```python
import streamlit as st
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
from joblib import dump, load
from gmdhpy import gmdh
from ModelSerialization import save_model, load_model
import os
import matplotlib.pyplot as plt

# Function to load data
@st.cache(allow_output_mutation=True)
def load_data(filepath):
    data = pd.read_csv(filepath)
    return data

# Function to train models and save them
def train_and_save_models(data, results_path):
    features = data.iloc[:, :13]
    outputs = data.iloc[:, -3:]
    scaler = StandardScaler()
    features_scaled = scaler.fit_transform(features)
    scaler_path = os.path.join(results_path, 'scaler.joblib')
    dump(scaler, scaler_path)  # Save the scaler

    X_train, X_test, y_train_all, y_test_all = train_test_split(features_scaled,
outputs, test_size=0.2, random_state=42)
    mses = []

    for i in range(outputs.shape[1]):
        y_train = y_train_all.iloc[:, i]
        y_test = y_test_all.iloc[:, i]
        model = gmdh.MultilayerGMDH()
        model.fit(X_train, y_train)
        model_path = os.path.join(results_path, f'gmdh_model_target_{i}.joblib')
        save_model(model, model_path)
        predictions = model.predict(X_test)
        mse = mean_squared_error(y_test, predictions)
        mses.append(mse)

    return mses, outputs.shape[1]

# Plot MSE values
```

```
def plot_mses(mses):
    plt.figure()
    plt.plot(mses, marker='o', linestyle='-')
    plt.title('Mean Squared Error for Each Target')
    plt.xlabel('Target Index')
    plt.ylabel('MSE')
    plt.grid(True)
    st.pyplot(plt)

# Streamlit interface
def main():
    st.title('Steel Alloy Properties Prediction')
    st.write('This application predicts material properties using the GMDH
approach.')

    data_folder = r'C:\Users\Elena\Documents\GitHub\steel_strength\Information'
    data_file = 'metals_data.csv'
    data_path = os.path.join(data_folder, data_file)

    results_folder =
r'C:\Users\Elena\Documents\GitHub\steel_strength\Results_data'

    if st.button('Load Data and Train Models'):
        data = load_data(data_path)
        st.dataframe(data)  # Display loaded data
        mses, num_targets = train_and_save_models(data, results_folder)
        st.write(f'MSEs for each target: {mses}')
        st.session_state.num_targets = num_targets  # Update session state
        plot_mses(mses)  # Plot MSE values

    st.write('Select a model to load:')
    selected_target = st.selectbox('Choose a target:', options=[f'Target {i}' for
i in range(st.session_state.get('num_targets', 3))])

    if st.button('Load Model and Make Predictions'):
        model_filename = f'gmdh_model_target_{selected_target[-1]}.joblib'
        model_path = os.path.join(results_folder, model_filename)
        model = load_model(model_path)
        st.write(f'Model {selected_target} loaded successfully.')

if __name__ == '__main__':
    main()
```