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

from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer

sns.set(style="whitegrid")
plt.rcParams["figure.figsize"] = (12, 6)
```

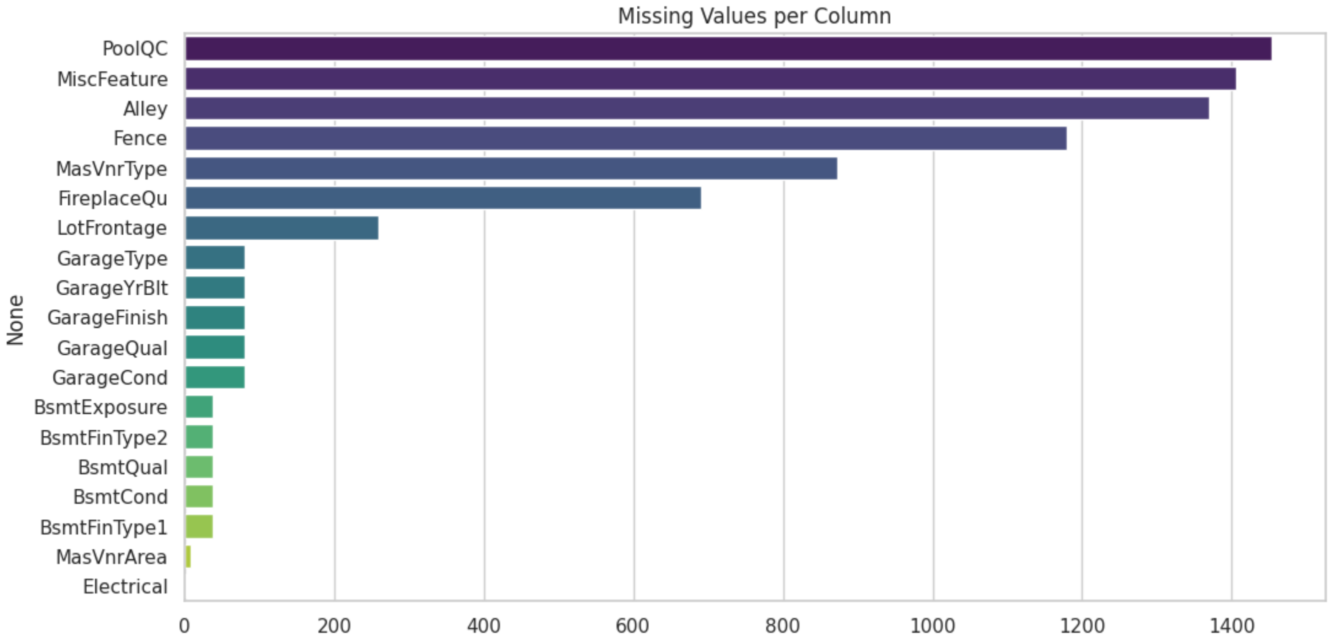
```
df = pd.read_csv('/content/train.csv')
print("Shape of dataset:", df.shape)
df.head()
```

```
Shape of dataset: (1460, 81)
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	La
0	1	60	RL	65.0	8450	Pave	NaN	Reg	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	

5 rows × 81 columns

```
# Visualize missing values
missing = df.isnull().sum()
missing = missing[missing > 0].sort_values(ascending=False)
sns.barplot(x=missing.values, y=missing.index, palette='viridis')
plt.title("Missing Values per Column")
plt.show()
```



```
# Drop columns with too many missing values
df = df.drop(['Alley', 'PoolQC', 'Fence', 'MiscFeature'], axis=1)

# Numeric columns – fill with median
num_cols = df.select_dtypes(include=[np.number]).columns
df[num_cols] = df[num_cols].fillna(df[num_cols].median())

# Categorical columns – fill with mode
cat_cols = df.select_dtypes(include=['object']).columns
df[cat_cols] = df[cat_cols].fillna(df[cat_cols].mode().iloc[0])
```

```
# Label encode ordinal columns
ordinal_cols = ['ExterQual', 'ExterCond', 'BsmtQual', 'BsmtCond',
                'HeatingQC', 'KitchenQual', 'FireplaceQu', 'GarageQual', 'Garag

le = LabelEncoder()
for col in ordinal_cols:
    if col in df.columns:
        df[col] = le.fit_transform(df[col])

# One-hot encode remaining categorical variables
df = pd.get_dummies(df, drop_first=True)
print("Dataset shape after encoding:", df.shape)
```

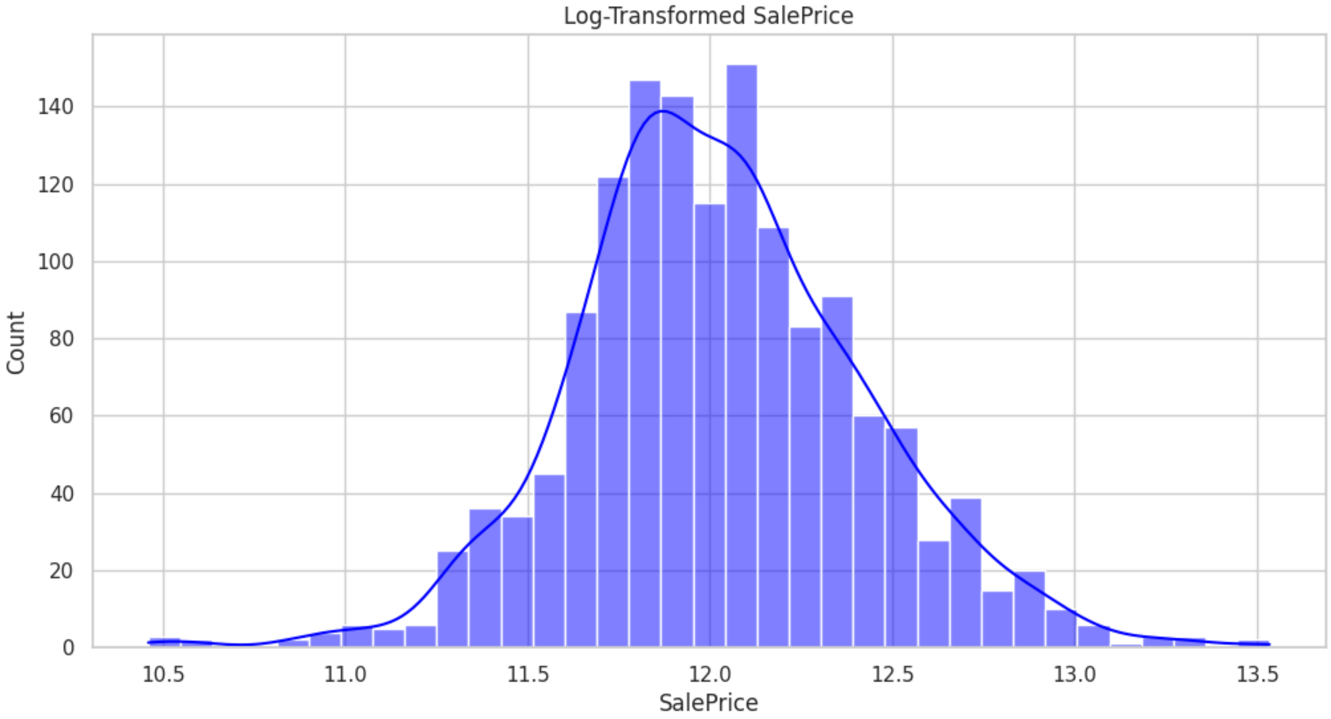
```
Dataset shape after encoding: (1460, 214)
```

```
# New features
df['TotalBathrooms'] = (df['FullBath'] + 0.5 * df['HalfBath'] +
                        df['BsmtFullBath'] + 0.5 * df['BsmtHalfBath'])

df['HouseAge'] = df['YrSold'] - df['YearBuilt']
df['Remodeled'] = (df['YearBuilt'] != df['YearRemodAdd']).astype(int)
```

```
scaler = StandardScaler()
scale_cols = ['GrLivArea', 'GarageArea', 'TotalBathrooms', 'HouseAge']
df[scale_cols] = scaler.fit_transform(df[scale_cols])
```

```
# Log-transform SalePrice
df['SalePrice'] = np.log1p(df['SalePrice'])
sns.histplot(df['SalePrice'], kde=True, color='blue')
plt.title("Log-Transformed SalePrice")
plt.show()
```



```
print("Final dataset shape:", df.shape)
df.head()
```

```
Final dataset shape: (1460, 217)
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt
0	1	60	65.0	8450	7	5	2003
1	2	20	80.0	9600	6	8	1976
2	3	60	68.0	11250	7	5	2001
3	4	70	60.0	9550	7	5	1915
4	5	60	84.0	14260	8	5	2000

5 rows × 217 columns

```
Warning: Total number of columns (217) exceeds max_columns (20) limiting to
```