In [1]:

```
import numpy as np  # Importing NumPy library
import pandas as pd  # Importing Pandas library
import matplotlib.pyplot as plt  # Importing Matplotlib library's "pyplot" modu
import seaborn as sns  # Importing Seaborn library

import warnings
warnings.filterwarnings('ignore')

import os
```

In [2]:

```
data = pd.read_csv("Regression_Dataset.csv")
data
```

Out[2]:

	S.No	Left-Lateral-Ventricle	Left-Inf-Lat-Vent	Left-Cerebellum-White-Matter	Left-Cerebellum-Cortex	Left-Thalamus	Left-Caudate	Left-Putamen	Lef Pallidu
0	1	22916.9	982.7	15196.7	55796.4	6855.5	2956.4	4240.7	2223
1	2	22953.2	984.5	15289.7	55778.6	6835.1	3064.2	4498.6	2354
2	3	23320.4	1062.1	15382.1	55551.2	7566.0	3231.7	4456.2	1995
3	4	24360.0	1000.5	14805.4	54041.8	8004.6	3137.3	4262.2	1983
4	5	25769.4	1124.4	16331.1	54108.6	6677.4	2964.4	4204.6	2409
...	...	...	...	...	...	...	...	...	
4221	4222	27065.6	532.4	12425.1	51042.9	6354.8	3822.6	4490.5	2019
4222	4223	28408.8	912.7	14024.8	43103.5	6060.7	3114.2	3731.0	1937
4223	4224	34467.9	1659.6	12744.5	54924.8	6256.7	3573.4	3526.6	2189
4224	4225	31627.5	1334.4	15883.2	57148.2	6982.4	4475.8	4464.4	2317
4225	4226	14879.4	704.2	11346.6	50468.5	6935.4	3258.5	3751.5	2226

4226 rows × 141 columns

In [3]:

```
X = data.drop(["Age"], axis=1)
y = data['Age']
```

In [4]:

```
from sklearn.linear_model import ElasticNet
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(X,y, test_size=0.2, random_sta


# create Elastic Net model
enet = ElasticNet(alpha=0.1, l1_ratio=0.5, max_iter=5000)

# fit model to training data
enet.fit(x_train, y_train)

# predict on test data
y_pred = enet.predict(x_test)
```

In [5]:

```
from sklearn.model_selection import cross_val_predict  # For K-Fold Cross Valic
from sklearn.metrics import r2_score  # For find accuracy with R2 Score
from sklearn.metrics import mean_squared_error  # For MSE
from math import sqrt  # For squareroot operation

y_pred_enet_train = enet.predict(x_train)
y_pred_enet_test = enet.predict(x_test)

r2_enet_train = r2_score(y_train, y_pred_enet_train)
print("Training R^2 for Elastic Net Model: ", r2_enet_train)

r2_enet_test = r2_score(y_test, y_pred_enet_test)
print("Testing R^2 for Elastic Net Model: ", r2_enet_test)

RMSE_enet_train = sqrt(mean_squared_error(y_train, y_pred_enet_train))
print("RMSE for Training Data: ", RMSE_enet_train)

RMSE_enet_test = sqrt(mean_squared_error(y_test, y_pred_enet_test))
print("RMSE for Testing Data: ", RMSE_enet_test)
```

```
Training R^2 for Elastic Net Model:  0.837897137872658
Testing R^2 for Elastic Net Model:  0.8254782321186127
RMSE for Training Data:  8.090466728155226
RMSE for Testing Data:  8.324266468884291
```

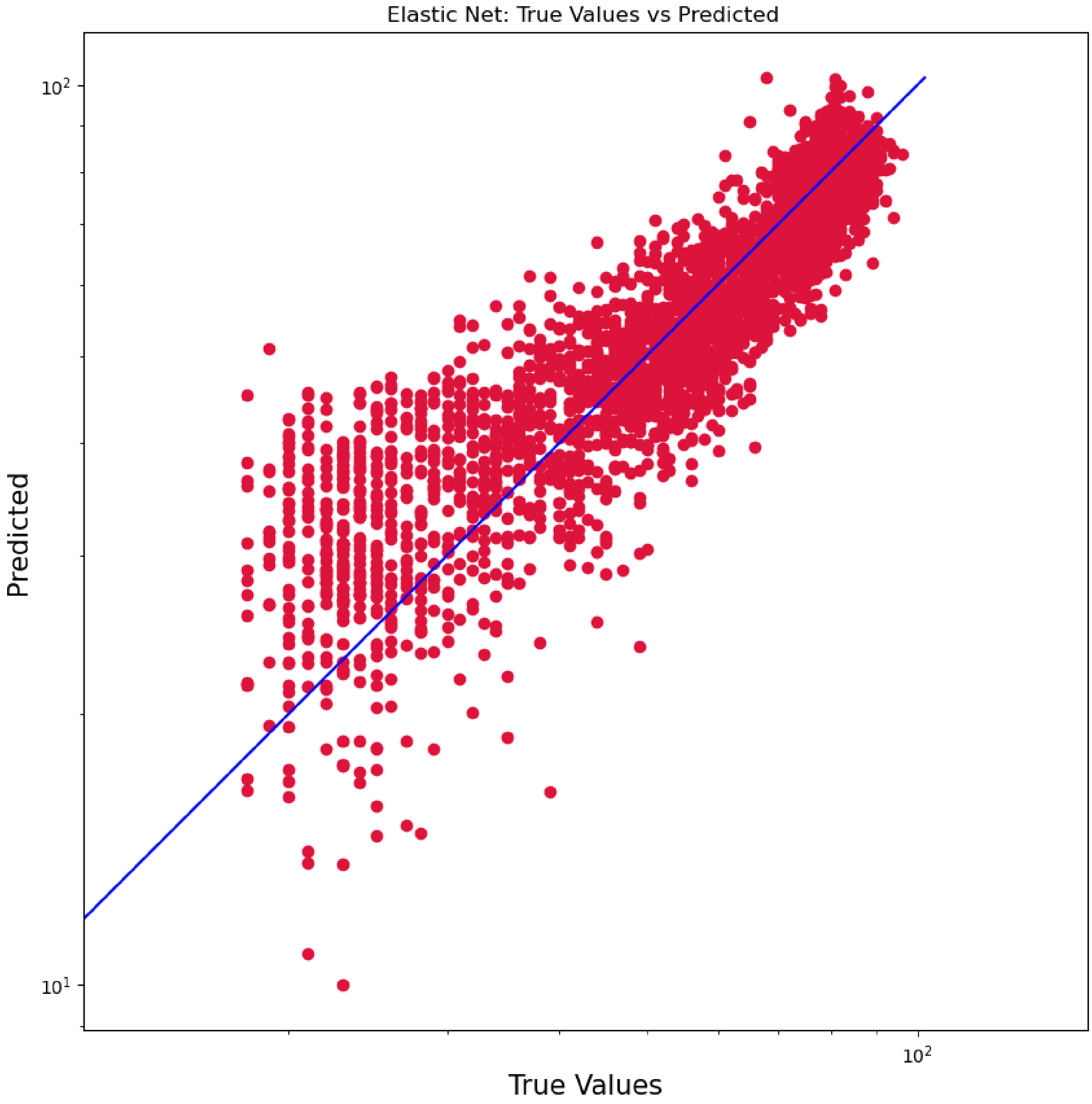
In [6]:

```
true_value = y_train
predicted_value = y_pred_enet_train
```

In [7]:

```
plt.figure(figsize=(10,10))
plt.scatter(true_value, predicted_value, c='crimson')
plt.yscale('log')
plt.xscale('log')

p1 = max(max(predicted_value), max(true_value))
p2 = min(min(predicted_value), min(true_value))
plt.plot([p1, p2], [p1, p2], 'b-')
plt.xlabel('True Values', fontsize=15)
plt.ylabel('Predicted', fontsize=15)
plt.title("Elastic Net: True Values vs Predicted ")
plt.axis('equal')
plt.show()
```


Elastic Net: True Values vs Predicted
10^2
10^1
Predicted
10^2
True Values

```
In [ ]:
```