In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Import necessary libraries:NumPy, Pandas, Matplotlib, and Seaborn.

In [2]:
```
pip install seaborn
```

```
Collecting seaborn
  Downloading seaborn-0.12.2-py3-none-any.whl (293 kB)
     ------------------------------------ 293.3/293.3 kB 566.1 kB/s eta 0:00:
00
Requirement already satisfied: numpy!=1.24.0,>=1.17 in c:\users\lenovo\anacon
da3\envs\rstudio\lib\site-packages (from seaborn) (1.21.6)
Requirement already satisfied: pandas>=0.25 in c:\users\lenovo\anaconda3\envs
\rstudio\lib\site-packages (from seaborn) (1.3.5)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in c:\users\lenovo\ana
conda3\envs\rstudio\lib\site-packages (from seaborn) (3.5.3)
Requirement already satisfied: typing_extensions in c:\users\lenovo\anaconda3
\envs\rstudio\lib\site-packages (from seaborn) (3.10.0.2)
Requirement already satisfied: cycler>=0.10 in c:\users\lenovo\anaconda3\envs
\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\lenovo\anaconda3
\envs\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (4.3
8.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\lenovo\anaconda3
\envs\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (1.4.
4)
Requirement already satisfied: packaging>=20.0 in c:\users\lenovo\anaconda3\e
nvs\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (21.3)
Requirement already satisfied: pillow>=6.2.0 in c:\users\lenovo\anaconda3\env
s\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (9.5.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\lenovo\anaconda3
\envs\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (3.0.
4)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\lenovo\anacon
da3\envs\rstudio\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(2.8.2)
Requirement already satisfied: pytz>=2017.3 in c:\users\lenovo\anaconda3\envs
\rstudio\lib\site-packages (from pandas>=0.25->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\lenovo\anaconda3\envs\rst
udio\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.1->s
eaborn) (1.16.0)
Installing collected packages: seaborn
Successfully installed seaborn-0.12.2
Note: you may need to restart the kernel to use updated packages.
```

Load The Data

In [4]:
```
df = pd.read_csv('database.csv')
```

Explore the data:

In [5]:
```
print(df.head())
```

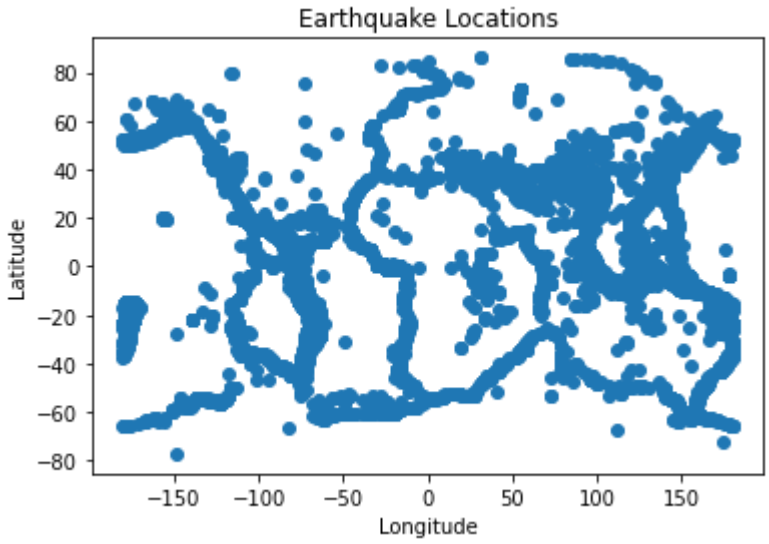
```
         Date      Time  Latitude  Longitude        Type  Depth  Depth Error
\
0  01/02/1965  13:44:18    19.246    145.616  Earthquake  131.6          NaN
1  01/04/1965  11:29:49     1.863    127.352  Earthquake   80.0          NaN
2  01/05/1965  18:05:58   -20.579   -173.972  Earthquake   20.0          NaN
3  01/08/1965  18:49:43   -59.076    -23.557  Earthquake   15.0          NaN
4  01/09/1965  13:32:50    11.938    126.427  Earthquake   15.0          NaN

   Depth Seismic Stations  Magnitude Magnitude Type  ...  \
0                     NaN        6.0             MW  ...
1                     NaN        5.8             MW  ...
2                     NaN        6.2             MW  ...
3                     NaN        5.8             MW  ...
4                     NaN        5.8             MW  ...

   Magnitude Seismic Stations  Azimuthal Gap  Horizontal Distance  \
0                         NaN            NaN                  NaN
1                         NaN            NaN                  NaN
2                         NaN            NaN                  NaN
3                         NaN            NaN                  NaN
4                         NaN            NaN                  NaN

   Horizontal Error  Root Mean Square            ID  Source Location Source
\
0               NaN               NaN  ISCGEM860706  ISCGEM          ISCGEM
1               NaN               NaN  ISCGEM860737  ISCGEM          ISCGEM
2               NaN               NaN  ISCGEM860762  ISCGEM          ISCGEM
3               NaN               NaN  ISCGEM860856  ISCGEM          ISCGEM
4               NaN               NaN  ISCGEM860890  ISCGEM          ISCGEM

  Magnitude Source     Status
0           ISCGEM  Automatic
1           ISCGEM  Automatic
2           ISCGEM  Automatic
3           ISCGEM  Automatic
4           ISCGEM  Automatic

[5 rows x 21 columns]
```

Clean the data

In [7]:
```
df_clean = df.dropna()
```

Visualize the data:

In [9]:
```
plt.scatter(df['Longitude'], df['Latitude'])
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.title('Earthquake Locations')
plt.show()
```



Compute the correlation

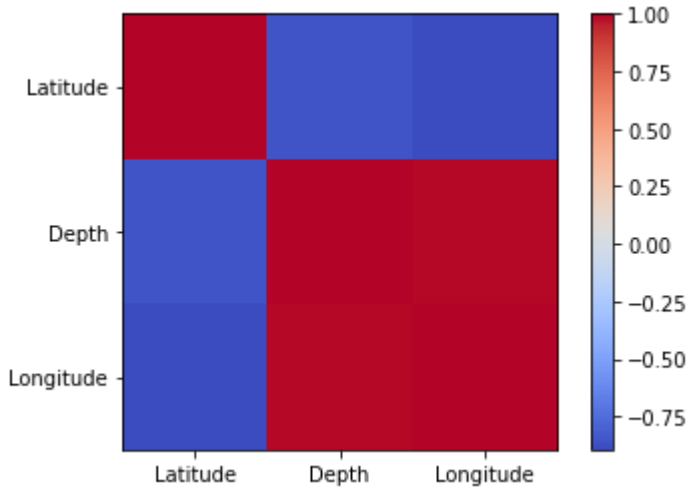
In [12]:
```
corr_matrix = df_clean[['Latitude', 'Depth', 'Longitude']].corr()
```

In [13]:
```
print(corr_matrix)
```

```
           Latitude     Depth  Longitude
Latitude   1.000000 -0.849861  -0.893475
Depth     -0.849861  1.000000   0.989156
Longitude -0.893475  0.989156   1.000000
```

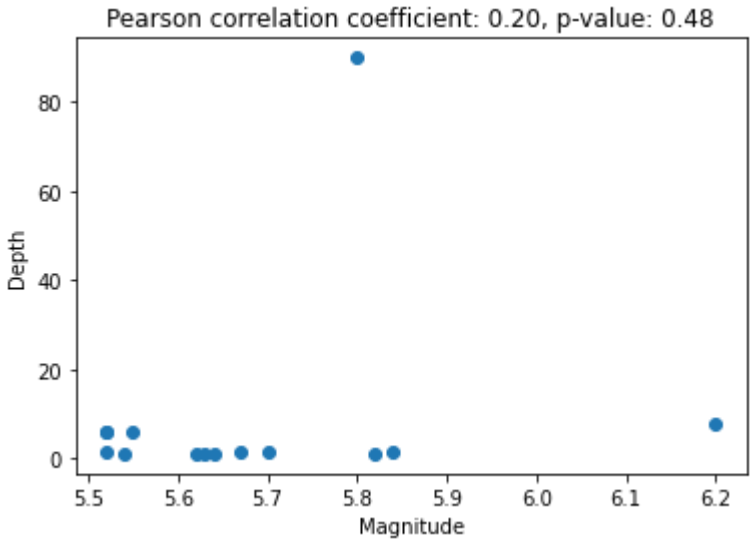
Visualize correlations:

In [14]:
```
plt.imshow(corr_matrix, cmap='coolwarm', interpolation='none')
plt.colorbar()
plt.xticks(range(len(corr_matrix)), corr_matrix.columns)
plt.yticks(range(len(corr_matrix)), corr_matrix.columns)
plt.show()
```



Pearson cofficient Relation

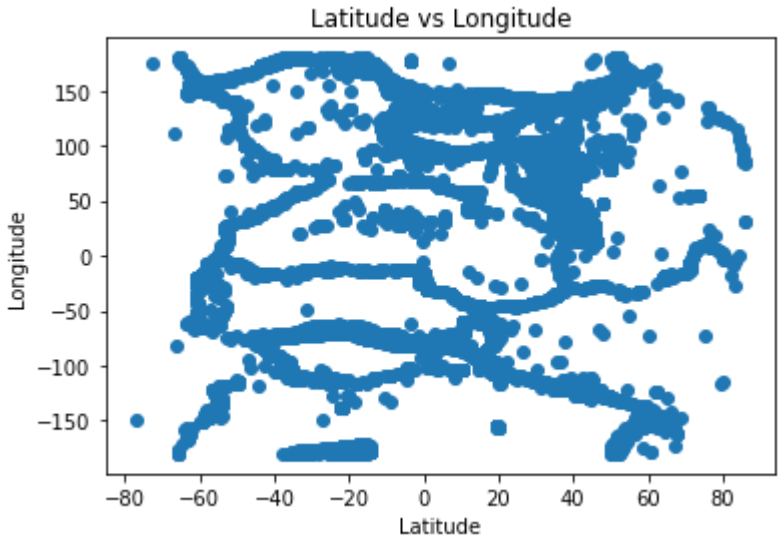
In [16]:
```
from scipy.stats import pearsonr

x = df_clean['Magnitude']
y = df_clean['Depth']

r, p = pearsonr(x, y)

plt.scatter(x, y)
plt.xlabel('Magnitude')
plt.ylabel('Depth')
plt.title(f'Pearson correlation coefficient: {r:.2f}, p-value: {p:.2f}')
plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

Applying Linear Regression

In [21]:
```
plt.scatter(df['Latitude'], df['Longitude'])
plt.title('Latitude vs Longitude')
plt.xlabel('Latitude')
plt.ylabel('Longitude')
plt.show()
```



In [4]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [5]:
```
# Load the earthquake data using pandas DataFrame
df = pd.read_csv('database.csv')
```

In [6]:
```
# Divide the data into training and testing datasets
X = df[['Latitude', 'Longitude']]
y = df['Magnitude']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rando
```

In [7]:
```
# Create a Linear Regression model and fit the training data
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[7]:
```
LinearRegression()
```

In [8]:
```
# Predict the magnitude of earthquakes using the testing dataset
y_pred = regressor.predict(X_test)
```

In [9]:
```
# Evaluate the performance of the model using Mean Squared Error (MSE)
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
```

```
Mean Squared Error: 0.18475232430091937
```

Applying Random Forest

In [10]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

In [11]:
```
# Create a Random Forest Regression model and fit the training data
regressor = RandomForestRegressor(n_estimators=100, random_state=42)
regressor.fit(X_train, y_train)
```

Out[11]:
```
RandomForestRegressor(random_state=42)
```

In [12]:
```
# Predict the magnitude of earthquakes using the testing dataset
y_pred = regressor.predict(X_test)
```

In [13]:
```
# Evaluate the performance of the model using Mean Squared Error (MSE)
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
```

```
Mean Squared Error: 0.21170610148313368
```

Applying Decision tree

In [14]:
```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

In [15]:
```
# Create a Decision Tree Regressor and fit the training data
regressor = DecisionTreeRegressor(random_state=42)
regressor.fit(X_train, y_train)
```

Out[15]:
```
DecisionTreeRegressor(random_state=42)
```

In [16]:
```
# Predict the magnitude of earthquakes using the testing dataset
y_pred = regressor.predict(X_test)
```

In [17]:
```
# Evaluate the performance of the model using mean squared error
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
```

```
Mean Squared Error: 0.35528641896220375
```