In [2]:

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

In [3]:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [4]:

```
data = pd.read_csv('Diabetes.csv')
data
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFu
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

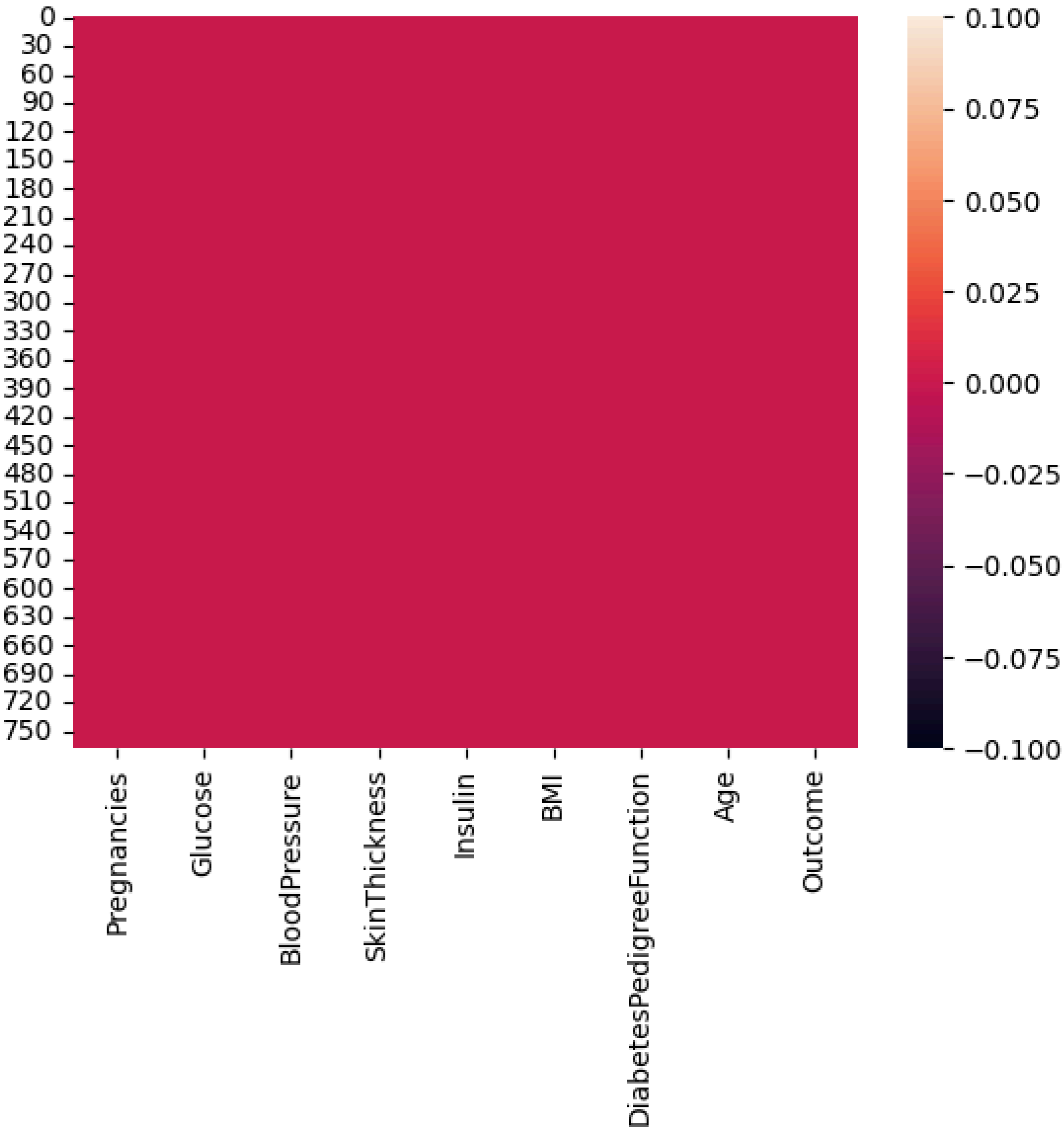
768 rows × 9 columns

In [5]:

```
sns.heatmap(data.isnull())
```

Out[5]:

```
<Axes: >
```



In [6]:

```
correlation =data.corr()
print(correlation)
```

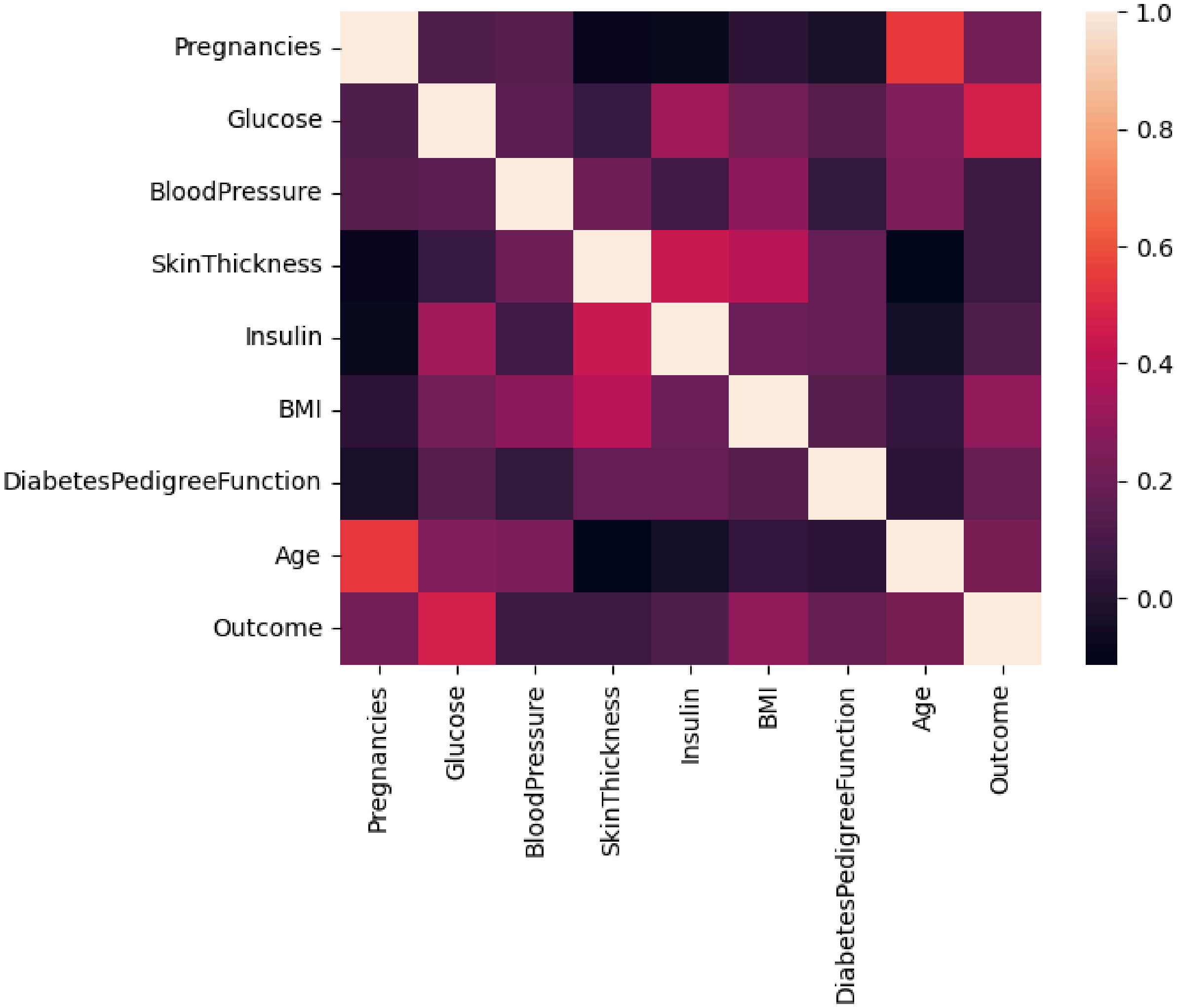
```
                          Pregnancies   Glucose  BloodPressure  SkinThickness  \
Pregnancies                  1.000000  0.129459       0.141282      -0.081672   
Glucose                      0.129459  1.000000       0.152590       0.057328   
BloodPressure                0.141282  0.152590       1.000000       0.207371   
SkinThickness               -0.081672  0.057328       0.207371       1.000000   
Insulin                     -0.073535  0.331357       0.088933       0.436783   
BMI                          0.017683  0.221071       0.281805       0.392573   
DiabetesPedigreeFunction    -0.033523  0.137337       0.041265       0.183928   
Age                          0.544341  0.263514       0.239528      -0.113970   
Outcome                      0.221898  0.466581       0.065068       0.074752   

                            Insulin       BMI  DiabetesPedigreeFunction  \
Pregnancies               -0.073535  0.017683                 -0.033523   
Glucose                    0.331357  0.221071                  0.137337   
BloodPressure              0.088933  0.281805                  0.041265   
SkinThickness              0.436783  0.392573                  0.183928   
Insulin                    1.000000  0.197859                  0.185071   
BMI                        0.197859  1.000000                  0.140647   
DiabetesPedigreeFunction   0.185071  0.140647                  1.000000   
Age                       -0.042163  0.036242                  0.033561   
Outcome                    0.130548  0.292695                  0.173844   

                                Age   Outcome  
Pregnancies                0.544341  0.221898  
Glucose                    0.263514  0.466581  
BloodPressure              0.239528  0.065068  
SkinThickness             -0.113970  0.074752  
Insulin                   -0.042163  0.130548  
BMI                        0.036242  0.292695  
DiabetesPedigreeFunction   0.033561  0.173844  
Age                        1.000000  0.238356  
Outcome                    0.238356  1.000000  
```

In [7]:

```
sns.heatmap(correlation)
```

Out[7]:

```
<Axes: >
```



In [8]:

```
X =data.drop("Outcome",axis=1)
Y =data['Outcome']
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)
```

In [9]:

```
model=LogisticRegression()
model.fit(X_train,Y_train)
```

```
C:\Users\Gupta's\AppData\Roaming\Python\Python312\site-packages\sklearn\linear_model
\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[9]:

```
▾ LogisticRegression
LogisticRegression()
```

In [10]:

```
prediction = model.predict(X_test)
```

In [11]:

```
print(prediction)
```

```
[0 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 1 0 0 1 1 1 0 0 0 1
 0 0 0 1 0 0 0 1 0 0 1 0 1 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1 0 1 0 1 1 1 0 0 1
 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0 1 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 1 1 0 1 0 1 0 0 0 0 0 0 1
 0 0 0 1 0 0]
```

In [12]:

```
accuracy = accuracy_score(prediction,Y_test)
```

In [13]:

```
print(accuracy)
```

```
0.7142857142857143
```

In [ ]: