

# CONTENTS

S.NO.	TITLE
1	Introduction 1.1 Overview 1.2 Purpose
2	Problem Definition & Design Thinking 2.1 Empathy Map 2.2 Ideation & Brainstorming Map
3	Result
4	Advantages & Disadvantages
5	Applications
6	Conclusion
7	Future Scope
8	Appendix

# **1.INTRODUCTION**

## **1.1 Overview**

Chronic Kidney Disease (CKD) is a serious health condition that affects millions of people worldwide. It is a long-term condition that occurs when the kidneys are damaged and cannot function properly, leading to a buildup of waste and fluids in the body.

Early prediction for CKD detection is crucial to prevent further kidney damage and to improve patient outcomes. Early detection can also help healthcare providers develop appropriate treatment plans and prevent the progression of the disease.

There are several methods for early prediction of CKD, including blood and urine tests, imaging tests, and genetic testing. Blood and urine tests can measure the levels of certain substances in the body that can indicate kidney function. Imaging tests, such as ultrasounds or CT scans, can also detect kidney damage. Genetic testing can identify genetic mutations that may increase the risk of CKD.

Machine learning algorithms have also been used to predict the risk of CKD development in patients. These algorithms use patient data, such as age, sex, medical history, and lab results, to develop predictive models that can identify patients at high risk of developing CKD.

Overall, early prediction for CKD detection is critical for early intervention and management of the disease. With early detection, healthcare providers can improve patient outcomes, prevent the progression of the disease, and reduce healthcare costs.

## 1.2 Purpose

The purpose of early prediction for Chronic Kidney Disease (CKD) detection is to identify individuals who are at risk of developing CKD or who have early-stage CKD before they experience significant kidney damage. Early detection of CKD can help healthcare providers develop appropriate treatment plans and prevent the progression of the disease, thereby reducing the risk of kidney failure, cardiovascular disease, and other complications associated with CKD.

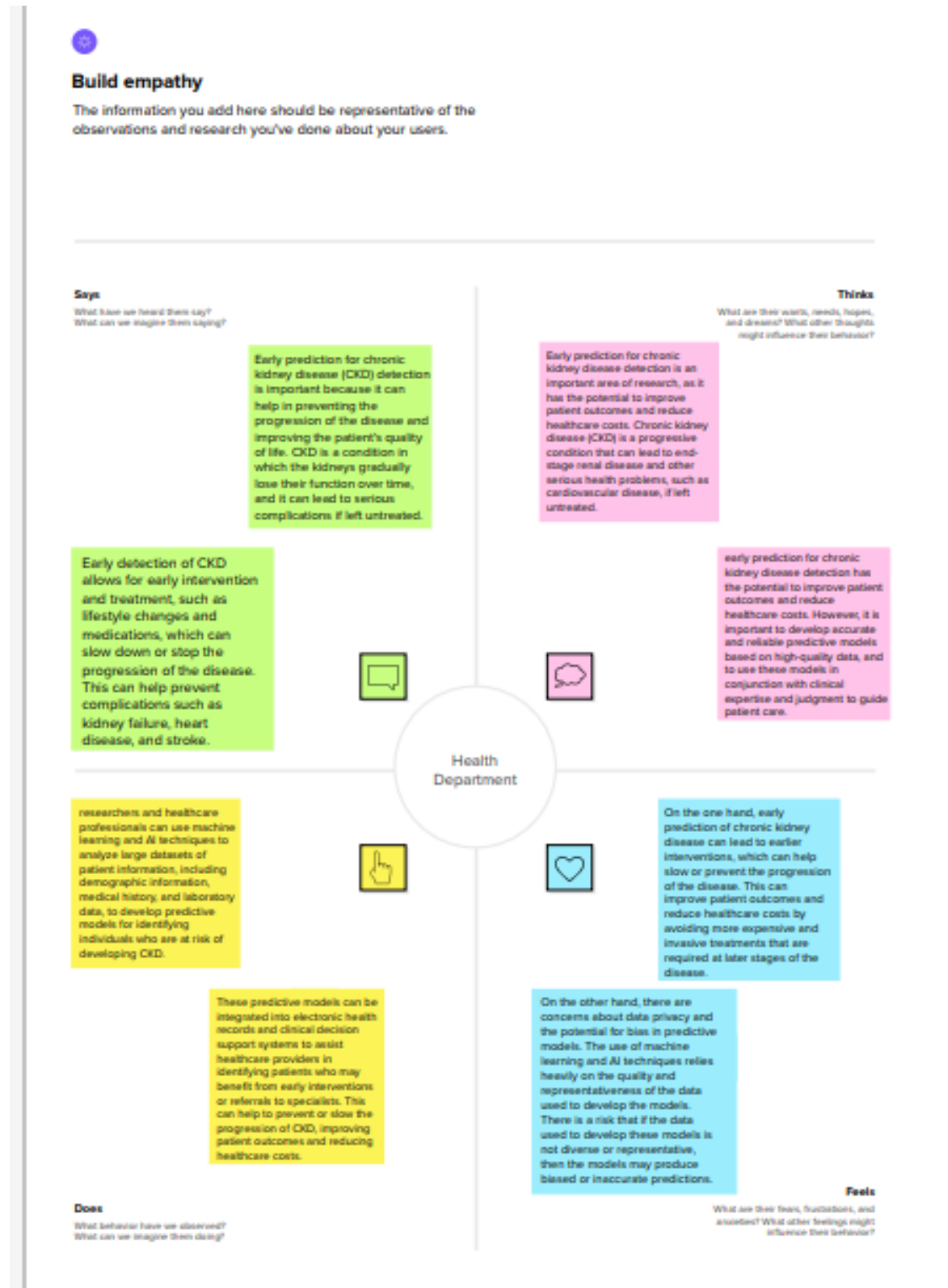
Early prediction of CKD can also help healthcare providers identify and manage underlying conditions that may contribute to the development or progression of CKD, such as high blood pressure, diabetes, and autoimmune disorders. By managing these conditions early, healthcare providers can help prevent further damage to the kidneys and improve patient outcomes.

Moreover, early prediction of CKD can lead to cost savings for healthcare systems by preventing the need for costly treatments such as dialysis and kidney transplantation. Early intervention and management of CKD can also help improve the quality of life for patients by reducing symptoms and complications associated with CKD.

In summary, the purpose of early prediction for CKD detection is to identify individuals at risk of developing CKD or who have early-stage CKD, allowing for early intervention and management, thereby reducing the risk of complications, improving patient outcomes, and reducing healthcare costs.

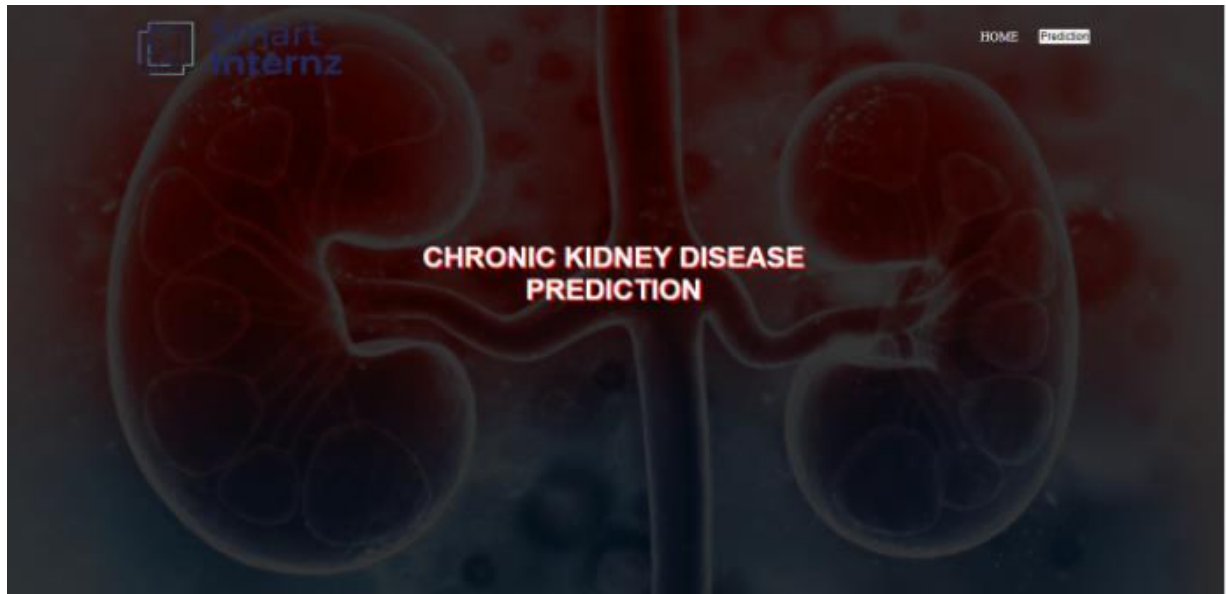
## 2. Problem Definition & Design Thinking

### 2.1 Empathy map



[illegible]

### 3. Result



The screenshot shows a prediction form titled 'Chronic Kidney Disease' in a white, cursive font, with the subtitle 'A Machine Learning Web App Built with Flask' in a smaller, white, sans-serif font below it. The form is set against a solid purple background. It contains seven input fields arranged vertically: two text inputs for 'Enter your blood\_urea' and 'Enter your blood glucose random', and five dropdown menus with labels 'Select anemia or not', 'Select coronary artery disease or not', 'Select pus\_cell or not', 'Select red\_blood\_cell\_level', and 'Select diabetes mellitus or not'. Each dropdown menu has a small downward arrow on its right side. Below these fields is a single, rounded rectangular button labeled 'Predict' in a white, sans-serif font.

## Chronic Kidney Disease

A Machine Learning Web App Built with Flask

**Prediction: Oops! You have Chronic Kidney Disease.**



## Chronic Kidney Disease

A Machine Learning Web App Built with Flask

1
1
NO
NO
NO
NO
NO
NO

Predict

25°C  
Sunny



ENG  
IN

# Chronic Kidney Disease

A Machine Learning Video App, Built with Flask

Prediction: **Great! You DON'T have Chronic Kidney Disease**



25°C  
Sunny



8:02  
18



## **4 Advantages & Disadvantages**

### **Advantages:**

Early prediction of chronic kidney disease (CKD) can help identify individuals who are at high risk of developing kidney disease, enabling them to take steps to prevent or slow the progression of the disease.

Early detection can lead to timely treatment and management of the disease, reducing the risk of complications such as kidney failure and cardiovascular disease.

Early prediction of CKD can help healthcare providers to plan appropriate interventions and allocate resources effectively.

Early prediction can help reduce healthcare costs associated with CKD treatment by preventing or delaying the onset of the disease.

### **Disadvantages:**

Early prediction models may not be accurate enough to accurately predict CKD in all individuals, leading to false positives and false negatives.

Early prediction may lead to unnecessary testing and procedures, resulting in increased healthcare costs and patient anxiety.

Early prediction may result in overdiagnosis, leading to treatment of individuals who may not have developed CKD without intervention.

Early prediction may not be feasible in all healthcare settings due to the availability of resources and technology required to implement predictive models.

## 5 Applications

There are several applications of early prediction of chronic kidney disease (CKD), including:

**Preventative healthcare:** Early prediction of CKD can identify individuals who are at high risk of developing the disease. By identifying these individuals early, healthcare providers can offer preventative measures such as lifestyle modifications, blood pressure control, and medication management to slow or prevent the progression of the disease.

**Patient management:** Early prediction of CKD can help healthcare providers to monitor patients more closely and provide targeted interventions to prevent or delay the onset of complications associated with the disease.

**Resource allocation:** Early prediction of CKD can help healthcare providers to allocate resources more effectively by identifying patients who require more intensive monitoring and management.

**Clinical research:** Early prediction of CKD can provide valuable insights into the disease process, enabling researchers to develop new treatments and interventions to prevent or delay the progression of the disease.

**Population health management:** Early prediction of CKD can be used to identify populations at high risk of developing the disease, enabling public health officials to develop targeted interventions to prevent or delay the onset of CKD in these populations.

## **6. Conclusion**

Early prediction of chronic kidney disease (CKD) is a valuable tool for identifying individuals who are at high risk of developing the disease. By identifying these individuals early, healthcare providers can offer preventative measures and targeted interventions to prevent or slow the progression of the disease, leading to better health outcomes and reduced healthcare costs. However, there are also potential disadvantages to early prediction, such as false positives and false negatives, unnecessary testing and procedures, and overdiagnosis. Therefore, healthcare providers should carefully consider the benefits and limitations of early prediction models and use them judiciously to improve patient outcomes and population health.

## **7. Future Scope**

The Future scope for early prediction of chronic kidney disease (CKD) is promising, as there are several emerging technologies and approaches that have the potential to improve the accuracy and accessibility of early prediction models. Some of the potential future developments in this field include:

**Use of machine learning:** Machine learning algorithms have shown promise in predicting CKD, and future research could explore ways to improve the accuracy of these models by incorporating additional data sources or developing more sophisticated algorithms.

**Integration of genetic data:** Advances in genomics have enabled the identification of genetic markers associated with CKD, and future research could explore ways to integrate this information into early prediction models.

**Wearable devices:** Wearable devices such as smartwatches and fitness trackers could be used to monitor key health metrics such as blood pressure and heart rate, providing valuable data for early prediction models.

**Telemedicine:** Telemedicine could be used to expand access to early prediction models in underserved communities, enabling healthcare providers to identify individuals at high risk of developing CKD and provide targeted interventions to prevent or slow the progression of the disease.

**Personalized medicine:** Personalized medicine approaches could be used to tailor interventions based on an individual's risk profile and disease progression, improving the effectiveness of early prediction models and reducing the risk of adverse outcomes.

Overall, the future of early prediction of CKD is likely to involve a combination of technological advancements and personalized approaches, enabling healthcare providers to identify and manage CKD more effectively and improve patient outcomes

## 8. Appendix

Home Page - Select or create a notebook | Untitled2 - Jupyter Notebook | Student

localhost8888/notebooks/Untitled2.ipynb?kernel\_name=python3

jupyter Untitled2 Last Checkpoint: 26 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Python 3 (pykernel)

```
In [3]: import pandas as pd
import numpy as np
from collections import Counter as c
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
import pickle
```

```
In [4]: data=pd.read_csv("C:\\Users\\ramesh kanna\\OneDrive\\Documents\\Data Set\\heart.csv")
data.head()
```

```
Out[4]:
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

```
In [5]: data.columns
```

```
Out[5]: Index(['Age', 'Sex', 'ChestPainType', 'RestingBP', 'Cholesterol', 'FastingBS',
      'RestingECG', 'MaxHR', 'ExerciseAngina', 'Oldpeak', 'ST_Slope',
      'HeartDisease'],
      dtype='object')
```

```
In [6]: data.columns=['age','gender','chestpaintype','restingBP','cholesterol','fastingBS','restingECG','maxHR','exerciseAngina','oldpeak']
data.columns
```

```
Out[6]: Index(['age', 'gender', 'chestpaintype', 'restingBP', 'cholesterol',
      'fastingBS', 'restingECG', 'maxHR', 'exerciseAngina', 'oldpeak',
      'ST_Slope', 'HeartDisease'],
      dtype='object')
```

```
In [7]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype
---  ---
 0   age                 918 non-null   int64
 1   gender              918 non-null   object
 2   chestpaintype       918 non-null   object
 3   restingBP           918 non-null   int64
 4   cholesterol         918 non-null   int64
 5   fastingBS           918 non-null   int64
 6   restingECG          918 non-null   object
 7   maxHR               918 non-null   int64
 8   exerciseAngina      918 non-null   object
 9   oldpeak             918 non-null   float64
10  ST_slope            918 non-null   object
11  HeartDisease        918 non-null   int64
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

```
In [8]: data.isnull().any()
```

```
Out[8]: age                False
gender                False
chestpaintype         False
restingBP              False
cholesterol            False
fastingBS              False
restingECG             False
maxHR                  False
exerciseAngina         False
oldpeak                False
ST_slope              False
```

```
Home Page - Select or create a ... x Untitled2 - Jupyter Notebook x - Student x +
localhost:8888/notebooks/Untitled2.ipynb?kernel_name=python3
Apps Gmail YouTube Maps chatgpt Debian.org Naanmudhalvan - S... Latest News Help

jupyter Untitled2 Last Checkpoint: 31 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (pykernel) O

In [9]: catcols=set(data.dtypes[data.dtypes=="O"].index.values)
print(catcols)
{'ST_slope', 'restingECG', 'gender', 'exerciseAngina', 'chestpaintype'}

In [10]: for i in catcols:
print("Columns :",i)
print(c(data[i]))
print("%*s128" % i)

Columns : ST_slope
Counter({'Flat': 460, 'Up': 395, 'Down': 63})
.....

Columns : restingECG
Counter({'Normal': 552, 'LVH': 188, 'ST': 178})
.....

Columns : gender
Counter({'M': 725, 'F': 193})
.....

Columns : exerciseAngina
Counter({'N': 547, 'Y': 371})
.....

Columns : chestpaintype
Counter({'ASY': 496, 'NAP': 203, 'ATA': 173, 'TA': 46})
.....

In [11]: catcols=['ST_slope','exerciseAngina','restingECG','gender','chestpaintype']

In [12]: from sklearn.preprocessing import LabelEncoder
for i in catcols:
print("LABEL ENCODING OF:",i)
LE1 = LabelEncoder()
print(c(data[i]))
data[i] = LE1.fit_transform(data[i])
print(c(data[i]))
```

```
Home Page - Select or create a ... x Untitled2 - Jupyter Notebook x - Student x +
localhost:8888/notebooks/Untitled2.ipynb?kernel_name=python3
Apps Gmail YouTube Maps chatgpt Debian.org Naanmudhalvan - S... Latest News Help

jupyter Untitled2 Last Checkpoint: 32 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (pykernel) O

LABEL ENCODING OF: ST_slope
Counter({'Flat': 460, 'Up': 395, 'Down': 63})
Counter({1: 460, 2: 395, 0: 63})
.....
LABEL ENCODING OF: exerciseAngina
Counter({'N': 547, 'Y': 371})
Counter({0: 547, 1: 371})
.....
LABEL ENCODING OF: restingECG
Counter({'Normal': 552, 'LVH': 188, 'ST': 178})
Counter({1: 552, 0: 188, 2: 178})
.....
LABEL ENCODING OF: gender
Counter({'M': 725, 'F': 193})
Counter({0: 725, 1: 193})
.....
LABEL ENCODING OF: chestpaintype
Counter({'ASY': 496, 'NAP': 203, 'ATA': 173, 'TA': 46})
Counter({0: 496, 2: 203, 1: 173, 3: 46})
.....

In [13]: contcols=set(data.dtypes[data.dtypes!="O"].index.values)
print(contcols)
{'heartdisease', 'age', 'restingECG', 'oldpeak', 'ST_slope', 'maxHR', 'restingBP', 'cholesterol', 'gender', 'exerciseAngina', 'chestpaintype', 'fastingBS'}

In [14]: contcols.remove('restingBP')
contcols.remove('cholesterol')
contcols.remove('fastingBS')
print(contcols)
{'heartdisease', 'age', 'restingECG', 'oldpeak', 'ST_slope', 'maxHR', 'gender', 'exerciseAngina', 'chestpaintype'}

In [15]: contcols.add('restingBP')
contcols.add('cholesterol')
contcols.add('fastingBS')
print(contcols)
{'heartdisease', 'age', 'restingECG', 'oldpeak', 'ST_slope', 'maxHR', 'restingBP', 'cholesterol', 'fastingBS', 'gender', 'exerc
```

Home Page - Select or create a notebook x Untitled2 - Jupyter Notebook x Student x +

localhost:8888/notebooks/Untitled2.ipynb?kernel\_name=python3

jupyter Untitled2 Last Checkpoint: 32 minutes ago (autosaved) Python 3 (pykernel)

```

In [15]: contcols.add('restingBP')
         contcols.add('cholesterol')
         contcols.add('fastingBS')
         print(contcols)

{'heartdisease', 'age', 'restingECG', 'oldpeak', 'ST_slope', 'maxHR', 'restingBP', 'cholesterol', 'fastingBS', 'gender', 'exerciseAngina', 'chestpainType'}

In [16]: data['exerciseAngina'] = data.exerciseAngina.replace('\to', 'N')
         c(data['exerciseAngina'])

Out[16]: Counter({0: 547, 1: 371})

In [17]: data.describe()

Out[17]:
```

	age	gender	chestpainType	restingBP	cholesterol	fastingBS	restingECG	maxHR	exerciseAngina	oldpeak	ST_slope	heartdisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510803	0.789780	0.781048	132.306514	198.709594	0.233115	0.989107	136.820288	0.404139	0.887364	1.391658	0.553377
std	9.432617	0.407701	0.958519	18.514154	109.384145	0.423048	0.631871	25.480334	0.480992	1.068570	0.807056	0.497414
min	28.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	80.000000	0.000000	-2.800000	0.000000	0.000000
25%	47.000000	1.000000	0.000000	120.000000	173.250000	0.000000	1.000000	120.000000	0.000000	0.000000	1.000000	0.000000
50%	54.000000	1.000000	0.000000	130.000000	223.000000	0.000000	1.000000	138.000000	0.000000	0.800000	1.000000	1.000000
75%	60.000000	1.000000	2.000000	140.000000	287.000000	0.000000	1.000000	186.000000	1.000000	1.500000	2.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	603.000000	1.000000	2.000000	202.000000	1.000000	8.200000	2.000000	1.000000

```

In [18]: sns.distplot(data.age)

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: 'distplot' is a deprecated function and
d will be removed in a future version. Please adapt your code to use either 'displot' (a figure-level function with similar file
xibility) or 'histplot' (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)

Out[18]: <AxesSubplot: xlabel='age', ylabel='Density'>

```

Home Page - Select or create a notebook x Untitled2 - Jupyter Notebook x Student x +

localhost:8888/notebooks/Untitled2.ipynb?kernel\_name=python3

jupyter Untitled2 Last Checkpoint: 33 minutes ago (autosaved) Python 3 (pykernel)

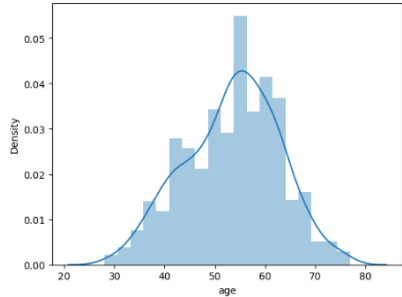
```

In [18]: sns.distplot(data.age)

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: 'distplot' is a deprecated function and
d will be removed in a future version. Please adapt your code to use either 'displot' (a figure-level function with similar file
xibility) or 'histplot' (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)

Out[18]: <AxesSubplot: xlabel='age', ylabel='Density'>

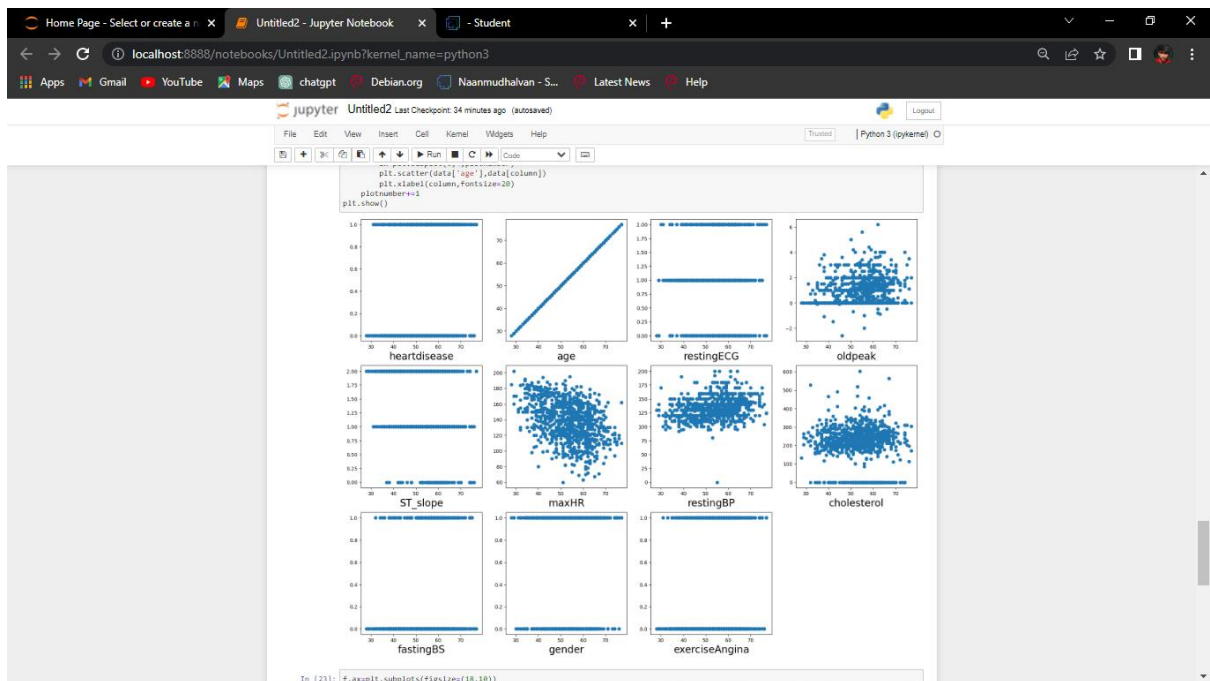
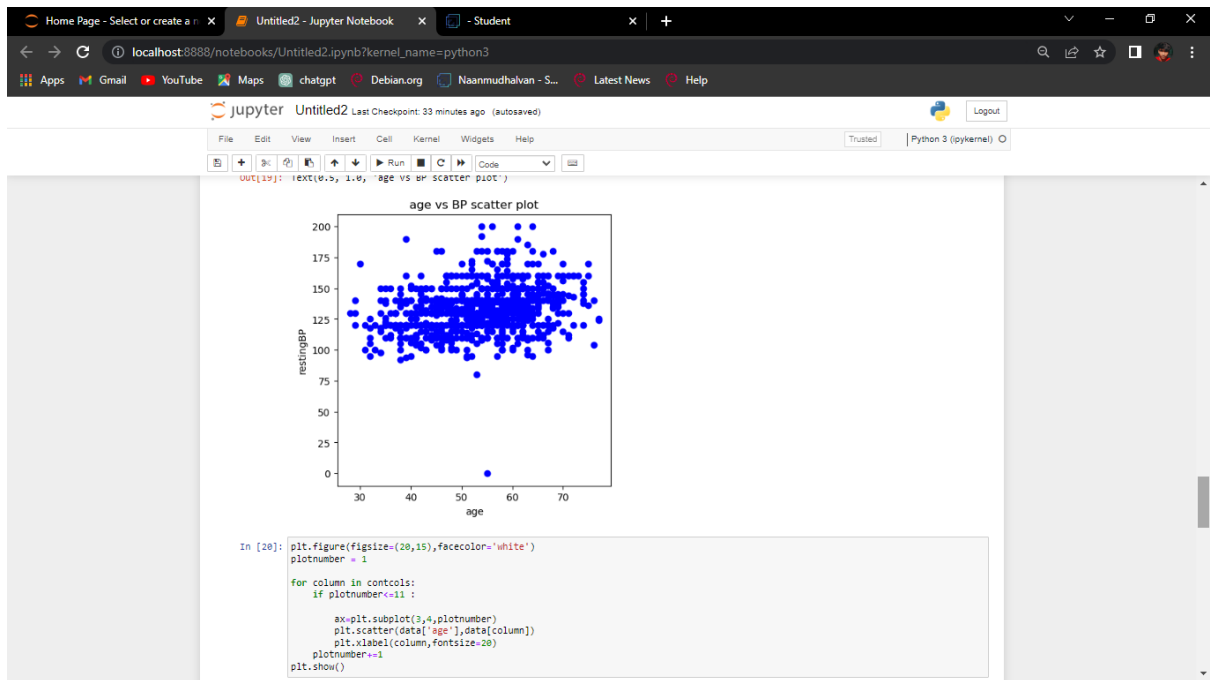
```



```

In [19]: import matplotlib.pyplot as plt
         fig=plt.figure(figsize=(5,5))
         plt.scatter(data['age'],data['restingBP'],color='blue')
         plt.xlabel('age')
         plt.ylabel('restingBP')
         plt.title("age vs BP scatter plot")

```





[illegible]