**User Acceptance Testing (UAT)**

| | |
|---|---|
| Date | 18 November 2022 |
| Team ID | PNT2022TMID21506 |
| Project Name | Visualizing and Predicting Heart Diseases with an Interactive Dashboard |
| Maximum Marks | 10 Marks |

**Purpose of Document**

The purpose of this document is to briefly explain the test coverage and open issues of the (Visualizing and Predicting Heart Diseases) project at the time of the release to User Acceptance Testing (UAT).

**1.Using ML model predicting whether the person having heart disease or not**

Total Number of Test cases tested: 15

Total Number of Test cases passed:12

Total Number of Test cases failed:3

## Using ML model predicting whether the person having heart disease or not

```
new_data=pd.DataFrame({
    'Age':70,
    'Sex':1,
    'Chest pain type':4,
    'BP':130,
    'Cholesterol':322,
    'FBS over 120':0,
    'EKG results':2,
    'Max HR':109,
    'Exercise angina':0,
    'ST depression':2.4,
    'Slope of ST':2,
    'Number of vessels fluro':3,
    'Thallium':3,
},index=[0])
```

```
new_data
```

| | Age | Sex | Chest pain type | BP | Cholesterol | FBS over 120 | EKG results | Max HR | Exercise angina | ST depression | Slope of ST | Number of vessels fluro | Thallium |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 70 | 1 | 4 | 130 | 322 | 0 | 2 | 109 | 0 | 2.4 | 2 | 3 | 3 |

```
p =rf.predict(new_data)
if p[0]==0:
  print("No disease")
else:
  print("disease")
```

```
disease
```

```
new_data1=pd.DataFrame({
    'Age':57,
    'Sex':1,
    'Chest pain type':4,
    'BP':140,
    'Cholesterol':192,
    'FBS over 120':0,
    'EKG results':0,
    'Max HR':148,
    'Exercise angina':0,
    'ST depression':0.4,
    'Slope of ST':2,
    'Number of vessels fluro':0,
    'Thallium':6,
},index=[0])
```

```
[44] new_data1
```

| | Age | Sex | Chest pain type | BP | Cholesterol | FBS over 120 | EKG results | Max HR | Exercise angina | ST depression | Slope of ST | Number of vessels fluro | Thallium |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 67 | 0 | 3 | 115 | 564 | 0 | 2 | 160 | 0 | 1.6 | 2 | 0 | 7 |

```
p1 =rf.predict(new_data1)
if p1[0]==0:
  print("disease")
else:
  print("No Disease")
```

```
No Disease
```

## Machine Learning Model Performance Testing

| | Model | Accuracy | Confusion Matrix |
|---|---|---|---|
| 0 | Logistic Regression | 0.870370 | [[30, 4], [3, 17]] |
| 1 | Support Vector Machine | 0.740741 | [[26, 8], [6, 14]] |
| 2 | Random Forest Classifier | 0.777778 | [[28, 6], [6, 14]] |
| 3 | K-Nearest Neighbour | 0.740741 | [[26, 8], [6, 14]] |
| 4 | Gaussian NB | 0.851852 | [[30, 4], [4, 16]] |
| 5 | Linear Discrimant Analysis | 0.888889 | [[32, 2], [4, 16]] |
| 6 | Adaboost | 0.740741 | [[28, 6], [8, 12]] |

## Confusing Matrix

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.

Let's assume we have a binary classification problem. We have some samples belonging to two classes : YES or NO. Also, we have our own classifier which predicts a class for a given input sample. On testing our model on 165 samples ,we get the following result.

| n=165 | **Predicted: NO** | **Predicted: YES** |
|---|---|---|
| **Actual: NO** | 50 | 10 |
| **Actual: YES** | 5 | 100 |

The are 4 important terms:

- **True Positives** : The cases in which we predicted YES and the actual output was also YES.
- **True Negatives** : The cases in which we predicted NO and the actual output was NO.
- **False Positives** : The cases in which we predicted YES and the actual output was NO.
- **False Negatives** : The cases in which we predicted NO and the actual output was YES.

1. Logistic Regression

```
[15] scLR=model.score(X_test,Y_test)
     scLR
```

```
0.8703703703703703
```

```
[17] ##from explainerdashboard.explainers import precision_score


     from sklearn.metrics import confusion_matrix,f1_score
     sclr1=confusion_matrix(Y_test,model.predict(X_test))

     print(sclr1)
```

```
[[25  4]
 [ 3 22]]
```

2. Support Vector Machine

```
[19] model1=SVC()
     model1.fit(X_train,Y_train)
     sc=model1.score(X_test,Y_test)

     sc
```

```
0.6851851851851852
```

```
[20] sclr2=confusion_matrix(Y_test,model1.predict(X_test))
     sclr2
```

```
array([[25,  4],
       [13, 12]])
```

3. Random Forest Classifier

```
[21] model2=RandomForestClassifier()
     model2.fit(X_train,Y_train)
     scrfc=model2.score(X_test,Y_test)
     scrfc
```

```
0.8888888888888888
```

```
[22] sclr3=confusion_matrix(Y_test,model2.predict(X_test))
     sclr3
```

```
array([[26,  3],
       [ 3, 22]])
```

4. K - Nearest Neighbour

```
23] model3=KNeighborsClassifier()
    model3.fit(X_train,Y_train)
    sckn=model3.score(X_test,Y_test)
    sckn
```

```
0.6111111111111112
```

```
24] sclr4=confusion_matrix(Y_test,model3.predict(X_test))
    sclr4
```

```
array([[18, 11],
       [10, 15]])
```

5. Gaussian NB

```
[25] model4=GaussianNB()
     model4.fit(X_train,Y_train)
     scnb=model4.score(X_test,Y_test)
     scnb

     0.8518518518518519
```

```
[26] sclr5=confusion_matrix(Y_test,model4.predict(X_test))
     sclr5

     array([[25,  4],
            [ 4, 21]])
```

6. Linear Discriminant Analysis

```
[27] model5=LinearDiscriminantAnalysis()
     model5.fit(X_train,Y_train)
     sclda=model5.score(X_test,Y_test)
     sclda

     0.8888888888888888
```

```
[28] sclr6=confusion_matrix(Y_test,model5.predict(X_test))
     sclr6

     array([[26,  3],
            [ 3, 22]])
```

7. AdaBoost Classifier

```
29] model6=AdaBoostClassifier()
    model6.fit(X_train,Y_train)
    scabc=model6.score(X_test,Y_test)
    scabc

    0.8148148148148148
```

```
30] sclr7=confusion_matrix(Y_test,model6.predict(X_test))
    sclr7

    array([[23,  6],
           [ 4, 21]])
```

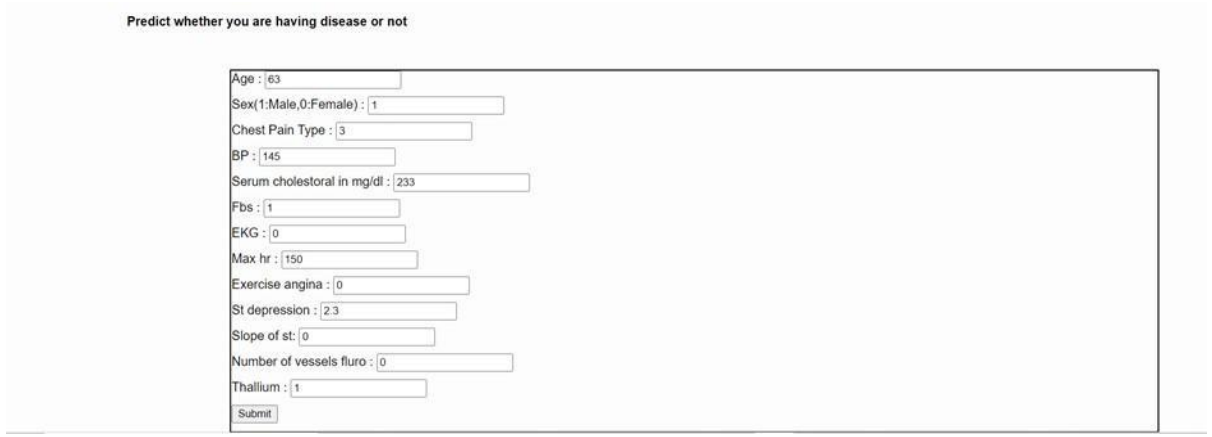
**Using ML Deployed model in Cloud predicting whether the person having heart disease or not**

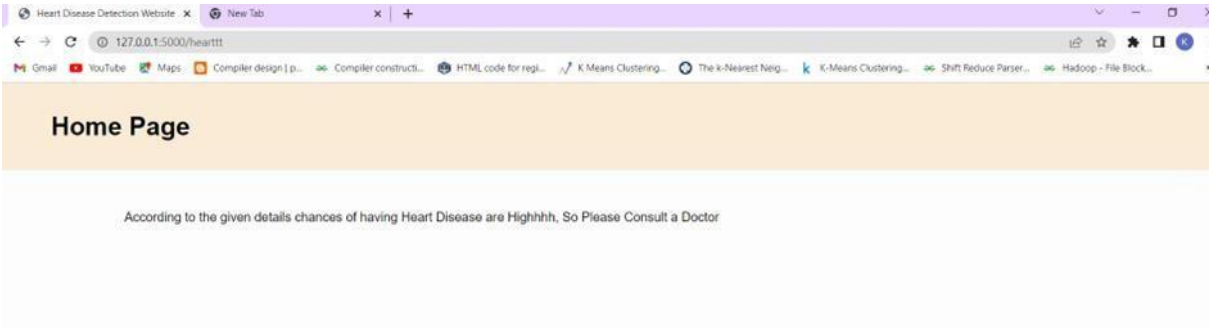
Total Number of Test cases tested: 10

Total Number of Test cases passed:10

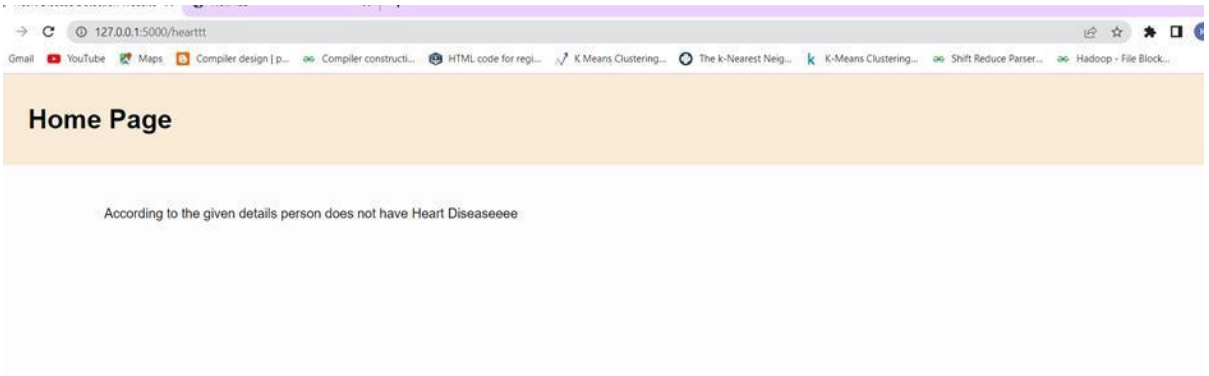
Total Number of Test cases failed:0

**Using the form data we can check whether the individual has heart disease or not Individual having disease:**

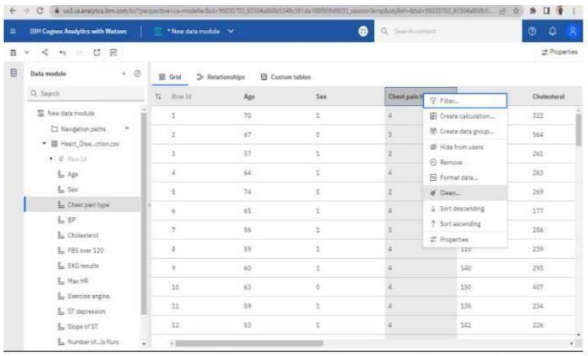


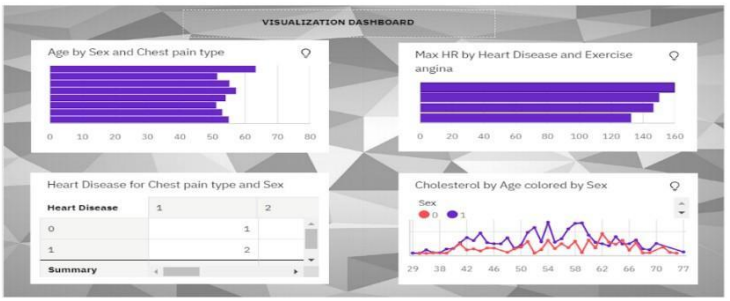




**Individual does not have heart disease:**




## Dashboard Performance Testing:

| S.No. | Parameter | Screenshot / Values |
|---|---|---|
| 1. | Dashboard designs | No of Visualizations / Graphs – 5 dashboard tabs with 6 visualizations in each dashboard   |
| 2. | Data Response | It hides certain aspects of the visualization if the size is limited, to maximize the space that is available to display data.<br>• Its Create with relationship with another explorations<br>• There was another data exploration with various continuous values ,those values was grouped as common. |
| 3. | Dataset collection and Modification | As per project ideation the dataset are collected from smartwatches by the and work the dataset through cleaning process. |
| | |  |
| 4. | Utilization of Data Filters | IN Cognos Dashboard utilization of the filtration to be filtered of all explorations of the dashboard |

| | | |
|---|---|---|
| 5. | Effective User Story | No of Scene Added – 15 stories with 2-3 visualizations in each story<br>  |
| 6. | Dashboard Deployment | The Dashboard finally get deployed through embedding dashboard with HTML |


The dashboard gets view when the user clicks the link