

LAW ADVISORY SERVICES AND MANAGEMENT SYSTEM

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

ANKIT KUMAR (Reg. no.37110041)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC |
Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **ANKIT KUMAR(Reg.No.37110041)** who carried out the project entitled "**LAW ADVISORY SERVICES AND MANAGEMENT SYSTEM**" under my supervision from November 2020 to March 2021.

Internal Guide

Mrs. M.D. Anto Praveena, M.C.A., M.E., (Ph.D).,

Head of the Department

**Dr.S.VIGNESHWARI M.E.,Ph.D.,
Dr.L.LAKSHMANAN M.E.,Ph.D.,**

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

We **ANKIT KUMAR** (Roll. No. 37110041) hereby declare that the Project Report entitled “**LAW ADVISORY SERVICES AND MANAGEMENT SYSTEM**” done by us under the guidance of **Mrs. M. D. Anto Praveena, M.C.A., M.E., (Ph.D).**, Department of Computer Science and Engineering at **Sathyabama Institute of Science and Technology** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. Anto Praveena M.C.A., M.E., (Ph.D.)**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Over the years, Artificial Intelligence (AI) is spreading its roots in different areas by utilizing the concept of making the computers learn and handle complex tasks. This project aims at designing an effective assistance system that can consult user with their query on law dispute"s in India. The system uses NLP(Natural Language Processing) to read and understand the query and derive meaning from it. Then the system uses the same NLP algorithm with web-scraping to extract meaningful data related to the query raised by user. Cosine similarity method is then used to get the possible result for the query. The main purpose of developing this project is to consult people who are involved in any law disputes, through online mode. It aims for guiding a user to every aspect of the law, which is managed by the government. These aspects are mainly related to the law dispute"s such as domestic violation, tenant & landlord agreement breach, debt, and lending agreement, etc. The user can also avail lawyer details who are near to the user location for further assistance.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	viii
1.	INTRODUCTION	01
	1.1. INTRODUCTION	01
	1.2. OUTLINE OF THE PROJECT	01
	1.3. PROBLEM STATEMENT	02
2.	LITERATURE SURVEY	03
	2.1. LITERATURE SURVEY	03
3	AIM AND SCOPE	04
	3.1. AIM OF THE PROJECT	04
	3.2. SCOPE OF THE PROJECT	04
4	SYSTEM IMPLEMENTATION	05
	4.1. SYSTEM ARCHITECTURE	05
	4.2. METHODS AND MODEL DETAILS	06
	4.2.1. QUERY ANALYSIS	06
	4.2.2. COLLECTING DATA FROM DIFFERENT WEB SOURCES	09
	4.2.3. PREDICT THE MOST APPROPRIATE SOLUTION FOR THE QUERY	10

5.	RESULT AND DISCUSSION	13
6.	SUMMARY AND CONCLUSION	17
7.	APPENDIX	18
	A) SOURCE CODE	18
	B) REFERENCES	33

LIST OF FIGURES

FIG.NO NO	FIGURE NAME	PAGE
4.1	ARCHITECTURE DIAGRAM	05
4.2	EQUATION OF COSINE SIMILARITY	11
4.3	COUNT VECTORIZER MATRIX FOR COUNTING OF SIMILAR WORDS	12
5.1	HOME PAGE	13
5.2	QUERY ON DAMAGED GOODS COMPLAINT	14
5.3	DAMAGED GOODS QUERY RESULT PAGE	14
5.4	QUERY ON ADVOCATE NEAR EGMORE	15
5.5	ADVOCATE NEAR EGMORE QUERY RESULT PAGE	16

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Legal information resources System in India can be categorised into primary, secondary and tertiary sources. Primary sources of information are being generated by Courts, Parliament, and State Assemblies in the form of judgments, acts, bills and rules. The Secondary and tertiary sources such as, commentaries, digests, directories, directory of directories, etc., are published mainly by the Government press and corporate publishers. In India many people does not want to go to court for their legal rights as they see to waste of their time and money. But our application help people to get an idea what they should do when they fell or don't know the legal matter of their accident which have happened with them. To many people of the country are unaware of their basic legal rights which help them in from getting into a trouble in any legal situation under Indian legislative law. Unless users know exactly which contents they want to know ,then they will get an help from a lawyer near them which are available to them at that point of time or can chat with them to get the suggestion that what they should do next as an legal step to get out the matter. A free text search facility is totally ineffective. Thus, it is not possible to do legal research with the help of existing open access legal information resources which are incomplete and not at all user-friendly. Moreover, each court"s database has to be searched separately for a comprehensive search of Indian law. Moreover, none of the open access legal information systems has developed a service to search all types of contents in a single shot. A web based online law advisory system is developed to provide the user solution to their query regarding law rights and how to proceed further in legal procedure.

1.2 OUTLINE OF THE PROJECT

The main goal of this application is to be able to provide a simple user – friendly means for setting a law related advises to their queries. The application is also set up for the connection between the user and lawyers to resolve the disputes related to the user query. It try to gives a better advice according to law of their issues.

It uses:

- Natural Language Processing Algorithm
- Cosine Similarity Algorithm

1.3 PROBLEM STATEMENT

Law is for all and equal to all in Indians. People always forget or have very few knowledge of their rights as an Indian citizen. When they fall in any legal issues, they don't know what to do and whom to contact. To get the clear idea of what is their respective rights according to the situation they are facing, they will give query or directly can contact to a lawyer near them to resolve their issues of their problem in legal terms.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATUER SURVEY

Various studies have been conducted on the design and development of information systems of the law advises over the internet. B. Morgan [1] discussed about how the lawyers support is available for sustainable economy initiatives for their legal professionals. He also discusses on the challenges for law or not-law relationalities to their legal professional identity. Now, according to Cheng[1], he postulates that to symbolise various laws and precedent, to adapt the fundamental logic system is required in the development of information System. In [8] ,they used a similar legal case retrieval system by using multiple speech question and answer their the required solution the user need. It also helps the user to get an alternat solution to their multiple question asked to the system. G. Greenleaf [11], states in their system about he building an environment for the sustainable free legal advisory system which does not indulge in any human activity and all the work is done by the system or the AI for their solution to the user. All the above authors have focused on the development of a system which is helpful to the user regarding their issues on legal issues.

.

CHAPTER 3

AIM AND SCOPE

3.1 AIM OF THE PROJECT

The aim of this project is to provide related solution to the user to get an idea what they should do when they fell or don't know the legal matter the related solution to the user related to their legal disputes or issues using an automation system or by advise of the lawyer. As in India many people did not have their basic rights which they know, so we are developing a web application to help them query raised by the user to provide an propriate solution to them.

3.2 SCOPE OF THE PROJECT

Law is for all and equal to all in india. People always forget or have verry few knowledge of their rights as an Indian citizen. When they fall in any legal issues, they don't know what to do and whom to contact .To get the clear idea of what is their respective rights according to the situation they are facing, they will give query or directly can contact to a lawyer near them to resolve their issues of their problem in legal terms.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 SYSTEM ARCHITECTURE :

The Proposed system aims to provides the results related to the type of query raised by the user in according to the legal issues they might be in or to find a lawyer near them available to resolve their query. The architecture of the system is as follows –

- User Query
- Data Processing using NLP
- Web Resources / Articles
- Cosine Similarity
- Query Result

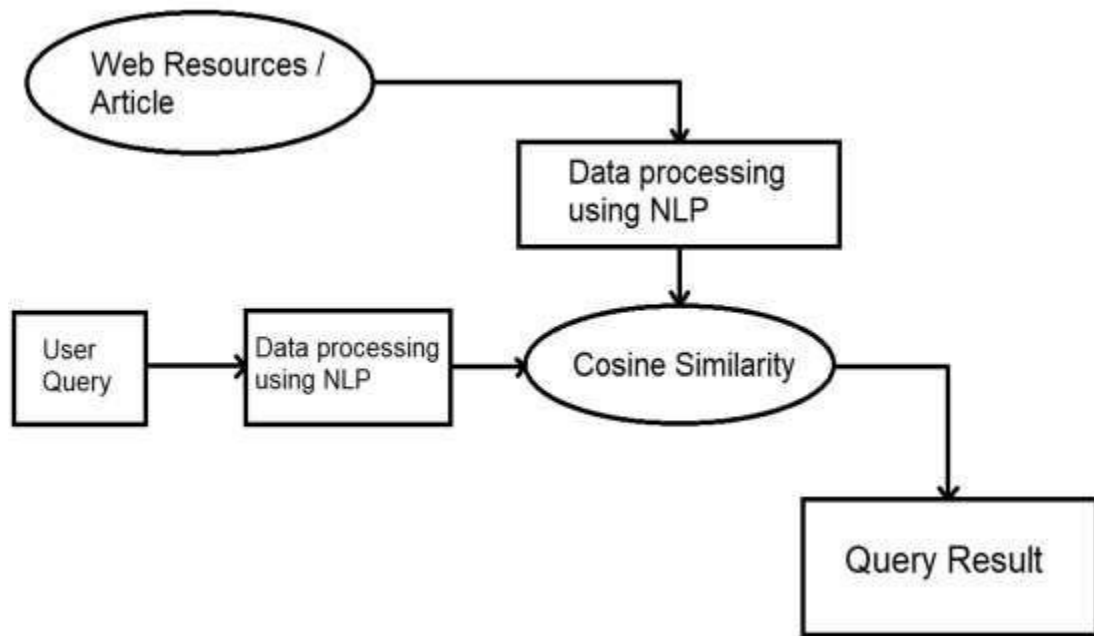


Figure 4.1 Architecture Diagram

4.2 METHODS AND MODEL DETAILS :

This project has mainly three parts:

- Query Analysis
- Collecting data from different web-sources
- Predict the most appropriate solution for the query

4.2.1 QUERY ANALYSIS:

In this part query raised by the user is pre-processed using NLP to generate meaning full insights in order to gather related information from different web sources related to the query. The modules used are NumPy, Django, NLTK, Speech Recognition, etc to plot different visualization on Query. The major task of this module of our project is to get the query from the user and try to manipulate that query with related machine learning algorithm to get he most appropriate and understandable language answer from our system.

NLP ALGORITHM (NATURAL LANGUAGE PROCESSING) :

Natural language processing (NLP) is a field of artificial intelligence in which computers analyse, understand, and derive meaning from human language in a smart and useful way. By utilizing NLP, developers can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation.

The field of study that focuses on the interactions between human language and computers is called natural language processing, or NLP for short. It sits at the intersection of computer science, artificial intelligence, and computational linguistics. "Natural Language Processing is a field that covers computer understanding and manipulation of human language, and it's ripe with possibilities for newsgathering," Anthony Pesce says in Natural Language Processing in Kitchen. "You usually hear

about it in the context of analysing large pools of legislation or other document sets, attempting to discover patterns or root out corruption.”

NLP is used to analyse, allowing machines to understand how humans speak. This human-computer interaction enables real-world applications like automatic text summarization, sentiment analysis, topic extraction, named entity recognition, parts of speech tagging, relationship extraction, stemming, and more. NLP is commonly used for text mining, machine translation, and automated question answering. NLP is characterized as a difficult problem in computer science. Human language is rarely precise, or plainly spoken. To understand human language is to understand not only the words, but the concepts and how they’re linked together to create meaning. Despite language being one of the easiest things for the human mind to learn, the ambiguity of language is what makes natural language processing a difficult problem for computers to master.

NLP algorithms have a variety of uses. Basically, they allow developers and businesses to create a software that understands human language. Due to the complicated nature of human language, NLP can be difficult to learn and implement correctly. However, with the knowledge gained from this article, you will be better equipped to use NLP successfully, no matter your use case.

NUMPY :

NumPy brings the computational power of languages like C and Fortran to Python, a language much easier to learn and use. NumPy is open source software and has many contributors.

NumPy targets the C and Python reference implementations of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most

operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter Kernels and many more vastly simplifies the programming workflow and debugging.

NLTK (NATURAL LANGUAGE TOOLKIT) :

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook

NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning. NLTK has been used successfully as a teaching tool, as an individual study tool, and as a platform for prototyping and building research systems. NLTK supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

SPEECH RECOGNITION:

Speech recognition is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers. It is also known as automatic speech recognition (ASR), computer speech recognition or speech to text (STT). From the technology perspective, speech recognition has a long history with several waves of major innovations. Most recently, the field has benefited from advances in deep learning and big data. The advances are evidenced not only by the surge of academic papers published in the field, but more importantly by the worldwide industry adoption of a variety of deep learning methods in designing and deploying speech recognition systems.

4.2.2 COLLECTION DATA FROM DIFFERENT WEB SOURCES:

In this part the data related to query are extracted from different web-sources and then get converted into human understandable language, the following are the steps followed here:

- Downloading the data from web-sources using modules like web-browser, newspaper-websites, Wikipedia, article.
- NLP is then applied on the downloaded data to get only meaningful data and rest of the data are get rejected.
- At last a corpus of data is prepared from the meaningful data and then tokenized sentence list is prepared from the corpus using modules like sklearn, string, speech-recognition.

SKLEARN MODULE :

Sklearn is probably the most useful library for machine learning in Python. The Sklearn library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction.

Scikit-learn comes loaded with a lot of features. Here are a few of them to help you understand the spread:

- **Supervised learning algorithms:** Think of any supervised machine learning algorithm you might have heard about and there is a very high chance that it is part of scikit-learn. Starting from Generalized linear models (e.g. Linear Regression), Support Vector Machines (SVM), Decision Trees to Bayesian methods – all of them are part of scikit-learn toolbox. The spread of machine learning algorithms is one of the big reasons for the high usage of scikit-learn. I started using scikit to solve supervised learning problems and would recommend that to people new to scikit / machine learning as well.
- **Cross-validation:** There are various methods to check the accuracy of supervised models on unseen data using sklearn.
- **Unsupervised learning algorithms:** Again there is a large spread of machine learning algorithms in the offering – starting from clustering, factor analysis, principal component analysis to unsupervised neural networks.
- **Various toy datasets:** This came in handy while learning scikit-learn. I had learned SAS using various academic datasets (e.g. IRIS dataset, Boston House prices dataset). Having them handy while learning a new library helped a lot.
- **Feature extraction:** Scikit-learn for extracting features from images and text (e.g. Bag of words).

4.2.3 PREDICT THE MOST APPROPRIATE SOLUTION FOR THE QUERY:

In this part the tokenized sentence list and query raised by user is compared and the paragraph (corpus) having maximum similarity with the query is returned as the consultant for the raised query. This is achieved using cosine similarity, count vectorizer. There are lot more variable which help to accumulate the most meaningful solution to the requested query. We uses cosine similarity which help us to make the query into tokens and find the

COSINE SIMILARITY :

A commonly used approach to match similar documents is based on counting the maximum number of common words between the documents. But this approach has

an inherent flaw. That is, as the size of the document increases, the number of common words tend to increase even if the documents talk about different topics. The cosine similarity helps overcome this fundamental flaw in the „count-the-common-words” or Euclidean distance approach. Cosine similarity is a metric used to determine how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. In this context, the two vectors I am talking about are arrays containing the word counts of two documents. As a similarity metric, how does cosine similarity differ from the number of common words? When plotted on a multi-dimensional space, where each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents and not the magnitude. If you want the magnitude, compute the Euclidean distance instead. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance because of the size (like, the word „cricket” appeared 50 times in one document and 10 times in another) they could still have a smaller angle between them. Smaller the angle, higher the similarity.

$$\text{Cos}\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a} \cdot \vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$ is the dot product of the two vectors.

Figure 2. Equation of the Cosine Similarity

COUNT VECTORIZER :

Count Vectorizer is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such

texts, and we wish to convert each word in each text into vectors (for using in further text analysis). In order to use textual data for predictive modelling, the text must be parsed to remove certain words – this process is called **tokenization**. These words need to then be encoded as integers, or floating-point values, for use as inputs in machine learning algorithms. This process is called feature extraction (or vectorization)

It creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix. The value of each cell is nothing but the count of the word in that particular text sample.

Inside Count Vectorizer, these words are not stored as strings. Rather, they are given a particular index value. In this case, „at’ would have index 0, „each’ would have index 1, „four’ would have index 2 and so on. The actual representation has been shown in the table below –

0	1	2	3	4	5	6	7	8	9	10	11
0	0	0	1	1	0	0	1	0	0	0	1
0	0	1	0	2	0	1	0	0	0	0	1
1	1	0	1	1	1	0	1	1	0	1	0

Figure 3: Count Vectorizer matrix for counting of similar words

CHAPTER 5

RESULT AND DISCUSSION

This paper focuses on providing real-time legal consultation, we have designed this web application to resolve the law disputes issues raised by any user. The application responds to users issues intelligently with the ability of reasoning. When compared with other automatic legal consultation websites, our system needs fewer questions to be asked from the user during a dialog, which in turn improves the efficiency of consultation provided by our system.

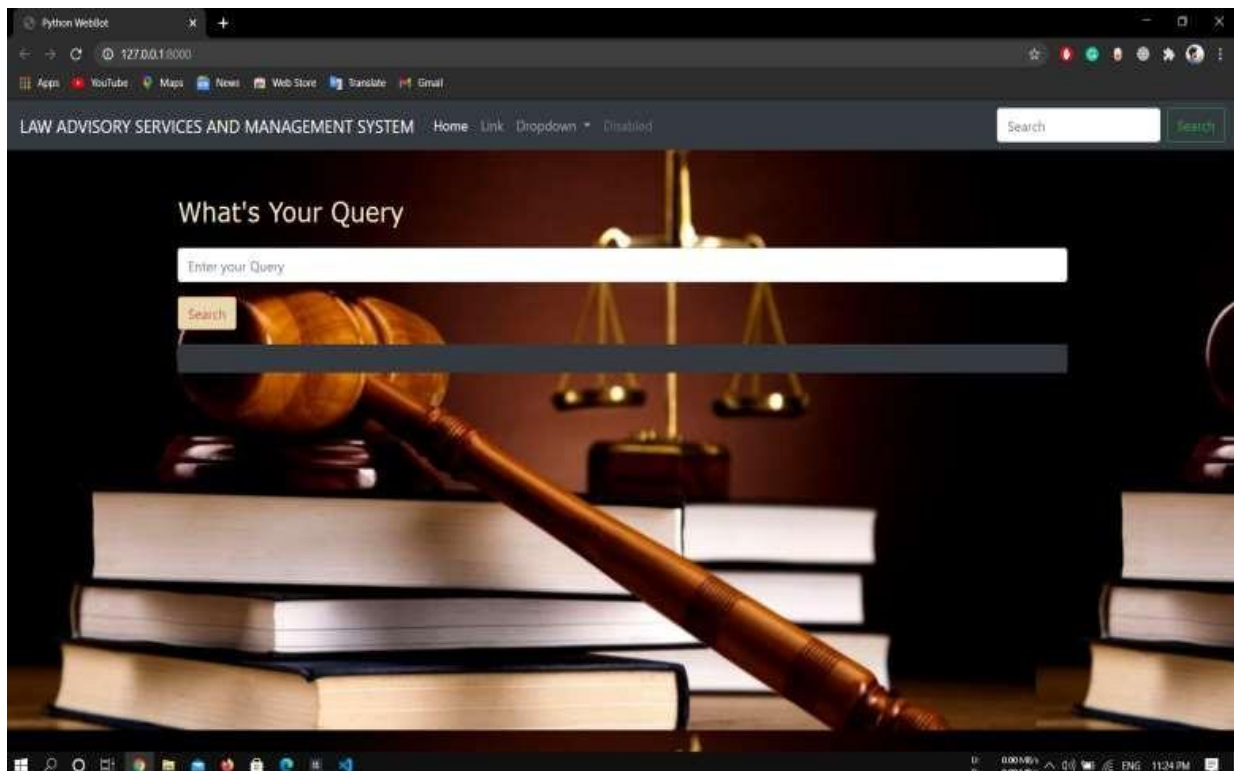


Figure 5.1: Home Page

Figure 5.1 represents the Home Page of the web application that can be used by the user to raise the query. When the user write his related query in the query text field and then enter search button. After entering the button the system try find the most understandable language answer of the query raised by the user.



Figure 5.2: Query on damaged goods complaint

Figure 5.2 represents the query raised by the user for the knowing of what are his/her rights when she file against the damaged good that he/she got product from the company.

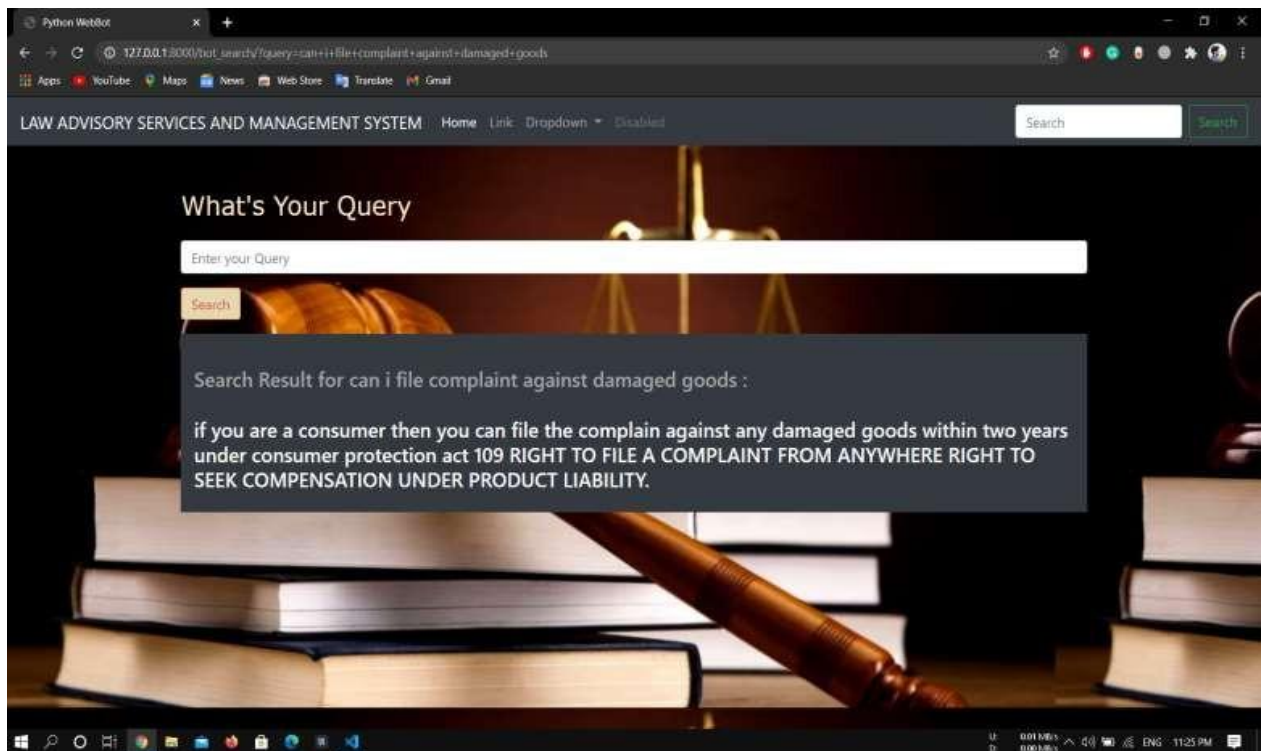


Figure 5.3: Damaged goods query result page

Figure 5.3 represents the query solution provided by the system is then shown in the dialog box where the user can see his solution. In the dialog box user can see the most probable outcome of the query.



Figure 5.4: Query on Advocate near Egmore

Figure 5.2 represents the query raised by the user to find the lawyers which are available in the specified region where the user wants to search to get help in their legal issues.

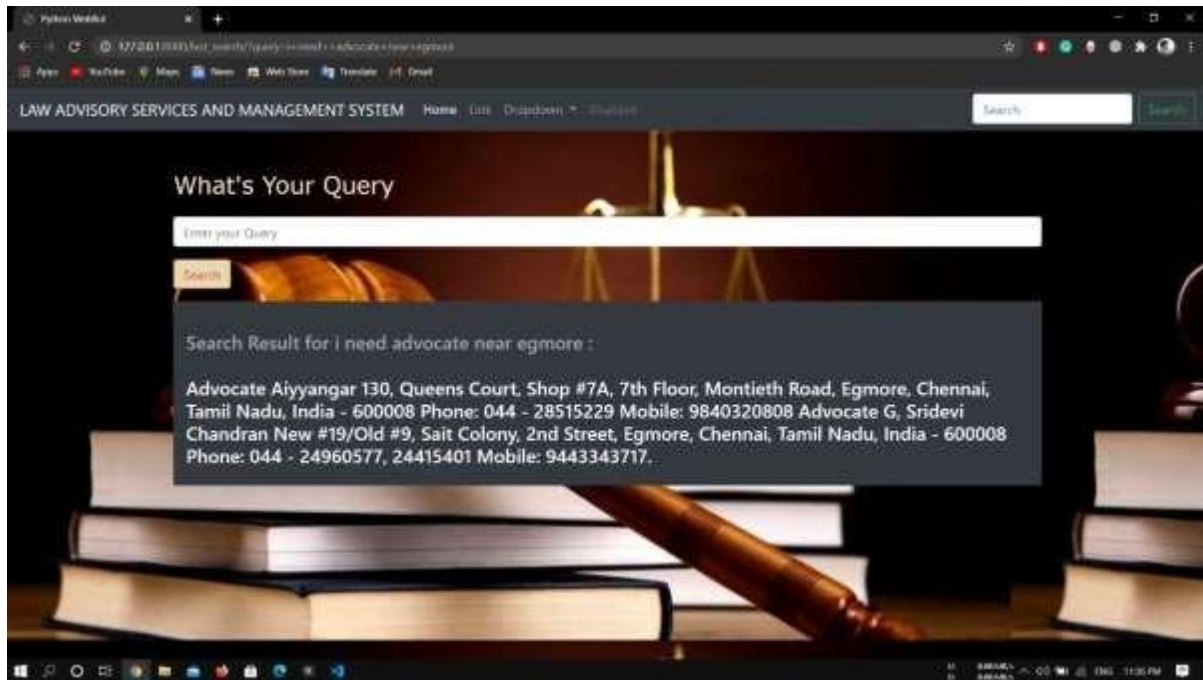


Figure 5.5: Advocate near Egmore query result page

Figure 5.3 represents the query solution provided by the system is then shown in the dialog box where the user can see his solution. In this dialog box user can see the lawyers and their offices available in the specified region.

CHAPTER 6

SUMMARY AND CONCLUSION

Legal information resources System in India can be categorised into primary, secondary and tertiary sources. Primary sources of information are being generated by Courts, Parliament, and State Assemblies in the form of judgments, acts, bills and rules. Moreover, none of the open access legal information systems has developed a service to search all types of contents in a single shot. A web based online law advisory system is developed to provide the user solution to their query regarding law rights and how to proceed further in legal procedure. When compared with other automatic legal consultation websites, our system needs fewer questions to be asked from the user during a dialog, which in turn improves the efficiency of consultation provided by our system. As of now our project is deployed as an web application in near future with the optimization of code an android can be developed of the existing system.

APPENDIX

A) SOURCE CODE

Views.py

```
from django.shortcuts import render

import speech_recognition as sr

import pyttsx3

import wolframalpha

import wikipedia

import webbrowser

from newspaper import Article

import random

import string

import nltk

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine_similarity

import numpy as np

import warnings

warnings.filterwarnings('ignore')

nltk.download('punkt', quiet=True)

links=['https://www.onlyadvocate.com/', 'https://www.chennaiindia.co.in/advocates-in-
chennai.php', 'http://www.irockit.in/a-comprehensive-guide-to-civil-and-criminal-
proceedings-in-
india/', 'https://data.gov.in/search/site?query=law', 'https://gouravkmr170.github.io/']

#Get The Article

for i in links:
```

```

    article = Article(url= "%s" %i)
article.download()
article.parse()
article.nlp()
corpus = article.text

#Print Article
#print(corpus)

#Tokenization
text = corpus
sentence_list = nltk.sent_tokenize(text) #Create list of sentences

def index_sort(list_var):
    lenght = len(list_var)
    list_index = list(range(0, lenght))

    x = list_var
    for i in range(lenght):
        for j in range(lenght):
            if x[list_index[i]] > x[list_index[j]]:
                #Swap
                temp = list_index[i]

```

```

        list_index[i] = list_index[j]

        list_index[j] = temp

    return list_index

def bot_response(query):
    query = query.lower()
    sentence_list.append(query)

    bot_response = ""

    cm = CountVectorizer().fit_transform(sentence_list)
    similarity_scores = cosine_similarity(cm[-1], cm)
    similarity_scores_list = similarity_scores.flatten()
    index = index_sort(similarity_scores_list)
    index = index[1:]
    response_flag = 0

    j = 0
    for i in range(len(index)):
        if similarity_scores_list[index[i]] > 0.1:
            bot_response = bot_response+' '+sentence_list[index[i]]
            response_flag = 1
            j = j+1
    if j > 3:
        break

```

```

if response_flag == 0:

    bot_response = bot_response+' '+ wikipedia.summary(query, sentences=10)

    sentence_list.remove(query)


    return bot_response


def index(request):

    return render(request, 'webbot/index.html')


def bot_search(request):

    query = request.GET.get('query')


    #try:

        #client = wolframalpha.Client("TPQYK3-RE6RKQ9T93")

        #res = bot_response(request)

    ans = bot_response(query)

    return render(request, 'webbot/index.html', {'ans': ans, 'query': query})


    #except Exception:

        #try:

```

```
#ans = wikipedia.summary(query, sentences=10)

#return render(request, 'webbot/index.htm', {'ans': ans, 'query': query})
```

```
#except Exception:
```

```
#ans = "FOUND NOTHING"
```

```
#return render(request, 'webbot/index.htm', {'ans': ans, 'query': query})
```

URLS.PY

```
from django.contrib import admin
```

```
from django.urls import path
```

```
from . import views
```

```
urlpatterns = [
```

```
    path('admin/', admin.site.urls),
```

```
    path("", views.index, name='index'),
```

```
    path('bot_search/', views.bot_search, name='bot_search')
```

```
]
```

Setting.py

```
import os
```

```
# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
```

```
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
```

```
# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/


# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'fq*&p!dk=xo8o(ljfft337&)9q@&$_3un!tk$d2paz6beu!)1i'


# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True


ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
```

```
'django.contrib.sessions.middleware.SessionMiddleware',  
'django.middleware.common.CommonMiddleware',  
'django.middleware.csrf.CsrfViewMiddleware',  
'django.contrib.auth.middleware.AuthenticationMiddleware',  
'django.contrib.messages.middleware.MessageMiddleware',  
'django.middleware.clickjacking.XFrameOptionsMiddleware',  
,
```

```
ROOT_URLCONF = 'webbot.urls'
```

```
TEMPLATES = [  
    {  
        'BACKEND': 'django.template.backends.django.DjangoTemplates',  
        'DIRS': ['templates'],  
        'APP_DIRS': True,  
        'OPTIONS': {  
            'context_processors': [  
                'django.template.context_processors.debug',  
                'django.template.context_processors.request',  
                'django.contrib.auth.context_processors.auth',  
                'django.contrib.messages.context_processors.messages',  
            ],  
        },  
    ],  
,
```

```
]
```

```
WSGI_APPLICATION = 'webbot.wsgi.application'
```

```
# Database
```

```
# https://docs.djangoproject.com/en/2.2/ref/settings/#databases
```

```
DATABASES = {
```

```
    'default': {
```

```
        'ENGINE': 'django.db.backends.sqlite3',
```

```
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
```

```
    }
```

```
}
```

```
# Password validation
```

```
# https://docs.djangoproject.com/en/2.2/ref/settings/#auth-password-validators
```

```
AUTH_PASSWORD_VALIDATORS = [
```

```
    {
```

```
        'NAME':
```

```
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
```

```
    },
```

```
    {
```

```
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
```

```
    },  
    {  
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',  
    },  
    {  
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',  
    },  
]
```

Internationalization

<https://docs.djangoproject.com/en/2.2/topics/i18n/>

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

Static files (CSS, JavaScript, Images)

<https://docs.djangoproject.com/en/2.2/howto/static-files/>

STATIC_URL = '/static/'

WSGI.py

```
import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'webbot.settings')

application = get_wsgi_application()
```

INDEX.html

```
<!doctype html>

<html lang="en">

  <head>

    <!-- Required meta tags -->

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">


    <!-- Bootstrap CSS -->

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
```

```
<title>Python WebBot</title>
```

```
</head>
```

```
<body style="background-image: url(https://wallpaperaccess.com/full/1558519.jpg)
;">
```

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">

  <a class="navbar-brand" href="#">LAW ADVISORY SERVICES AND
MANAGEMENT SYSTEM</a>
```

```
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
```

```
<span class="navbar-toggler-icon"></span>
```

```
</button>
```

```
<div class="collapse navbar-collapse" id="navbarSupportedContent">
```

```
<ul class="navbar-nav mr-auto">
```

```
<li class="nav-item active">
```

```
<a class="nav-link" href="#">Home <span class="sr-
only">(current)</span></a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link" href="#">Link</a>
```

```
</li>
```

```
<li class="nav-item dropdown">
```

```
<a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
```

```
Dropdown
```

```
</a>
```

```
<div class="dropdown-menu" aria-labelledby="navbarDropdown">
```

```
<a class="dropdown-item" href="#">Action</a>
```

```
<a class="dropdown-item" href="#">Another action</a>
```

```
<div class="dropdown-divider"></div>
```

```
<a class="dropdown-item" href="#">Something else here</a>
```

```

    </div>

    </li>

    <li class="nav-item">

        <a class="nav-link disabled" href="#" tabindex="-1" aria-
disabled="true">Disabled</a>

    </li>

</ul>

<form class="form-inline my-2 my-lg-0">

    <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">

    <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>

</form>

</div>

</nav>

```

```

<div class="container" style="padding-top: 3%;">

<form action="/bot_search/" method="GET">

    <div class="form-group">

        <label for="exampleInputEmail1" style="font-size: 3.5ch; padding-bottom: 1%;
color:blanchedalmond;font-family: Verdana, Geneva, Tahoma, sans-serif;">What's
Your Query</label>

        <input type="text" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" placeholder="Enter your Query" name="query">

    </div>

```

```
<button type="submit" class="btn btn-outline-danger" style="background:
rgb(230, 217, 173);">Search</button>
```

```
</form>
```

```
<div class="container p-3 my-3 bg-dark text-white">
```

```
{% if ans %}
```

```
<h4 style="color: darkgray; padding-top: 2%;">Search Result for {{query}} :</h4>
```

```
<h4 style="padding-top: 2%;">{{ans}}</h4>
```

```
{% endif %}
```

```
</div>
```

```
</div>
```

```
<!-- Optional JavaScript -->
```

```
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
```

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
```

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
```

UO2eT0CpHqdSJK6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dlHNDz0W
1" crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIlly6OrQ6VrjIEaFf/nJGzlxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

</body>

</html>

REFERENCES

- [1] B. Morgan, "Lawyers, legal advice and relationality in sustainable economy initiatives," *Oñati Socio-Legal Ser.*, vol.no. 7, page no. 22, 2017
- [2] <https://algorithmia.com/blog/introduction-natural-language-processing-nlp>
- [3] <https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/>
- [4] https://en.wikipedia.org/wiki/Natural_Language_Toolkit
- [5] <https://en.wikipedia.org/wiki/NumPy>
- [6] <https://www.dataquest.io/blog/sci-kit-learn-tutorial/>
- [7] <https://www.machinelearningplus.com/nlp/cosine-similarity/>
- [8] L. Chen, J. M. Jose, H. Yu, and F. Yuan, "A hybrid approach for question retrieval in community question answer in," *Computer. J.*, vol. 60, no. 7. pp. 1019-1031, 2016.
- [9] Y. Liu and X. Yang, "A similar legal case retrieval system by multiple speech question and answer." in *Proc. 18th Int.Conf. Electron. Bus. (ICEB)*, Guilin, China, 2018, pp. 72-81.
- [10] Z. Li, "A classification retrieval approach for English legal texts," in *Proc. Int. Conf. Intell. Transp., Big Data Smart City (ICITBS)*, Jan. 2019, Pp. 220-223
- [11] G. Greenleaf, A. Mowbray, and P. Chung, „Building sustainable free legal advisory systems: Experiences from the history of AI & law," *Comput. Law Secur. Rev.*, vol. 34, no. 2, pp. 314–326,2018