In [1]:
```
import pandas as pd
import seaborn as sns
```

In [6]:
```
data = pd.read_csv(r'C:\Users\deepa\Downloads\diabetes.csv')
data.head
data.info
```

Out[6]:
```
<bound method DataFrame.info of      Pregnancies  Glucose  BloodPressure  SkinThickne
ss  Insulin   BMI  \
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

     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
1                       0.351   31        0
2                       0.672   32        1
3                       0.167   21        0
4                       2.288   33        1
..                        ...  ...      ...
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[768 rows x 9 columns]>
```

In [7]:
```
data.isnull().sum()
```

Out[7]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [8]:
```
data
```

Out[8]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Ag
0	6	148	72	35	0	33.6	0.627	5
1	1	85	66	29	0	26.6	0.351	3
2	8	183	64	0	0	23.3	0.672	3
3	1	89	66	23	94	28.1	0.167	2
4	0	137	40	35	168	43.1	2.288	3
...	...	...	...	...	...	...	...	.
763	10	101	76	48	180	32.9	0.171	6
764	2	122	70	27	0	36.8	0.340	2
765	5	121	72	23	112	26.2	0.245	3
766	1	126	60	0	0	30.1	0.349	4
767	1	93	70	31	0	30.4	0.315	2

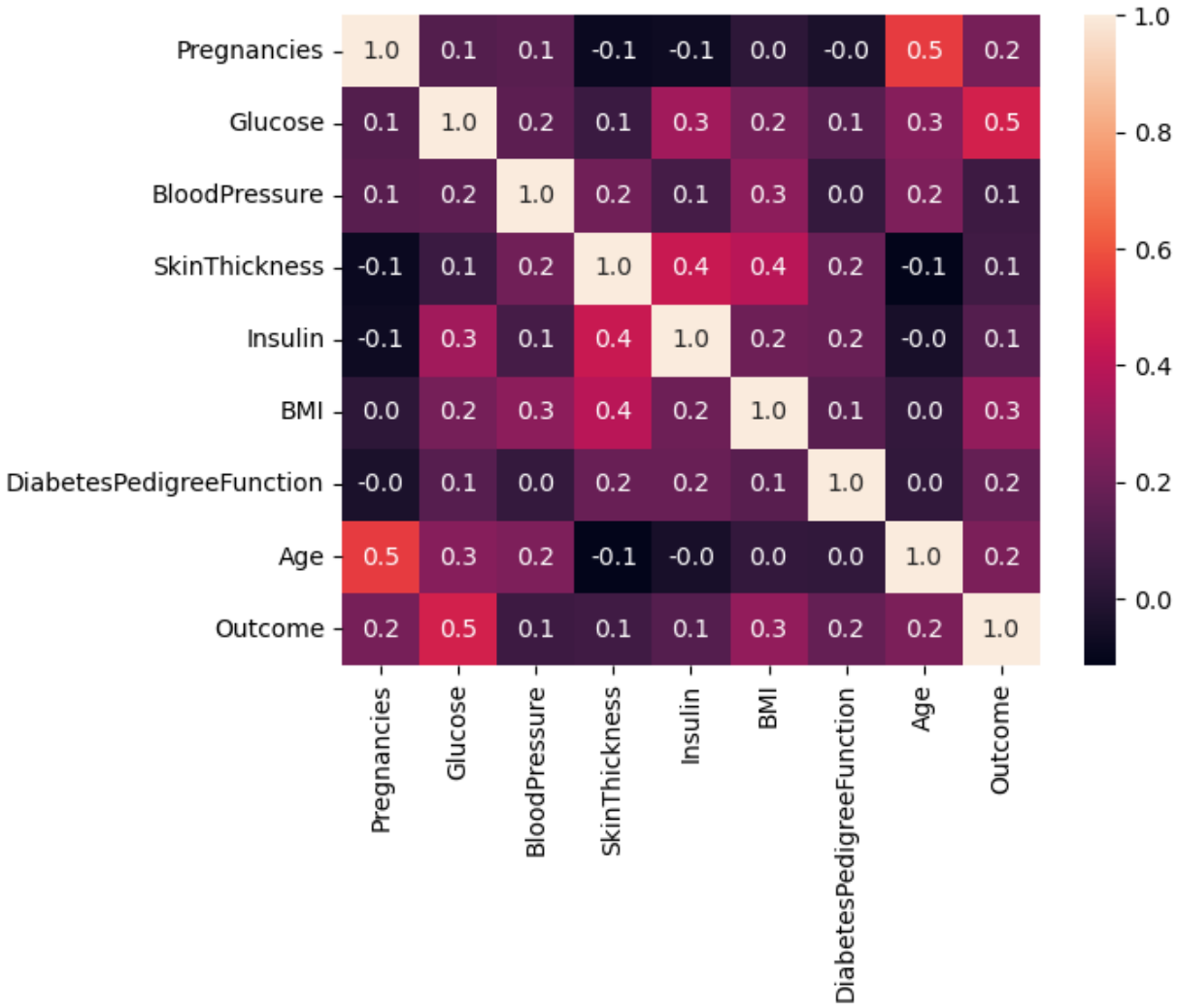
768 rows × 9 columns

In [9]:
```
data['Outcome'].unique()
```

Out[9]:
```
array([1, 0], dtype=int64)
```

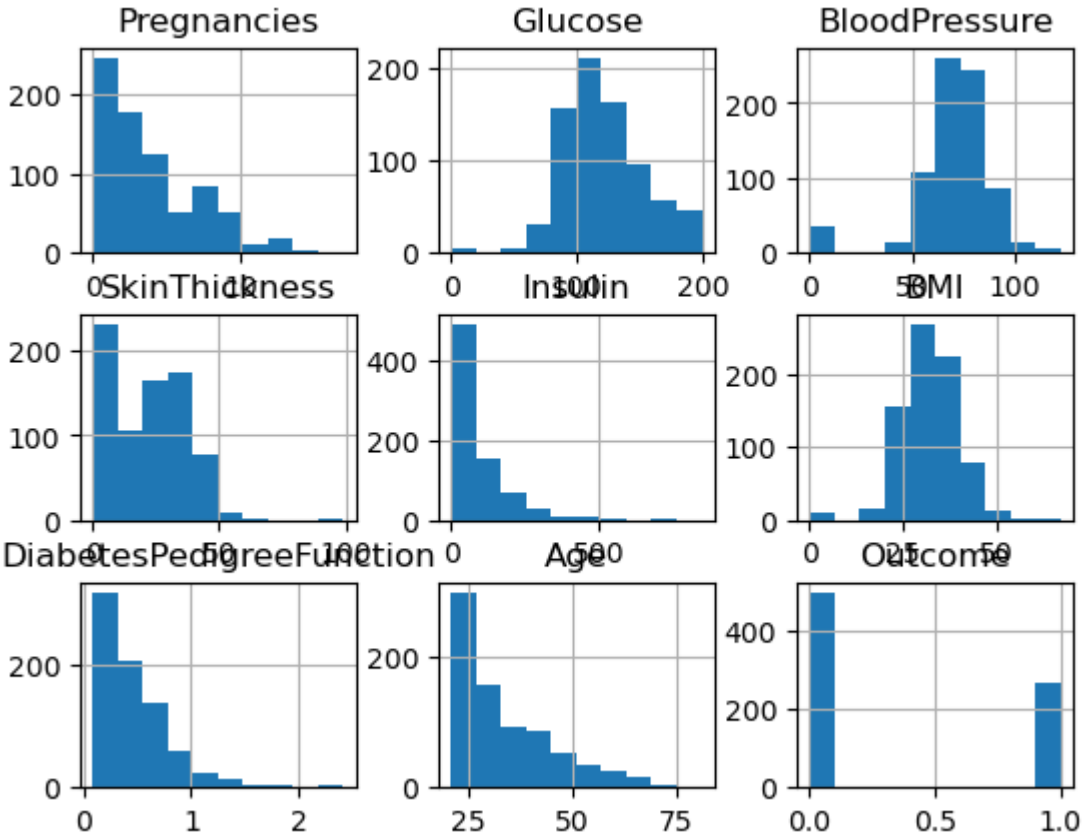
In [22]:
```
data.corr()
sns.heatmap(data.corr(),annot=True,fmt="0.1f")
```

Out[22]:
```
<AxesSubplot:>
```



In [43]:
```
data.hist()
```

Out[43]:
```
array([[<AxesSubplot:title={'center':'Pregnancies'}>,
        <AxesSubplot:title={'center':'Glucose'}>,
        <AxesSubplot:title={'center':'BloodPressure'}>],
       [<AxesSubplot:title={'center':'SkinThickness'}>,
        <AxesSubplot:title={'center':'Insulin'}>,
        <AxesSubplot:title={'center':'BMI'}>],
       [<AxesSubplot:title={'center':'DiabetesPedigreeFunction'}>,
        <AxesSubplot:title={'center':'Age'}>,
        <AxesSubplot:title={'center':'Outcome'}>]], dtype=object)
```



In [10]:
```
x = data.drop('Outcome',axis=1)
x
```

Out[10]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Ag
0	6	148	72	35	0	33.6	0.627	5
1	1	85	66	29	0	26.6	0.351	3
2	8	183	64	0	0	23.3	0.672	3
3	1	89	66	23	94	28.1	0.167	2
4	0	137	40	35	168	43.1	2.288	3
...	...	...	...	...	...	...	...	.
763	10	101	76	48	180	32.9	0.171	6
764	2	122	70	27	0	36.8	0.340	2
765	5	121	72	23	112	26.2	0.245	3
766	1	126	60	0	0	30.1	0.349	4
767	1	93	70	31	0	30.4	0.315	2

768 rows × 8 columns

In [12]:
```
y = data['Outcome']
y
```

Out[12]:
```
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

In [13]:
```
from sklearn.model_selection import train_test_split
xtrain,xt,ytrain,yt = train_test_split(x,y,test_size=0.2,random_state=30)
```

In [14]:
```
xtrain.shape,xt.shape,ytrain.shape,yt.shape
```

Out[14]:
```
((614, 8), (154, 8), (614,), (154,))
```

In [15]:
```
from sklearn.linear_model import LogisticRegression
LR = LogisticRegression()
model = LR.fit(xtrain,ytrain)
```

```
C:\Users\deepa\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: Con
vergenceWarning: 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(
```

In [17]:
```
model.score(xt,yt)*100
```

Out[17]:
```
78.57142857142857
```

In [19]:
```
z = LR.predict(xt)
z
```

Out[19]:
```
array([0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1,
       1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0,
       1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0],
      dtype=int64)
```

In [34]:
```
import gradio as gr
import numpy as np
```

In [40]:
```
def expense(Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,DiabetesPedigr
    x = np.array([Pregnancies,Glucose,BloodPressure,SkinThickness,Insulin,BMI,Diabetes
    x = np.array(x).reshape(1,-1)
 #df = sc.transform(df)
    prediction = LR.predict(x)
```

```
    prediction = float(prediction)
    return prediction
```

In [41]:
```
app = gr.Interface(fn=expense,
 inputs=[gr.inputs.Number(label="Pregnancies"),
 gr.inputs.Number(label="Glucose"),
 gr.inputs.Number(label="BloodPressure"),
 gr.inputs.Number(label="SkinThickness"),
 gr.inputs.Number(label="Insulin"),
 gr.inputs.Number(label="BMI"),
 gr.inputs.Number(label="DiabetesPedigreeF"),
 gr.inputs.Number(label="Age")
        ],
 outputs= "label",
 title="Developing an ML Model for Diabestes prediction"
 )
```

```
C:\Users\deepa\anaconda3\lib\site-packages\gradio\inputs.py:59: UserWarning: Usage of
gradio.inputs is deprecated, and will not be supported in the future, please import y
our component from gradio.components
  warnings.warn(
C:\Users\deepa\anaconda3\lib\site-packages\gradio\deprecation.py:40: UserWarning: `op
tional` parameter is deprecated, and it has no effect
  warnings.warn(value)
```

In [42]:
```
app.launch(show_error=True)
```

```
Running on local URL:  http://127.0.0.1:7863

To create a public link, set `share=True` in `launch()`.
```

Out[42]:

```
C:\Users\deepa\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does n
ot have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```

In [ ]: