# Feature Engineering Dream House Project

In [1]:
```
## Import the libraries

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

## Display all the columns of the dataframe

pd.pandas.set_option('display.max_columns',None)
```

In [3]:
```
#read the dataset
dataset=pd.read_csv('C:\\SimpliLearn\\2. Data science with Python\\Feature Engineering\\pep1.csv')
```

In [4]:
```
dataset.shape
```

Out[4]:
```
(1460, 81)
```

In [5]:
```
dataset.head()
```

Out[5]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Nei
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	Corner	Gtl	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	FR2	Gtl	

In [6]:
```
#1. a. a.   Identify the shape of the dataset
dataset.shape
```

Out[6]:
```
(1460, 81)
```

In [7]:
```
#1. b.  Identify variables with null values
dataset.isnull().sum()
```

Out[7]:
```
Id                 0
MSSubClass         0
MSZoning           0
LotFrontage      259
LotArea            0
                ...
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64
```

In [10]:
```
## Fill Missing Values

dataset['LotFrontage']=pep['LotFrontage'].fillna(dataset['LotFrontage'].mean())
```

In [11]:

```
dataset['LotFrontage']
```

Out[11]:

```
0       65.0
1       80.0
2       68.0
3       60.0
4       84.0
        ...
1455    62.0
1456    85.0
1457    66.0
1458    68.0
1459    75.0
Name: LotFrontage, Length: 1460, dtype: float64
```

In [12]:

```
dataset.drop(['Alley'],axis=1,inplace=True)
```

In [13]:

```
dataset['BsmtCond']=dataset['BsmtCond'].fillna(dataset['BsmtCond'].mode()[0])
dataset['BsmtQual']=dataset['BsmtQual'].fillna(dataset['BsmtQual'].mode()[0])
```

In [14]:

```
dataset['FireplaceQu']=dataset['FireplaceQu'].fillna(dataset['FireplaceQu'].mode()[0])
dataset['GarageType']=dataset['GarageType'].fillna(dataset['GarageType'].mode()[0])
```

In [15]:

```
dataset.drop(['GarageYrBlt'],axis=1,inplace=True)
```

In [16]:

```
dataset['GarageFinish']=dataset['GarageFinish'].fillna(dataset['GarageFinish'].mode()[0])
dataset['GarageQual']=dataset['GarageQual'].fillna(dataset['GarageQual'].mode()[0])
dataset['GarageCond']=dataset['GarageCond'].fillna(dataset['GarageCond'].mode()[0])
```

In [17]:

```
dataset.shape
```

Out[17]:

```
(1460, 79)
```

In [18]:

```
dataset.columns
```

Out[18]:

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'LandSlope',
       'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle',
       'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'RoofStyle',
       'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'MasVnrArea',
       'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1', 'BsmtFinType2',
       'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating', 'HeatingQC',
       'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchebvGr', 'KitchenQual', 'TotRmsAbvGrd', 'Functiol',
       'Fireplaces', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageCars',
       'GarageArea', 'GarageQual', 'GarageCond', 'PavedDrive', 'WoodDeckSF',
       'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea',
       'PoolQC', 'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold',
       'SaleType', 'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [19]:

```
dataset.drop(['Id'],axis=1,inplace=True)
```

In [20]:

```
dataset.shape
```

Out[20]:

```
(1460, 78)
```

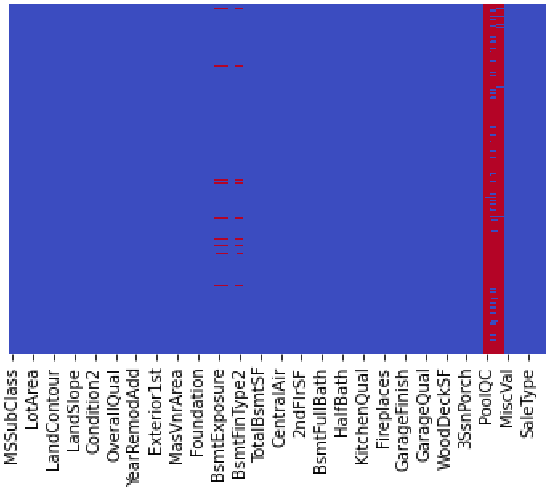
In [21]:
```
dataset.isna().sum()
```

Out[21]:
```
MSSubClass       0
MSZoning         0
LotFrontage      0
LotArea          0
Street           0
                ..
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
SalePrice        0
Length: 78, dtype: int64
```

In [22]:
```
dataset['MasVnrType']=dataset['MasVnrType'].fillna(dataset['MasVnrType'].mode()[0])
dataset['MasVnrArea']=dataset['MasVnrArea'].fillna(dataset['MasVnrArea'].mode()[0])
```

In [23]:
```
sns.heatmap(dataset.isnull(),yticklabels=False,cbar=False,cmap='coolwarm')
```

Out[23]:
```
<AxesSubplot:>
```



In [24]:
```
dataset['BsmtExposure']=dataset['BsmtExposure'].fillna(dataset['BsmtExposure'].mode()[0])
```

In [25]:
```
dataset.shape
```

Out[25]:
```
(1460, 78)
```

# 2. Generate a separate dataset for numerical and categorical variables

In [26]:

```
# list of numerical variables
numerical_features = [feature for feature in dataset.columns if dataset[feature].dtypes != 'O']

print('Number of numerical variables: ', len(numerical_features))

# visualise the numerical variables
dataset[numerical_features].head()
```

```
Number of numerical variables:  36
```

Out[26]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinS
0	60	65.0	8450	7	5	2003	2003	196.0	706	
1	20	80.0	9600	6	8	1976	1976	0.0	978	
2	60	68.0	11250	7	5	2001	2002	162.0	486	
3	70	60.0	9550	7	5	1915	1970	0.0	216	
4	60	84.0	14260	8	5	2000	2000	350.0	655	

In [27]:

```
# list of variables that contain year information
year_feature = [feature for feature in numerical_features if 'Yr' in feature or 'Year' in feature]

year_feature
```

Out[27]:

```
['YearBuilt', 'YearRemodAdd', 'YrSold']
```

In [28]:

```
# let's explore the content of these year variables
for feature in year_feature:
    print(feature, dataset[feature].unique())
```

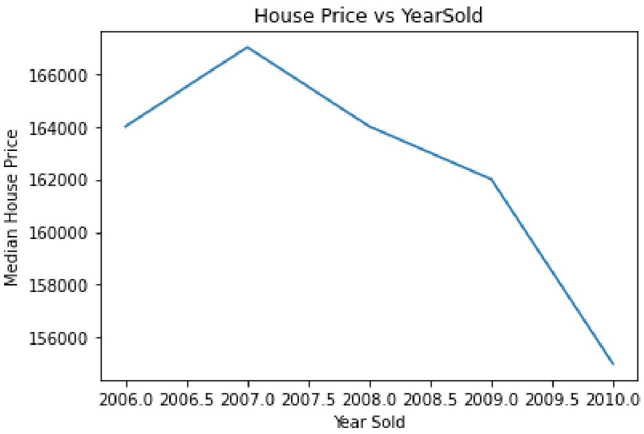
```
YearBuilt [2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 1965 2005 1962 2006
 1960 1929 1970 1967 1958 1930 2002 1968 2007 1951 1957 1927 1920 1966
 1959 1994 1954 1953 1955 1983 1975 1997 1934 1963 1981 1964 1999 1972
 1921 1945 1982 1998 1956 1948 1910 1995 1991 2009 1950 1961 1977 1985
 1979 1885 1919 1990 1969 1935 1988 1971 1952 1936 1923 1924 1984 1926
 1940 1941 1987 1986 2008 1908 1892 1916 1932 1918 1912 1947 1925 1900
 1980 1989 1992 1949 1880 1928 1978 1922 1996 2010 1946 1913 1937 1942
 1938 1974 1893 1914 1906 1890 1898 1904 1882 1875 1911 1917 1872 1905]
YearRemodAdd [2003 1976 2002 1970 2000 1995 2005 1973 1950 1965 2006 1962 2007 1960
 2001 1967 2004 2008 1997 1959 1990 1955 1983 1980 1966 1963 1987 1964
 1972 1996 1998 1989 1953 1956 1968 1981 1992 2009 1982 1961 1993 1999
 1985 1979 1977 1969 1958 1991 1971 1952 1975 2010 1984 1986 1994 1988
 1954 1957 1951 1978 1974]
YrSold [2008 2007 2006 2009 2010]
```

In [29]:

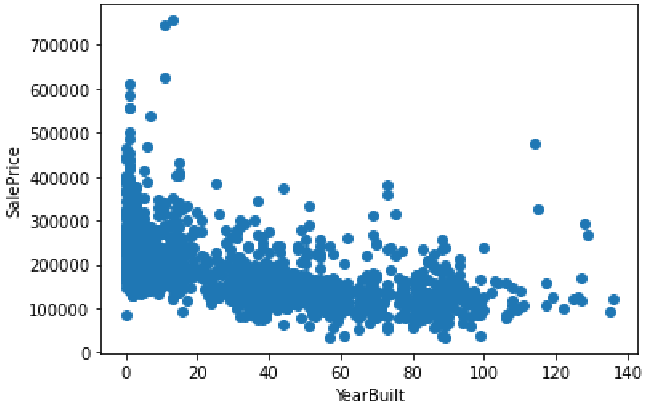
```
## Lets analyze the Temporal Datetime Variables
## We will check whether there is a relation between year the house is sold and the sales price

dataset.groupby('YrSold')['SalePrice'].median().plot()
plt.xlabel('Year Sold')
plt.ylabel('Median House Price')
plt.title("House Price vs YearSold")
```

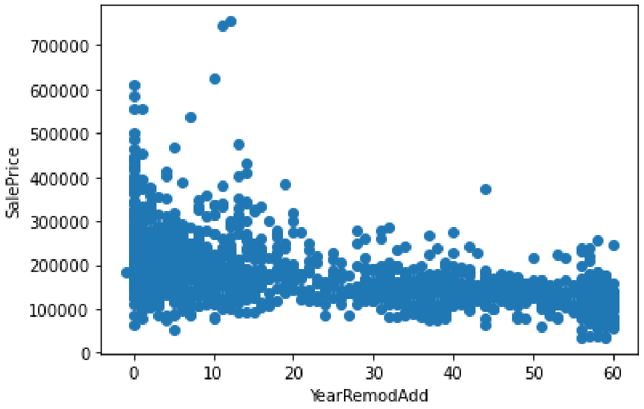
Out[29]:

```
Text(0.5, 1.0, 'House Price vs YearSold')
```



In [30]:
```
## Here we will compare the difference between All years feature with SalePrice

for feature in year_feature:
    if feature!='YrSold':
        data=dataset.copy()
        ## We will capture the difference between year variable and year the house was sold for
        data[feature]=data['YrSold']-data[feature]

        plt.scatter(data[feature],data['SalePrice'])
        plt.xlabel(feature)
        plt.ylabel('SalePrice')
        plt.show()
```





In [31]:
```
## Numerical variables are usually of 2 type
## 1. Continous variable and Discrete Variables

discrete_feature=[feature for feature in numerical_features if len(dataset[feature].unique())<25 and feat
print("Discrete Variables Count: {}".format(len(discrete_feature)))
```

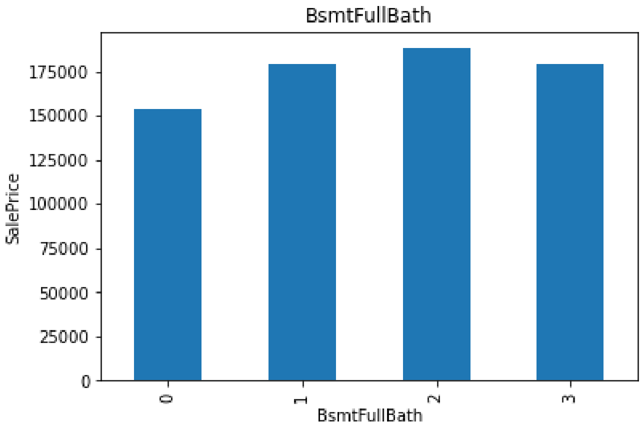
```
Discrete Variables Count: 17
```

In [32]:
```
dataset[discrete_feature].head()
```

Out[32]:

	MSSubClass	OverallQual	OverallCond	LowQualFinSF	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	Kitch
0	60	7	5	0	1	0	2	1	3	
1	20	6	8	0	0	1	2	0	3	
2	60	7	5	0	1	0	2	1	3	
3	70	7	5	0	1	0	1	0	3	
4	60	8	5	0	1	0	2	1	4	

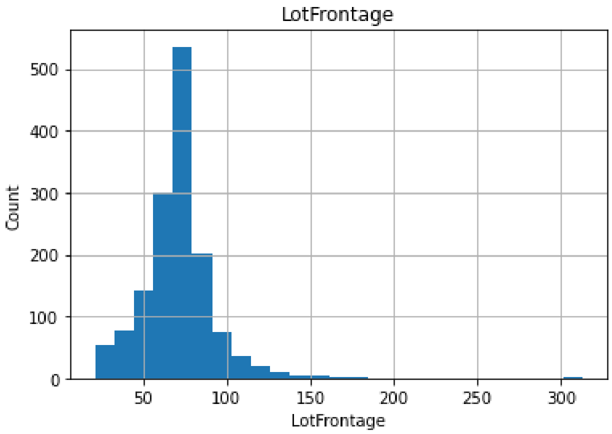
In [33]:
```
## Lets Find the realtionship between them and Sale PRice

for feature in discrete_feature:
    data=dataset.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```



In [34]:
```
continuous_feature=[feature for feature in numerical_features if feature not in discrete_feature+year_fea
print("Continuous feature Count {}".format(len(continuous_feature)))
```

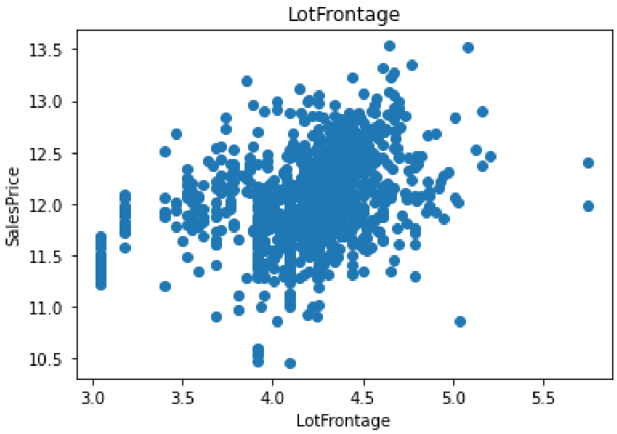
```
Continuous feature Count 16
```

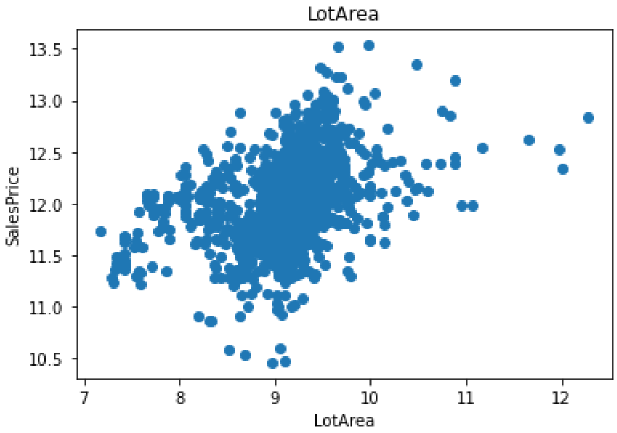
In [35]:
```
## Lets analyse the continuous values by creating histograms to understand the distribution

for feature in continuous_feature:
    data=dataset.copy()
    data[feature].hist(bins=25)
    plt.xlabel(feature)
    plt.ylabel("Count")
    plt.title(feature)
    plt.show()
```

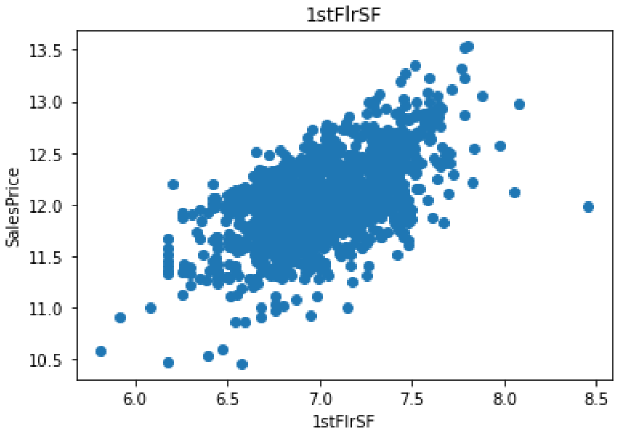


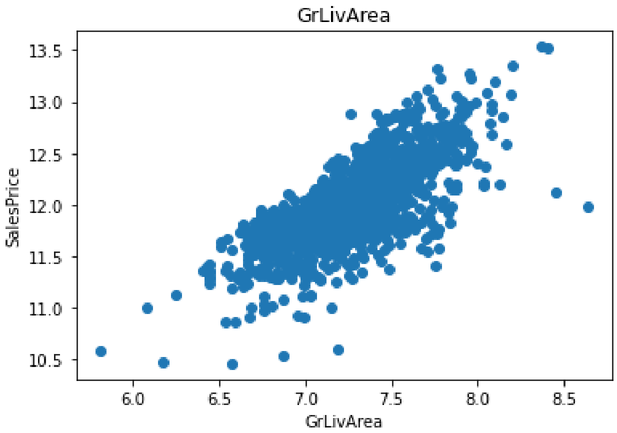
# 3.   EDA of numerical variables:

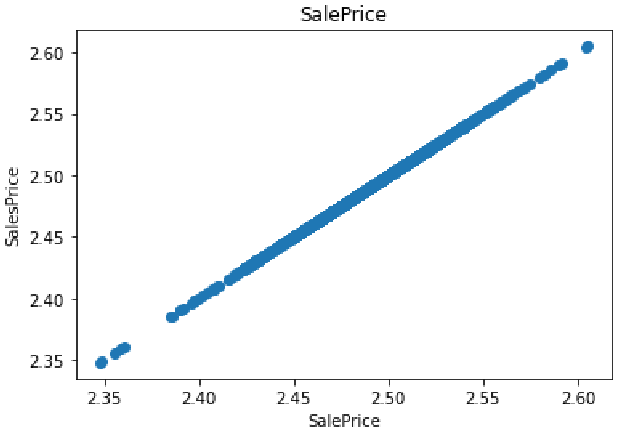
In [36]:
```
## We will be using logarithmic transformation


for feature in continuous_feature:
    data=dataset.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data['SalePrice']=np.log(data['SalePrice'])
        plt.scatter(data[feature],data['SalePrice'])
        plt.xlabel(feature)
        plt.ylabel('SalesPrice')
        plt.title(feature)
        plt.show()
```











# 5. Combine all the significant categorical and numerical variables Categorical Variables

In [37]:

```
categorical_features=[feature for feature in dataset.columns if data[feature].dtypes=='O']
categorical_features
```

Out[37]:

```
['MSZoning',
 'Street',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinType2',
 'Heating',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 'KitchenQual',
 'Functiol',
 'FireplaceQu',
 'GarageType',
 'GarageFinish',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'PoolQC',
 'Fence',
 'MiscFeature',
 'SaleType',
 'SaleCondition']
```

In [38]:

```
dataset[categorical_features].head()
```

Out[38]:

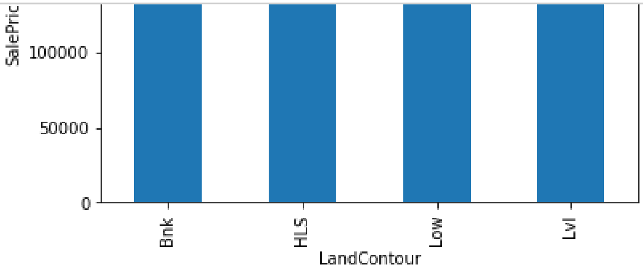
	MSZoning	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType
0	RL	Pave	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam
1	RL	Pave	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fam
2	RL	Pave	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam
3	RL	Pave	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	Norm	Norm	1Fam
4	RL	Pave	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	Norm	Norm	1Fam

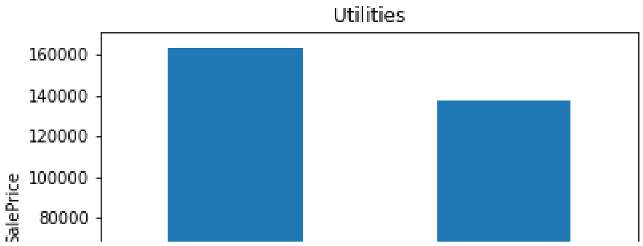
In [39]:

```
for feature in categorical_features:
    print('The feature is {} and number of categories are {}'.format(feature,len(dataset[feature].unique(
```

```
The feature is MSZoning and number of categories are 5
The feature is Street and number of categories are 2
The feature is LotShape and number of categories are 4
The feature is LandContour and number of categories are 4
The feature is Utilities and number of categories are 2
The feature is LotConfig and number of categories are 5
The feature is LandSlope and number of categories are 3
The feature is Neighborhood and number of categories are 25
The feature is Condition1 and number of categories are 9
The feature is Condition2 and number of categories are 8
The feature is BldgType and number of categories are 5
The feature is HouseStyle and number of categories are 8
The feature is RoofStyle and number of categories are 6
The feature is RoofMatl and number of categories are 8
The feature is Exterior1st and number of categories are 15
The feature is Exterior2nd and number of categories are 16
The feature is MasVnrType and number of categories are 4
The feature is ExterQual and number of categories are 4
The feature is ExterCond and number of categories are 5
The feature is Foundation and number of categories are 6
The feature is BsmtQual and number of categories are 4
The feature is BsmtCond and number of categories are 4
The feature is BsmtExposure and number of categories are 4
The feature is BsmtFinType1 and number of categories are 7
The feature is BsmtFinType2 and number of categories are 7
The feature is Heating and number of categories are 6
The feature is HeatingQC and number of categories are 5
The feature is CentralAir and number of categories are 2
The feature is Electrical and number of categories are 6
The feature is KitchenQual and number of categories are 4
The feature is Functiol and number of categories are 7
The feature is FireplaceQu and number of categories are 5
The feature is GarageType and number of categories are 6
The feature is GarageFinish and number of categories are 3
The feature is GarageQual and number of categories are 5
The feature is GarageCond and number of categories are 5
The feature is PavedDrive and number of categories are 3
The feature is PoolQC and number of categories are 4
The feature is Fence and number of categories are 5
The feature is MiscFeature and number of categories are 5
The feature is SaleType and number of categories are 9
The feature is SaleCondition and number of categories are 6
```

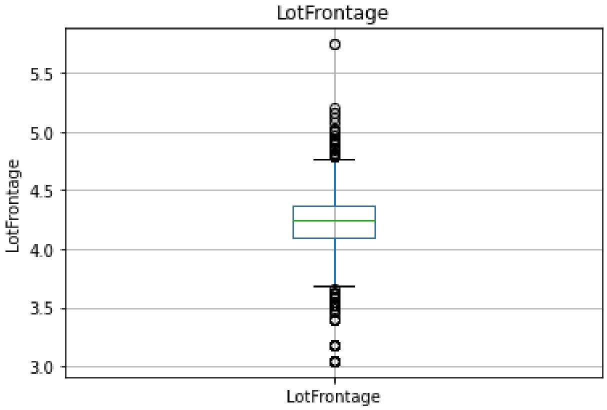
In [40]:
```python
for feature in categorical_features:
    data=dataset.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```

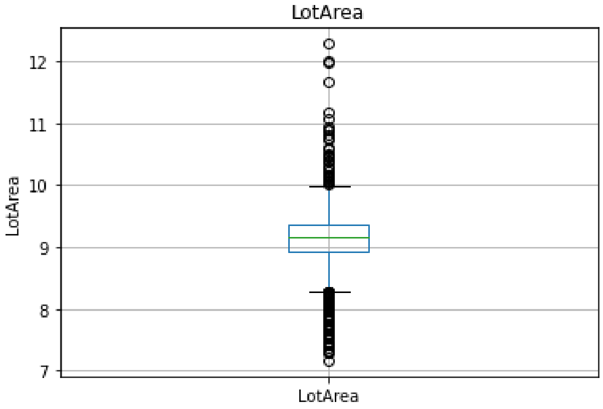


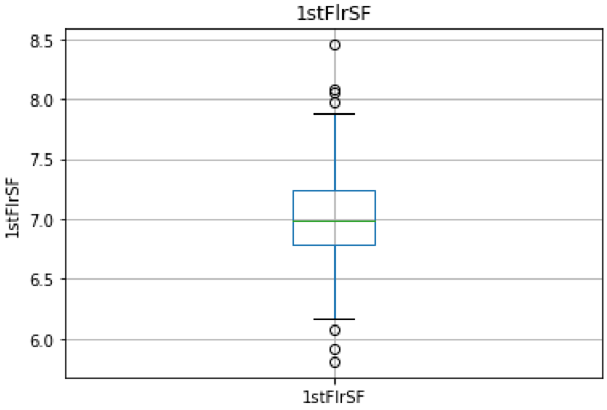


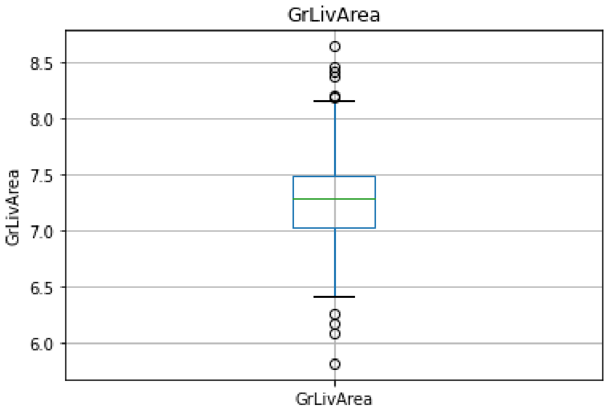
# 6. Plot box plot for the new dataset to find the variables with outliers

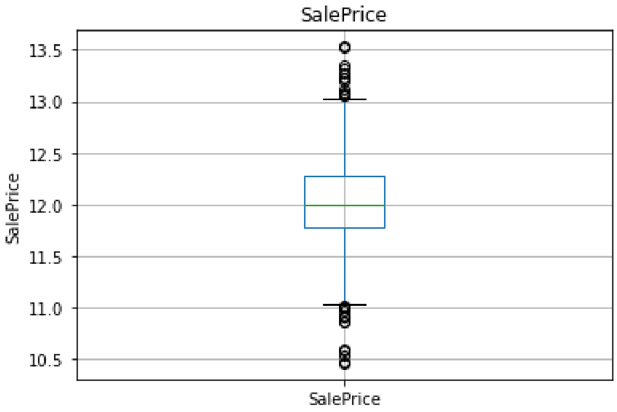
In [41]:
```
for feature in continuous_feature:
    data=dataset.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data.boxplot(column=feature)
        plt.ylabel(feature)
        plt.title(feature)
        plt.show()
```








GrLivArea
8.5
8.0
7.5
7.0
6.5
6.0
GrLivArea
GrLivArea


SalePrice
13.5
13.0
12.5
12.0
11.5
11.0
10.5
SalePrice
SalePrice

In [ ]: