Website Recommendation:

The website name and description are saved in an excel sheet in format Name, Description. The excel sheet is read using pandas.read_excel() and stored in pandas DataFrame df_train. We consider each row as a separate document and recommend websites ie document as per the required description. The given descriptions are preprocessed such as removing punctuations and converting to lower cases.

In order to recommend these websites, they must be organized in the form of clusters. Hence, K means clustering is used to form n number of clusters, where each cluster would have websites with a similar context. Currently, we have assumed the number of clusters to be 2 for the given dataset.

Term Frequency i.e TF, is a measure of how many times a given token is present in a string. Whereas, IDF is used to measure how much information a token or word gives in the whole corpus. TF-IDF is a product of combining TF and IDF. It generates the weights for each word across all documents.

For the given problem, I have used the scikit-learn implementation of TF-IDF and KMeans.

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
```

After calculating TF-IDF for training corpus, train the K means model with k=2. Once the model is trained, predict the clusters for all the training corpus and update the DataFrame df_train, by assigning predicted clusters to each document. Further, we predict the cluster for unseen data and recommend the websites present in the similar cluster.

One of the drawbacks of K means is to assign a number of clusters. This can be solved using the elbow method or visualizing the data set and decide the number of clusters. A visualized description of clusters is shown in the code, however since we have only 4 data points, the points in the same cluster look a bit far.

The executable code for the above problem can be accessed through Github Link.

https://github.com/mhaiskarshridhar/Document-Classificaiton/blob/master/Kmeans-TFIDF.ipynb

Another method to classify documents is Doc2Vec. I have implemented the same and can be accessed through the below link:

https://github.com/mhaiskarshridhar/Document-Classificaiton/blob/master/Gensim%20-%20Doc2Vec.ipynb