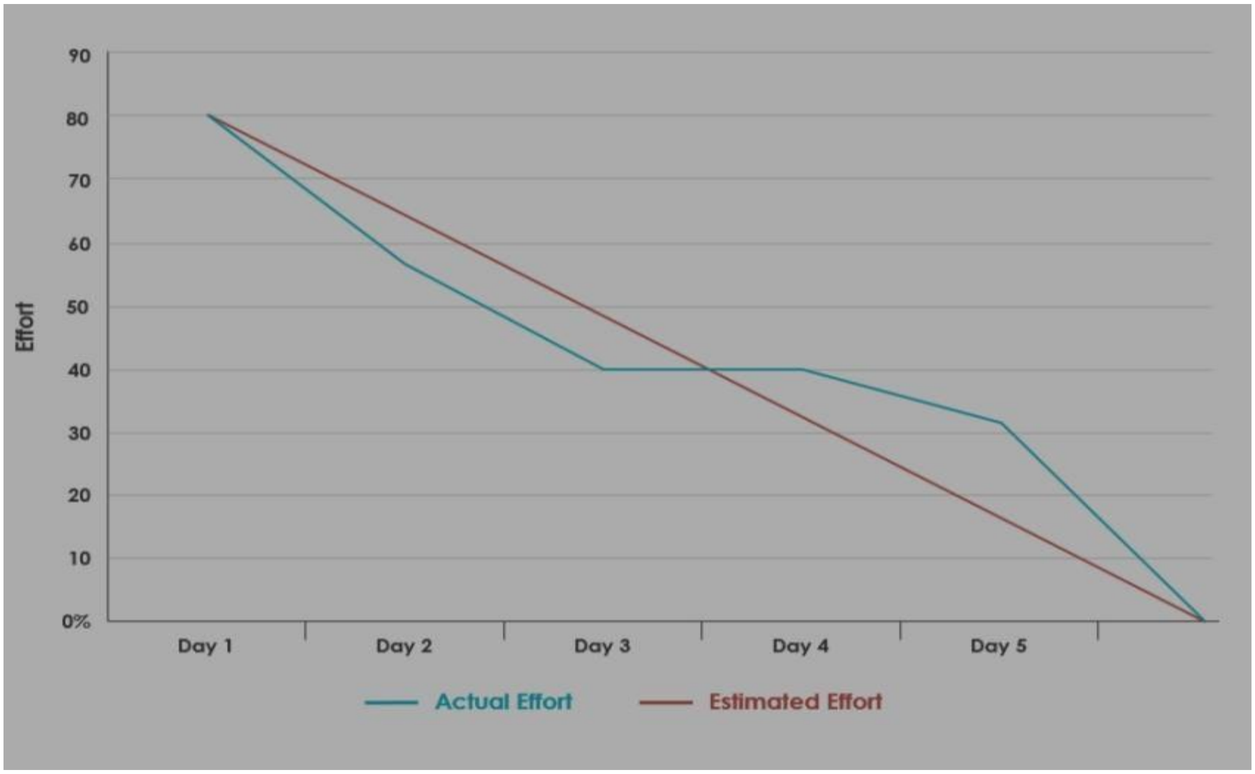
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	8	High	Kumar, Kumaraguru
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	5	High	Manigandan, Kiran Kumar
Sprint-1		USN-3	As a user, I can register for the application through Email, Google account and mobile number	2	Medium	Kumaraguru Logeshwaran
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	5	High	Manigandan Kumar
Sprint-2	Dashboard	USN-5	As a user, I can update my profile and medical records for analysis	10	High	Kiran Kumar, Logeshwaran
Sprint-2		USN-6	As a user, I can view the accuracy of occurrence of heart disease through the report generation	10	High	Kumaraguru, Manigandan
Sprint-3	Guidelines	USN-7	As a user, they can view the guidelines and perform the required actions	10	Medium	Kumaraguru, Kumar
Sprint-4	User profile	USN-9	As an admin, he/she can update the health details of the users	5	High	Kiran Kumar, Manikandan
Sprint-4		USN-10	As an admin, he/she can add or delete users	5	High	Manigandan, Logeshwaran
Sprint-4		USN-11	As an admin, he/she can manage the user details	10	High	Kumar, Kiran Kumar

6.2. Sprint Delivery Schedule:



Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	04 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	10 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	18 Nov 2022	20	19 Nov 2022

6.3.Reports From Jira



7.CODING AND SOLUTIONING

7.1.Feature

The objective of this project is to check whether the patient is likely to be diagnosed with any cardio vascular heart diseases based on their medical attributes such as gender, age, chest, pain, fasting sugar level, etc

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
import seaborn as sns
%matplotlib inline

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn import tree
from warnings import filterwarnings
filterwarnings(\"ignore\")

from sklearn.metrics import
log_loss,roc_auc_score,precision_score,f1_score,recall_score,roc_curve,
auc,plot_roc_curve
```

```
   from sklearn.metrics import classification_report,
confusion_matrix,accuracy_score,fbeta_score,matthews_corrcoef

   from sklearn import metrics

   from mlxtend.plotting import plot_confusion_matrix



    from sklearn.pipeline import make_pipeline, make_union

    from sklearn.preprocessing import PolynomialFeatures

    from sklearn.feature_selection import SelectFwe, f_regression


    "from sklearn.ensemble import RandomForestClassifier\n",

    "from sklearn.neighbors import KNeighborsClassifier\n",

    "from sklearn.tree import DecisionTreeClassifier\n",

    "from sklearn.naive_bayes import GaussianNB"


    "dataset = pd.read_csv('dataset.csv',sep=',',encoding=\"utf-8\")"


    "type(dataset)"


    "dataset.shape"


      "RangeIndex: 270 entries, 0 to 269\n",

      "Data columns (total 14 columns):\n",
      " #   Column    Non-Null Count  Dtype  \n",
      "---  ------    --------------  -----  \n",
      " 0   age       270 non-null    int64  \n",
      " 1   sex       270 non-null    int64  \n",
      " 2   cp        270 non-null    int64  \n",
      " 3   trestbps  270 non-null    int64  \n",
      " 4   chol      270 non-null    int64  \n",
      " 5   fbs       270 non-null    int64  \n",
      " 6   restecg   270 non-null    int64  \n",
      " 7   thalach   270 non-null    int64  \n",
      " 8   exang     270 non-null    int64  \n",
      " 9   oldpeak   270 non-null    float64\n",
      " 10  slope     270 non-null    int64  \n",
      " 11  ca        270 non-null    int64  \n",
      " 12  thal      270 non-null    int64  \n",
      " 13  target    270 non-null    int64  \n",
      "dtypes: float64(1), int64(13)\n",
      "memory usage: 29.7 KB\n"

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

```
   },
   "execution_count": 10,
   "metadata": {},
   "output_type": "execute_result"
  }
\
     "</style>\n",
     "<table border=\"1\" class=\"dataframe\">\n",
     "  <thead>\n",
     "    <tr style=\"text-align: right;\">\n",
     "      <th></th>\n",
     "      <th>age</th>\n",
     "      <th>sex</th>\n",
     "      <th>cp</th>\n",
     "      <th>trestbps</th>\n",
     "      <th>chol</th>\n",
     "      <th>fbs</th>\n",
     "      <th>restecg</th>\n",
     "      <th>thalach</th>\n",
     "      <th>exang</th>\n",
     "      <th>oldpeak</th>\n",
     "      <th>slope</th>\n",
     "      <th>ca</th>\n",
    "      <th>thal</th>\n",
    "      <th>target</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
    "      <th>count</th>\n",
    ,


    "              age         sex           cp    trestbps         chol          fbs  \\\n",

   ],
   "source": [
    "dataset.describe()"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "9ece7ed8",
   "metadata": {},
   "source": [
    "The scale of each feature column is different and quite varied as well. While the maximum for age reaches 77, the maximum of chol (serum cholestoral) is 564."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 12,
   "id": "227e54aa",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
,
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>age</th>\n",
"      <th>sex</th>\n",
"      <th>cp</th>\n",
"      <th>trestbps</th>\n",
"      <th>chol</th>\n",
"      <th>fbs</th>\n",
"      <th>restecg</th>\n",
"      <th>thalach</th>\n",
"      <th>exang</th>\n",
"      <th>oldpeak</th>\n",
    "      <th>slope</th>\n",
    "      <th>ca</th>\n",
    "      <th>thal</th>\n",
    "      <th>target</th>\n",


   "cell_type": "code",
   "execution_count": 14,
   "id": "acd9bb35",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "age         0\n",
       "sex         0\n",
       "cp          0\n",
       "trestbps    0\n",
       "chol        0\n",
       "fbs         0\n",
       "restecg     0\n",
       "thalach     0\n",
       "exang       0\n",
   "oldpeak     0\n",
   "slope       0\n",
   "ca          0\n",
   "thal        0\n",
   "target      0\n",
   "dtype: int64"
  ]

{
 "data": {
  "text/plain": [
   "age          41\n",
   "sex           2\n",
   "cp            4\n",
   "trestbps     47\n",
   "chol        144\n",
   "fbs           2\n",
   "restecg       3\n",
   "thalach      90\n",
   "exang         2\n",
   "oldpeak      39\n",
   "slope         3\n",
   "ca            4\n",
     "thal         3\n",
     "target       2\n",
     "dtype: int64"
    ]

  "source": [
   "print('cp ',dataset['cp'].unique())\n",
   "print('fbs ',dataset['fbs'].unique())\n",
   "print('restecg ',dataset['restecg'].unique())\n",
   "print('exang ',dataset['exang'].unique())\n",
   "print('slope ',dataset['slope'].unique())\n",
   "print('ca ',dataset['ca'].unique())\n",
   "print('thal ',dataset['thal'].unique())\n"
  ]
 },

    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
  "source": [
    "# Display age distribution based on heart disease\n",
    "sns.distplot(dataset[dataset['target'] == 1]['age'], label='Do not have heart disease')\n",
    "sns.distplot(dataset[dataset['target'] == 2]['age'], label = 'Have heart disease')\n",
    "plt.xlabel('Frequency')\n",
    "plt.ylabel('Age')\n",
    "plt.title('Age Distribution based on Heart Disease')\n",
    "plt.legend()\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "fb21e14c",
   "metadata": {},
   "source": [
    "Get min, max and average of the age of the people do not have heart disease"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "id": "c0da4bb5",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Min age of people who do not have heart disease:  29\n",
      "Max age of people who do not have heart disease:  76\n",
      "Average age of people who do not have heart disease:
52.70666666666666\n"
     ]
    }
   ],
   "source": [
    "print('Min age of people who do not have heart disease: ',
min(dataset[dataset['target'] == 1]['age']))\n",
    "print('Max age of people who do not have heart disease: ',
max(dataset[dataset['target'] == 1]['age']))\n",
    "print('Average age of people who do not have heart disease: ',
dataset[dataset['target'] == 1]['age'].mean())"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "a8e8f507",
   "metadata": {},
   "source": [
    "Get min, max and average of the age of the people have heart disease"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 27,
   "id": "388a33d3",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Min age of people who have heart disease:  35\n",
      "Max age of people who have heart disease:  77\n",
      "Average age of people who have heart disease:  56.59166666666667\n"
     ]
    }
   ],
   "source": [
    "print('Min age of people who have heart disease: ', min(dataset[dataset['target'] == 2]['age']))\n",
    "print('Max age of people who have heart disease: ', max(dataset[dataset['target'] == 2]['age']))\n",
    "print('Average age of people who have heart disease: ', dataset[dataset['target'] == 2]['age'].mean())"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "327be9df",
   "metadata": {},
   "source": [
    "From the data, we can say that the heart disease infects the old and young people, and the probability of the old people te be infected is higher than young people.  \n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "e3961af5",
   "metadata": {},
   "source": [
    "**Heart Disease Frequency for ages**"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "id": "a73172bb",
   "metadata": {},
   "outputs": [


     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
"pd.crosstab(dataset.age,dataset.target).plot(kind=\"bar\",figsize=(20,6))\n",
    "plt.title('Heart Disease Frequency for Ages')\n",
    "plt.xlabel('Age')\n",
    "plt.ylabel('Frequency')\n",
    "plt.savefig('heartDiseaseAndAges.png')\n",
    "plt.show()"
   ]
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.9.12"
```

7.2.FEATURE 2

```
"import pandas as pd\n",
"import numpy as np\n",
"import matplotlib.pyplot as plt\n",
"import seaborn as sns"
  "</style>\n",
  "<table border=\"1\" class=\"dataframe\">\n",
  "  <thead>\n",
  "    <tr style=\"text-align: right;\">\n",
  "      <th></th>\n",
  "      <th>Age</th>\n",
  "      <th>Sex</th>\n",
  "      <th>Chest pain type</th>\n",
  "      <th>BP</th>\n",
  "      <th>Cholesterol</th>\n",
  "      <th>FBS over 120</th>\n",
  "      <th>EKG results</th>\n",
  "      <th>Max HR</th>\n",
  "      <th>Exercise angina</th>\n",
  "      <th>ST depression</th>\n",
  "      <th>Slope of ST</th>\n",
  "      <th>Number of vessels fluro</th>\n",
       "      <th>Thallium</th>\n",
       "      <th>Heart Disease</th>\n",
    "execution_count": 3,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df.head()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 4,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "Age                        0\n",
       "Sex                        0\n",
       "Chest pain type            0\n",
       "BP                         0\n",
       "Cholesterol                0\n",
       "FBS over 120               0\n",
       "EKG results                0\n",
       "Max HR                     0\n",
       "Exercise angina            0\n",
       "ST depression              0\n",
       "Slope of ST                0\n",
       "Number of vessels fluro    0\n",
       "Thallium                   0\n",
       "Heart Disease              0\n",
       "dtype: int64"
      ]
     },
     "execution_count": 4,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "df.isnull().sum()"
   ]
  },
  {
```

```
"cell_type": "code",
"execution_count": 5,
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "<class 'pandas.core.frame.DataFrame'>\n",
   "RangeIndex: 270 entries, 0 to 269\n",
   "Data columns (total 14 columns):\n",
   " #   Column                   Non-Null Count  Dtype  \n",
   "---  ------                   --------------  -----  \n",
   " 0   Age                      270 non-null    int64  \n",
   " 1   Sex                      270 non-null    int64  \n",
   " 2   Chest pain type          270 non-null    int64  \n",
   " 3   BP                       270 non-null    int64  \n",
   " 4   Cholesterol              270 non-null    int64  \n",
   " 5   FBS over 120             270 non-null    int64  \n",
   " 6   EKG results              270 non-null    int64  \n",
   " 7   Max HR                   270 non-null    int64  \n",
   " 8   Exercise angina          270 non-null    int64  \n",
   " 9   ST depression            270 non-null    float64\n",
     " 10  Slope of ST              270 non-null    int64  \n",
     " 11  Number of vessels fluro  270 non-null    int64  \n",
     " 12  Thallium                 270 non-null    int64  \n",
     " 13  Heart Disease            270 non-null    object \n",
     "dtypes: float64(1), int64(12), object(1)\n",
     "memory usage: 29.7+ KB\n",
     "None\n"
    ]
   }
  ],
  "source": [
   "print(df.info())"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 6,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "<AxesSubplot:>"
      ]
     },
     "execution_count": 6,
     "metadata": {},
     "output_type": "execute_result"
    },
   ,
  {
   "cell_type": "code",
   "execution_count": 7,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<seaborn.axisgrid.PairGrid at 0x1896d9664f0>"
      ]
     },
     "execution_count": 7,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "text/plain": [
     "<Figure size 2340x2340 with 182 Axes>"
    ]
   },
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
 "source": [
  "sns.pairplot(data=df)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 8,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[<AxesSubplot:title={'center':'Age'}>,\n",
    "       <AxesSubplot:title={'center':'Sex'}>,\n",
    "       <AxesSubplot:title={'center':'Chest pain type'}>,\n",
    "       <AxesSubplot:title={'center':'BP'}>],\n",
    "      [<AxesSubplot:title={'center':'Cholesterol'}>,\n",
    "       <AxesSubplot:title={'center':'FBS over 120'}>,\n",
    "       <AxesSubplot:title={'center':'EKG results'}>,\n",
    "       <AxesSubplot:title={'center':'Max HR'}>],\n",
    "      [<AxesSubplot:title={'center':'Exercise angina'}>,\n",
    "       <AxesSubplot:title={'center':'ST depression'}>,\n",
    "       <AxesSubplot:title={'center':'Slope of ST'}>,\n",
    "       <AxesSubplot:title={'center':'Number of vessels fluro'}>],\n",
    "      [<AxesSubplot:title={'center':'Thallium'}>,
<AxesSubplot:>,\n",
    "       <AxesSubplot:>, <AxesSubplot:>],\n",
    "      [<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>,
<AxesSubplot:>]],\n",
    "     dtype=object)"
   ]
  },
  "execution_count": 8,
  "metadata": {},
  "output_type": "execute_result"
 },
  "<Figure size 720x864 with 20 Axes>"
```

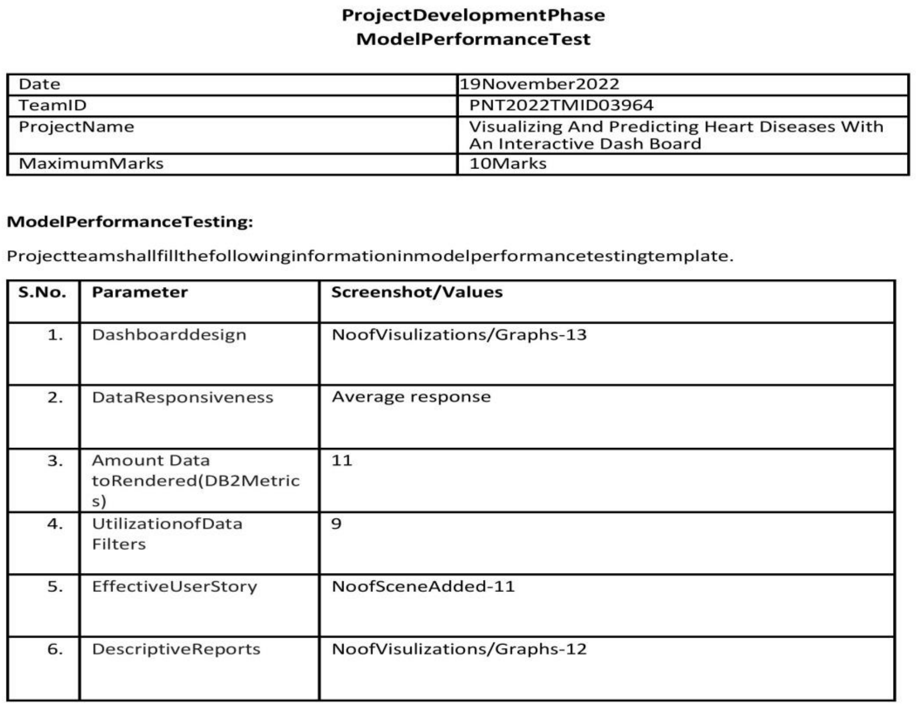
```
    ]
   },
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
 "source": [
  "df.hist(figsize=(10,12), layout=(5,4))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 9,
 "metadata": {},
 "outputs": [
  {
      "<Figure size 720x720 with 18 Axes>"
    ]
   },
   "metadata": {
    "needs_background": "light"
```

```
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "df.plot(kind='box', subplots=True, layout=(6,3), figsize=(10,10))\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<seaborn.axisgrid.FacetGrid at 0x189764cc430>"
      ]
     },
     "execution_count": 10,
     "metadata": {},
     "output_type": "execute_result"
```

```
 },
 {
  "data": {
  "execution_count": 18,
"metadata": {},
"outputs": [
 {
  "data": {
   "text/html": [
    "<div>\n",
    "<style scoped>\n",
    "    .dataframe tbody tr th:only-of-type {\n",
    "        vertical-align: middle;\n",
    "    }\n",
    "\n",
    "    .dataframe tbody tr th {\n",
    "        vertical-align: top;\n",
    "    }\n",
    "\n",
    "    .dataframe thead th {\n",
    "        text-align: right;\n",
    "    }\n",
    "</style>\n",
    "<table border=\"1\" class=\"dataframe\">\n",
    "  <thead>\n",
    "    <tr style=\"text-align: right;\">\n",
    "      <th></th>\n",
    "      <th>Age</th>\n",
    "      <th>Sex</th>\n",
    "      <th>Chest pain type</th>\n",
    "      <th>BP</th>\n",
    "      <th>Cholesterol</th>\n",
    "      <th>FBS over 120</th>\n",
    "      <th>EKG results</th>\n",
    "      <th>Max HR</th>\n",
    "      <th>Exercise angina</th>\n",
    "      <th>ST depression</th>\n",
    "      <th>Slope of ST</th>\n",
    "      <th>Number of vessels fluro</th>\n",
    "      <th>Thallium</th>\n",
    "      <th>Heart Disease</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
  "cell_type": "code",
   "execution_count": 19,
   "metadata": {},
   "outputs": [],
   "source": [
    "x=df.drop(['Heart Disease'], axis=1)\n",
    "y=df['Heart Disease']"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 20,
   "metadata": {},
   "outputs": [],
   "source": [
    "x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3, random_state=40)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 21,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "x_train- 2457\n",
      "x_test- 1053\n",
      "y_train- 189\n",
      "x_test- 1053\n"
     ]
    }
   ],
   "source": [
    "print('x_train-', x_train.size)\n",
    "print('x_test-', x_test.size)\n",
    "print('y_train-', y_train.size)\n",
    "print('x_test-', x_test.size)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 22,
   "metadata": {},
   "outputs": [
 {
   "text": [
   "Testing Accuracy: 0.9382716049382716\n"
  ]
 }
],
"source": [
 "TP=cm[0][0]\n",
 "TN=cm[1][1]\n",
 "FN=cm[1][0]\n",
 "FP=cm[0][1]\n",
 "print('Testing Accuracy:', (TP+TN+FN)/(TP+TN+FN+FP))"
]
```

9.RESULTS

9.1 PERFORMANCE METRICS



ProjectDevelopmentPhase
ModelPerformanceTest

Date	19November2022
TeamID	PNT2022TMID03964
ProjectName	Visualizing And Predicting Heart Diseases With An Interactive Dash Board
MaximumMarks	10Marks

ModelPerformanceTesting:

Projectteamshallfillthefollowinginformationinmodelperformancetestingtemplate.

S.No.	Parameter	Screenshot/Values
1.	Dashboarddesign	NoofVisulizations/Graphs-13
2.	DataResponsiveness	Average response
3.	Amount Data toRendered(DB2Metrics)	11
4.	UtilizationofData Filters	9
5.	EffectiveUserStory	NoofSceneAdded-11
6.	DescriptiveReports	NoofVisulizations/Graphs-12

10.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- The EHDPS predicts the likelihood of patients getting heart disease. It enables significant knowledge, eg, relationships between medical factors related to heart disease and patterns, to be established.
- Heart disease is one of the biggest causes of morbidity and mortality among the population of the world. Prediction of cardiovascular disease is regarded as one of the most important subjects in the section of clinical data analysis. The amount of data in the healthcare industry is huge
- guidelines of blood pressure, total cholesterol, and LDL cholesterol effectively predict CHD risk in a middle-aged white population sample.
- Heart attacks can be predicted months in advance by assessing the risk factors of the patient, which include hypercholesterolemia, hypertension, diabetes and tobacco use, along with obesity, lack of exercise, and elevated inflammatory markers such as CRP.
- Cardiology information system or CIS is a collection of clinical information using various software. This information is collected digitally on the software. It helps the end users in taking various decisions and to

help in advancement of science and sharing of information in the field of cardiology.

DISADVANTAGES

- Prediction of cardiovascular disease results is not accurate. 2. Data mining techniques does not help to provide effective decision making.
- Those with heart failure can develop swelling, dizziness, and other symptoms that can affect their ability to complete daily tasks. A person with diagnosed heart disease must also live with the stress of knowing they have a long-term illness that could result in a cardiac event, such as heart attack or stroke.
- Heart disease and stroke can be fatal, but they can also lead to serious illness, disability, and lower quality of life. Suffering a stroke may lead to significant disability, such as paralysis, speech difficulties, and emotional problems.

11.CONCLUSION:

This overview of the project conveys the idea that numerous methods have been investigated for diagnosing cardio vascular disease. Big data machine learning, data mining can be used to great success to analyse the prediction model with the highest degree of accuracy. The primary goal of this projectis to diagnose cardiovascular disease or heart utilizing a variety of techniques and procedures to obtain a progonosis

12.FUTURE SCOPE:

The objective of this project is to check whether the patient is likely to be diagnosed with any cardiovascular heart diseases based on their medical attributes such as gender, age, chest pain, fasting sugar level, etc. A dataset is selected from the UCI repository with patient's medical history and attributes. The proposed work predicts the chances of Heart Disease and classifies patient's risk level by implementing different data mining techniques such as Naive Bayes, Decision Tree, Logistic Regression and Random Forest.

12 APPENDIX:

SOURCE CODE:

```
          import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
import seaborn as sns
%matplotlib inline

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn import tree
from warnings import filterwarnings
filterwarnings(\"ignore\")
```

```
   from sklearn.metrics import
log_loss,roc_auc_score,precision_score,f1_score,recall_score,roc_curve,
auc,plot_roc_curve
   from sklearn.metrics import classification_report,
confusion_matrix,accuracy_score,fbeta_score,matthews_corrcoef
   from sklearn import metrics
   from mlxtend.plotting import plot_confusion_matrix



    from sklearn.pipeline import make_pipeline, make_union
    from sklearn.preprocessing import PolynomialFeatures
    from sklearn.feature_selection import SelectFwe, f_regression

    "from sklearn.ensemble import RandomForestClassifier\n",
    "from sklearn.neighbors import KNeighborsClassifier\n",
    "from sklearn.tree import DecisionTreeClassifier\n",
    "from sklearn.naive_bayes import GaussianNB"

    "dataset = pd.read_csv('dataset.csv',sep=',',encoding=\"utf-8\")"

    "type(dataset)"

    "dataset.shape"
```

```
"RangeIndex: 270 entries, 0 to 269\n",
"Data columns (total 14 columns):\n",
" #   Column    Non-Null Count  Dtype  \n",
"---  ------    --------------  -----  \n",
" 0   age       270 non-null    int64  \n",
" 1   sex       270 non-null    int64  \n",
" 2   cp        270 non-null    int64  \n",
" 3   trestbps  270 non-null    int64  \n",
" 4   chol      270 non-null    int64  \n",
" 5   fbs       270 non-null    int64  \n",
" 6   restecg   270 non-null    int64  \n",
" 7   thalach   270 non-null    int64  \n",
" 8   exang     270 non-null    int64  \n",
" 9   oldpeak   270 non-null    float64\n",
" 10  slope     270 non-null    int64  \n",
" 11  ca        270 non-null    int64  \n",
" 12  thal      270 non-null    int64  \n",
" 13  target    270 non-null    int64  \n",
"dtypes: float64(1), int64(13)\n",
"memory usage: 29.7 KB\n"
```

```
 "Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',\n",
 "       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],\n",
      "      dtype='object')"
     ]
    },
    "execution_count": 10,
    "metadata": {},
    "output_type": "execute_result"
   }
\
      "</style>\n",
      "<table border=\"1\" class=\"dataframe\">\n",
      "  <thead>\n",
      "    <tr style=\"text-align: right;\">\n",
      "      <th></th>\n",
      "      <th>age</th>\n",
      "      <th>sex</th>\n",
      "      <th>cp</th>\n",
      "      <th>trestbps</th>\n",
      "      <th>chol</th>\n",
      "      <th>fbs</th>\n",
      "      <th>restecg</th>\n",
      "      <th>thalach</th>\n",
      "      <th>exang</th>\n",
      "      <th>oldpeak</th>\n",
    "      <th>slope</th>\n",
    "      <th>ca</th>\n",
    "      <th>thal</th>\n",
    "      <th>target</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
    "      <th>count</th>\n",
    ,

    "              age        sex         cp    trestbps       chol         fbs  \\\n",

   ],
   "source": [
    "dataset.describe()"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "9ece7ed8",
   "metadata": {},
   "source": [
    "The scale of each feature column is different and quite varied as well. While the maximum for age reaches 77, the maximum of chol (serum cholestoral) is 564."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 12,
   "id": "227e54aa",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
 "\n",
 "    .dataframe thead th {\n",
 "        text-align: right;\n",
 "    }\n",
 "</style>\n",
 "<table border=\"1\" class=\"dataframe\">\n",
 "  <thead>\n",
 "    <tr style=\"text-align: right;\">\n",
,
 "</style>\n",
 "<table border=\"1\" class=\"dataframe\">\n",
 "  <thead>\n",
 "    <tr style=\"text-align: right;\">\n",
 "      <th></th>\n",
 "      <th>age</th>\n",
 "      <th>sex</th>\n",
 "      <th>cp</th>\n",
 "      <th>trestbps</th>\n",
 "      <th>chol</th>\n",
 "      <th>fbs</th>\n",
 "      <th>restecg</th>\n",
 "      <th>thalach</th>\n",
 "      <th>exang</th>\n",
    "      <th>oldpeak</th>\n",
    "      <th>slope</th>\n",
    "      <th>ca</th>\n",
    "      <th>thal</th>\n",
    "      <th>target</th>\n",


"cell_type": "code",
"execution_count": 14,
"id": "acd9bb35",
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "age         0\n",
    "sex         0\n",
    "cp          0\n",
    "trestbps    0\n",
    "chol        0\n",
    "fbs         0\n",
    "restecg     0\n",
    "thalach     0\n",
   "exang       0\n",
   "oldpeak     0\n",
   "slope       0\n",
   "ca          0\n",
   "thal        0\n",
   "target      0\n",
   "dtype: int64"
  ]

{
 "data": {
  "text/plain": [
   "age         41\n",
   "sex          2\n",
   "cp           4\n",
   "trestbps    47\n",
   "chol       144\n",
   "fbs          2\n",
   "restecg      3\n",
   "thalach     90\n",
   "exang        2\n",
   "oldpeak     39\n",
   "slope        3\n",
     "ca          4\n",
     "thal        3\n",
     "target      2\n",
     "dtype: int64"
    ]

 "source": [
  "print('cp ',dataset['cp'].unique())\n",
  "print('fbs ',dataset['fbs'].unique())\n",
  "print('restecg ',dataset['restecg'].unique())\n",
  "print('exang ',dataset['exang'].unique())\n",
  "print('slope ',dataset['slope'].unique())\n",
  "print('ca ',dataset['ca'].unique())\n",
  "print('thal ',dataset['thal'].unique())\n"
 ]
},

   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  }
 ],
   "source": [
    "# Display age distribution based on heart disease\n",
    "sns.distplot(dataset[dataset['target'] == 1]['age'], label='Do not have heart disease')\n",
    "sns.distplot(dataset[dataset['target'] == 2]['age'], label = 'Have heart disease')\n",
    "plt.xlabel('Frequency')\n",
    "plt.ylabel('Age')\n",
    "plt.title('Age Distribution based on Heart Disease')\n",
    "plt.legend()\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "fb21e14c",
   "metadata": {},
   "source": [
    "Get min, max and average of the age of the people do not have heart disease"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "id": "c0da4bb5",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Min age of people who do not have heart disease:  29\n",
      "Max age of people who do not have heart disease:  76\n",
      "Average age of people who do not have heart disease:  52.70666666666666\n"
     ]
    }
   ],
   "source": [
    "print('Min age of people who do not have heart disease: ', min(dataset[dataset['target'] == 1]['age']))\n",
    "print('Max age of people who do not have heart disease: ', max(dataset[dataset['target'] == 1]['age']))\n",
    "print('Average age of people who do not have heart disease: ', dataset[dataset['target'] == 1]['age'].mean())"
   ]
  },
 {
  "cell_type": "markdown",
  "id": "a8e8f507",
  "metadata": {},
  "source": [
   "Get min, max and average of the age of the people have heart disease"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 27,
  "id": "388a33d3",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Min age of people who have heart disease:  35\n",
     "Max age of people who have heart disease:  77\n",
     "Average age of people who have heart disease: 56.591666666666667\n"
    ]
    }
   ],
   "source": [
    "print('Min age of people who have heart disease: ',
min(dataset[dataset['target'] == 2]['age']))\n",
    "print('Max age of people who have heart disease: ',
max(dataset[dataset['target'] == 2]['age']))\n",
    "print('Average age of people who have heart disease: ',
dataset[dataset['target'] == 2]['age'].mean())"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "327be9df",
   "metadata": {},
   "source": [
    "From the data, we can say that the heart disease infects the old and
young people, and the probability of the old people te be infected is
higher than young people.  \n",
    "\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "e3961af5",
   "metadata": {},
   "source": [
    "**Heart Disease Frequency for ages**"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "id": "a73172bb",
   "metadata": {},
   "outputs": [


     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [

"pd.crosstab(dataset.age,dataset.target).plot(kind=\"bar\",figsize=(20,6))
```

```
"import pandas as pd\n",
"import numpy as np\n",
"import matplotlib.pyplot as plt\n",
"import seaborn as sns"
   "</style>\n",
   "<table border=\"1\" class=\"dataframe\">\n",
   "  <thead>\n",
   "    <tr style=\"text-align: right;\">\n",
   "      <th></th>\n",
   "      <th>Age</th>\n",
   "      <th>Sex</th>\n",
   "      <th>Chest pain type</th>\n",
   "      <th>BP</th>\n",
   "      <th>Cholesterol</th>\n",
   "      <th>FBS over 120</th>\n",
   "      <th>EKG results</th>\n",
   "      <th>Max HR</th>\n",
   "      <th>Exercise angina</th>\n",
   "      <th>ST depression</th>\n",
   "      <th>Slope of ST</th>\n",
   "      <th>Number of vessels fluro</th>\n",
   "      <th>Thallium</th>\n",
   "      <th>Heart Disease</th>\n",
   "execution_count": 3,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "df.head()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 4,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "Age                        0\n",
      "Sex                        0\n",
      "Chest pain type            0\n",
      "BP                         0\n",
      "Cholesterol                0\n",
      "FBS over 120               0\n",
      "EKG results                0\n",
      "Max HR                     0\n",
      "Exercise angina            0\n",
      "ST depression              0\n",
      "Slope of ST                0\n",
      "Number of vessels fluro    0\n",
      "Thallium                   0\n",
      "Heart Disease              0\n",
      "dtype: int64"
     ]
    },
    "execution_count": 4,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "df.isnull().sum()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 5,
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "<class 'pandas.core.frame.DataFrame'>\n",
   "RangeIndex: 270 entries, 0 to 269\n",
   "Data columns (total 14 columns):\n",
   " #   Column                   Non-Null Count  Dtype  \n",
   "---  ------                   --------------  -----  \n",
   " 0   Age                      270 non-null    int64  \n",
   " 1   Sex                      270 non-null    int64  \n",
   " 2   Chest pain type          270 non-null    int64  \n",
   " 3   BP                       270 non-null    int64  \n",
   " 4   Cholesterol              270 non-null    int64  \n",
   " 5   FBS over 120             270 non-null    int64  \n",
   " 6   EKG results              270 non-null    int64  \n",
   " 7   Max HR                   270 non-null    int64  \n",
   " 8   Exercise angina          270 non-null    int64  \n",
   " 9   ST depression            270 non-null    float64\n",
   " 10  Slope of ST              270 non-null    int64  \n",
   " 11  Number of vessels fluro  270 non-null    int64  \n",
      " 12  Thallium                 270 non-null    int64  \n",
      " 13  Heart Disease            270 non-null    object \n",
      "dtypes: float64(1), int64(12), object(1)\n",
      "memory usage: 29.7+ KB\n",
      "None\n"
     ]
    }
   ],
   "source": [
    "print(df.info())"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 6,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "<AxesSubplot:>"
      ]
     },
   "execution_count": 6,
   "metadata": {},
   "output_type": "execute_result"
  },
 ,
{
 "cell_type": "code",
 "execution_count": 7,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "<seaborn.axisgrid.PairGrid at 0x1896d9664f0>"
    ]
   },
   "execution_count": 7,
   "metadata": {},
   "output_type": "execute_result"
  },
  {
     "text/plain": [
     "<Figure size 2340x2340 with 182 Axes>"
```

```
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "sns.pairplot(data=df)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 8,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([[<AxesSubplot:title={'center':'Age'}>,\n",
       "        <AxesSubplot:title={'center':'Sex'}>,\n",
       "        <AxesSubplot:title={'center':'Chest pain type'}>,\n",
      "       <AxesSubplot:title={'center':'BP'}>],\n",
      "      [<AxesSubplot:title={'center':'Cholesterol'}>,\n",
      "       <AxesSubplot:title={'center':'FBS over 120'}>,\n",
      "       <AxesSubplot:title={'center':'EKG results'}>,\n",
      "       <AxesSubplot:title={'center':'Max HR'}>],\n",
      "      [<AxesSubplot:title={'center':'Exercise angina'}>,\n",
      "       <AxesSubplot:title={'center':'ST depression'}>,\n",
      "       <AxesSubplot:title={'center':'Slope of ST'}>,\n",
      "       <AxesSubplot:title={'center':'Number of vessels fluro'}>],\n",
      "      [<AxesSubplot:title={'center':'Thallium'}>, <AxesSubplot:>,\n",
      "       <AxesSubplot:>, <AxesSubplot:>],\n",
      "      [<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>]],\n",
      "     dtype=object)"
     ]
    },
    "execution_count": 8,
    "metadata": {},
    "output_type": "execute_result"
   },
     "<Figure size 720x864 with 20 Axes>"
     ]
    },
    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
  "source": [
   "df.hist(figsize=(10,12), layout=(5,4))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 9,
  "metadata": {},
  "outputs": [
   {
       "<Figure size 720x720 with 18 Axes>"
     ]
    },
    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
  "source": [
   "df.plot(kind='box', subplots=True, layout=(6,3), figsize=(10,10))\n",
   "plt.show()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 10,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "<seaborn.axisgrid.FacetGrid at 0x189764cc430>"
     ]
    },
    "execution_count": 10,
    "metadata": {},
    "output_type": "execute_result"
   },
   {
    "data": {
    "execution_count": 18,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/html": [
      "<div>\n",
      "<style scoped>\n",
      "    .dataframe tbody tr th:only-of-type {\n",
      "        vertical-align: middle;\n",
      "    }\n",
      "\n",
      "    .dataframe tbody tr th {\n",
      "        vertical-align: top;\n",
      "    }\n",
      "\n",
      "    .dataframe thead th {\n",
      "        text-align: right;\n",
      "    }\n",
      "</style>\n",
      "<table border=\"1\" class=\"dataframe\">\n",
      "  <thead>\n",
    "    <tr style=\"text-align: right;\">\n",
    "      <th></th>\n",
    "      <th>Age</th>\n",
    "      <th>Sex</th>\n",
    "      <th>Chest pain type</th>\n",
    "      <th>BP</th>\n",
    "      <th>Cholesterol</th>\n",
    "      <th>FBS over 120</th>\n",
    "      <th>EKG results</th>\n",
    "      <th>Max HR</th>\n",
    "      <th>Exercise angina</th>\n",
    "      <th>ST depression</th>\n",
    "      <th>Slope of ST</th>\n",
    "      <th>Number of vessels fluro</th>\n",
    "      <th>Thallium</th>\n",
    "      <th>Heart Disease</th>\n",
    "    </tr>\n",
    "  </thead>\n",
    "  <tbody>\n",
    "    <tr>\n",
 "cell_type": "code",
"execution_count": 19,
"metadata": {},
   "outputs": [],
   "source": [
    "x=df.drop(['Heart Disease'], axis=1)\n",
    "y=df['Heart Disease']"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 20,
   "metadata": {},
   "outputs": [],
   "source": [
    "x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3, random_state=40)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 21,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
    "output_type": "stream",
    "text": [
     "x_train- 2457\n",
     "x_test- 1053\n",
     "y_train- 189\n",
     "x_test- 1053\n"
    ]
   }
  ],
  "source": [
   "print('x_train-', x_train.size)\n",
   "print('x_test-', x_test.size)\n",
   "print('y_train-', y_train.size)\n",
   "print('x_test-', x_test.size)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 22,
  "metadata": {},
  "outputs": [
   {
     "text": [
     "Testing Accuracy: 0.9382716049382716\n"
    ]
   }
  ],
  "source": [
   "TP=cm[0][0]\n",
   "TN=cm[1][1]\n",
   "FN=cm[1][0]\n",
   "FP=cm[0][1]\n",
   "print('Testing Accuracy:', (TP+TN+FN)/(TP+TN+FN+FP))"
  ]
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-21889-1659795107