RED WINE QUALITY ANALYSIS

In [4]:
```
import pandas as pd
```

In [5]:
```
data = pd.read_csv("D:\Data Analytics Project\Red Wine Quality/winequality-r
```

In [7]:
```
data.head(10)
```

Out[7]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alco
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	
5	7.4	0.66	0.00	1.8	0.075	13.0	40.0	0.9978	3.51	0.56	
6	7.9	0.60	0.06	1.6	0.069	15.0	59.0	0.9964	3.30	0.46	
7	7.3	0.65	0.00	1.2	0.065	15.0	21.0	0.9946	3.39	0.47	1
8	7.8	0.58	0.02	2.0	0.073	9.0	18.0	0.9968	3.36	0.57	
9	7.5	0.50	0.36	6.1	0.071	17.0	102.0	0.9978	3.35	0.80	1

In [8]:
```
data.tail(10)
```

Out[8]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates
1589	6.6	0.725	0.20	7.8	0.073	29.0	79.0	0.99770	3.29	0.54
1590	6.3	0.550	0.15	1.8	0.077	26.0	35.0	0.99314	3.32	0.82
1591	5.4	0.740	0.09	1.7	0.089	16.0	26.0	0.99402	3.67	0.56
1592	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75
1593	6.8	0.620	0.08	1.9	0.068	28.0	38.0	0.99651	3.42	0.82
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66

DATA PREPROCESSING

In [10]:

```
print('   Size of the table')
print('No.of Rows:',data.shape[0])
print('No.of Columns:',data.shape[1])
```

```
   Size of the table
No.of Rows: 1599
No.of Columns: 12
```

In [11]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [12]:

```
data.describe()
```

Out[12]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total di
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.0
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.4
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.8
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.0
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.0
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.0
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.0
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.0

In [13]:

```
# Checking Null values
data.isnull().sum()
```

Out[13]:

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

In [15]:

```
quality = data['quality'].value_counts()
quality
```

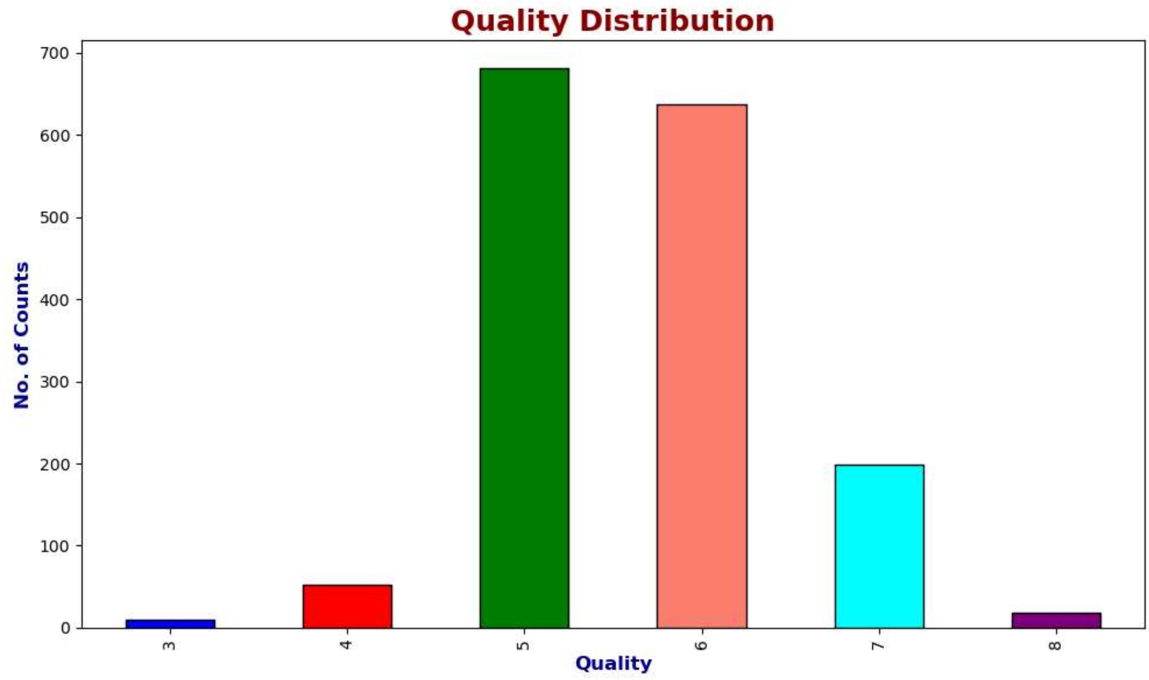
Out[15]:

```
5    681
6    638
7    199
4     53
8     18
3     10
Name: quality, dtype: int64
```

DATA VISUALIZATION AND EDA

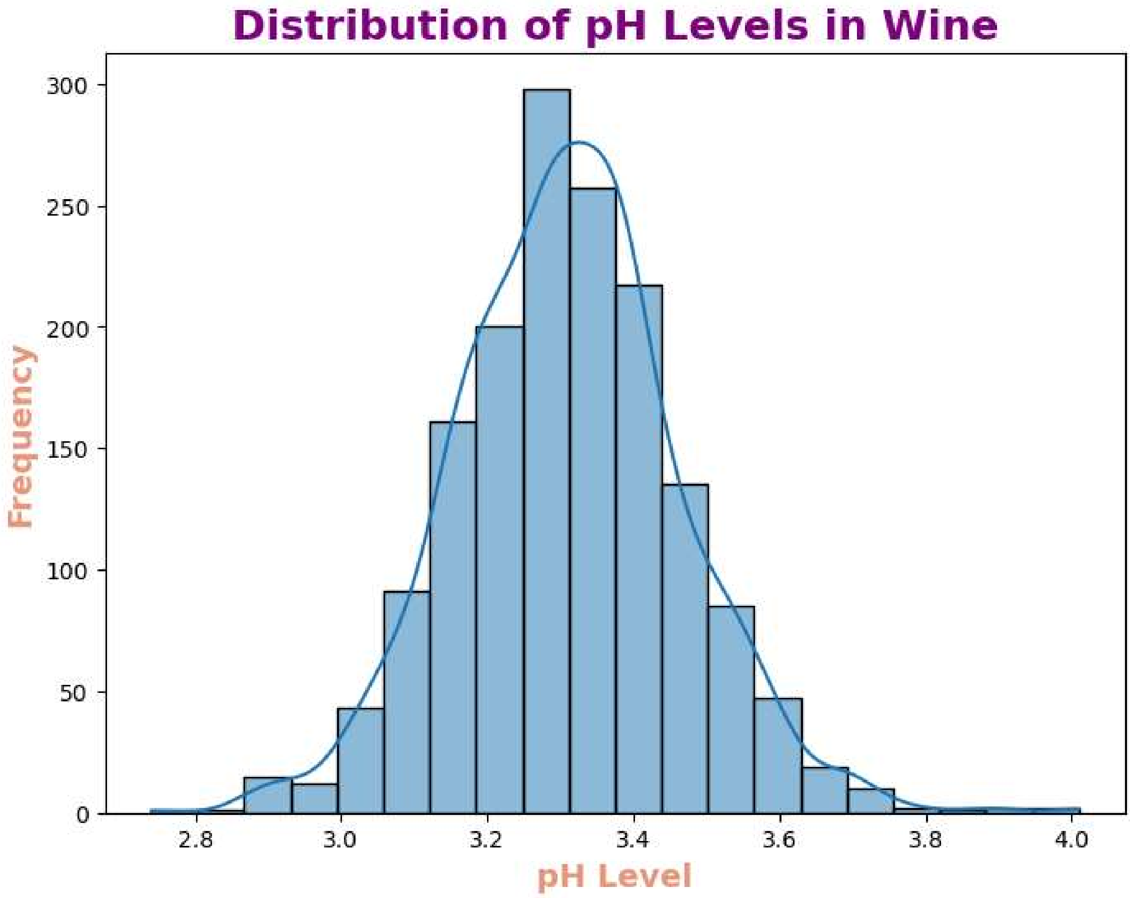
In [30]:

```
# Quality Distribution
quality = data.groupby('quality').size()
import matplotlib.pyplot as plt

plt.figure(figsize=(10,6))
quality.plot.bar(color=['blue','red','green','salmon','cyan','purple'],edge
plt.title('Quality Distribution',fontsize=18,fontweight='bold',color='darkre
plt.xlabel('Quality',fontsize=12,fontweight='bold',color='darkblue')
plt.ylabel('No. of Counts',fontsize=12,fontweight='bold',color='darkblue')
plt.tight_layout()
plt.show()
```

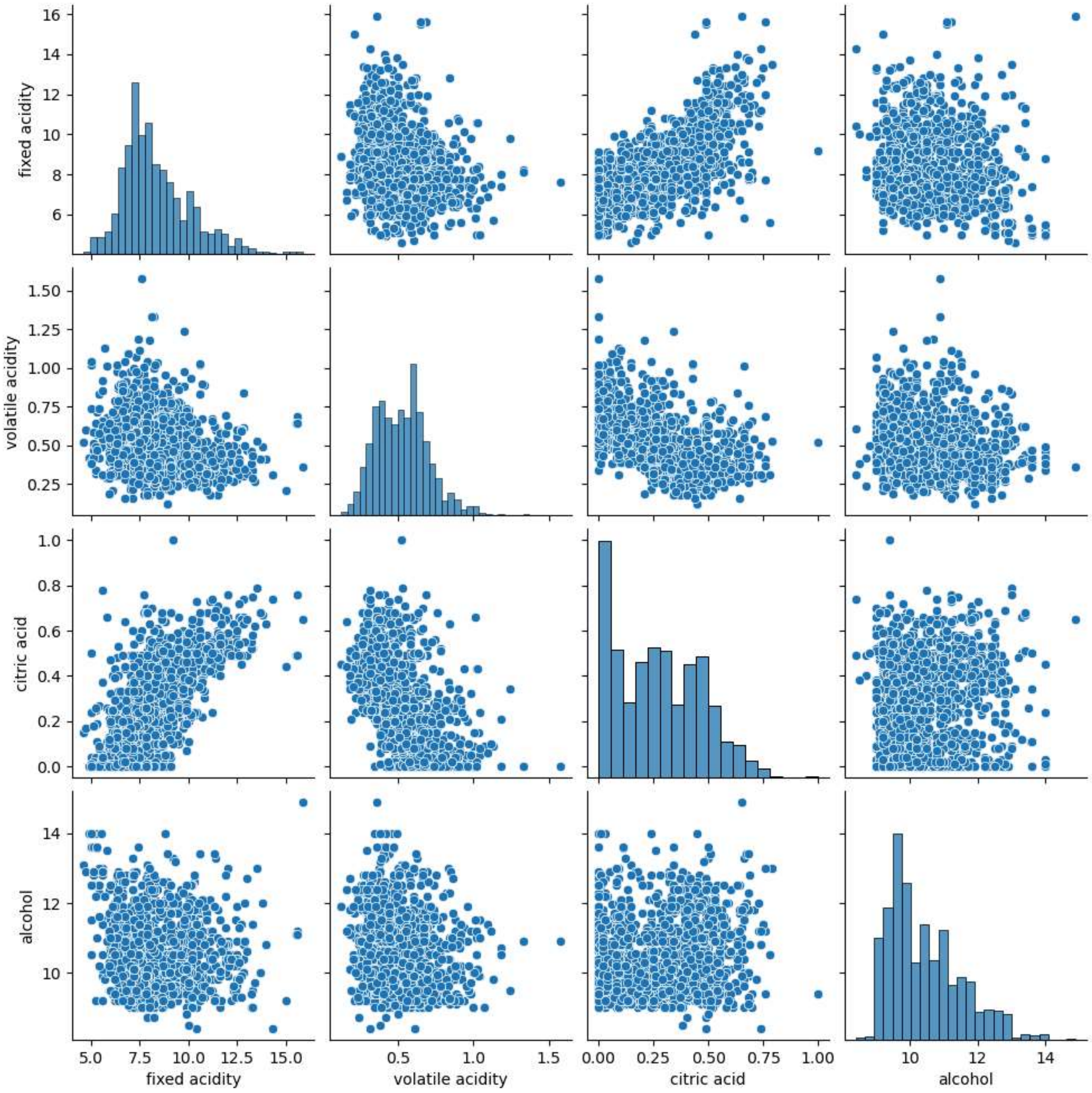


In [47]:

```
# Distribution of pH level
import seaborn as sns
plt.figure(figsize=(8,6))
sns.histplot(data['pH'],kde=True,bins=20)
plt.title('Distribution of pH Levels in Wine',fontsize=18,fontweight='bold'
plt.xlabel('pH Level',fontsize=14,color='darksalmon',fontweight='bold')
plt.ylabel('Frequency',fontsize=14,color='darksalmon',fontweight='bold')
plt.show()
```



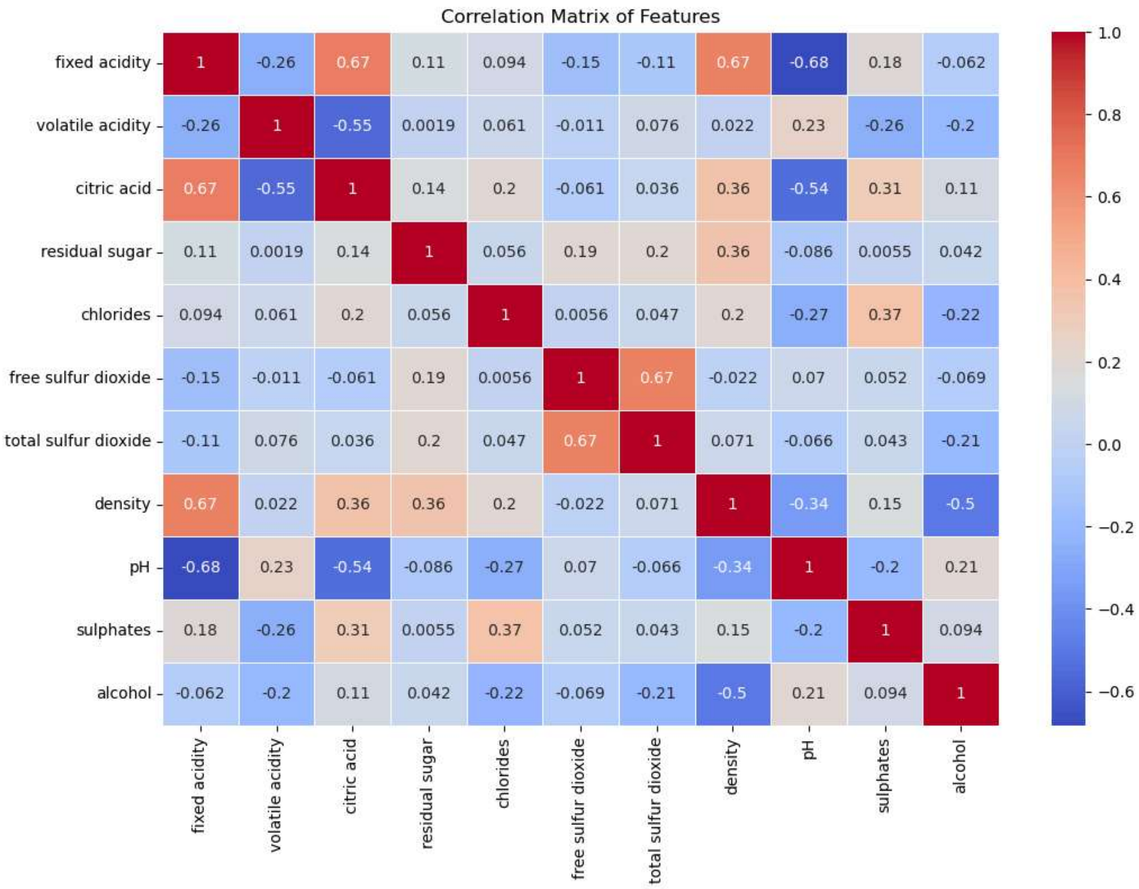
In [51]:
```
# Relationship between key features
sns.pairplot(data[['fixed acidity','volatile acidity','citric acid','alcohol
plt.show()
```



In [53]:

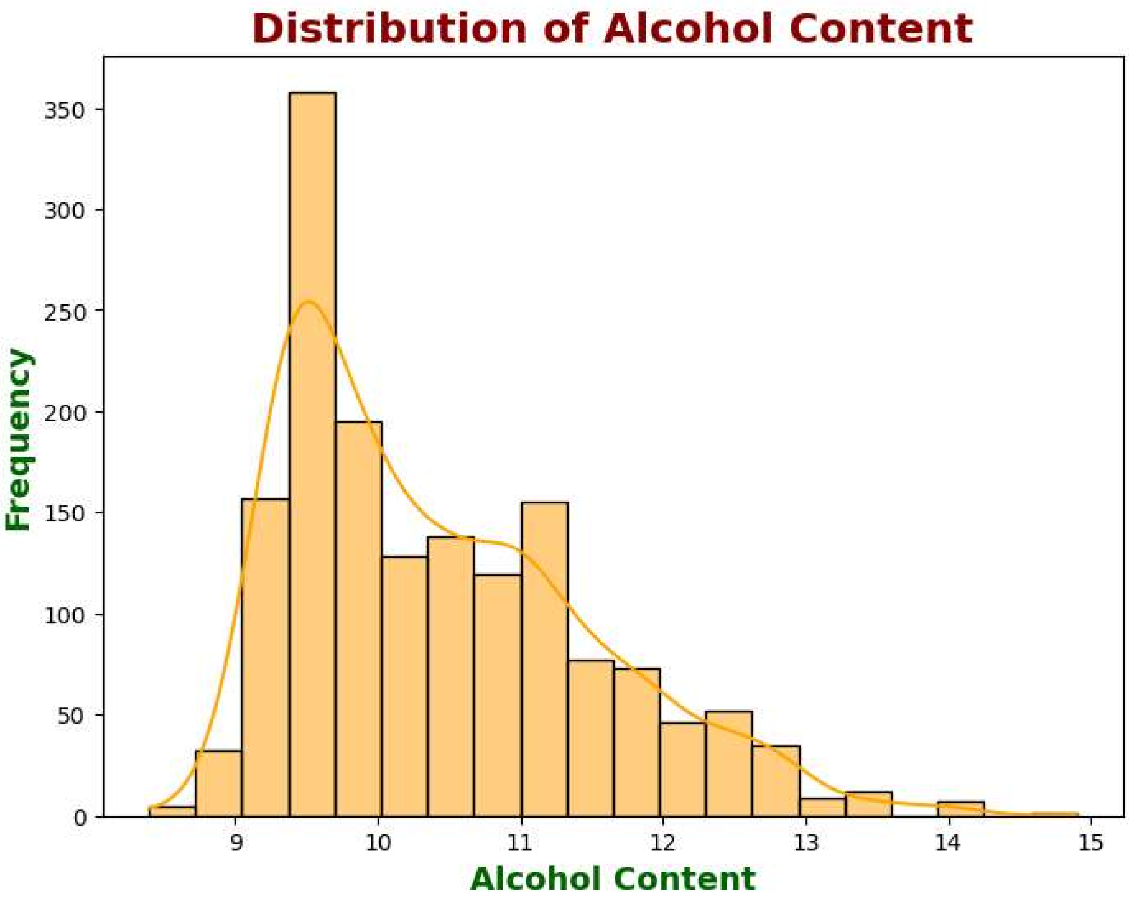
```
# Correlation Matrix
corr_matrix = data.corr()
plt.figure(figsize=(12,8))
sns.heatmap(corr_matrix,annot=True,cmap='coolwarm',linewidth=0.5)
plt.title('Correlation Matrix of Features')
plt.show()
```

```
C:\Users\ROHITH DP\AppData\Local\Temp\ipykernel_7932\2098915487.py:2: Futu
reWarning: The default value of numeric_only in DataFrame.corr is deprecat
ed. In a future version, it will default to False. Select only valid colum
ns or specify the value of numeric_only to silence this warning.
  corr_matrix = data.corr()
```



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
fixed acidity	1	-0.26	0.67	0.11	0.094	-0.15	-0.11	0.67	-0.68	0.18	-0.062
volatile acidity	-0.26	1	-0.55	0.0019	0.061	-0.011	0.076	0.022	0.23	-0.26	-0.2
citric acid	0.67	-0.55	1	0.14	0.2	-0.061	0.036	0.36	-0.54	0.31	0.11
residual sugar	0.11	0.0019	0.14	1	0.056	0.19	0.2	0.36	-0.086	0.0055	0.042
chlorides	0.094	0.061	0.2	0.056	1	0.0056	0.047	0.2	-0.27	0.37	-0.22
free sulfur dioxide	-0.15	-0.011	-0.061	0.19	0.0056	1	0.67	-0.022	0.07	0.052	-0.069
total sulfur dioxide	-0.11	0.076	0.036	0.2	0.047	0.67	1	0.071	-0.066	0.043	-0.21
density	0.67	0.022	0.36	0.36	0.2	-0.022	0.071	1	-0.34	0.15	-0.5
pH	-0.68	0.23	-0.54	-0.086	-0.27	0.07	-0.066	-0.34	1	-0.2	0.21
sulphates	0.18	-0.26	0.31	0.0055	0.37	0.052	0.043	0.15	-0.2	1	0.094
alcohol	-0.062	-0.2	0.11	0.042	-0.22	-0.069	-0.21	-0.5	0.21	0.094	1

In [56]:
```
# Alcohol Distribution
plt.figure(figsize=(8,6))
sns.histplot(data['alcohol'],kde=True,bins=20,color='orange')
plt.title('Distribution of Alcohol Content',fontsize=18,
          fontweight='bold',color='darkred')
plt.xlabel('Alcohol Content',fontsize=14,fontweight='bold',color='darkgreen
plt.ylabel('Frequency',fontsize=14,fontweight='bold',color='darkgreen')
plt.show()
```



CLASSIFICATION

In [58]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,confusion_
```

In [37]:

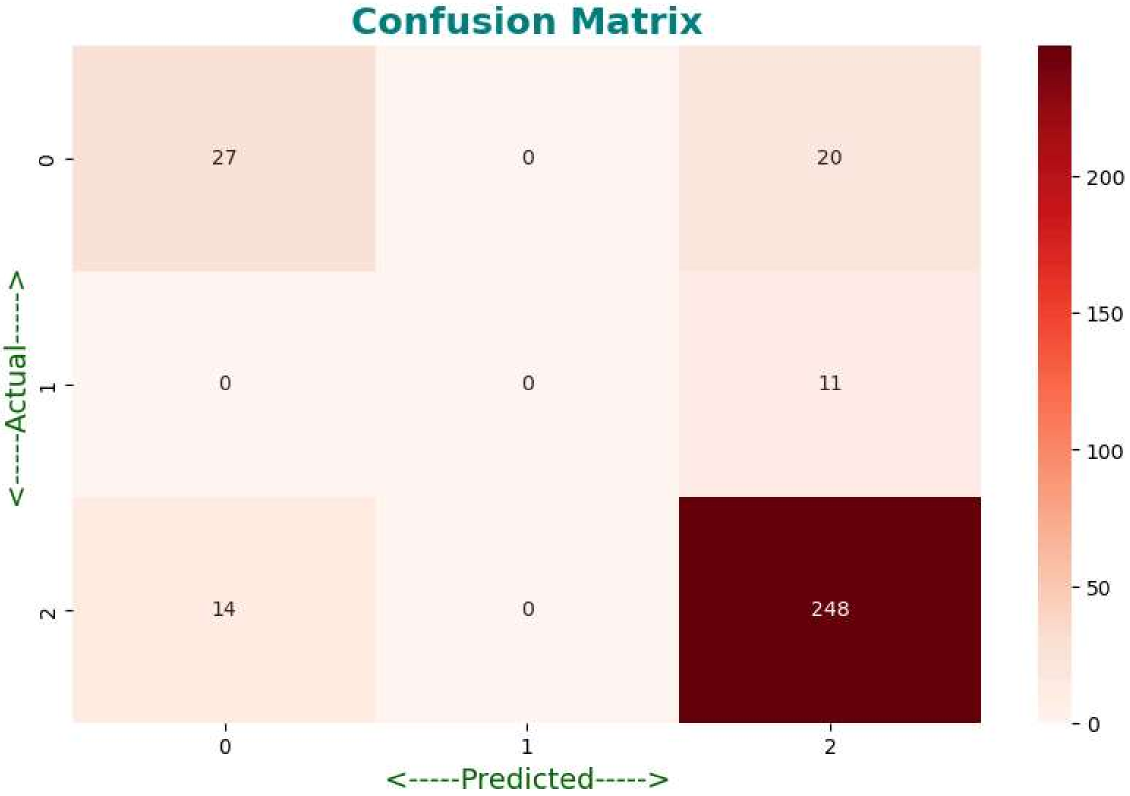
```
bins = [2,4,6,8]
labels = ['low','medium','high']
data['quality_category'] = pd.cut(data['quality'],bins=bins,labels=labels,in
data = data.drop('quality',axis=1)

# Separate features and target variables
x = data.drop('quality_category',axis=1)
y = data['quality_category']

# Split, Train and Test
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_
```

Random Forest Classifier

In [67]:

```
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
y_pred = rfc.predict(x_test)
rfc_accuracy = accuracy_score(y_test,y_pred)
rfc_report = classification_report(y_test,y_pred)
print("Accuracy:",rfc_accuracy)
print("Classification Report:")
print(rfc_report)

rfc_conf_matrix = confusion_matrix(y_test,y_pred)
print("Confusion Matrix:\n",rfc_conf_matrix)
plt.figure(figsize=(10,6))
sns.heatmap(rfc_conf_matrix,annot=True,fmt='d',cmap='Reds')
plt.title('Confusion Matrix',fontsize=18,fontweight='bold',color='teal')
plt.xlabel('<-----Predicted----->',fontsize=14,color='darkgreen')
plt.ylabel("<-----Actual----->",fontsize=14,color='darkgreen')
plt.show()
```

```
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

Accuracy: 0.859375
Classification Report:
              precision    recall  f1-score   support

        high       0.66      0.57      0.61        47
         low       0.00      0.00      0.00        11
      medium       0.89      0.95      0.92       262

    accuracy                           0.86       320
   macro avg       0.52      0.51      0.51       320
weighted avg       0.82      0.86      0.84       320

Confusion Matrix:
 [[ 27   0  20]
 [  0   0  11]
 [ 14   0 248]]
```

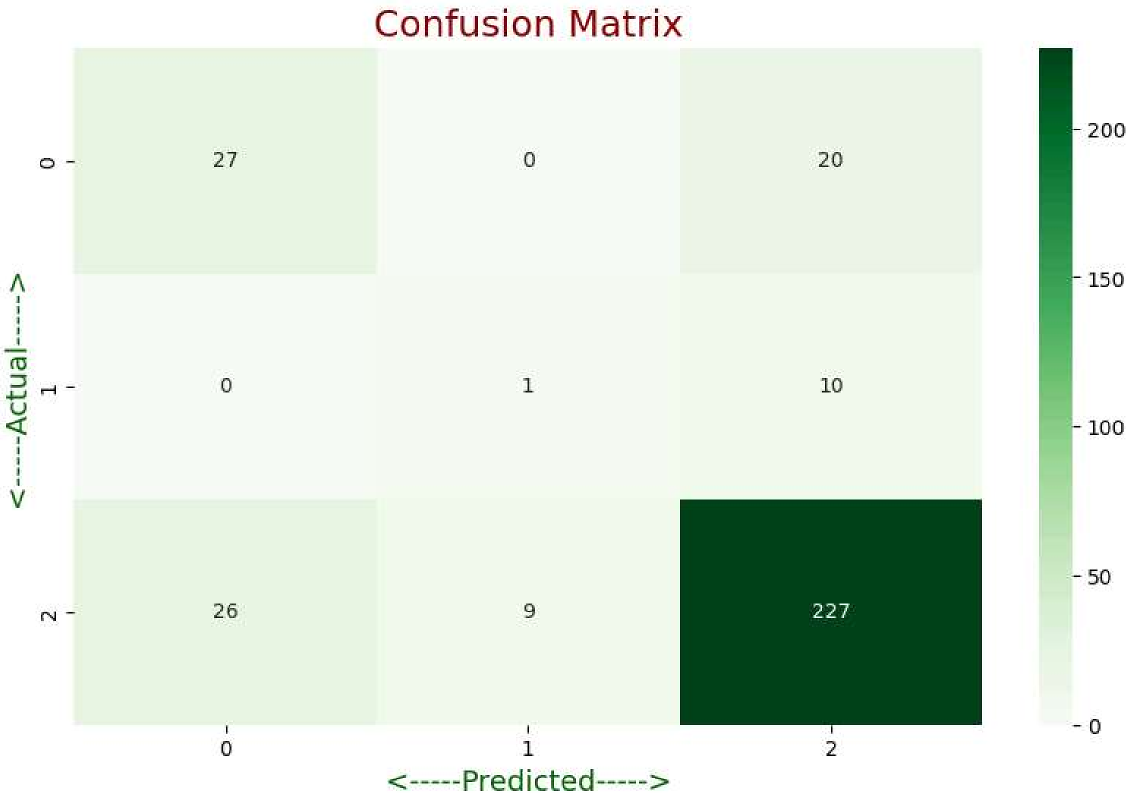


Decision Tree Classifier

In [69]:

```
dtc = DecisionTreeClassifier()
dtc.fit(x_train,y_train)
y_pred = dtc.predict(x_test)
dtc_accuracy = accuracy_score(y_test,y_pred)
dtc_report = classification_report(y_test,y_pred)
print('Accuracy:',dtc_accuracy)
print("Classification Report:")
print(dtc_report)
dtc_conf_matrix = confusion_matrix(y_test,y_pred)
print('Confusion Matrix:\n',dtc_conf_matrix)
plt.figure(figsize=(10,6))
sns.heatmap(dtc_conf_matrix,annot=True,fmt='d',cmap='Greens')
plt.title('Confusion Matrix',fontsize=18,color='darkred')
plt.xlabel('<-----Predicted----->',fontsize=14,color='darkgreen')
plt.ylabel('<-----Actual----->',fontsize=14,color='darkgreen')
plt.show()
```

```
Accuracy: 0.796875
Classification Report:
              precision    recall  f1-score   support

        high       0.51      0.57      0.54        47
         low       0.10      0.09      0.10        11
      medium       0.88      0.87      0.87       262

    accuracy                           0.80       320
   macro avg       0.50      0.51      0.50       320
weighted avg       0.80      0.80      0.80       320

Confusion Matrix:
 [[ 27   0  20]
 [  0   1  10]
 [ 26   9 227]]
```



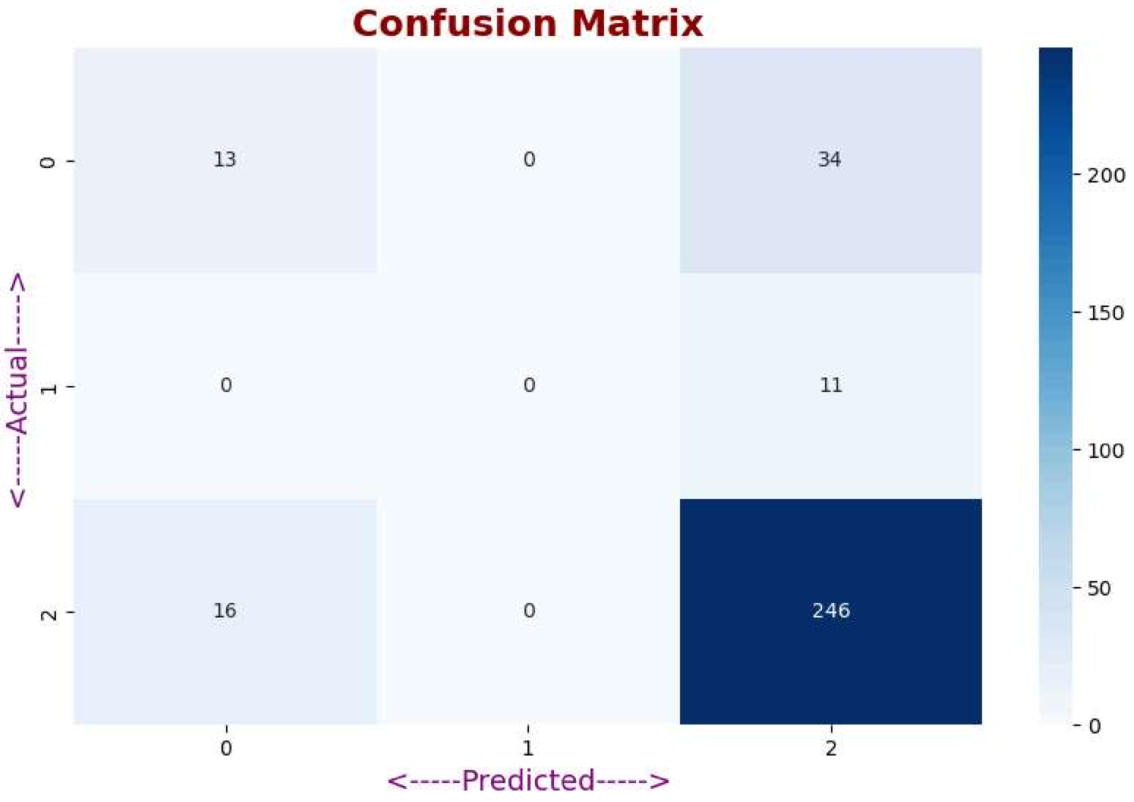
Logistic Regression

In [100]:

```
lr = LogisticRegression()
lr.fit(x_train,y_train)
y_pred = lr.predict(x_test)
lr_accuracy = accuracy_score(y_test,y_pred)
lr_report = classification_report(y_test,y_pred)
print("Accuracy:",lr_accuracy)
print("Classification Report")
print(lr_report)
lr_conf_matrix = confusion_matrix(y_test,y_pred)
print('Confusion Matrix:\n',lr_conf_matrix)
plt.figure(figsize=(10,6))
sns.heatmap(lr_conf_matrix,annot=True,fmt='d',cmap='Blues')
plt.title('Confusion Matrix',fontsize=18,fontweight='bold',color='darkred')
plt.xlabel('<-----Predicted----->',fontsize=14,color='purple')
plt.ylabel('<-----Actual----->',fontsize=14,color='purple')
plt.show()
```

```
E:\Anaconda Software\Lib\site-packages\sklearn\linear_model\_logistic.py:4
58: 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 i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sc
ikit-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-reg
ression (https://scikit-learn.org/stable/modules/linear_model.html#logisti
c-regression)
  n_iter_i = _check_optimize_result(
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\Anaconda Software\Lib\site-packages\sklearn\metrics\_classification.py:
1344: UndefinedMetricWarning: Precision and F-score are ill-defined and be
ing set to 0.0 in labels with no predicted samples. Use `zero_division` pa
rameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
Accuracy: 0.809375
Classification Report
              precision    recall  f1-score   support

        high       0.45      0.28      0.34        47
         low       0.00      0.00      0.00        11
      medium       0.85      0.94      0.89       262

    accuracy                           0.81       320
   macro avg       0.43      0.41      0.41       320
weighted avg       0.76      0.81      0.78       320

Confusion Matrix:
 [[ 13   0  34]
 [  0   0  11]
 [ 16   0 246]]
```



In [99]:

```
plt.figure()
plt.text(0.5,0.6,'Thank You',fontsize=30,ha='center',va='center')
plt.text(0.5,0.4,'CognoRise Infotech',fontsize=40,fontweight='bold',color='
plt.axis('off')
plt.show()
```

Thank You

CognoRise Infotech