```
spam_detector_flask/
│
├── app.py
├── model.pkl
├── vectorizer.pkl
└── templates/
    └── index.html
```

```
#app.py — Flask App
from flask import Flask, render_template, request
import pickle

app = Flask(__name__)

# Load the trained model and vectorizer
with open('model.pkl', 'rb') as model_file:
    model = pickle.load(model_file)

with open('vectorizer.pkl', 'rb') as vec_file:
```

```
    vectorizer = pickle.load(vec_file)


@app.route('/', methods=['GET', 'POST'])
def index():
    prediction = None
    if request.method == 'POST':
        message = request.form['message']
        message_vector =
vectorizer.transform([message])
        pred = model.predict(message_vector)[0]
        prediction = "Spam" if pred == 1 else "Not
Spam"
    return render_template('index.html',
prediction=prediction)


if __name__ == '__main__':
    app.run(debug=True)


#index.html — Input UI (inside templates/
folder)
<!DOCTYPE html>
```

```
<html>
<head>
    <title>Spam Detector</title>
</head>
<body>
    <h2>Spam Detection - Naive Bayes</h2>
    <form method="POST">
        <label for="message">Enter your
message:</label><br>
        <input type="text" name="message"
required style="width: 300px;"><br><br>
        <button type="submit">Predict</button>
    </form>
    {% if prediction %}
        <h3>Prediction: <span
style="color:blue">{{ prediction
}}</span></h3>
    {% endif %}
</body>
</html>


#train_and_save_model.py — Script to train
and save model

import pandas as pd
from sklearn.feature_extraction.text import
CountVectorizer
from sklearn.model_selection import
train_test_split
```

```
from sklearn.naive_bayes import
MultinomialNB
import pickle

# Sample data
data = {
    'text': [
        'Win money now', 'Limited offer just for
you', 'Hi, how are you?',
        'Call me tomorrow', 'Free tickets available',
'Congratulations, you won!',
        'Are you coming to the party?', 'Let’s grab
lunch today', 'Earn extra cash fast', 'Meeting at
10 am'
    ],
    'label': [1, 1, 0, 0, 1, 1, 0, 0, 1, 0]
}
df = pd.DataFrame(data)

# Vectorization
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df['text'])
y = df['label']

# Model training
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=42)
model = MultinomialNB()
model.fit(X_train, y_train)
```

```
# Save model and vectorizer
with open('model.pkl', 'wb') as model_file:
    pickle.dump(model, model_file)

with open('vectorizer.pkl', 'wb') as vec_file:
    pickle.dump(vectorizer, vec_file)

print("Model and vectorizer saved!")
```

To Run the Application
Train and Save Model

```
python train_and_save_model.py
```

Start Flask App

```
python app.py
```