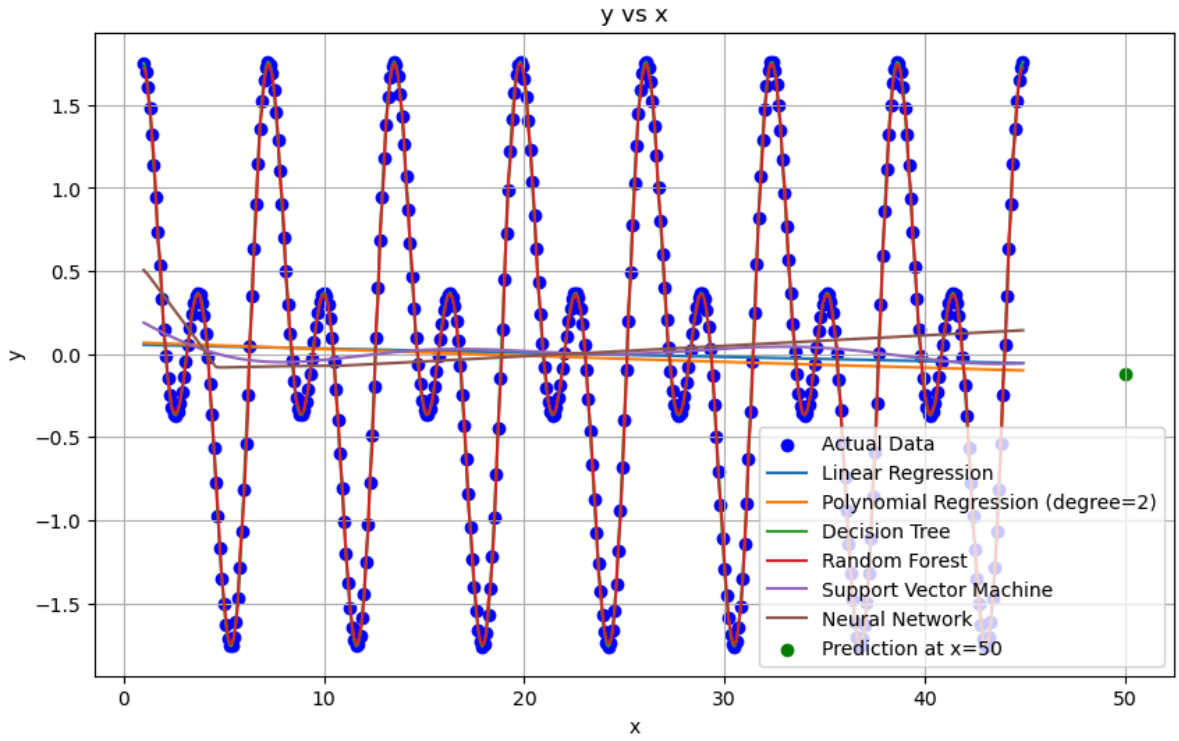
In [25]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error, r2_score


df = pd.read_csv('scr-dataset.csv')


X = df[['x']].values
y = df['y'].values


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran


models = {
    'Linear Regression': LinearRegression(),
    'Polynomial Regression (degree=2)': PolynomialFeatures(degree=2),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100, random_state=42
    'Support Vector Machine': SVR(),
    'Neural Network': MLPRegressor(max_iter=1000, random_state=42)
}


def fit_and_predict(model, X_train, y_train, X_test, x_value):
    if isinstance(model, PolynomialFeatures):
        X_poly_train = model.fit_transform(X_train)
        X_poly_test = model.transform(X_test)
        lin_reg = LinearRegression().fit(X_poly_train, y_train)
        y_pred = lin_reg.predict(X_poly_test)
        y_value_pred = lin_reg.predict(model.transform([[x_value]]))[0]
    else:
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        y_value_pred = model.predict([[x_value]])[0]
    return y_pred, y_value_pred


x_value = 50
predictions = {}
for name, model in models.items():
    y_pred, y_value_pred = fit_and_predict(model, X_train, y_train, X_test,
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    predictions[name] = (y_value_pred, mse, r2)


for name, (y_value_pred, mse, r2) in predictions.items():
    print(f"{name}:\n - Predicted y at x = {x_value}: {y_value_pred}\n - MSE


plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='blue', label='Actual Data')
```

```
for name, model in models.items():
    if isinstance(model, PolynomialFeatures):
        X_poly = model.transform(X)
        lin_reg = LinearRegression().fit(model.transform(X_train), y_train)
        plt.plot(X, lin_reg.predict(X_poly), label=name)
    else:
        model.fit(X, y)
        plt.plot(X, model.predict(X), label=name)

plt.scatter(x_value, predictions['Linear Regression'][0], color='green', lab
plt.title('y vs x')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.legend()
plt.show()
```

```
Linear Regression:
 - Predicted y at x = 50: -0.12139300595210278
 - MSE: 0.816
 - R^2: -0.021

Polynomial Regression (degree=2):
 - Predicted y at x = 50: -0.11411178551411022
 - MSE: 0.815
 - R^2: -0.021

Decision Tree:
 - Predicted y at x = 50: 1.7594025983376151
 - MSE: 0.034
 - R^2: 0.957

Random Forest:
 - Predicted y at x = 50: 1.6997102509493058
 - MSE: 0.017
 - R^2: 0.979

Support Vector Machine:
 - Predicted y at x = 50: 0.012436371614998845
 - MSE: 0.849
 - R^2: -0.063

Neural Network:
 - Predicted y at x = 50: -0.08630193546583581
 - MSE: 0.798
 - R^2: 0.001
```


y vs x
y
x
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
10
20
30
40
50
Actual Data
Linear Regression
Polynomial Regression (degree=2)
Decision Tree
Random Forest
Support Vector Machine
Neural Network
Prediction at x=50