```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sb
        import matplotlib.pyplot as plt
        from sklearn.model_selection import train_test_split
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.metrics import mean_squared_error
```

```
In [2]: df = pd.read_csv('Alumni Giving Regression (Edited).csv')
```

```
In [3]: df.head()
```

Out[3]:

	A	B	C	D	E	F
0	24	0.42	0.16	0.59	0.81	0.08
1	19	0.49	0.04	0.37	0.69	0.11
2	18	0.24	0.17	0.66	0.87	0.31
3	8	0.74	0.00	0.81	0.88	0.11
4	8	0.95	0.00	0.86	0.92	0.28

```
In [4]: df.describe()
```

Out[4]:

	A	B	C	D	E	F
count	123.000000	123.000000	123.000000	123.000000	123.000000	123.000000
mean	17.772358	0.403659	0.136260	0.645203	0.841138	0.141789
std	4.517385	0.133897	0.060101	0.169794	0.083942	0.080674
min	6.000000	0.140000	0.000000	0.260000	0.580000	0.020000
25%	16.000000	0.320000	0.095000	0.505000	0.780000	0.080000
50%	18.000000	0.380000	0.130000	0.640000	0.840000	0.130000
75%	20.000000	0.460000	0.180000	0.785000	0.910000	0.170000
max	31.000000	0.950000	0.310000	0.960000	0.980000	0.410000

```
In [5]: X = df[['A', 'B', 'C', 'D', 'E']]
        y = df['F']
```

```
In [6]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randor
```

```
In [7]: model = RandomForestRegressor(n_estimators=100, random_state=42)
```

In [8]:
```python
model.fit(X_train, y_train)
```

Out[8]:   RandomForestRegressor(random_state=42)

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [9]:
```python
y_pred = model.predict(X_test)
```

In [10]:
```python
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

Mean Squared Error: 0.0038105840000000005

In [12]:
```python
plt.scatter(X_test['A'], y_test, color='blue', label='Actual Values')
plt.scatter(X_test['A'], y_pred, color='red', label='Predicted Values')
plt.xlabel("Feature A")
plt.ylabel("Target F")
plt.title("Actual vs Predicted Values for Feature A and Target F (Testing Set)"
plt.legend()
plt.show()
```