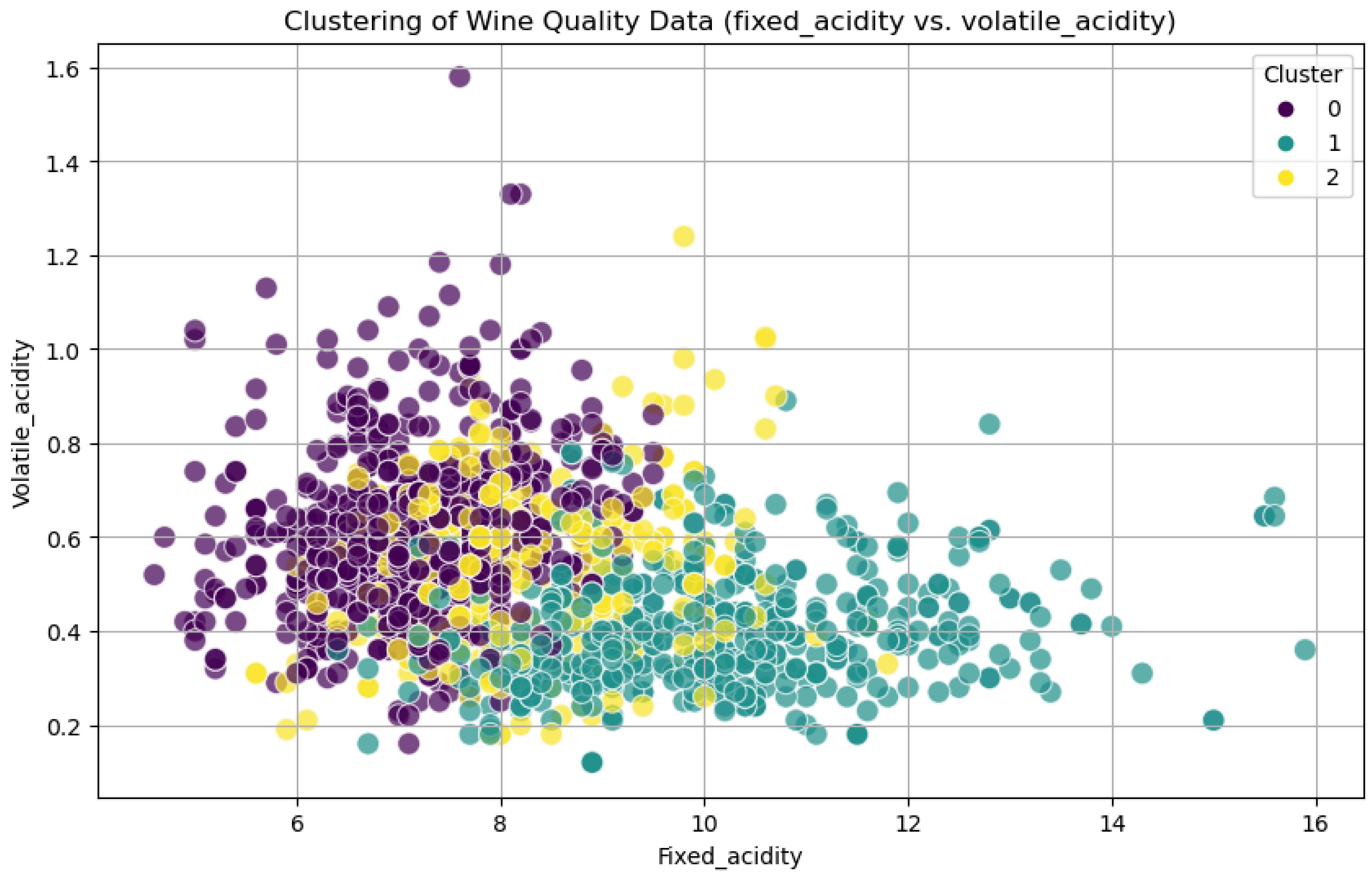
In [2]:

```python
# Import necessary libraries
import pandas as pd
from sklearn.datasets import fetch_openml
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns

# Step 1: Load the Dataset
# Fetch the Wine Quality dataset and save it as a CSV file
data = fetch_openml(name='wine-quality-red', as_frame=True)
df = pd.concat([data.data, data.target.rename('quality')], axis=1)
df.to_csv("wine_quality.csv", index=False)  # Save as CSV

# Load the CSV file
df = pd.read_csv("wine_quality.csv")

# Print the dataset structure to verify column names
print(df.head())
print(df.columns)

# Step 2: Preprocess the Data
# Separate features and target variable
features = df.drop(columns=['quality'])
target = df['quality']

# Standardize the features
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

# Step 3: Apply KMeans Clustering
# Initialize KMeans with 3 clusters
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
kmeans.fit(features_scaled)

# Add cluster labels to the dataframe
df['cluster'] = kmeans.labels_

# Step 4: Visualize the Clusters
# Use the correct column names from the dataset
# Replace 'fixed acidity' and 'alcohol' with the actual column names
if 'fixed acidity' in features.columns and 'alcohol' in features.columns:
    x_col = 'alcohol'
    y_col = 'fixed acidity'
else:
    # Replace with alternative columns if those do not exist
    x_col = features.columns[0]
    y_col = features.columns[1]

plt.figure(figsize=(10, 6))
sns.scatterplot(
    x=features[x_col],
    y=features[y_col],
    hue=df['cluster'],
    palette='viridis',
    s=100,
    alpha=0.7
)
plt.title(f"Clustering of Wine Quality Data ({x_col} vs. {y_col})")
plt.xlabel(x_col.capitalize())
plt.ylabel(y_col.capitalize())
plt.legend(title="Cluster")
plt.grid(True)
plt.show()
```

```
C:\Users\Kritika Agrahari\anaconda3\Lib\site-packages\sklearn\datasets\_openml.py:1002: FutureWarning: The default value of `parser` will change from
`'liac-arff'` to `'auto'` in 1.4. You can set `parser='auto'` to silence this warning. Therefore, an `ImportError` will be raised from 1.4 if the datase
t is dense and pandas is not installed. Note that the pandas parser may return different data types. See the Notes Section in fetch_openml's API doc for
details.
  warn(
   fixed_acidity  volatile_acidity  citric_acid  residual_sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free_sulfur_dioxide  total_sulfur_dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
Index(['fixed_acidity', 'volatile_acidity', 'citric_acid', 'residual_sugar',
       'chlorides', 'free_sulfur_dioxide', 'total_sulfur_dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```



In [ ]: