In [147]:
```
w# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [148]:
```
insurancedata = pd.read_csv(r'C:\Users\alharbi\Downloads\insurance.csv')
```

In [149]:
```
insurancedata.head(5)
```

Out[149]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	NaN	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [150]:
```
insurancedata.shape
```

Out[150]: (1338, 7)

In [151]:
```
insurancedata.describe()
```

Out[151]:

	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

In [228]:
```
plt.figure(figsize=(10, 6))
sns.regplot(x='age', y='charges', data=insurancedata)
plt.title('Regression Plot of charges by age')
```

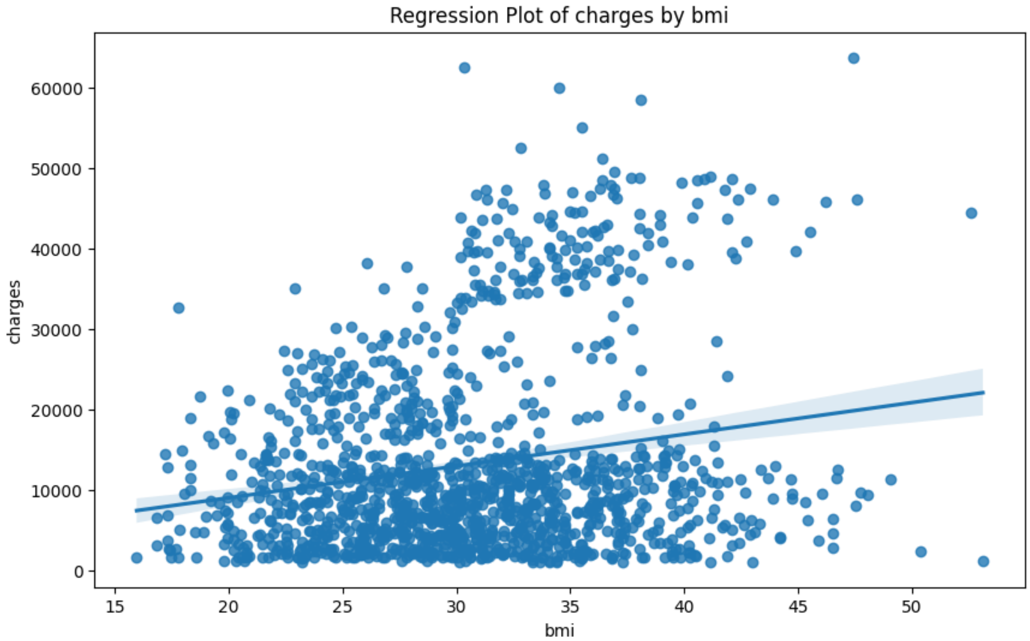
Out[228]: Text(0.5, 1.0, 'Regression Plot of charges by age')

In [225]:
```
from scipy import stats
pearson_coef_age, p_value_age = stats.pearsonr(insurancedata['age'], insurancedata['charges'])
print("The Pearson Correlation Coefficient between age and charges is", pearson_coef_age, "with a P-value of P=", p_value_age)
```

The Pearson Correlation Coefficient between age and charges is 0.29900819333064754 with a P-value of P= 4.886693331718298e-29

In [229]:
```
plt.figure(figsize=(10, 6))
sns.regplot(x='bmi', y='charges', data=insurancedata)
plt.title('Regression Plot of charges by bmi')
```

Out[229]: `Text(0.5, 1.0, 'Regression Plot of charges by bmi')`

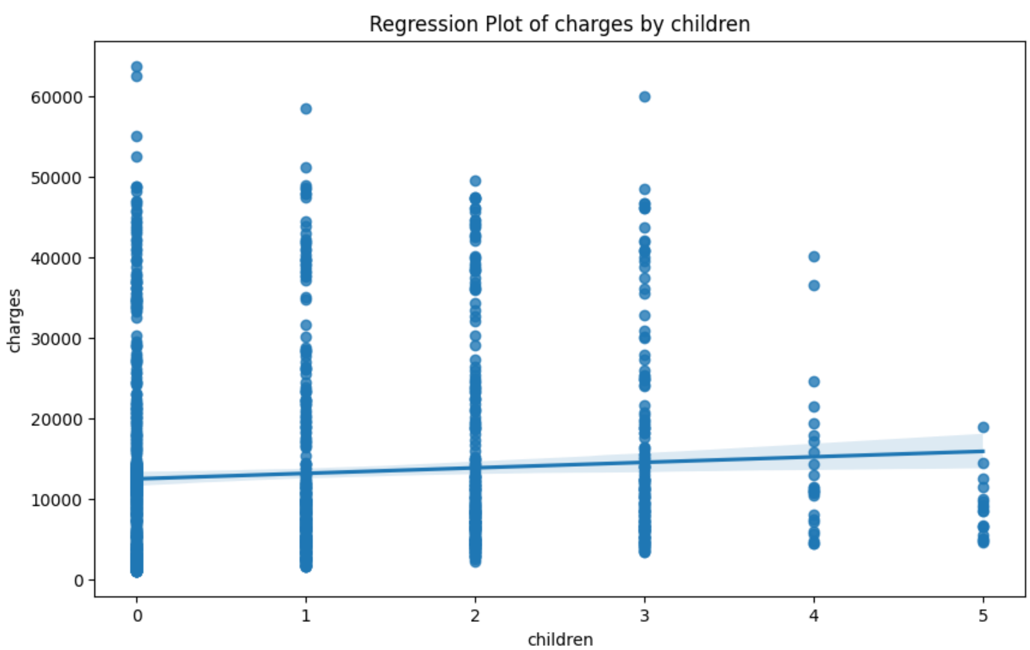


In [90]:
```
pearson_coef_bmi, p_value_bmi = stats.pearsonr(insurancedata['bmi'], insurancedata['charges'])
print("The Pearson Correlation Coefficient between bmi and charges is", pearson_coef_bmi, "with a P-value of P=", p_value_bmi)
```

```
The Pearson Correlation Coefficient between bmi and charges is 0.19834096883362878 with a P-value of P= 2.459085535116766e-13
```

In [230]:
```
plt.figure(figsize=(10, 6))
sns.regplot(x='children', y='charges', data=insurancedata)
plt.title('Regression Plot of charges by children')
```

Out[230]: `Text(0.5, 1.0, 'Regression Plot of charges by children')`

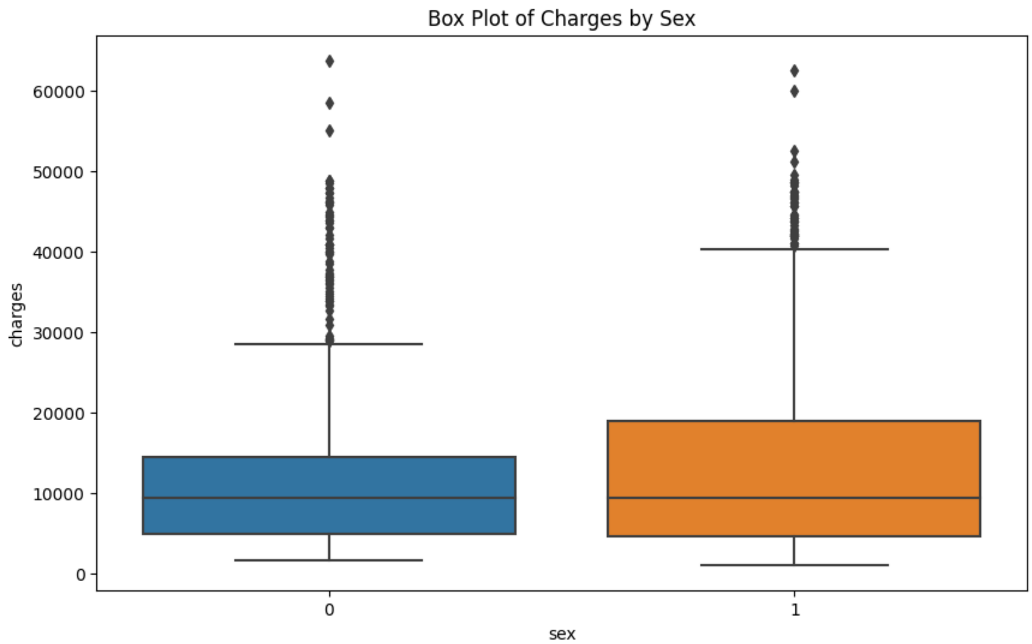


In [231]:
```
pearson_coef_children, p_value_children = stats.pearsonr(insurancedata['children'], insurancedata['charges'])
print("The Pearson Correlation Coefficient between children and charges is", pearson_coef_children, "with a P-value of P=", p_value_children)
```

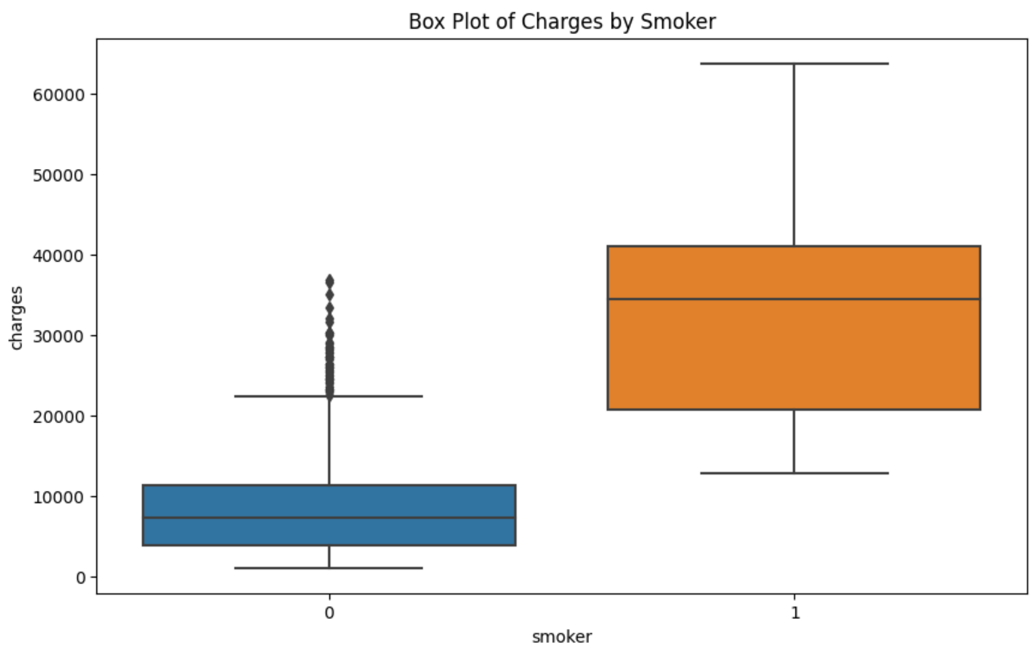
```
The Pearson Correlation Coefficient between children and charges is 0.06799822684790464 with a P-value of P= 0.012852128520136508
```

In [232]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

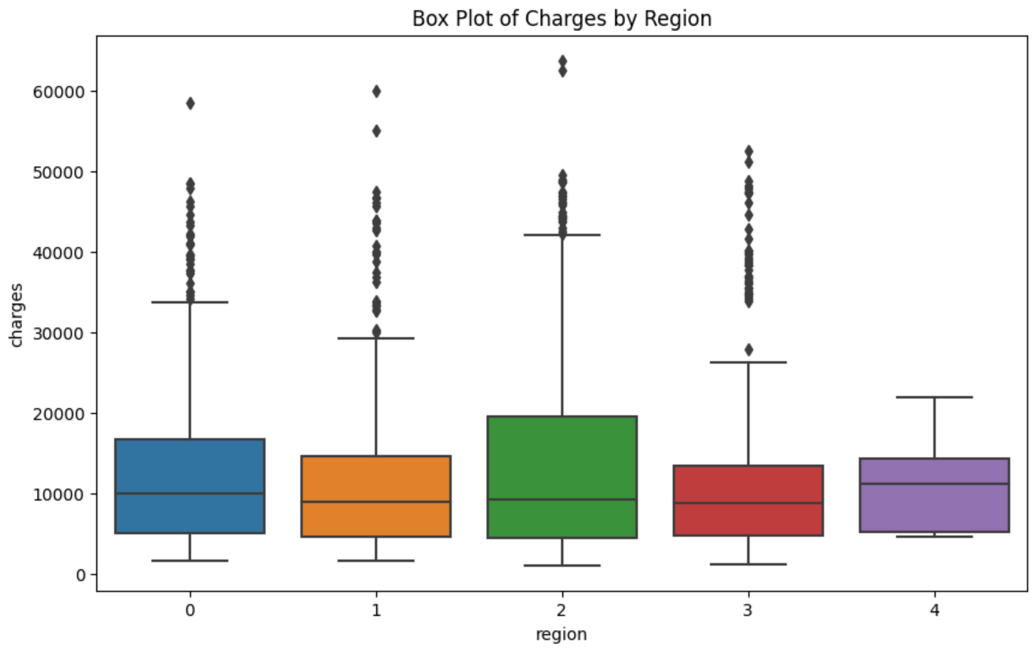
In [233]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='sex', y='charges', data=insurancedata)
plt.title('Box Plot of Charges by Sex')
plt.show()
```



In [234]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='smoker', y='charges', data=insurancedata)
plt.title('Box Plot of Charges by Smoker')
plt.show()
```



In [235]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='region', y='charges', data=insurancedata)
plt.title('Box Plot of Charges by Region')
plt.show()
```



In [236]:
```
data = insurancedata.drop(['region', 'sex', 'smoker'], axis=1)
```

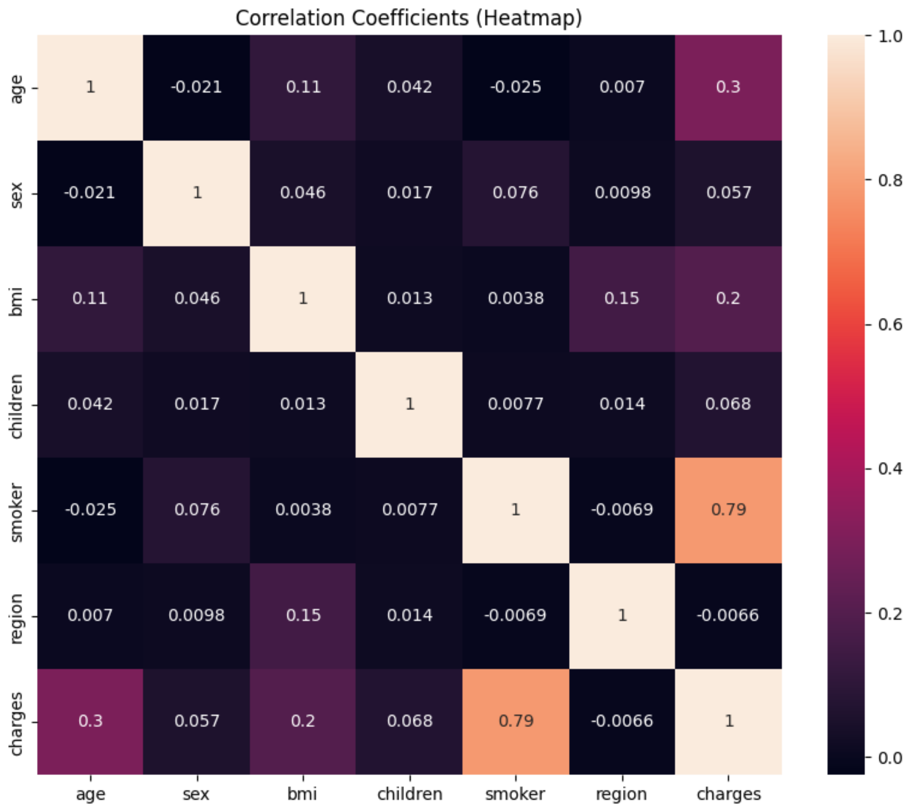
In [237]:
```
data.head(5)
```

Out[237]:

	age	bmi	children	charges
0	19	27.900	0	16884.92400
1	18	33.770	1	1725.55230
2	28	33.000	3	4449.46200
3	33	22.705	0	21984.47061
4	32	28.880	0	3866.85520

In [238]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [226]:
```python
plt.figure(figsize=(10, 8))
sns.heatmap(data.corr(), annot=True)
plt.title('Correlation Coefficients (Heatmap)')
plt.show()
```



In [165]:
```python
insurancedata.head(5)
```

Out[165]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	NaN	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [166]:
```python
insurancedata.shape
```

Out[166]: (1338, 7)

In [170]:
```python
# Check for missing values
print(insurancedata.isnull().sum())
```

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      5
charges     0
dtype: int64
```

In [173]:
```python
data=insurancedata.dropna()
```

In [175]:
```python
# Check for missing values
print(data.isnull().sum())
```

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [176]:
```
data.dtypes
```
Out[176]:
```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

In [177]:
```
from sklearn.preprocessing import LabelEncoder
# Label encoding
label_encoder = LabelEncoder()
insurancedata['sex'] = label_encoder.fit_transform(insurancedata['sex'])
insurancedata['smoker'] = label_encoder.fit_transform(insurancedata['smoker'])
insurancedata['region'] = label_encoder.fit_transform(insurancedata['region'])
```

In [178]:
```
data.head(10)
```
Out[178]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.90	0	yes	southwest	16884.92400
1	18	male	33.77	1	no	southeast	1725.55230
2	28	male	33.00	3	no	southeast	4449.46200
4	32	male	28.88	0	no	northwest	3866.85520
5	31	female	25.74	0	no	southeast	3756.62160
6	46	female	33.44	1	no	southeast	8240.58960
7	37	female	27.74	3	no	northwest	7281.50560
8	37	male	29.83	2	no	northeast	6406.41070
9	60	female	25.84	0	no	northwest	28923.13692
10	25	male	26.22	0	no	northeast	2721.32080

In [243]:
```
import scipy.stats as stats
data = stats.zscore(insurancedata)
```

In [244]:
```
data
```
Out[244]:

	age	sex	bmi	children	smoker	region	charges
0	-1.438764	-1.010519	-0.453320	-0.908614	1.970587	1.324162	0.298584
1	-1.509965	0.989591	0.509621	-0.078767	-0.507463	0.425719	-0.953689
2	-0.797954	0.989591	0.383307	1.580926	-0.507463	0.425719	-0.728675
3	-0.441948	0.989591	-1.305531	-0.908614	-0.507463	2.222604	0.719843
4	-0.513149	0.989591	-0.292556	-0.908614	-0.507463	-0.472723	-0.776802
...	...	...	...	...	...	...	...
1333	0.768473	0.989591	0.050297	1.580926	-0.507463	-0.472723	-0.220551
1334	-1.509965	-1.010519	0.206139	-0.908614	-0.507463	-1.371165	-0.914002
1335	-1.509965	-1.010519	1.014878	-0.908614	-0.507463	0.425719	-0.961596
1336	-1.296362	-1.010519	-0.797813	-0.908614	-0.507463	1.324162	-0.930362
1337	1.551686	-1.010519	-0.261388	-0.908614	1.970587	-0.472723	1.311053

1338 rows × 7 columns

In [259]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import seaborn as sns
import matplotlib.pyplot as plt
```

In [260]:
```
X = insurancedata.drop(['charges'], axis=1)
y = insurancedata['charges']
```

In [261]:
```
# Split the data into training and testing datasets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

In [262]:
```
# Multiple Linear Regression
lr = LinearRegression()
lr.fit(X_train, y_train)
```
Out[262]:
```
▾ LinearRegression
LinearRegression()
```

In [263]:
```python
# Making predictions
y_pred_lr = lr.predict(X_test)
```

In [264]:
```python
# Evaluate the model
mse_lr = mean_squared_error(y_test, y_pred_lr)
print('MSE for Linear Regression: ', mse_lr)
```

```
MSE for Linear Regression:  31882830.66682946
```

In [265]:
```python
# Random Forest Regressor
rf = RandomForestRegressor()
rf.fit(X_train, y_train)
```

Out[265]:
```
▾ RandomForestRegressor
RandomForestRegressor()
```

In [266]:
```python
# Making predictions
y_pred_rf = rf.predict(X_test)
```

In [267]:
```python
# Evaluate the model
mse_rf = mean_squared_error(y_test, y_pred_rf)
print('MSE for Random Forest: ', mse_rf)
```

```
MSE for Random Forest:  19699405.91022278
```

In [268]:
```python
# LASSO Regression
lasso = Lasso()
lasso.fit(X_train, y_train)
```

Out[268]:
```
▾ Lasso
Lasso()
```

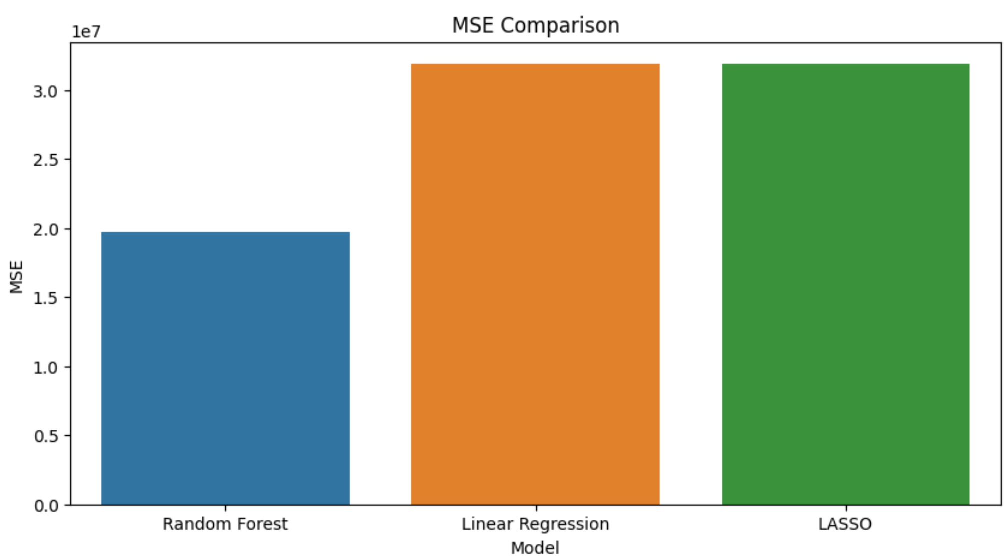
In [269]:
```python
# Making predictions
y_pred_lasso = lasso.predict(X_test)
```

In [270]:
```python
# Evaluate the model
mse_lasso = mean_squared_error(y_test, y_pred_lasso)
print('MSE for LASSO is: ', mse_lasso)
```

```
MSE for LASSO is:  31887412.635049313
```

In [271]:
```python
# Compare performance of models using bar plot
mse_scores = [('Linear Regression', mse_lr), ('Random Forest', mse_rf), ('LASSO', mse_lasso)]
mse_df = pd.DataFrame(data = mse_scores, columns=['Model', 'MSE Score'])
mse_df.sort_values(by='MSE Score', ascending=True, inplace=True)
```

In [272]:
```python
f, axe = plt.subplots(1,1, figsize=(10,5))
sns.barplot(x = mse_df['Model'], y = mse_df['MSE Score'], ax = axe)
plt.title('MSE Comparison')
plt.xlabel('Model')
plt.ylabel('MSE')
plt.show()
```



In [ ]:

In [ ]: