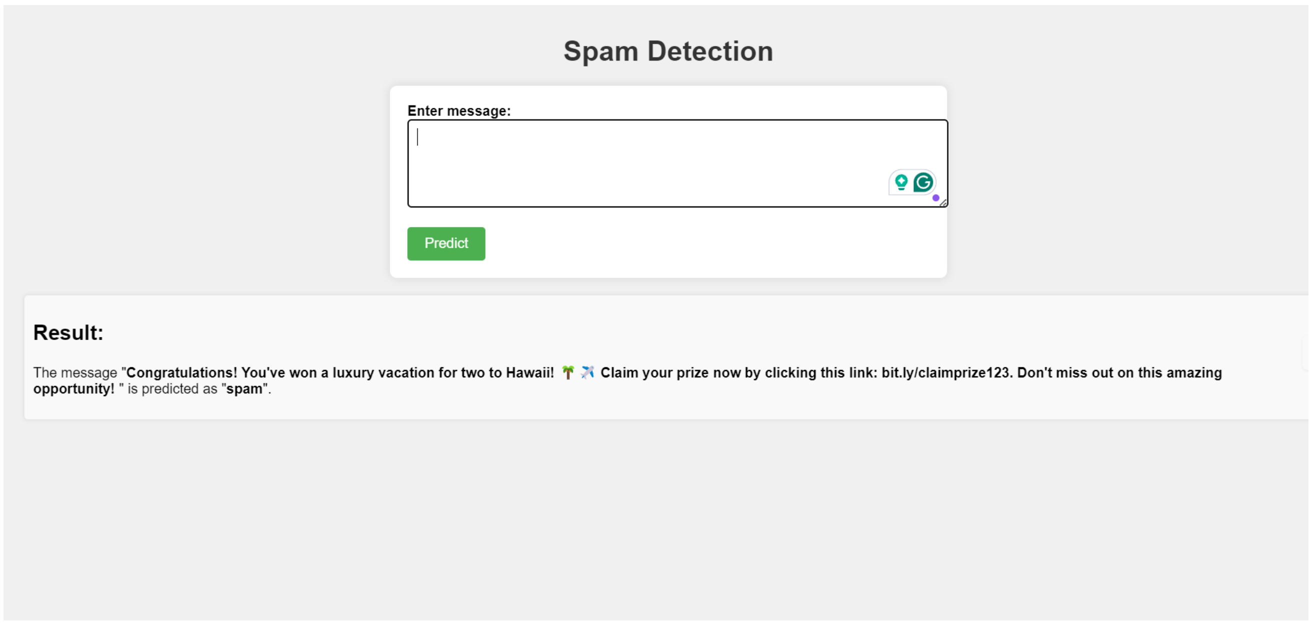
```python
import pandas as pd
import string
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
import pickle

# Load and preprocess data
df = pd.read_csv('spam.csv', encoding='latin-1')
df = df.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)
df.columns = ["label", "message"]

# Preprocessing function
def preprocess_content(text):
    stemmer = PorterStemmer()
    nopunc = ''.join([char for char in text if char not in
string.punctuation])
    words = word_tokenize(nopunc.lower())
    nostop = [stemmer.stem(word) for word in words if word not in
stopwords.words('english') and word.isalpha()]
    return ' '.join(nostop)

# Apply preprocessing
df['cleaned_text'] = df['message'].apply(preprocess_content)

tfidf = TfidfVectorizer()
X = tfidf.fit_transform(df['cleaned_text'])
y = df['label']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Train model
rf_model = RandomForestClassifier()
rf_model.fit(X_train, y_train)

# Evaluate model
rf_preds = rf_model.predict(X_test)
rf_accuracy = accuracy_score(y_test, rf_preds)
rf_report = classification_report(y_test, rf_preds)

print("Random Forest Model:")
print(f'Accuracy: {rf_accuracy:.4f}')
print(f'Classification Report:\n{rf_report}\n')
```

```
Random Forest Model:
Accuracy: 0.9758
Classification Report:
```

```
              precision    recall  f1-score   support

         ham       0.97      1.00      0.99       965
        spam       1.00      0.82      0.90       150

    accuracy                           0.98      1115
   macro avg       0.99      0.91      0.94      1115
weighted avg       0.98      0.98      0.97      1115



# Prediction function
def predict_class(input_text):
    cleaned_input = preprocess_content(input_text)
    X_new = tfidf.transform([cleaned_input])
    return rf_model.predict(X_new)[0]

# Test prediction
input_text = 'free entri wkli comp win fa cup final tkt may text fa
receiv entri questionstd txt ratetc appli'
predicted_class = predict_class(input_text)
print(f"Predicted class for '{input_text}': {predicted_class}")

Predicted class for 'free entri wkli comp win fa cup final tkt may
text fa receiv entri questionstd txt ratetc appli': spam

# Pickle dump
with open('rf_model.pkl', 'wb') as f:
    pickle.dump(rf_model, f)

with open('tfidf_vectorizer.pkl', 'wb') as f:
    pickle.dump(tfidf, f)
```



Result:

The message "**Congratulations! You've won a luxury vacation for two to Hawaii! 🌴 ✈️ Claim your prize now by clicking this link: bit.ly/claimprize123. Don't miss out on this amazing opportunity!** " is predicted as "**spam**".