Admission Counselling



for Direct Second Year

**TABLE OF CONTENTS**

Page No.

Acknowledgement i

Abstract ii

List of Tables iii

List of Figures iv

Chapter 1: INTRODUCTION

Chapter 2: LITERATURE REVIEW

2.1 What is Django?

2.2 Functions of Project.

2.3 MySQL Database.

2.4 Web-Scrapping using Beautiful Soup and Request.

2.5 Extracting Data from PDF to CSV file using Camelot.

2.5.1 Why another PDF table extraction library?

2.5.2 Why name Camelot?

Chapter 3: PRESENT WORK

3.1 Main Home Page Urls.py File

3.2 Main Home Page views.py File

3.3 All India Colleges views.py File

3.4 All India Colleges urls.py File

3.5 All India Colleges admin.py File

3.6 All India Colleges models.py File

3.7 Web-Scrapping Extracting All India 200 Colleges:

3.8 Extracting Data from PDF to CSV file for Final Merit List

3.9 Storing the Data of Diploma Final Year Merit List In Database

Chapter 4: RESULTS AND DISCUSSION

4.1 Home Page

4.2 Display Colleges in Maharashtra

4.3 Cut-Off List

4.4 Show Seat Matrix

4.5 Top Colleges in Maharashtra for Admission

4.6 All India Colleges

4.7 All India Top 200 Colleges

4.8 Which College I Can Get Admission?

4.9 UML Diagram

Chapter 5: CONCLUSION AND FUTURE SCOPE

References

**List of Tables**

**-**

**List of Figures**

Table Title

4.1 Home Page

4.2 Display Colleges in Maharashtra

4.3 Cut-Off List

4.4 Show Seat Matrix

4.5 Top Colleges in Maharashtra for Admission

4.6 All India Colleges

4.7 All India Top 200 Colleges

4.8 Which College I Can Get Admission?

4.9 UML Diagram

**Abstraction**

Students reach online to the respective sites of their college and enter the preferences according to their interests. This type of setup is usually bottom up approach, where the last choice is given the first preference and then the latter going up the series.

Now as according to our markings when you are assigned courses then the courses which has been skipped can't be taken in most of the colleges. And then in next round of counselling it's time for upgrade if there is any scope of improvement based on your marks.

So as for spot round counselling they are carried out by colleges on their own, if there is any vacant seat then student personally approach the respective college and check by themselves and fill up the form and it's generally first come first serve.

To solve all this problem where you can get admission, which colleges to be filled in which order so that you can get best out of best college based on percentage you got.

**Introduction**

One of the reasons that students wasted their time in searching which college he/she can get admission but now with the help of this website they can easily find out how many colleges are available, where they can take admission and bunch of other things.

As per the programming perspective here we are going to use Python and its some of the framework like Django, Camelot and Machine Learning techniques. In this we are also going to use HTML, CSS and Bootstrap for look and feel of the website and JavaScript for Client-Side Validation to reduce the load of server.

The main challenge for creating this website is to create backend database as the data available in PDF format it is very hard to extract that kind of data and store into database.

**2.1 What is Django?**

Django is a free and open source web application framework, written in Python. A web framework is a set of components that helps you to develop websites faster and easier.

When you're building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc.

Luckily for you, other people long ago noticed that web developers face similar problems when building a new site, so they teamed up and created frameworks (Django being one of them) that give you ready-made components to use.

Frameworks exist to save you from having to reinvent the wheel and to help alleviate some of the overhead when you’re building a new site.

**2.2 Functions of Project:**

            There are seven functionalities provided by the Admission Counselling for Direct Second Year website.

• **Display colleges in Maharashtra**: This page will display colleges available in Maharashtra based on city and college type.

• **Top Colleges Admission:** This page will show top 10 colleges of Maharashtra where last year student have taken admission.

• **All India Colleges:** This page will show all engineering colleges of India.

• **Outside Maharashtra Colleges:** This page will display outside Maharashtra colleges which are giving admission for lateral entry.

• **Which College I can take Admission:** This page will display colleges where you can take admission based on certain parameter.

**2.3 MySQL Database:**

The name of MySQL is the combination of My and SQL, MySQL.

MySQL is a database management system that allows you to manage relational databases. It is open source software backed by Oracle. It means you can use MySQL without paying a dime. Also, if you want, you can change its source code to suit your needs.

Even though MySQL is open source software, you can buy a commercial license version from Oracle to get premium support services.

MySQL is pretty easy to master in comparison with other database software like Oracle Database, or Microsoft SQL Server.

MySQL can run on various platforms UNIX, Linux, Windows, etc. You can install it on a server or even in a desktop. Besides, MySQL is reliable, scalable, and fast.

The following information is to be stored in the database:

1) All India College’s list

2) Allocated College’s details

3) List of College’s in Maharashtra

4) Outside Maharashtra College List

5) Diploma Result List

**2.4 Web-Scrapping using Beautiful Soup and Request:**

According to Ryan Mitchell’s book, Web Scraping with Python (O’Reilly), it is the practice of gathering data through any means other than API. One can write a program that queries web servers, requests and retrieves data, parses it to extract information, and stores it to be analyzed later.

The first step into web scraping is to take a deep look at the page you are trying to scrape, you will need to open “Show/View Page Source” in the developer menu of the web browser of your choice. As Mitchell says, if you can see it in your browser, you can access it via a Python script. And, if you can access it, you can store it in a database to retrieve and analyze.

While inspecting the CFC webpage, few things become relevant. The CFC site’s provinces are enclosed in <h2> HTML headings, while the links and names of the foundations are in <h3> headings. Also, the links include the text ‘cfc\_locations’, which will help distinguish them from any other link.

How about parsing the information we just obtained? This is where the Beautiful Soup (BS4) library comes in. BS4 is a Python library for parsing HTML and XML documents — even for pages with malformed markup or poorly designed. It provides simple methods to navigate, search, and modify parse trees.

**2.5 Extracting Data From PDF to CSV file using Camelot:**

Camelot is a Python library that makes it easy for anyone to extract tables from PDF files.

You are in control. Unlike other libraries and tools which either give a nice output or fail miserably (with no in-between), Camelot gives you the power to tweak table extraction. (This is important since everything in the real world, including PDF table extraction, is fuzzy.)

Bad tables can be discarded based on metrics like accuracy and whitespace, without ever having to manually look at each table.

Each table is a pandas DataFrame, which seamlessly integrates into ETL and data analysis workflows.

Export to multiple formats, including JSON, Excel and HTML.

The PDF (Portable Document Format) was born out of The Camelot Project to create “a universal way to communicate documents across a wide variety of machine configurations, operating systems and communication networks”. The goal was to make these documents viewable on any display and printable on any modern printers. The invention of the PostScript page description language, which enabled the creation of fixed-layout flat documents (with text, fonts, graphics, images encapsulated), solved this problem.

At a high level, PostScript defines instructions, such as “place this character at this x,y coordinate on a plane”. Spaces can be simulated by placing characters relatively far apart. Extending from that, tables can be simulated by placing characters (which constitute words) in two-dimensional grids. A PDF viewer just takes these instructions and draws everything for the user to view. Since a PDF is just characters on a plane, there is no table data structure that can be extracted and used for analysis!

2.5.1 Why another PDF table extraction library?

There are both open (Tabula, pdf-table-extract) and closed-source (smallpdf, PDFTables) tools that are widely used to extract tables from PDF files. They either give a nice output or fail miserably. There is no in between. This is not helpful since everything in the real world, including PDF table extraction, is fuzzy. This leads to the creation of ad-hoc table extraction scripts for each type of PDF table.

Camelot was created to offer users complete control over table extraction. If you can’t get your desired output with the default settings, you can tweak them and get the job done.

2.5.2 Why name Camelot?

As you can already guess, this library is named after The Camelot Project.

Fun fact: In the British comedy film Monty Python and the Holy Grail (and in the Arthurian legend depicted in the film), “Camelot” is the name of the castle where Arthur leads his men, the Knights of the Round Table, and then sets off elsewhere after deciding that it is “a silly place”. Interestingly, the language in which this library is written (Python) was named after Monty Python.

**Project Source Code**

**3.1 Main Home Page Urls.py File:**

#AdmissionDirectSecondYear URL Configuration

from django.contrib import admin

from django.urls import path, include

from django.conf.urls import url

from . import views

from django.views.generic import TemplateView

urlpatterns = [

path('',include('AllIndiaColleges.urls')),

path('',include('CuttOffList.urls')),

path('',include('DisplayCollegesInMaharashtra.urls')),

path('',include('TopColleges.urls')),

path('',include('ShowSeatMatrix.urls')),

path('',include('TopMaharashtraCollegeList.urls')),

path('',include('WhereICanGetAdmission.urls')),

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

path('',views.home\_page,name="home"),

]

**3.2 Main Home Page views.py File:**

from django.shortcuts import render

from django.http import HttpResponse

# Create your views here.

def home\_page(request):

return render(request,'home\_page.html');

**3.3 All India Colleges views.py File:**

from django.shortcuts import render

from django.http import HttpResponse

from . models import AllIndiaCollegeModel

from django.db import connection

# Create your views here.

def all\_india\_college(request):

cursor = connection.cursor()

cursor.execute('select distinct state from allindiacollege2019')

row = cursor.fetchall()

state\_list = [item for i in row for item in i]

cursor.execute('select distinct city from allindiacollege2019')

row = cursor.fetchall()

city\_list = [item for i in row for item in i]

city\_get = ""

state\_get = ""

result = zip()

if request.method == "POST":

if request.POST.get("Vise"):

city\_get = request.POST['city']

state\_get = request.POST['state']

if city\_get != "Select City":

#cursor.execute("select \* from allindiacollege2019 where city = %s ",(city\_get,))

result = AllIndiaCollegeModel.objects.raw("select \* from allindiacollege2019 where city = %s order by state ",(city\_get,))

elif state\_get != "Select State":

#cursor.execute("select \* from allindiacollege2019 where state = %s ",(state\_get,))

result = AllIndiaCollegeModel.objects.raw("select \* from allindiacollege2019 where state = %s order by city ",(state\_get,))

if request.POST.get("All"):

result = AllIndiaCollegeModel.objects.raw("select \* from allindiacollege2019 order by state")

#result = cursor.fetchall()

return render(request,'all\_india\_college.html',{'city\_list':city\_list,'state\_list':state\_list,'result':result})

**3.4 All India Colleges urls.py File:**

from django.urls import path

from . import views

urlpatterns = [

path('All-India-Colleges/',views.all\_india\_college,name="all\_india\_college")

]

**3.5 All India Colleges admin.py File:**

from django.contrib import admin

from . models import AllIndiaCollegeModel

# Register your models here.

admin.site.register(AllIndiaCollegeModel)

**3.6 All India Colleges models.py File:**

from django.db import models

# Create your models here.

class AllIndiaCollegeModel(models.Model):

name = models.CharField(max\_length=300)

city = models.CharField(max\_length=300)

state = models.CharField(max\_length=300)

class Meta:

db\_table = "ALLINDIACOLLEGE2019"

**3.7 Web-Scrapping Extracting All India 200 Colleges:**

'''All India Top 200 Colleges by NIRF

Extracting Data from web

'''

import requests

import bs4

import mysql.connector

res = requests.get("https://www.nirfindia.org/2019/EngineeringRanking.html", verify=False)

#print(res.text)

soup = bs4.BeautifulSoup(res.text,'lxml');

#print(soup)

tabledata = soup.select('td')

seats = []

count = 0

s = ['','','','','','']

r = 0

b = 0

for i in tabledata:

count+=1

b +=1

if(count<=11):

if( b <= 7 and count > 2 ):

pass

else:

s[r] = i.getText()

r += 1

if(count == 11):

count = 0

b = 0

if(r == 6):

s[1] = s[1][:s[1].find('More')]

seats.append(s)

s = ['','','','','', '']

r=0

try:

# Connecting to Database

connection = mysql.connector.connect(host='localhost', database='Diploma', user='Atharva',

password='Password')

cursor = connection.cursor()

for i in seats:

cursor.execute(""" INSERT INTO IndiaTop200College2018 VALUES (%s,%s,%s,%s,%s,%s)""",(i[0], i[1], i[2], i[3], i[4], i[5]))

connection.commit()

except mysql.connector.Error as error:

connection.rollback() # rollback if any exception occured

print("Failed inserting record into IndiaTop200College2018 table {}".format(error))

finally:

if (connection.is\_connected()):

cursor.close()

connection.close()

# print("MySQL connection is closed")

**3.8 Extracting Data From PDF to CSV file for Final Merit List**

import camelot

count=1

for i in range(1,565):

# First Converting 4 page in pair to csv

no\_of\_pages = str(count)+"-"+str(count+3);

tables = camelot.read\_pdf('2018.pdf', pages=no\_of\_pages, parallel=True);

for j in range(4):

tables[j].df[2] =tables[j].df[2].str.replace("\n", " ")

path='Y:\\Direct Second Year\\Analysis\\Rec\\'+str(i)+'.csv'

tables.export(path, f='csv')

print(count)

count = count + 4

**3.9 Storing the Data of Diploma Final Year Merit List In Database**

import csv

import mysql.connector

import os.path

import pandas as pd

from mysql.connector import Error

from mysql.connector import errorcode

from datetime import datetime

def executeStore():

print("Hello")

count = 1

try:

# Connecting to Database

connection = mysql.connector.connect(host='localhost', database='Diploma', user='Atharva',

password='Password')

cursor = connection.cursor()

count = 1

g = 1

for i in range(1,173):

for j in range(4):

path = "Y:\\Direct Second Year\\Analysis\\Rec\\"+ str(count) +"-page-" + str(g) + "-table-1.csv"

g = g + 1

file = open(path, newline='')

reader = csv.reader(file)

header = next(reader) # the first line is the header

data = []

for row in reader:

MG = int(row[9])

SubGrop = int(row[11])

if (SubGrop == 301): # For Computer Branch Only

SLGMN = int(row[0])

APPID = row[1]

Name = row[2]

Gender = row[3]

DiplomaPer = float(row[4])

SSC = float(row[5])

SSCM = float(row[6])

SSCS = float(row[7])

SSCE = float(row[8])

CATEGORY = row[13]

data.append([SLGMN, APPID, Name, Gender, DiplomaPer, SSC, SSCM, SSCS, SSCE, CATEGORY])

print(data)

file.close();

if len(data) != 0:

# Loop to store data

for i in data:

cursor.execute(

""" INSERT INTO Diploma2018 (SLGMN , APPID , NAME , GENDER ,DIPLOMAMARKS ,SSC,SCCMATH ,SSCSCIENCE ,SSCENGLISH ,CATEGORY) VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)""",

(i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8], i[9]))

connection.commit()

# print("Record inserted successfully into Diploma2018 table")

print(count)

count += 1

except mysql.connector.Error as error:

connection.rollback() # rollback if any exception occured

print("Failed inserting record into Diploma2018 table {}".format(error))

finally:

if (connection.is\_connected()):

cursor.close()

connection.close()

# print("MySQL connection is closed")

executeStore()

**Output**

4.1 Home Page:

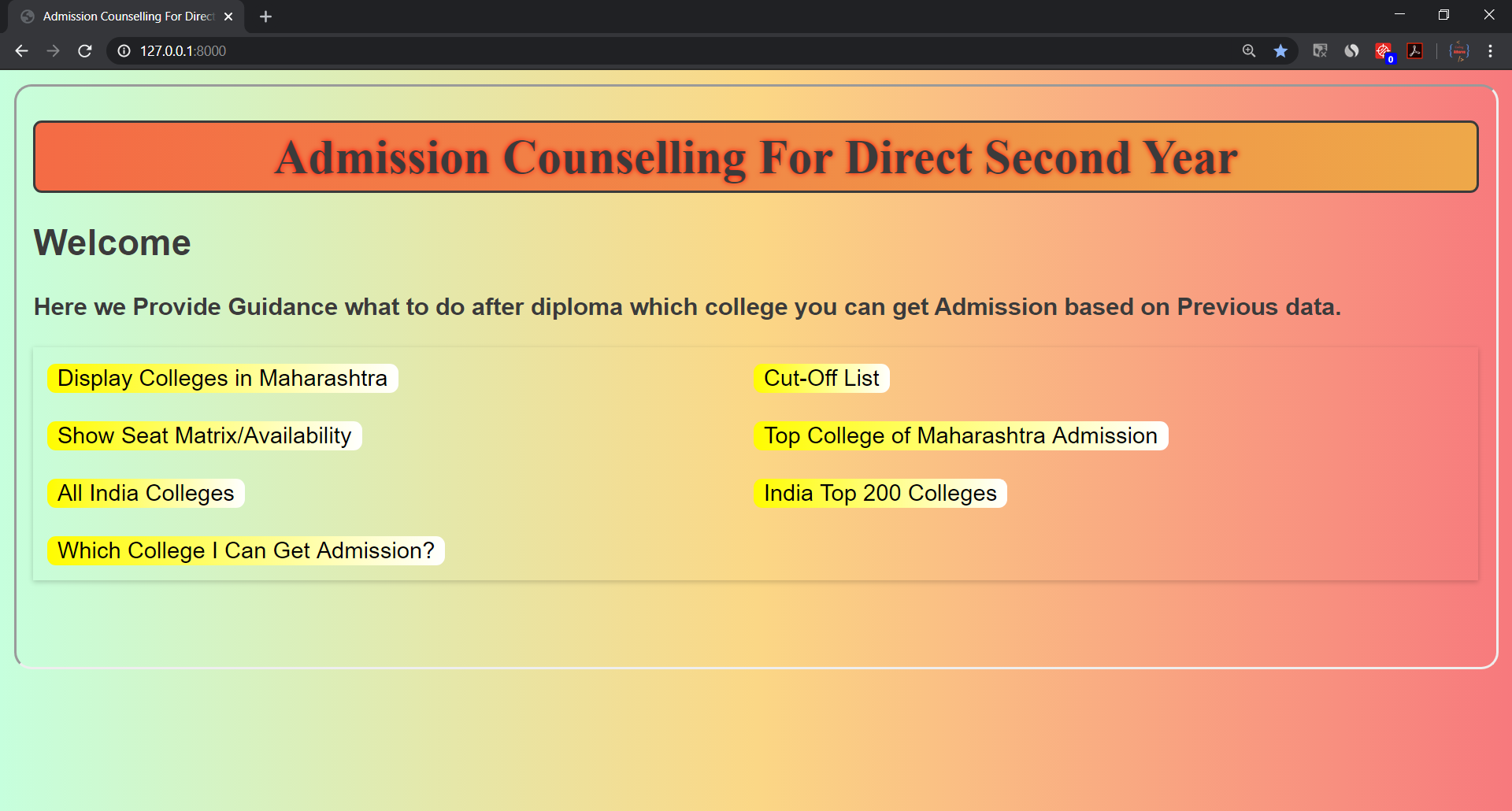


Fig: 4.1 Home Page

2) Display Colleges in Maharashtra:

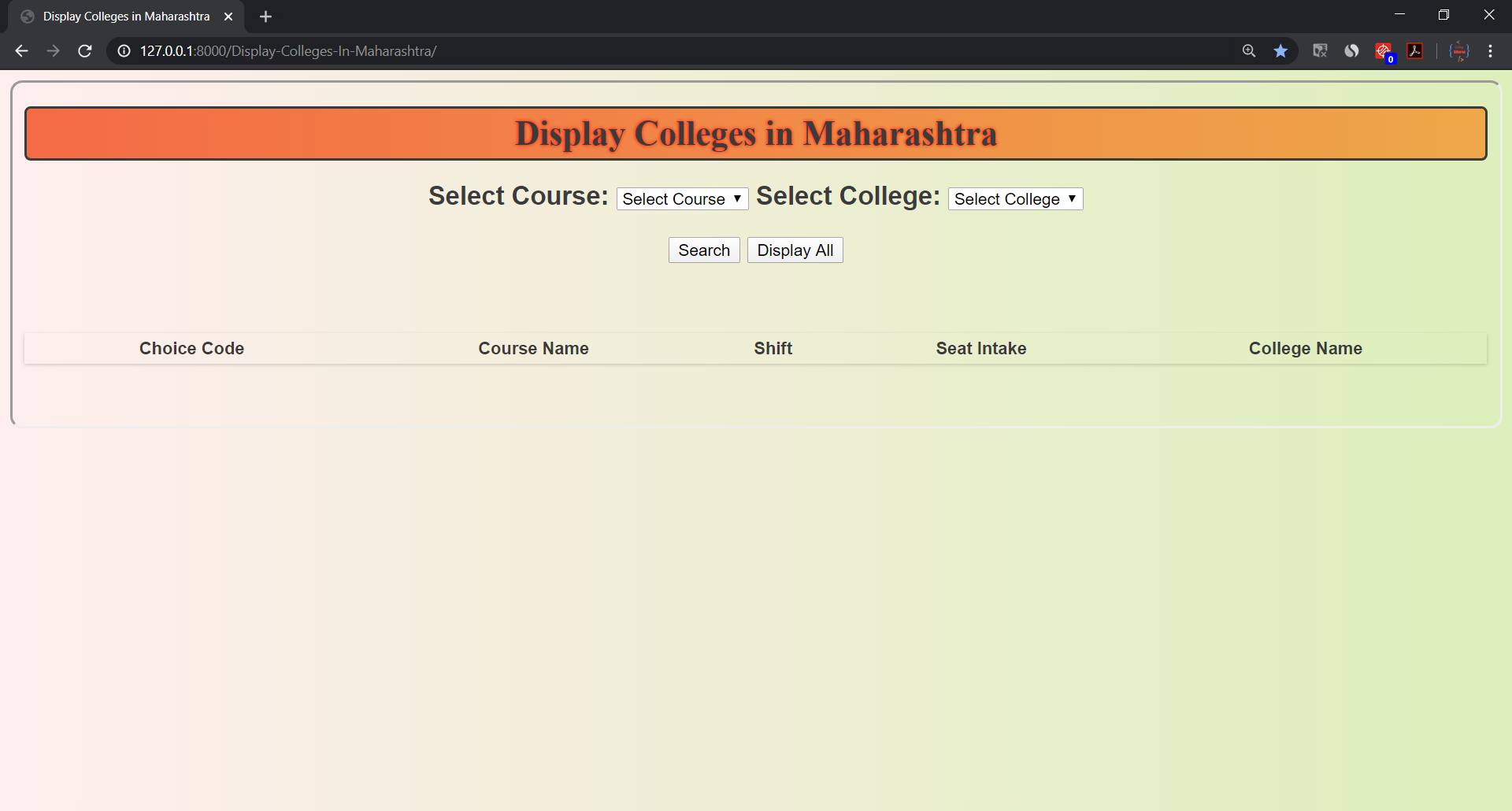


Fig: 4.2 Display Colleges in Maharashtra

3) Cut-Off List:

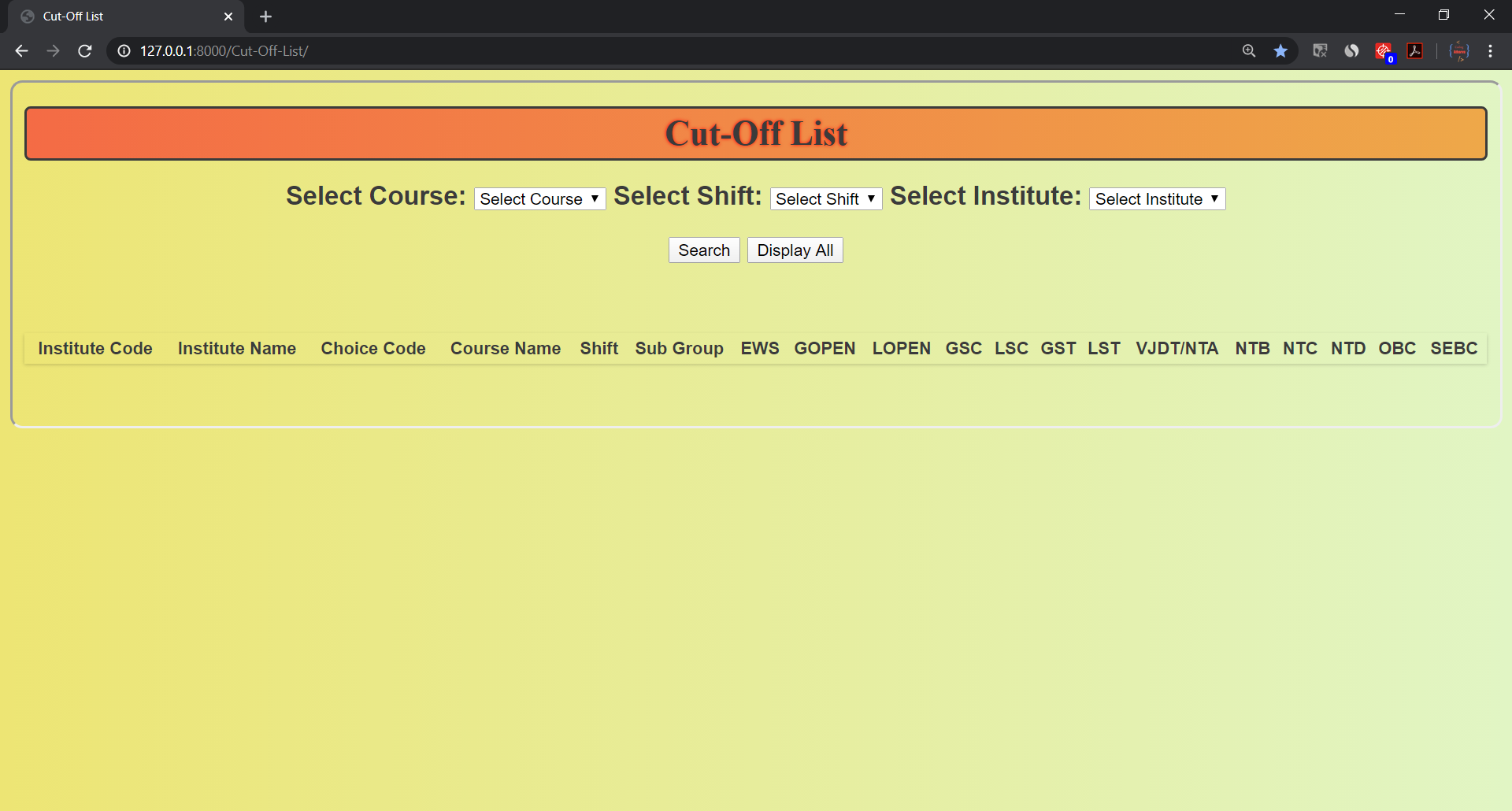


Fig: 4.3 Cut-Off List

4) Show Seat Matrix:

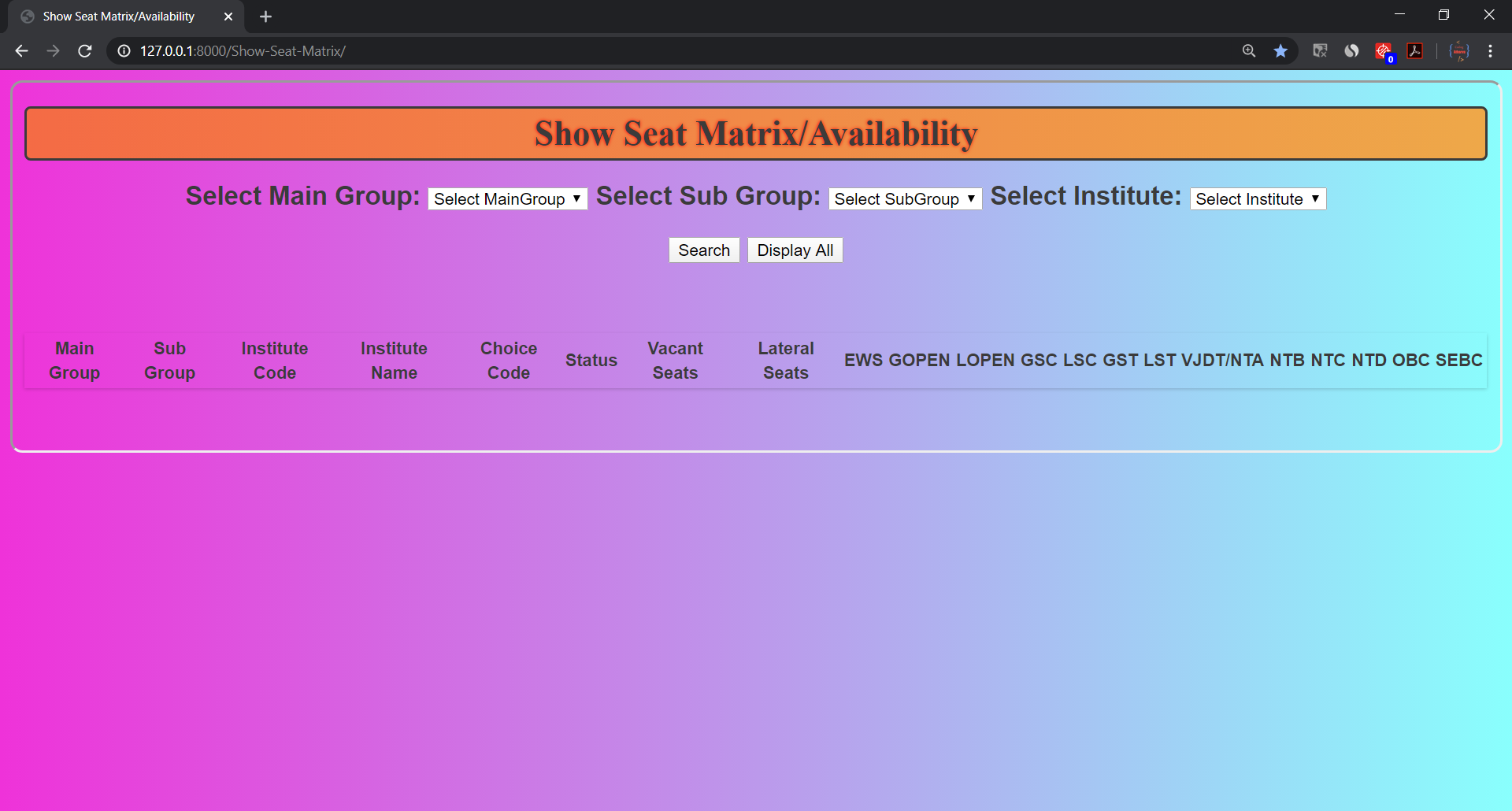


Fig: 4.4 Show Seat Matrix

5) Top Colleges in Maharashtra for Admission:

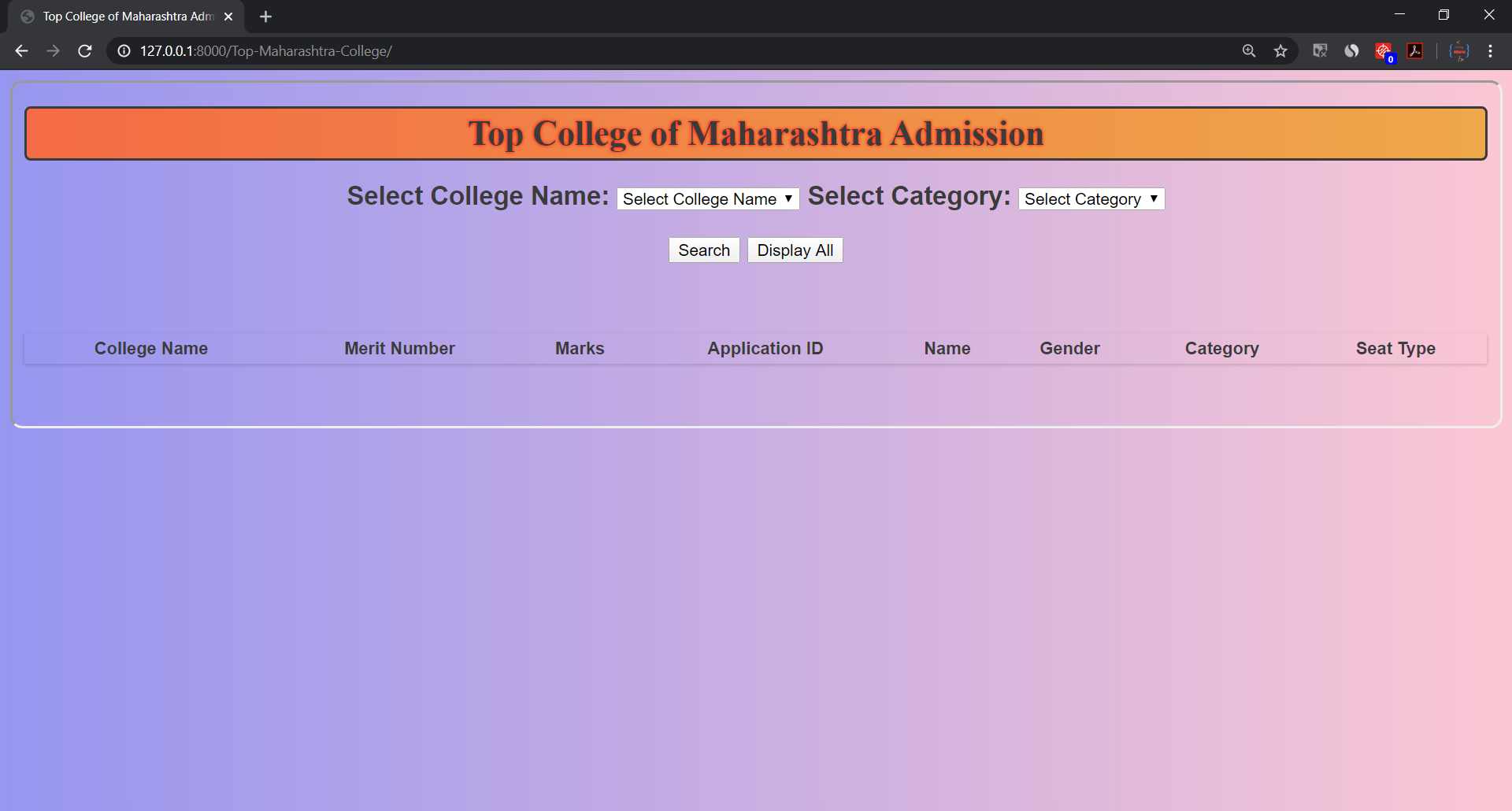


Fig: 4.5 Top Colleges in Maharashtra for Admission

6) All India Colleges

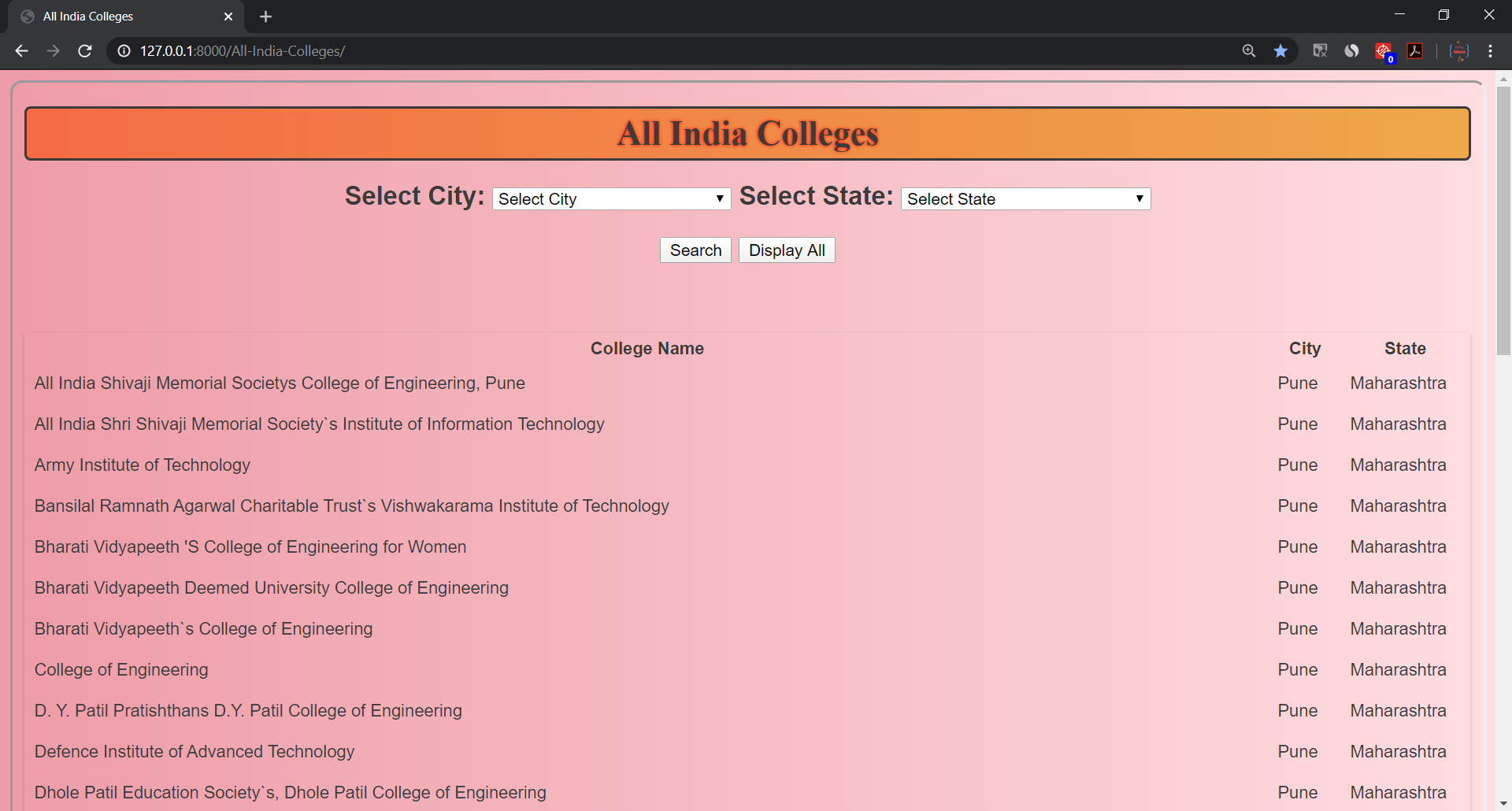


Fig: 4.6 All India Colleges

7) All India Top 200 Colleges

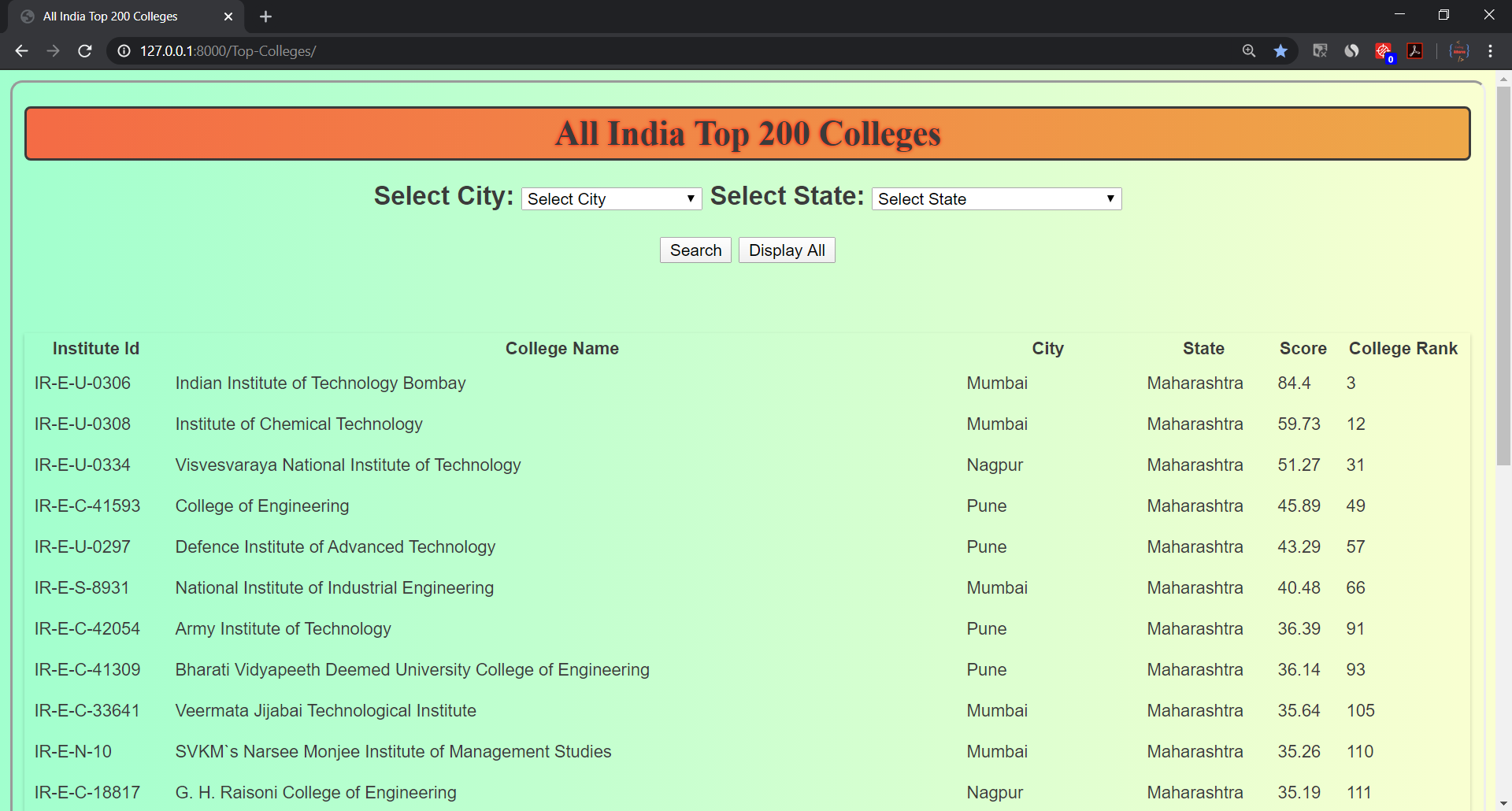


Fig: 4.7 All India Top 200 Colleges

8) Which College I Can Get Admission?

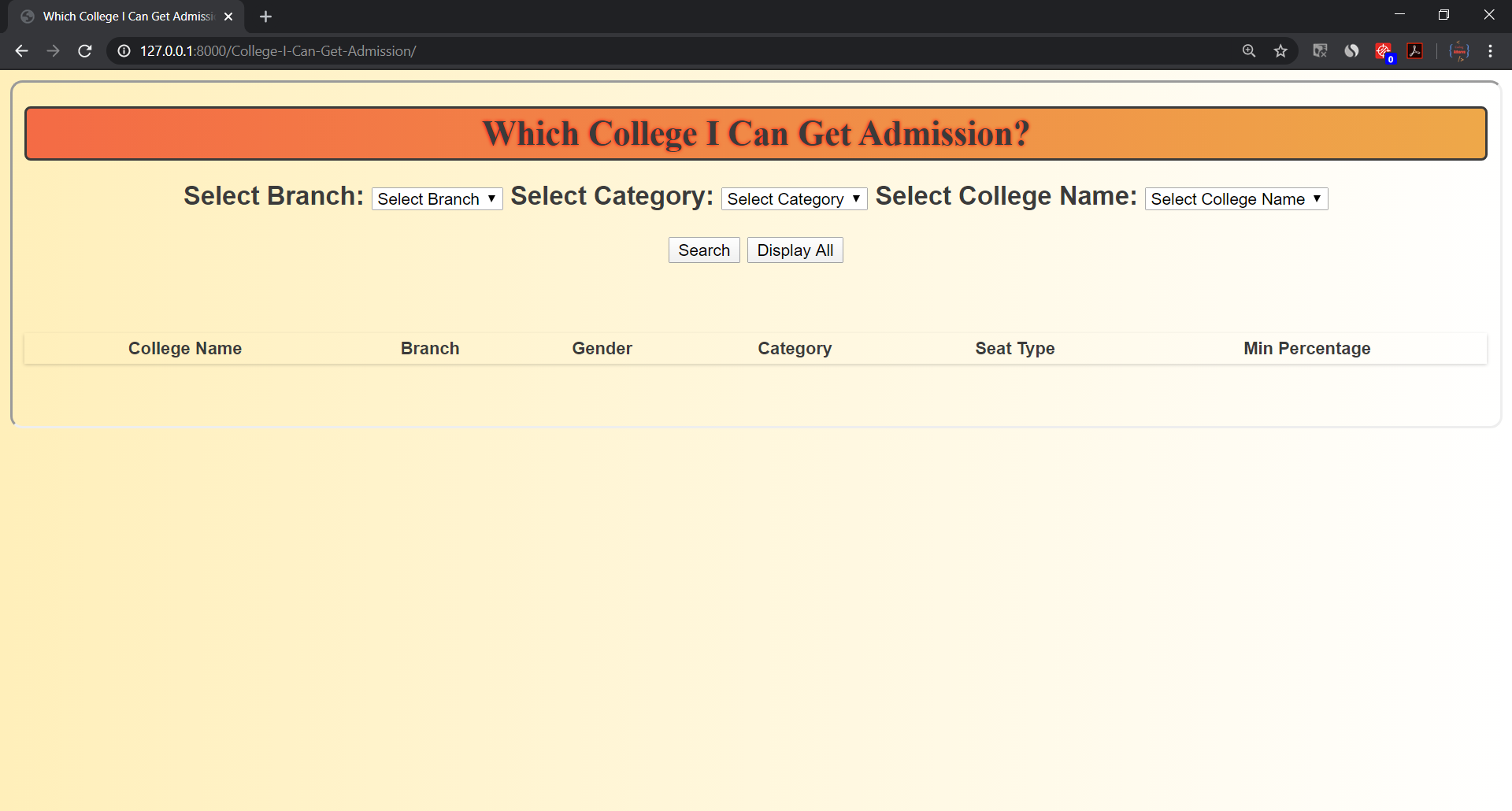


Fig: 4.8 Which College I Can Get Admission?

**UML Diagram:**

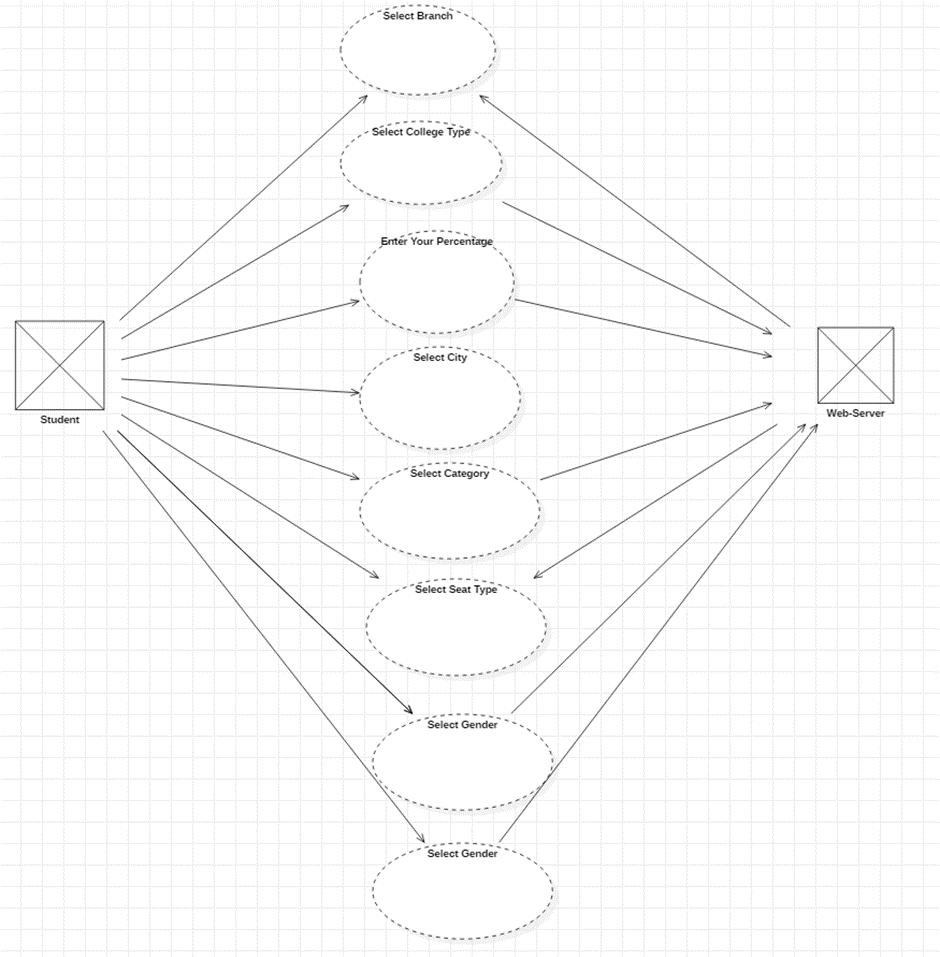


Fig: 4.9 UML Diagram

**CONCLUSION**

In this project, we had created Admission Counselling for Direct Second Year. We had abled to implement various function and classes in the program. We had used Visual Studio Code for creating Admission Counselling for Direct Second Year website. We also understood how to link different views and database. And we understand how to read data from MySQL database into Django views.

**FUTURE SCOPE**

The project is regarding admission analysis and as in this we had implement for diploma. We can add in future for NEET, JEE, BTech admission. We can create student login and provide special services for save data for college selected.

**REFERENCES**

1. python.org

2. dse19.mahacet.org.in

3. djangoproject.com

4. quora.com

5. stackoverflow.com