In [1]:

```
#importing essential libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [4]:

```
#reading the csv file
amazon_data = pd.read_csv("Amazon.csv")
```

In [5]:

```
amazon_data
```

Out[5]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	1997-05-15	2.437500	2.500000	1.927083	1.958333	1.958333	72156000
1	1997-05-16	1.968750	1.979167	1.708333	1.729167	1.729167	14700000
2	1997-05-19	1.760417	1.770833	1.625000	1.708333	1.708333	6106800
3	1997-05-20	1.729167	1.750000	1.635417	1.635417	1.635417	5467200
4	1997-05-21	1.635417	1.645833	1.375000	1.427083	1.427083	18853200
...	...	...	...	...	...	...	...
6150	2021-10-21	3414.250000	3440.280029	3403.000000	3435.010010	3435.010010	1881400
6151	2021-10-22	3421.000000	3429.840088	3331.300049	3335.550049	3335.550049	3133800
6152	2021-10-25	3335.000000	3347.800049	3297.699951	3320.370117	3320.370117	2226000
6153	2021-10-26	3349.510010	3416.120117	3343.979980	3376.070068	3376.070068	2693700
6154	2021-10-27	3388.000000	3412.000000	3371.453369	3396.189941	3396.189941	1080291

6155 rows × 7 columns

In [6]:

```
# reading first five rows

amazon_data.head()
```

Out[6]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	1997-05-15	2.437500	2.500000	1.927083	1.958333	1.958333	72156000
1	1997-05-16	1.968750	1.979167	1.708333	1.729167	1.729167	14700000
2	1997-05-19	1.760417	1.770833	1.625000	1.708333	1.708333	6106800
3	1997-05-20	1.729167	1.750000	1.635417	1.635417	1.635417	5467200
4	1997-05-21	1.635417	1.645833	1.375000	1.427083	1.427083	18853200

In [7]:

```
# reading last five rows

amazon_data.tail()
```

Out[7]:

	Date	Open	High	Low	Close	Adj Close	Volume
6150	2021-10-21	3414.25000	3440.280029	3403.000000	3435.010010	3435.010010	1881400
6151	2021-10-22	3421.00000	3429.840088	3331.300049	3335.550049	3335.550049	3133800
6152	2021-10-25	3335.00000	3347.800049	3297.699951	3320.370117	3320.370117	2226000
6153	2021-10-26	3349.51001	3416.120117	3343.979980	3376.070068	3376.070068	2693700
6154	2021-10-27	3388.00000	3412.000000	3371.453369	3396.189941	3396.189941	1080291

In [8]:

```
amazon_data.shape
```

Out[8]:

```
(6155, 7)
```

In [9]:

```
amazon_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6155 entries, 0 to 6154
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       6155 non-null   object 
 1   Open       6155 non-null   float64
 2   High       6155 non-null   float64
 3   Low        6155 non-null   float64
 4   Close      6155 non-null   float64
 5   Adj Close  6155 non-null   float64
 6   Volume     6155 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 336.7+ KB
```

In [10]:

```
amazon_data.describe()
```

Out[10]:

	Open	High	Low	Close	Adj Close	Volume
count	6155.000000	6155.000000	6155.000000	6155.000000	6155.000000	6.155000e+03
mean	520.556302	526.216132	514.277282	520.429832	520.429832	7.329010e+06
std	857.161696	865.821041	847.270905	856.668492	856.668492	7.149521e+06
min	1.406250	1.447917	1.312500	1.395833	1.395833	4.872000e+05
25%	38.750000	39.514999	38.104999	38.821251	38.821251	3.579350e+06
50%	92.669998	94.190002	90.750000	92.639999	92.639999	5.470000e+06
75%	528.949982	535.304993	521.950012	529.450012	529.450012	8.294950e+06
max	3744.000000	3773.080078	3696.790039	3731.409912	3731.409912	1.043292e+08

In [11]:

```
amazon_data.columns
```

Out[11]:

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='objec
t')
```

In [12]:

```
amazon_data.isnull().sum()
```

Out[12]:

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

In [13]:

```python
#dropping 'adj close' column
amazon_data = amazon_data.drop(columns = ['Adj Close'])
```

In [14]:

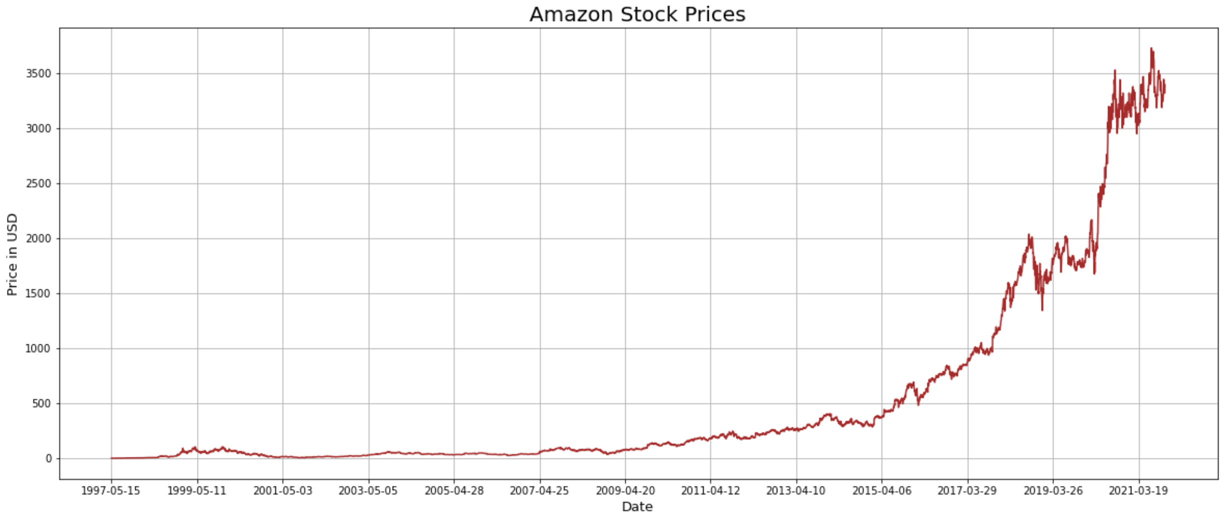
```python
amazon_data.head()
```

Out[14]:

	Date	Open	High	Low	Close	Volume
0	1997-05-15	2.437500	2.500000	1.927083	1.958333	72156000
1	1997-05-16	1.968750	1.979167	1.708333	1.729167	14700000
2	1997-05-19	1.760417	1.770833	1.625000	1.708333	6106800
3	1997-05-20	1.729167	1.750000	1.635417	1.635417	5467200
4	1997-05-21	1.635417	1.645833	1.375000	1.427083	18853200

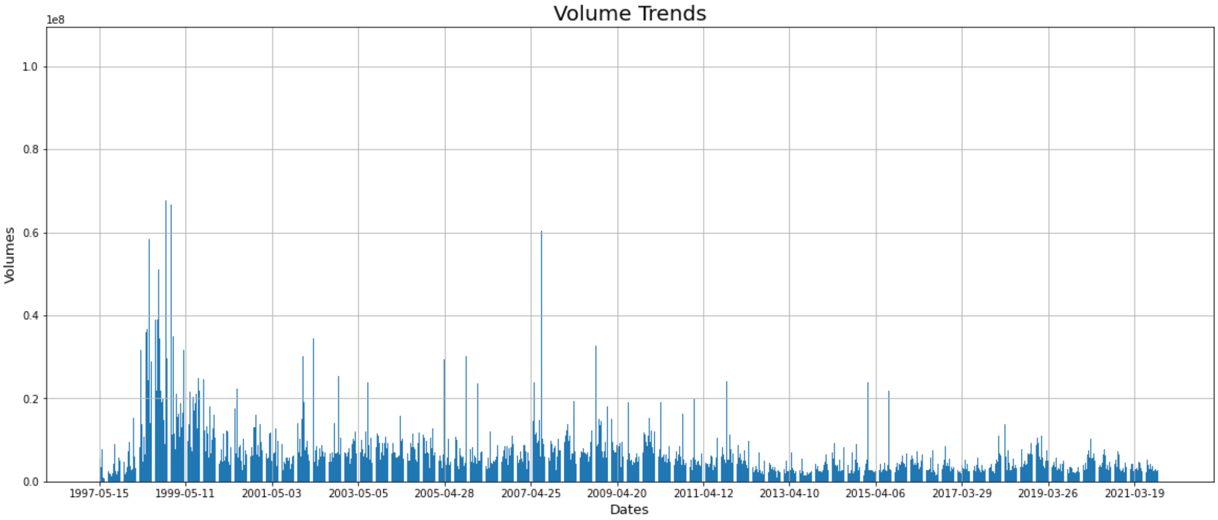
In [15]:

```python
fig,ax = plt.subplots(figsize=(20,8))
ax.plot(amazon_data['Date'],amazon_data['Close'], color='Brown')
ax.xaxis.set_major_locator(plt.MaxNLocator(15))
ax.set_xlabel('Date',fontsize='13')
ax.set_ylabel('Price in USD',fontsize='13')
plt.title('Amazon Stock Prices',fontsize='20')
plt.grid()
plt.show()
```



In [16]:

```python
fig,ax=plt.subplots(figsize=(20,8))
ax.bar(amazon_data['Date'],amazon_data['Volume'])
ax.xaxis.set_major_locator(plt.MaxNLocator(15))
ax.set_xlabel('Dates',fontsize='13')
ax.set_ylabel('Volumes',fontsize='13')
plt.title('Volume Trends',fontsize='20')
plt.grid()
plt.show()
```

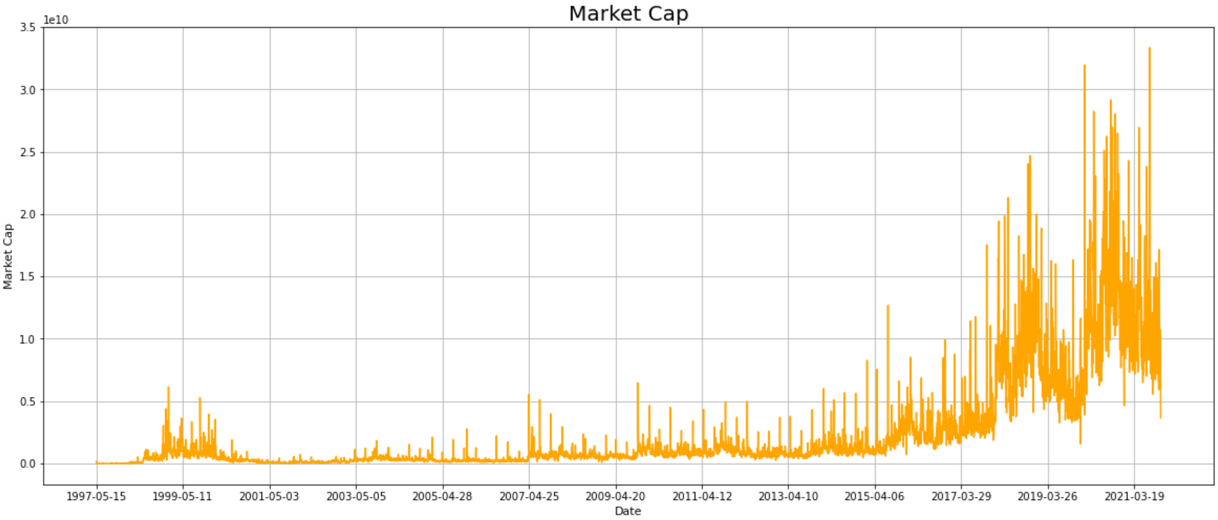


In [17]:

```
amazon_data['Market Cap'] = amazon_data['Open']*amazon_data['Volume']
```

In [18]:

```
fig,ax = plt.subplots(figsize=(20,8))
ax.plot(amazon_data['Date'],amazon_data['Market Cap'],color='Orange')
ax.xaxis.set_major_locator(plt.MaxNLocator(15))
ax.set_xlabel('Date',fontsize='11')
ax.set_ylabel('Market Cap',fontsize='11')
plt.title('Market Cap',fontsize='20')
plt.grid()
plt.show()
```



In [19]:

```
amazon_data.iloc[amazon_data['Market Cap'].argmax()]
```

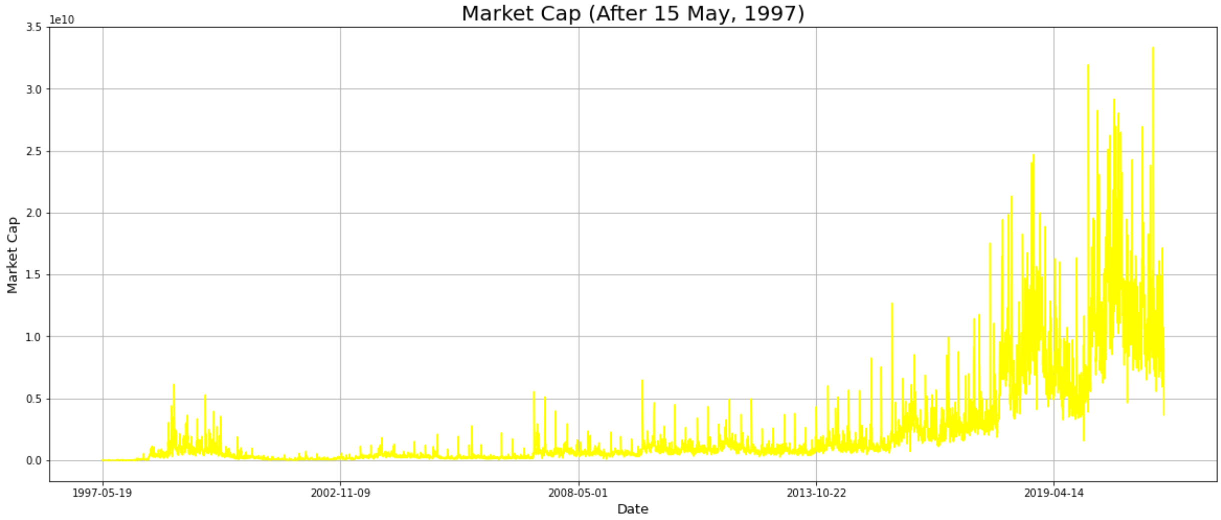
Out[19]:

```
Date                    2021-07-30
Open                   3347.949951
High                   3368.139893
Low                     3306.97998
Close                  3327.590088
Volume                     9957100
Market Cap       33335872457.1021
Name: 6092, dtype: object
```

In [21]:

```
ohlc = amazon_data[(amazon_data['Date'] > '1997-05-15')]
ohlc= ohlc.loc[:,['Date', 'Open', 'High', 'Low', 'Close', 'Volume',
'Market Cap']]
ohlc['Date'] = pd.to_datetime(ohlc['Date'],format = '%Y-%m-%d')
```

```
fig,ax = plt.subplots(figsize=(20,8))
ax.plot(ohlc['Date'],ohlc['Market Cap'], color = 'Yellow')
ax.xaxis.set_major_locator(plt.MaxNLocator(5))
ax.set_xlabel('Date',fontsize='13')
ax.set_ylabel('Market Cap',fontsize='13')
plt.title('Market Cap (After 15 May, 1997)',fontsize='20')
plt.grid()
plt.show()
```

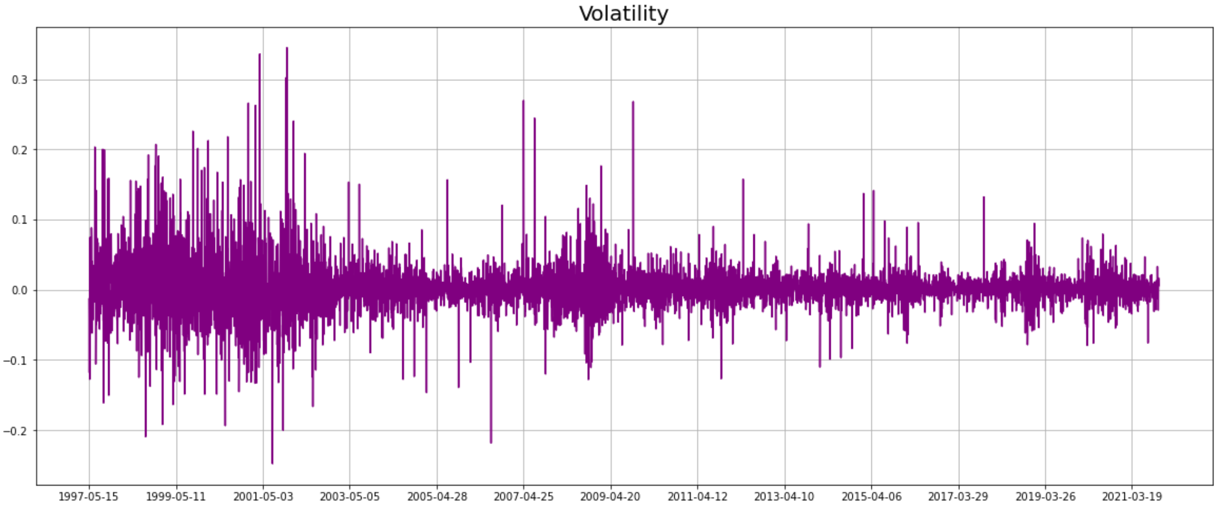


In [22]:

```
amazon_data['vol'] = (amazon_data['Close']/amazon_data['Close'].shift(1))-1
```

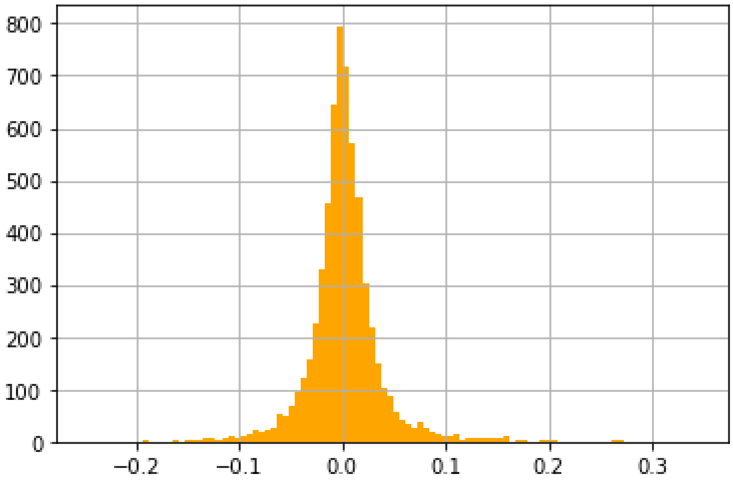
In [23]:

```
fig,ax = plt.subplots(figsize=(20,8))
ax.plot(amazon_data['Date'],amazon_data['vol'],color='Purple')
ax.xaxis.set_major_locator(plt.MaxNLocator(15))
plt.title('Volatility',fontsize='20')
plt.grid()
plt.show()
```



In [24]:

```
amazon_data['vol'].hist(bins=100,color='Orange');
```

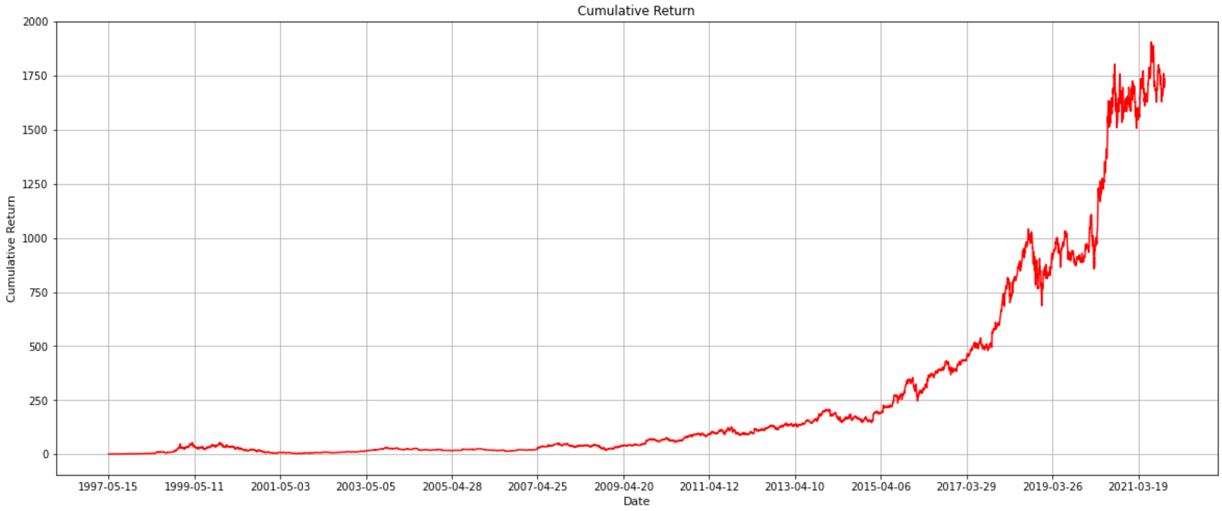


In [25]:

```
amazon_data['Cumulative Return'] = (1 + amazon_data['vol']).cumprod()
```

In [26]:

```
fig, ax = plt.subplots(figsize=(20,8))
ax.plot(amazon_data['Date'], amazon_data['Cumulative Return'], color='red')
ax.xaxis.set_major_locator(plt.MaxNLocator(15))
ax.set_xlabel('Date', fontsize='11')
ax.set_ylabel('Cumulative Return', fontsize='11')
plt.title('Cumulative Return')
plt.grid()
plt.show()
```



In [28]:

```
amazon_data.iloc[amazon_data['Cumulative Return'].argmax()]
```

Out[28]:

```
Date                          2021-07-08
Open                         3643.560059
High                          3759.98999
Low                          3621.120117
Close                        3731.409912
Volume                           5180600
Market Cap           18875827241.655399
vol                             0.009422
Cumulative Return             1905.40113
Name: 6076, dtype: object
```

In [31]:

```
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM
import math
```

In [32]:

```
amazon_data['Date'] = pd.to_datetime(amazon_data['Date'])
amazon_data.set_index('Date',inplace=True)
```

In [33]:

```
data = amazon_data.filter(['Close'])
dataset = data.values
training_data_len = math.ceil(len(dataset)*.8)
training_data_len
```

Out[33]:

```
4924
```

In [34]:

```
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(dataset)
scaled_data
```

Out[34]:

```
array([[1.50803720e-04],
       [8.93653463e-05],
       [8.37798446e-05],
       ...,
       [8.89802079e-01],
       [9.04734986e-01],
       [9.10129033e-01]])
```

In [35]:

```
train_data = scaled_data[0:training_data_len, :]
x_train = []
y_train = []
for i in range(60,len(train_data)):
    x_train.append(train_data[i-60:i,0])
    y_train.append(train_data[i,0])
    if i<=60:
        print(x_train)
        print(y_train)
        print()
```

```
[array([1.50803720e-04, 8.93653463e-05, 8.37798446e-05, 6.42313929e-05,
       8.37798446e-06, 0.00000000e+00, 2.79267042e-05, 5.02679068e-05,
       3.63046887e-05, 2.93229456e-05, 2.79267042e-05, 3.07194551e-05,
       2.23414706e-05, 5.58550171e-06, 3.90974396e-05, 6.98166266e-05,
       7.81946110e-05, 5.02679068e-05, 3.90974396e-05, 5.58534085e-05,
       5.02679068e-05, 4.74754240e-05, 2.93229456e-05, 3.07194551e-05,
       3.07194551e-05, 3.49084473e-05, 2.79267042e-05, 3.07194551e-05,
       3.07194551e-05, 3.07194551e-05, 2.51339534e-05, 3.90974396e-05,
       3.21156965e-05, 5.16644163e-05, 1.38236744e-04, 1.61974456e-04,
       2.42961549e-04, 2.48547051e-04, 3.12778176e-04, 2.40169067e-04,
       1.98279144e-04, 2.79266238e-04, 2.51339534e-04, 2.20620347e-04,
       2.03864378e-04, 2.10846121e-04, 1.98279144e-04, 2.40169067e-04,
       2.31791082e-04, 2.23413098e-04, 2.48547051e-04, 2.90436973e-04,
       2.82058989e-04, 2.68095503e-04, 2.73681005e-04, 2.45754300e-04,
       2.17827596e-04, 2.28998331e-04, 2.09449612e-04, 2.40169067e-04])]
[0.00024994329250626934]
```

In [36]:

```
x_train,y_train = np.array(x_train), np.array(y_train)
x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
x_train.shape
```

Out[36]:

```
(4864, 60, 1)
```

In [37]:

```
model =Sequential()
model.add(LSTM(64,return_sequences=True, input_shape=(x_train.shape[1],1)))
model.add(LSTM(64, return_sequences= False))
model.add(Dense(32))
model.add(Dense(1))
```

In [38]:

```
model.compile(optimizer='adam', loss='mean_squared_error')
```

In [39]:

```
model.fit(x_train,y_train, batch_size=1, epochs=10)
```

```
Epoch 1/10
4864/4864 [==============================] - 256s 51ms/step - loss: 6.2195e-05
Epoch 2/10
4864/4864 [==============================] - 251s 52ms/step - loss: 2.4151e-05
Epoch 3/10
4864/4864 [==============================] - 258s 53ms/step - loss: 1.8679e-05
Epoch 4/10
4864/4864 [==============================] - 257s 53ms/step - loss: 1.6038e-05
Epoch 5/10
4864/4864 [==============================] - 262s 54ms/step - loss: 1.1240e-05
Epoch 6/10
4864/4864 [==============================] - 251s 52ms/step - loss: 1.0635e-05
Epoch 7/10
4864/4864 [==============================] - 244s 50ms/step - loss: 9.1616e-06
Epoch 8/10
4864/4864 [==============================] - 246s 51ms/step - loss: 1.0213e-05
Epoch 9/10
4864/4864 [==============================] - 249s 51ms/step - loss: 8.5540e-06
Epoch 10/10
4864/4864 [==============================] - 249s 51ms/step - loss: 8.7978e-06
```

Out[39]:

```
<keras.callbacks.History at 0x1e52b810850>
```

In [48]:

```
test_data= scaled_data[training_data_len-60:, :]
x_test = []

y_test = dataset[training_data_len:,:]
for i in range(60,len(test_data)):
    x_test.append(test_data[i-60:i,0])
```

In [49]:

```
x_test = np.array(x_test)
```

In [50]:

```
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1],1))
x_test.shape
```

Out[50]:

```
(1231, 60, 1)
```

In [51]:

```
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)
```

In [52]:

```
rmse = np.sqrt(np.mean(predictions - y_test)**2)
```

```
rmse
```

Out[52]:

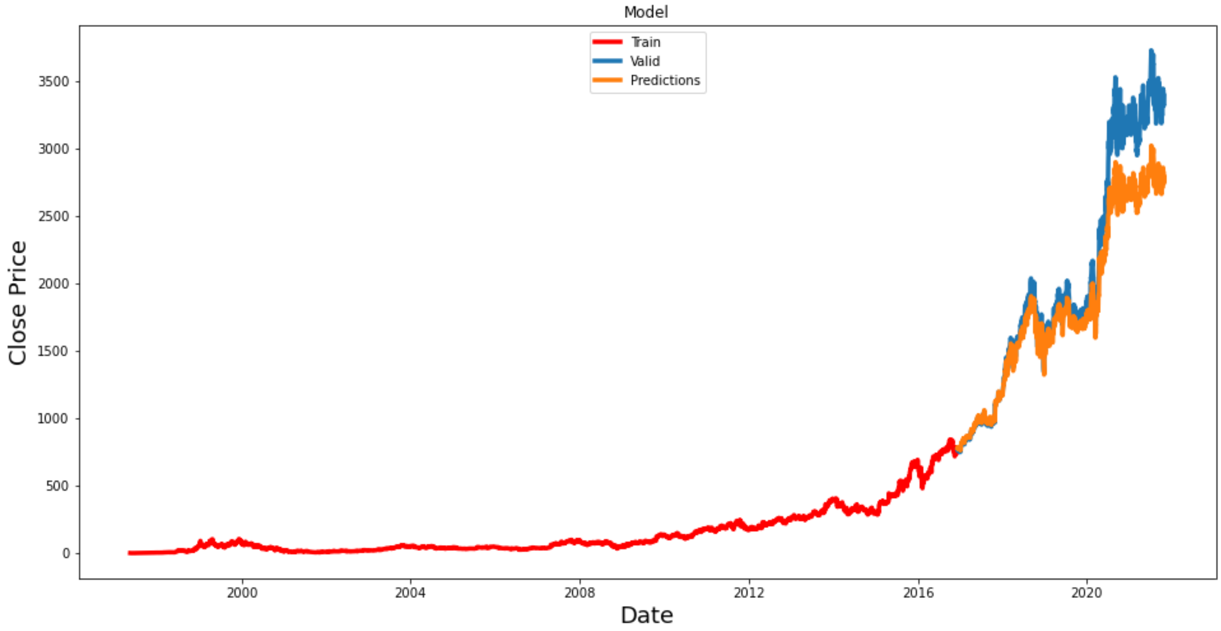
```
191.2957627177536
```

In [53]:

```
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price' ,fontsize=18)
plt.plot(train['Close'],linewidth=3.5,color='red')
plt.plot(valid[['Close','Predictions']],linewidth=3.5)
plt.legend(['Train','Valid','Predictions'], loc='upper center')
```

Out[53]:

```
<matplotlib.legend.Legend at 0x1e52e9ebdc0>
```



In [54]:

```
valid
```

Out[54]:

	Close	Predictions
Date		
2016-12-07	770.419983	782.066650
2016-12-08	767.330017	786.743958
2016-12-09	768.659973	782.235535
2016-12-12	760.119995	782.452393
2016-12-13	774.340027	773.541809
...	...	...
2021-10-21	3435.010010	2818.171875
2021-10-22	3335.550049	2829.681396
2021-10-25	3320.370117	2757.317139
2021-10-26	3376.070068	2743.766846

	Close	Predictions
Date		
2021-10-27	3396.189941	2793.502930

1231 rows × 2 columns

In [55]:

```
amazon_quote = pd.read_csv("Amazon.csv")
new_amazon_data = amazon_quote.filter(['Close'])
last_60_days = new_amazon_data[-60:].values
last_60_days_scaled = scaler.transform(last_60_days)
X_test = []
X_test.append(last_60_days_scaled)
X_test = np.array(X_test)
X_test = np.reshape(X_test,(X_test.shape[0], X_test.shape[1],1))
pred_price= model.predict(X_test)
pred_price = scaler.inverse_transform(pred_price)
pred_price
```

Out[55]:

```
array([[2814.3782]], dtype=float32)
```

In [ ]: