SMS SPAM DETECTION

Introduction to project:

Overview:

In this technical era , most of us are aware of spam messages which acts as gateway for many of malware attacks on our systems . To decrease these type of attacks we came forward with the concept of sms spam detection . Using this sms spam detection we are able to which of those messages we have received are spam . To detect , At first we have to enter the message in the provided space and then by clicking on the button to check we are able to know wheather it is spam or not.This is the main overview of the project.

Purpose:

The main purpose of sms spam detector is to detect the spam messages . By identifying spam messages we can protect our data and another important information by stopping it from going into the hands of hackers .

Literature survey:

Existing problem:

Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS, Mobile service providers suffer from some sort of financial problems as well as it reduces calling time for users. Unfortunately, if the user accesses such Spam SMS they may face the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.

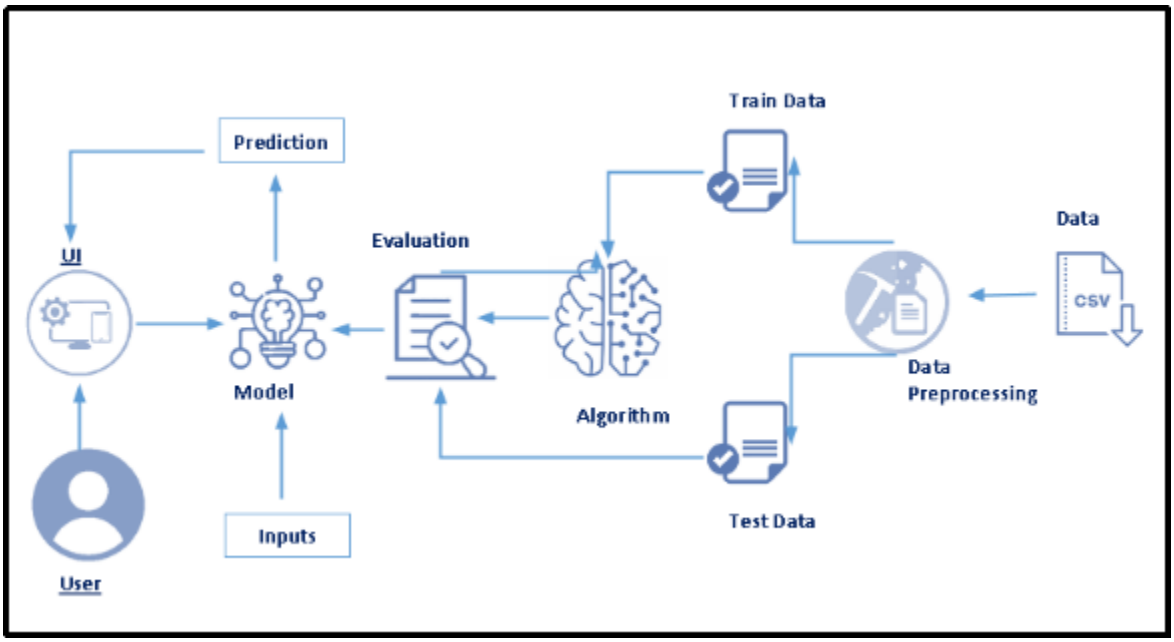
SMS SPAM DETECTION

Proposed solution:

To avoid such Spam SMS people use white and black list of numbers. But this technique is not adequate to completely avoid Spam SMS. To tackle this problem it is needful to use a smarter technique which correctly identifies Spam SMS. Natural language processing technique is useful for Spam SMS identification. It analyses text content and finds patterns which are used to identify Spam and Non-Spam SMS.

Theoritical Analysis:

Block diagram:



Hardware/software Design:

- Hardware
- operating system : Windows 10
- processor : Intel core i5
- Ram : 8GB

SMS SPAM DETECTION

- Software
- Anakonda navigator
- Anakonda prompt

Experimental Investigations:

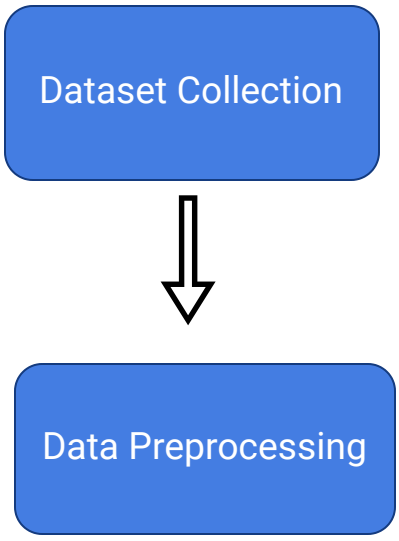
Through this project , we are trying to build a sms spam detector which takes message as input and expected to produce output stating either it is spam message or not .

Evaluating Model

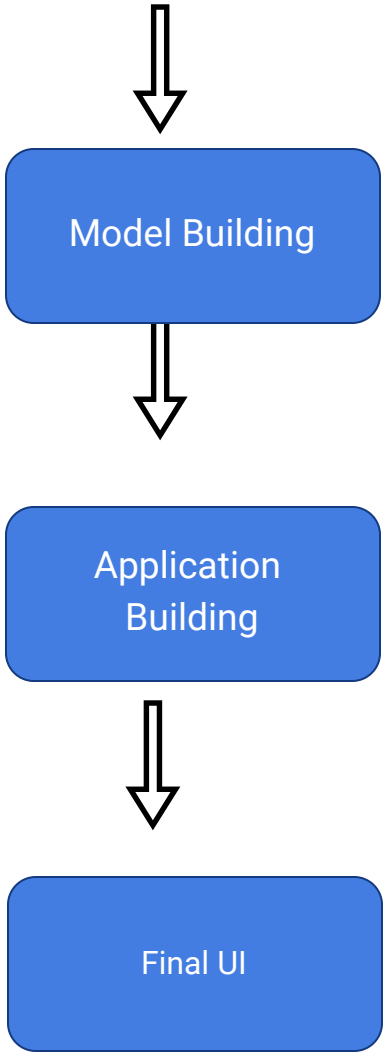
```
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
score = accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is:- ' ,score*100)
```

```
[[716  16]
 [ 17 286]]
Accuracy Score Is:-  96.81159420289856
```

Flow chart:



SMS SPAM DETECTION



Result:

The result of this project is detecting spam messages and also creating awareness about spam messages and data protection amoung people .

Advantages & Disadvantages:

Advantages:

- Data Protection
- Identifying spam messages

SMS SPAM DETECTION

Disadvantages:

- The main disadvantage is this sms spam detector may not avoid spam messages completely but can be reduced to certain limit .

Applications:

This machine learning web app

- can be used in mobiles .
- can be used in personal computers/laptops .

Conclusion:

we are hereby concluding that our project named sms spam detection is able to detect spam message whenever that particular message is given as input to the detector .

Future Scope:

These days technology is increasing day by day , along with the increase technology thefts are also being increased . Definently there is a need to differenciate amoung real and fake . Our project definently has a wider scope in future .

Bibliography:

- Research paper on spam sms detection by Lutfun Nahar Lota and B.M Mainul Hossian .
- Book named "Detection of sms spam botnets in mobile devices" by Mashael Al-Omany and Abdelouahid Derhab .

SMS SPAM DETECTION

Source Code:

```
from flask import Flask,render_template,request
import pickle
#from gevent.pywsgi import WSGIServer
import os
filename='spam_sms_mnb_model.pkl'
classifier=pickle.load(open(filename,'rb'))
cv=pickle.load(open('cv-transform.pkl','rb'))
app=Flask(__name__)
@app.route('/')
def home():
    return render_template('index,html')
@app.route('/Spam',methods=['POST','GET'])
def prediction():
    return render_template('spam.html')
@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        message = request.form['message']
        data = [message]
        vect = cv.transform(data).toarray()
        my_prediction = classifier.predict(vect)
        return render_template('result.html',prediction=my_prediction)
#port=os.getenv('VCAP_APP_PORT','8080')
if __name__=="__main__":
    port=int(os.environ.get('PORT',5000))
    app.run(port=port,debug=True,use_reloader=False)
    #app.secret_key=os.urandom(12)
    #app.run(debug=True,host='0.0.0.0', port=port)
```

Importing necessary libraries

```
: import numpy as np # scientific computation
  import pandas as pd # loading dataset file
  import matplotlib.pyplot as plt # Visulization
  import nltk  # Preprocessing our text
  from nltk.corpus import stopwords # removing all the stop words
  from nltk.stem.porter import PorterStemmer # stemming of words
  from sklearn.naive_bayes import MultinomialNB
  import re
  from sklearn.model_selection import train_test_split
  from sklearn.metrics import confusion_matrix,accuracy_score
  from sklearn.svm import SVC
  from sklearn.feature_extraction.text import CountVectorizer
  from sklearn.tree import DecisionTreeClassifier
  import pickle
```

SMS SPAM DETECTION

Load our dataset

```
#Load our dataset
df = pd.read_csv("spam_ham_dataset.csv")
```

```
# top 5 rows of the dataframe
df.head()
```

	label	text	label_num
0	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	spam	Subject: photoshop , windows , office . cheap ...	1
4	ham	Subject: re : indian springs\r\nthis deal is t...	0

```
# bottom 5 rows of the dataframe
df.tail()
```

	label	text	label_num
5166	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	ham	Subject: industrial worksheets for august 2000...	0
5170	spam	Subject: important online banking alert\r\ndea...	1

SMS SPAM DETECTION

EDA on Dataset

```
df=df.drop("Unnamed: 0",axis=1)
```

```
# Return the shape of data
df.shape
```

```
(5171, 3)
```

```
df.ndim
```

```
2
```

```
#Return the size of data
df.size
```

```
15513
```

```
#Returns the sum fo all na values
df.isna().sum()
```

```
label        0
text         0
label_num    0
dtype: int64
```

```
#Give concise summary of a DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5171 entries, 0 to 5170
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   label      5171 non-null   object
 1   text       5171 non-null   object
 2   label_num  5171 non-null   int64 
dtypes: int64(1), object(2)
memory usage: 121.3+ KB
```

SMS SPAM DETECTION

```
df.describe() #return improtant values for continous column data
```

	label_num
count	5171.000000
mean	0.289886
std	0.453753
min	0.000000
25%	0.000000
50%	0.000000
75%	1.000000
max	1.000000

```
#it will return true if any  column having null values
df.isnull().any()
```

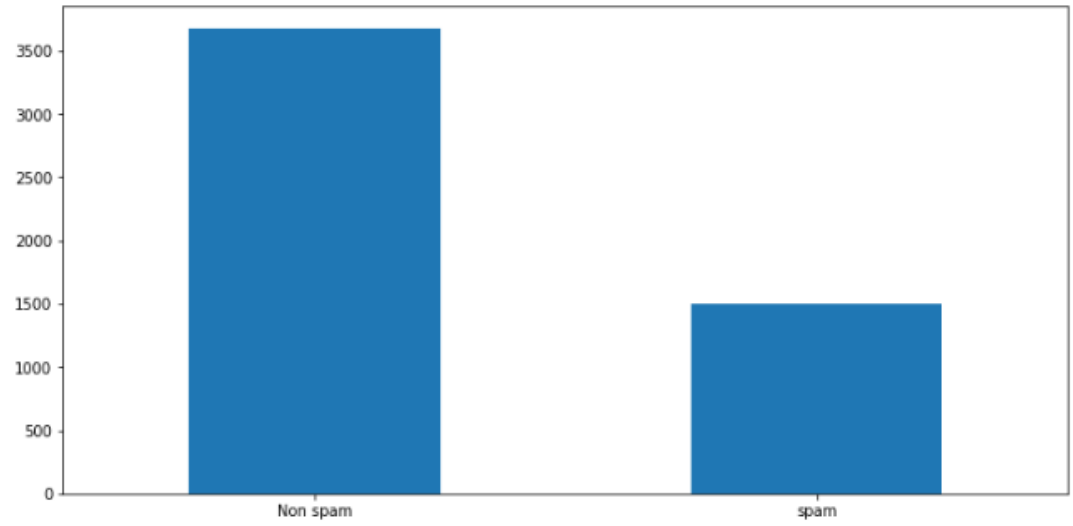
```
label        False
text         False
label_num    False
dtype: bool
```

```
#Find null values
df.isnull().sum()
```

```
label        0
text         0
label_num    0
dtype: int64
```

Let's Visualize the Column label_num

```
df["label_num"].value_counts().plot(kind="bar",figsize=(12,6))
plt.xticks(np.arange(2), ('Non spam', 'spam'),rotation=0);
```



SMS SPAM DETECTION

Cleaning The Text

```
import re
corpus = []
length = len(df)
```

```
for i in range(0,length):
    text = re.sub("[^a-zA-Z0-9]"," ",df["text"][i])
    text = text.lower()
    text = text.split()
    pe = PorterStemmer()
    stopword = stopwords.words("english")
    text = [pe.stem(word) for word in text if not word in set(stopword)]
    text = " ".join(text)
    corpus.append(text)
```

```
corpus
```

```
['subject enron methanol meter 988291 follow note gave monday 4 3 00 preliminari flow data provid daren pleas overrid pop dai
li volum present zero reflect daili activ obtain ga control chang need asap econom purpos',
 'subject hpl nom januari 9 2001 see attach file hplnol 09 xl hplnol 09 xl',
 'subject neon retreat ho ho ho around wonder time year neon leader retreat time know time year extrem hectic tough think any
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=35000)
X = cv.fit_transform(corpus).toarray()
```

```
# Extracting dependent variable from the dataset
y = pd.get_dummies(df['label'])
y = y.iloc[:, 1].values
```

```
import pickle ## importing pickle used for dumping models
pickle.dump(cv, open('cv-transform.pkl', 'wb')) ## saving to into cv-transform.pkl file
```

```
#Splitting data into train and validation sets using train_test_split

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
```

Creating a model using Multinomial NaiveBayes

```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
```

```
#Fitting the model to the training sets
model.fit(X_train, y_train)
```

```
MultinomialNB()
```

SMS SPAM DETECTION

Prediction

```
y_pred=model.predict(X_test)
y_pred
```

```
array([0, 0, 0, ..., 0, 1, 0], dtype=uint8)
```

Evaluating Model

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
score = accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is:- ' ,score*100)
```

```
[[716  16]
 [ 17 286]]
Accuracy Score Is:-  96.81159420289856
```

SMS SPAM DETECTION

```
from sklearn.svm import SVC
svm1=SVC(kernel='rbf')
svm1.fit(X_train,y_train)
```

```
SVC()
```

```
y_pred4=svm1.predict(X_test)
from sklearn.metrics import accuracy_score
svm_rbf=accuracy_score(y_test,y_pred4)
svm_rbf
```

```
0.9623188405797102
```

```
svm2=SVC(kernel='sigmoid')
svm2.fit(X_train,y_train)
```

```
SVC(kernel='sigmoid')
```

```
y_pred5=svm2.predict(X_test)
from sklearn.metrics import accuracy_score
svm_sig=accuracy_score(y_test,y_pred5)
svm_sig
```

```
0.9652173913043478
```

```
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
```

```
DecisionTreeClassifier()
```

```
y_pred6=dt.predict(X_test)
from sklearn.metrics import accuracy_score
dec_tree=accuracy_score(y_test,y_pred6)
dec_tree
```

```
0.9468599033816425
```

```
models = pd.DataFrame({
    'Model': [ 'MultinomialNB','SVM-rbf','SVM-sigmoid','Decision Tree'],
    'Test Score': [ score,svm_rbf,svm_sig,dec_tree,]})
models.sort_values(by='Test Score', ascending=False)
```

	Model	Test Score
0	MultinomialNB	0.968116
2	SVM-sigmoid	0.965217
1	SVM-rbf	0.962319
3	Decision Tree	0.946860

Saving our model

```
import pickle
pickle.dump(model, open("spam-sms-mnb-model.pkl", "wb"))
```

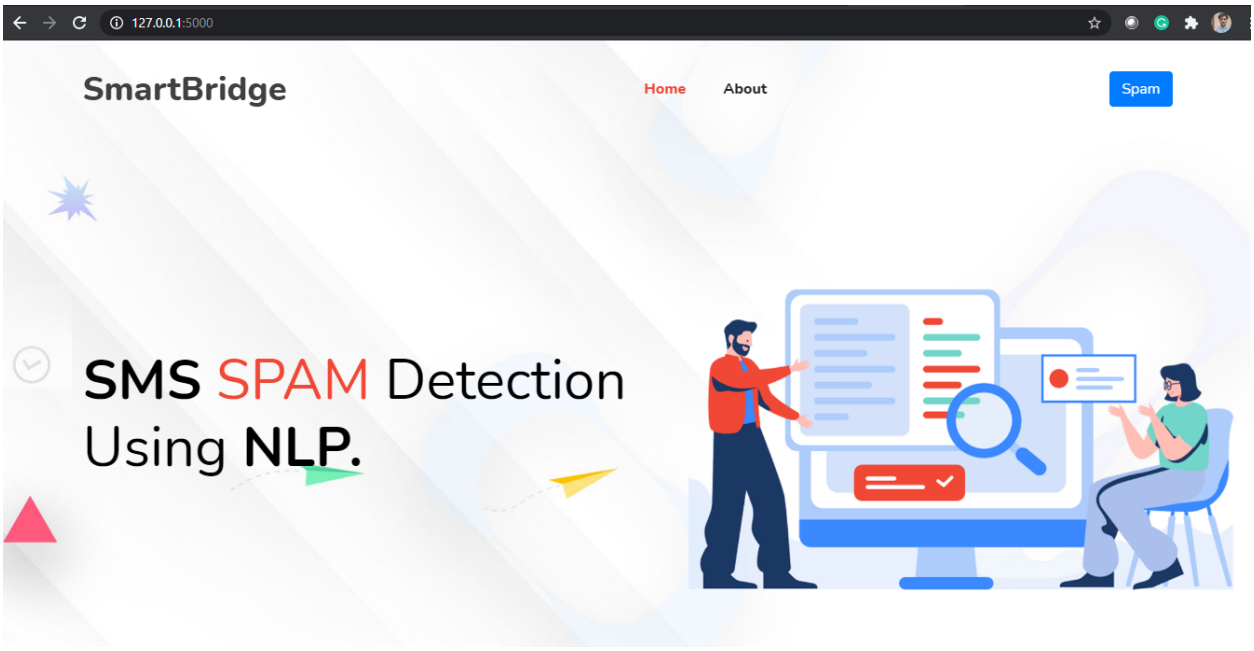
SMS SPAM DETECTION

Html codes:

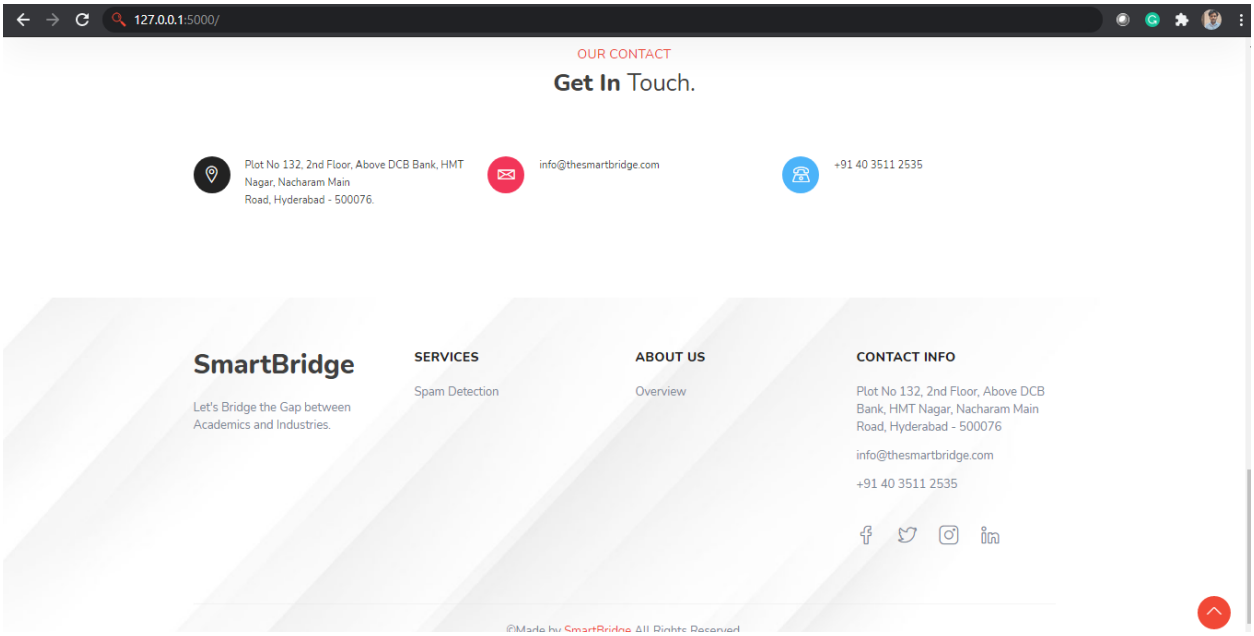
This project requires three html files.The links to html codes are provided below

- index.html : view-source:file:///C:/Users/Jahnavi%20Nelluri/Desktop/internz%20python/Spam-SMS-Classifier-Deployment-main/Flask/templates/index.html
- result.html : view-source:file:///C:/Users/Jahnavi%20Nelluri/Desktop/internz%20python/Spam-SMS-Classifier-Deployment-main/Flask/templates/result.html
- spam.html : view-source:file:///C:/Users/Jahnavi%20Nelluri/Desktop/internz%20python/Spam-SMS-Classifier-Deployment-main/Flask/templates/spam.html

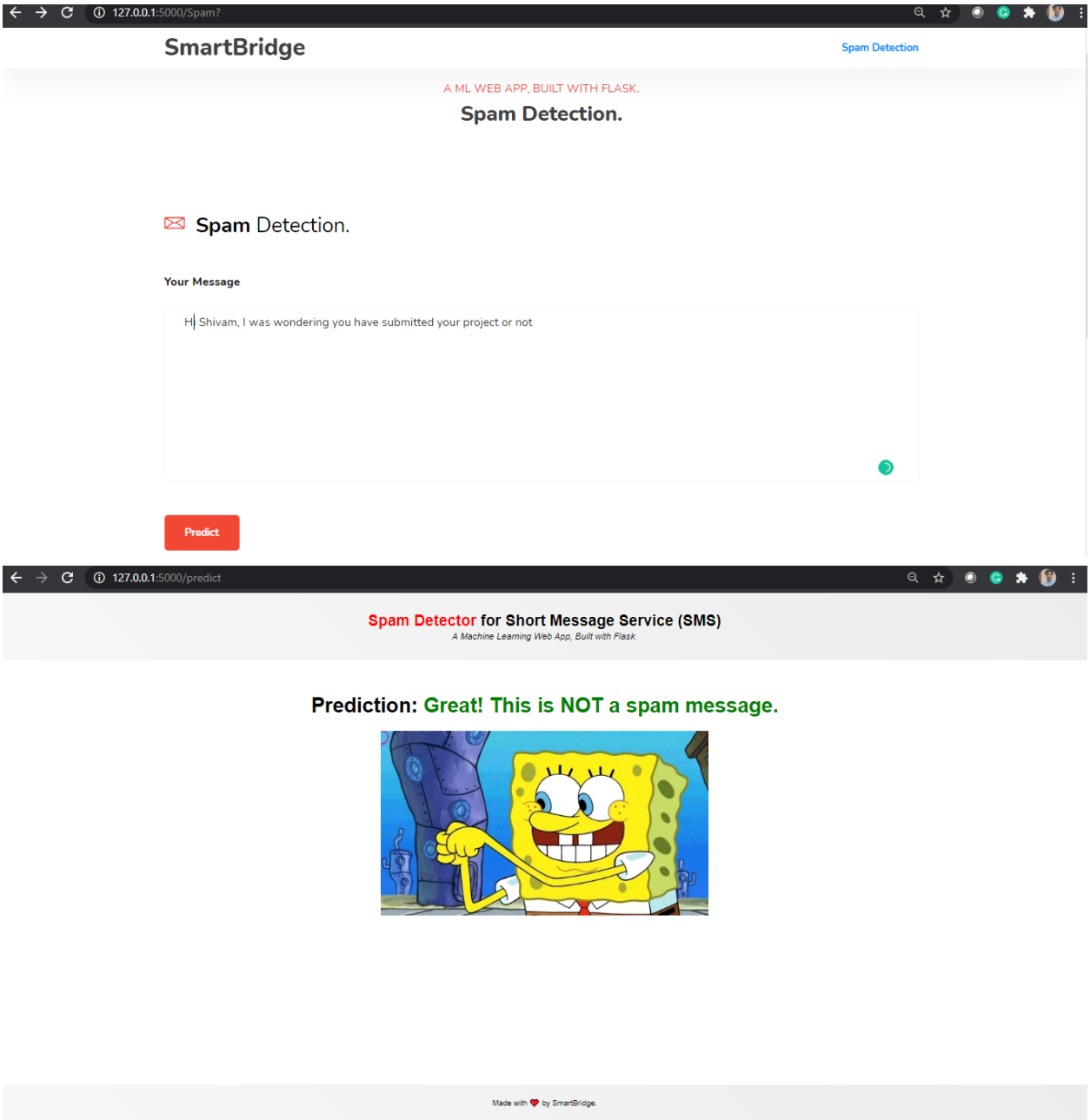
Final UI:



SMS SPAM DETECTION



SMS SPAM DETECTION



SMS SPAM DETECTION