LEGAL RECOURSE

"Not just a lawyer, find your legal coach"

MINOR PROJECT REPORT

Under the Guidance of:
Prof. Sudesh Rani
Assistant Professor
Computer Science Department
PEC University of Technology

Submitted by:
Vyom Tayal (14103021)
Akash Attri (14103048)
Abhishek Kasana (14103049)

Department of Computer Science and Engineering

PEC University of Technology, Chandigarh

August to November, 2016

DECLARATION

We, hereby, declare that the project work entitled **"LEGAL RECOURSE"** is an authentic record of our own work carried out as requirements of six months Minor project during 5th Semester of degree of B.E. Computer Science and Engineering, PEC University of Technology, Chandigarh, under the able guidance of **Prof. Sudesh Rani**, during August to November, 2016.

Date: _____

Vyom Tayal (14103021)
Akash Attri (14103048)
Abhishek Kasana
(14103049)

Certified that the above statement made by the students is correct to the best of my knowledge and belief.

_____

Prof Sudesh Rani
Assistant Professor
CSE Department
PEC University of Technology

ACKNOWLEDGEMENT

We avail this opportunity to express our profound sense of sincerity and deep gratitude to all those who have played an integral role in the accomplishment of the project work by providing their willing guidance and help.

We wish to express our deep gratitude for **Prof. Sudesh Rani**, Assistant Professor and Capstone Coordinator for her continual kind words of encouragement and motivation throughout the project. Without his abiding inspiration, generous guidance and encouragement, we would not have been able to cope with the project work. The faith and confidence he kept has been the driving force behind each step taken for the completion of this project.

We are also grateful to the college and staff of CSE Department, who taught the fundamental essentials and gave us an opportunity to take up this project which has been a great learning experience for us.

Vyom Tayal (14103021)
Akash Attri (14103048)
Abhishek Kasana(14103049)

ABSTRACT

The main objective of **Legal Recourse System** is to develop a smart **Lawyer Recommendation System (LRS)** for judicial matters. The system designed aims at making Lawyer Selection for any course fully automatic, simple and effective. Getting a competent lawyer for any type of case, be it Adoption, Murder, Embezzlement etc, and also getting legal advice on such matters at a reasonable price , can be of significance. A traditional approach to consult your family and friends to get recommendations but then our choice is based purely on trust and nothing else. Through our portal, the recommendation will be provided by our system which gets its data from the long and diverse history of court cases in India. The autonomous data crawler traverses the site : www.ecourtservices.com and collects data of cases handled by district level courts along the previous years. The User can put in his case into our portal along with the type of the case which will then be matched with the types of cases previously stored in our database. Therefore Lawyers which specialize in that particular type of court cases will be considered and provided a position in the Result list according to Our Advanced Ranking System.

The User will get effective legal advice without having to consult anyone and will also get some contact info on the subsequent lawyer so as to approach him directly. Thus without spending a penny he will get to know the best suited lawyer for his case , that too based on the data extracted from the court cases that have happened over several years.

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

Legal Recourse is an interactive online platform that makes it faster and easier to find and hire the best Lawyers in any city / court in India, because you deserve access to first-rate, professional legal advice from the best Lawyers out there. We are on a mission to make the legal experience remarkable. By making legal services high quality, cost-effective and on-demand for every need.

Today, there is a plethora of lawyers in our society and the number is so vast that it makes choosing an effective lawyer nearly impossible. Our project utilizes a data on Civil court cases along these years to provide a ranking system to the lawyers according to their win-loss percentage making choosing an effective lawyer a lot more easier for the not-so legally sound indian.

1.1 DATA CRAWLING:

A Web crawler is an Internet bot which systematically browses the World Wide Web, typically for the purpose of Web indexing (web spidering).

Web search engines and some other sites use Web crawling or spidering software to update their web content or indices of others sites' web content. Web crawlers can copy all the pages they visit for later processing by a search engine which indexes the downloaded pages so the users can search much more efficiently.

Crawlers consume resources on the systems they visit and often visit sites without tacit approval. Issues of schedule, load, and "politeness" come into play when large collections of pages are accessed. Mechanisms exist for public sites not wishing to be crawled to make this known to the crawling agent. For instance, including a robots.txt file can request bots to index only parts of a website, or nothing at all.

How does web crawling work

Web crawlers for data extraction are built by technical personnel with programming skills. The first step in the process is identifying sources for data extraction. The sources have to be reliable sites since the quality of data and smoothness of the process will depend on the source websites. Once the sources are defined, the data points that have to be extracted from these sources must be defined. Next step is to program the crawler to navigate through the list of websites and extract the required data points. In order to extract data points, the person setting up the crawler has to find out html tags associated with every data point that is required. Once the setup is done, the crawler can be run in desired frequency depending upon the specific data requirements.

What data can you acquire with web crawling ?

There is no limit as to what data you can get from the web using a web crawler. Some great applications of web crawling are in ecommerce, recruitment, content aggregation, brand monitoring, business intelligence, manufacturing and market research. With its automation capabilities, robustness, speed and flexibility to scale up, web crawling is the best solution for acquiring data for any of these domains.

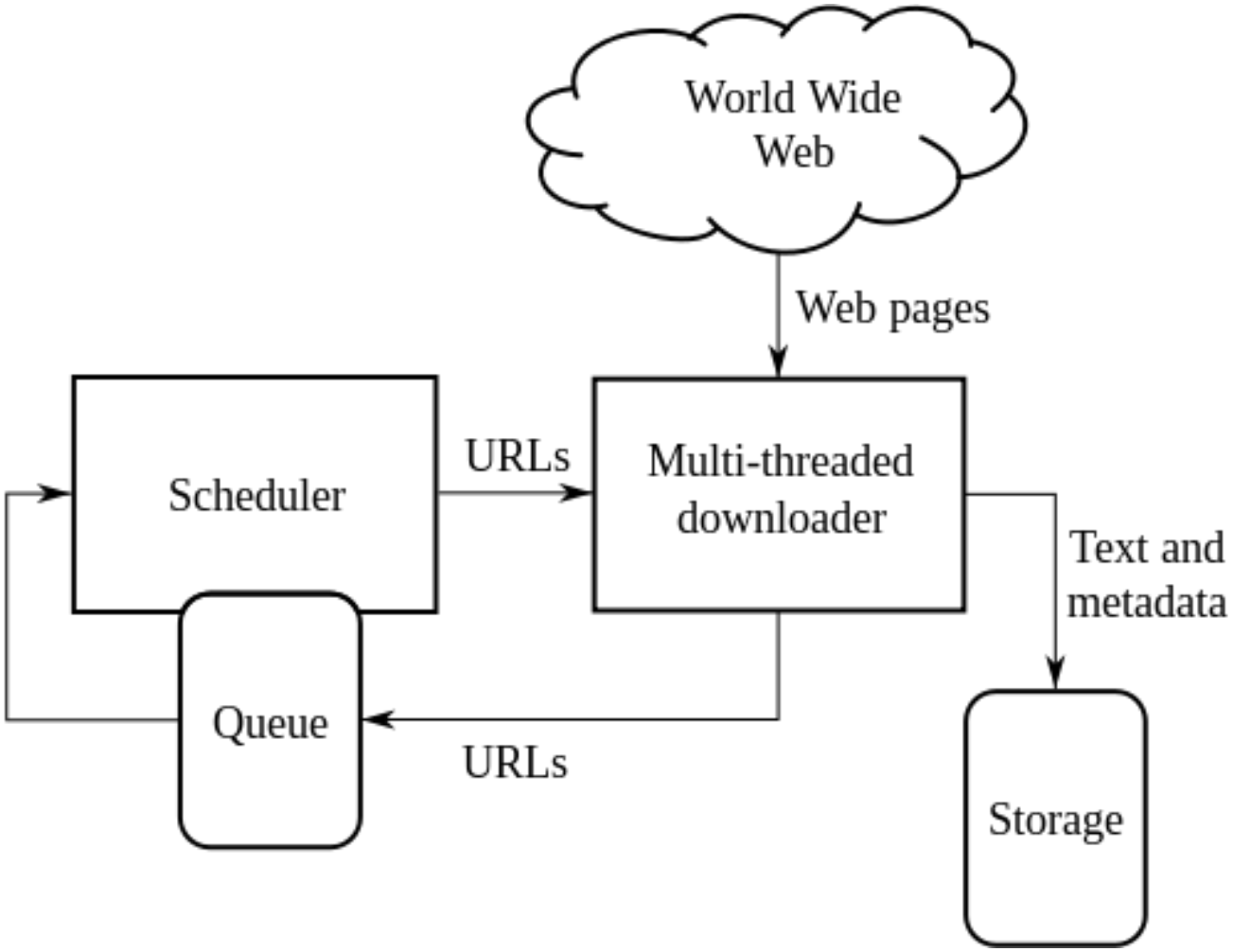


Figure 1.1 : Flow Chart of working of Web Crawler

1.2 LAYOUT OF THE PROJECT

- **Court Case Info Crawler:** Initially, the Court Case info crawler will crawl over data of E-court services website.
- **Lawyer Info Received :** The crawler will return the info on the lawyers to the database which will be saved in .csv format.
- **Rating Assigned** : A rating is assigned to each lawyer based on the system discussed on the following pages.
- **INFO Crawler** : This is the second crawler that picks lawyer name from the database and searches for the name in justdial.com to get contact info on that particular lawyer.
- **Info found :** If the contact info is found by the Info Crawler then that info is sent back to the database and it is updated accordingly. But if no data is found then there are no modifications made to the database.
- **Ranking List finalized** : A final ranking list is made on the ratings that were assigned to different lawyers of that particular domain. The list is output to the user through the medium of the web interface.
- **Output to the user** : Here, the final result is shown to the user that consists of the lawyer suggestion list on the user's specific case type.

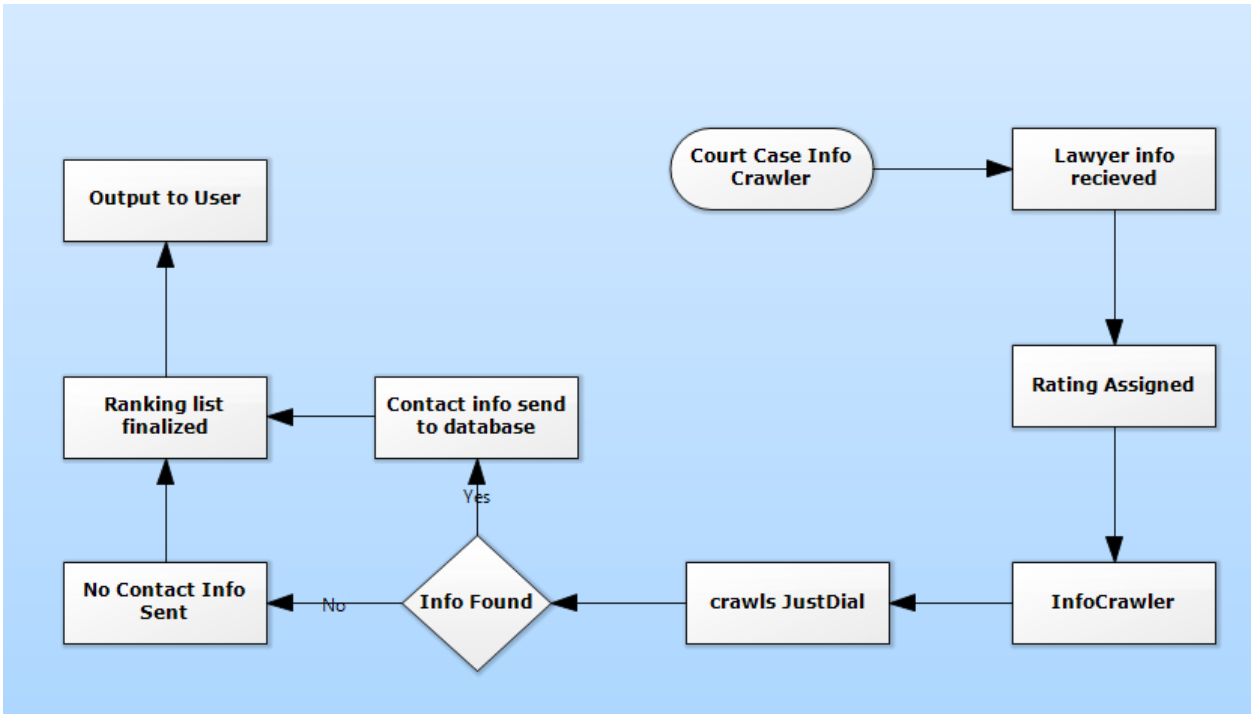


Fig 1.2 : Flowchart describing steps of the project

2. MOTIVATION BEHIND THE PROJECT

The judiciary (also known as the judicial system or court system) is the system of courts that interprets and applies the law in the name of the state. The judiciary also provides a mechanism for the resolution of disputes. In some nations, under doctrines of separation of powers, the judiciary generally does not make law (which is the responsibility of the legislature) or enforce law (which is the responsibility of the executive), but rather interprets law and applies it to the facts of each case. In other nations, the judiciary can make law, known as Common Law, by setting precedent for other judges to follow, as opposed to Statutory Law made by the legislature. The Judiciary is often tasked with ensuring equal justice under law. There are a variety of ways that lawyers can benefit humanity:

In any society, disputes between people will arise. In a *civil* society, we expect people to manage these disagreements in a non-violent, non-confrontational way. Lawyers (and judges as an extension) have the knowledge, the skill, and the position in society to intervene between two parties to a dispute, and to resolve the situation equitably. That is the job of a lawyer.

- There is a large body of the world that cannot speak for itself - the environment, animals, neighborhoods, etc. All of these groups deserve a voice in the decisions and policies that are being made which may affect them. Lawyers are able to take up these interests and present them before the bodies that are making such decisions in a way that they can understand and hopefully agree with. That is the job of the lawyer.
- In any society where the State has the power to remove someone's liberty, there must be process in place to ensure that such an act is only done based on reasonable and valid evidence of wrongdoing. No individual citizen should be expected to be able to stand toe-to-toe with the government and ensure that their rights are protected. That is the job of the lawyer

After thorough research and analysis, we decided that we want to do something for the betterment of the condition of legally absent people in our country via technology, and hence came the idea for making a Lawyer Recommendation System (LRS) or LEGAL RECOURSE as we call it. The project aims at helping the user through an efficient, fully automatic, very simple, effective to manage and affordable platform for giving free legal advice to the user on which lawyer is best suited for their job. They are saved the overhead of making frequent visits to different lawyer's houses just for consulting them on their cases to choose one among them. Our platform uses Data Cawling and a ranking system to automate the entire process. The First crawler first crawls the website of www.ecoutservices.com and collects all the data it can on the type of court case, which lawyer won and which lost. There are some cases where there is an in-between settlement and therefore we have an appropriate ranking system which gives rating to the lawyers on their performance in previous court cases and present the lawyer names in that order to the general user. Thereon the second crawler crawls various lawyer info websites for extracting data on the lawyer that was presented to the user and provides various info such as address, contact number etc.

This project aims at providing free aid in lawyer selection to the legally unsound Indian.

3. OBJECTIVE OF THE PROJECT

Lawyer Recommendation System (LRS) seeked motivation for building an efficient and automatic portal to provide legal advice on lawyers to the general public. The objectives of this project are to:

- Develop an autonomous system for the society which will process data of the various court cases in the history of India at its backend.
- Using that processed data it will come up with a suggestion list of viable lawyers (ordered in descending order of proficiency in that field can who can fight that particular type of case.
- Develop an appropriate ranking system that can provide rating to each lawyer in accordance to his win loss percentage making special arrangement for cases that are in-between settlements.
- Develop another web crawler for collecting contact info on the lawyers so as to equip the user with means to contact the lawyer immediately.
- To provide an interactive interface to the user where he will put in his case and make his lawyer selection more interactive.

Legal Recourse (Lawyer Recommendation system) aims at making the lawyer selection process for any user to be autonomous. The web portal is equipped with 2 web crawlers that can crawl their respective sites and provide our portal with updated feed on the current lawyer situation in India. All this is done using libraries in python and Html/CSS for the design of the website. Also the Lawyers are recommended on the basis of an Appropriate Ranking System and not in any random order. This gives the user surety that the recommended lawyer is well versed in the field of the user's case as the position of the lawyer in the list is directly proportional to his win percentage which is the most viable option to determine a lawyer's proficiency in fighting court cases.

Also another problem faced by the society is “How to get appropriate contact info on the suggested lawyer?”. This is solved by the second data crawler that is employed by our website to crawl lawyer info websites to get info regarding the lawyers in the list provided to the users making the automation process complete.

Therefore the person will get the best possible advice on the lawyer that he should employ to fight your court case and also appropriate info on the lawyer on how to contact him immediately.

4. BACKGROUND

4.1 Problems with the existing system

- Locating a good lawyer who can efficiently help with your particular problem may not be easy. Don't expect to locate a good lawyer by simply looking in the phone book or reading an advertisement. There's not enough information in these sources to help you make a valid judgment.
- Making a decision about a lawyer solely on the basis of someone else's recommendation is not appropriate. Different people will have different responses to a lawyer's style and personality.
- In a legal system as complex as India’s the problem of getting the right legal representation can have a major impact on the timely resolution of your problem.
- The best lawyer with whom you have the maximum chance of winning is the top priority of an individual entangled in any legal case.
- Therefore our system solves this exact problem.

5. PROPOSED WORK

- The project provides a comprehensive profile for each attorney with information that will help you select the right attorney. The profiles tell you about the lawyer's experience, contact information.
- This Project confirms that every listed attorney has a valid license and is in good standing with their bar association.
- This project also suggest the most well suited lawyer for your particular case on the basis of all the data mined from various judiciary websites.
- This data consists of 'related' past cases, their verdict, the lawyers involved and other information that would help in suggesting a suitable lawyer.
- Another problem faced by the society is "How to get appropriate contact info on the suggested lawyer?". This is solved by the second data crawler that is employed by our website to crawl lawyer info websites to get info regarding the lawyers in the list provided to the users making the automation process complete. The figure 5.1 refers to the first level DFD of the project.

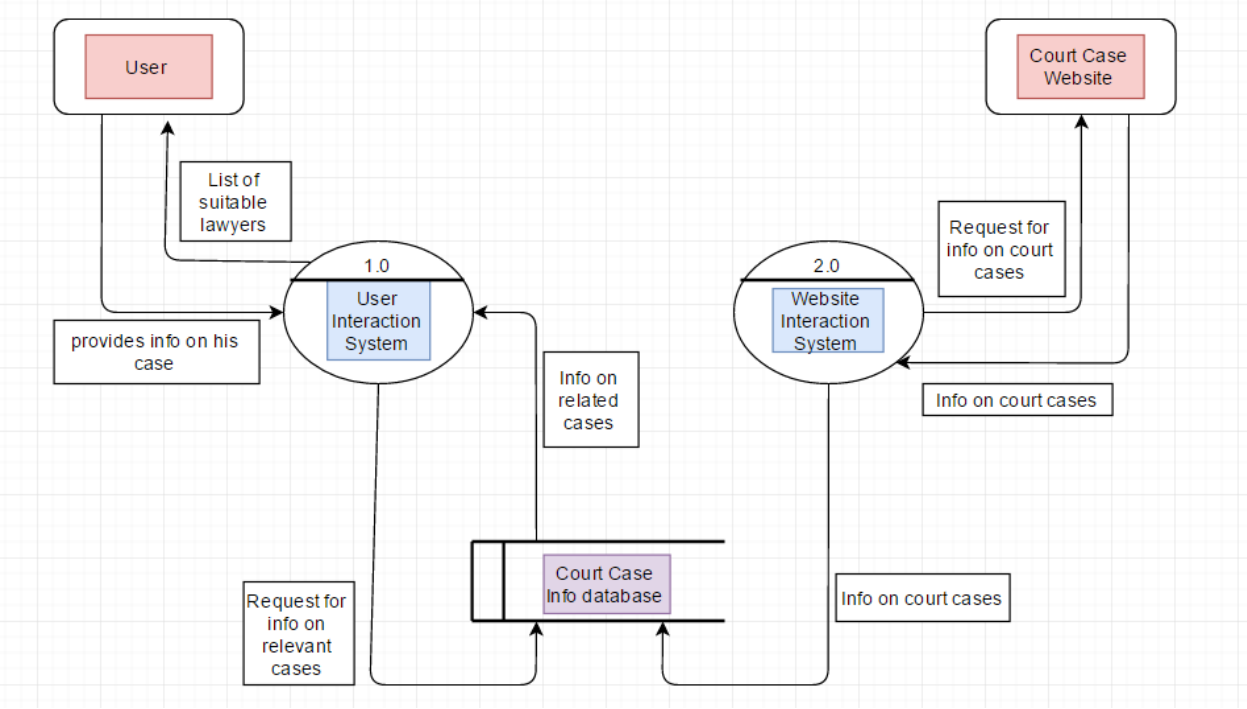


Figure 5.1 First level DFD of the project

6. IMPLEMENTATION DETAILS

6.1 Court Case Crawler

The data crawler made crawls the services.ecourt.gov.in/ecourtindia which provides data on the court cases of district level in India. This site needs the following information for the data to be displayed:

- Duration of the time for which court case data is needed.
- Type of the court Case.
- Captcha
- The state name for which court case info is needed.
- The district name for which court case info is needed.

And the crawler uses the following libraries for it's working.

6.1.1 SELENIUM LIBRARY

Selenium is an umbrella project for a range of tools and libraries that enable and support the automation of web browsers.

6.1.1.1 Working

Due to limitations in conventional crawling techniques for crawling of dynamically rendered pages, we used the browser-automation tool , Selenium.

The working of selenium is as follows :
At the core of Selenium is WebDriver, an interface to write instruction sets that can be run interchangeably in many browsers
Here is one of the simplest instructions you can make:

```
from selenium import webdriver
driver = webdriver.Firefox()
driver.get("http://google.com/?hl=en")
search_box = driver.find_element_by_id("q")
search_box.send_keys("cheese")
search_box.submit()
```

After going to a particular page, it locates elements based on their HTML attributes like class name, ID name, tags etc. It also provides actions like button click, filling forms etc.

The project uses a python library Selenium which handles all the crawling business.

It provides extensions to emulate user interaction with browsers, a distribution server for scaling browser allocation, and the infrastructure for implementations of the W3C WebDriver specification that lets you write interchangeable code for all major web browsers. The project is made possible by volunteer contributors who have put in thousands of hours of their own time, and made the source code freely available for anyone to use, enjoy, and improve.

6.1.2 CAPTCHA

The Project uses a Python Library Tesseract which handled the Captcha by recognizing the screenshot of the page having the captcha.

6.1.2.1 Working

The following steps were performed by our crawler using the library :

- First the page containing the captcha is opened.
- A screenshot of the page is taken by the crawler.
- Screenshot is cropped till we get only the image of the captcha.
- Image Recognition is done using Tesseract Library.
- The result text of the Captcha is returned back to our crawler which then inserts it into the captcha text field.
- Then it advances forward to the next page containing the info of cases

6.1.2.2 Tesseract

Tesseract is an open source Optical Character Recognition (OCR) Engine, available under the Apache 2.0 license. It can be used directly, or (for programmers) using an API to extract typed, handwritten or printed text from images. It supports a wide variety of languages.

Tesseract doesn't have a built-in GUI, but there are several available from the 3rdParty page.

6.1.3 . CSV file

A Comma separated values File is used by us as a backend database to store values :

- Name of the Lawyer
- Rating of the Lawyer
- Type of Case
- Location

In computing, a comma-separated values (CSV) file stores tabular data (numbers and text) in plain text. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. The figure 6.1.1 refers to the flow chart of court case info crawler.

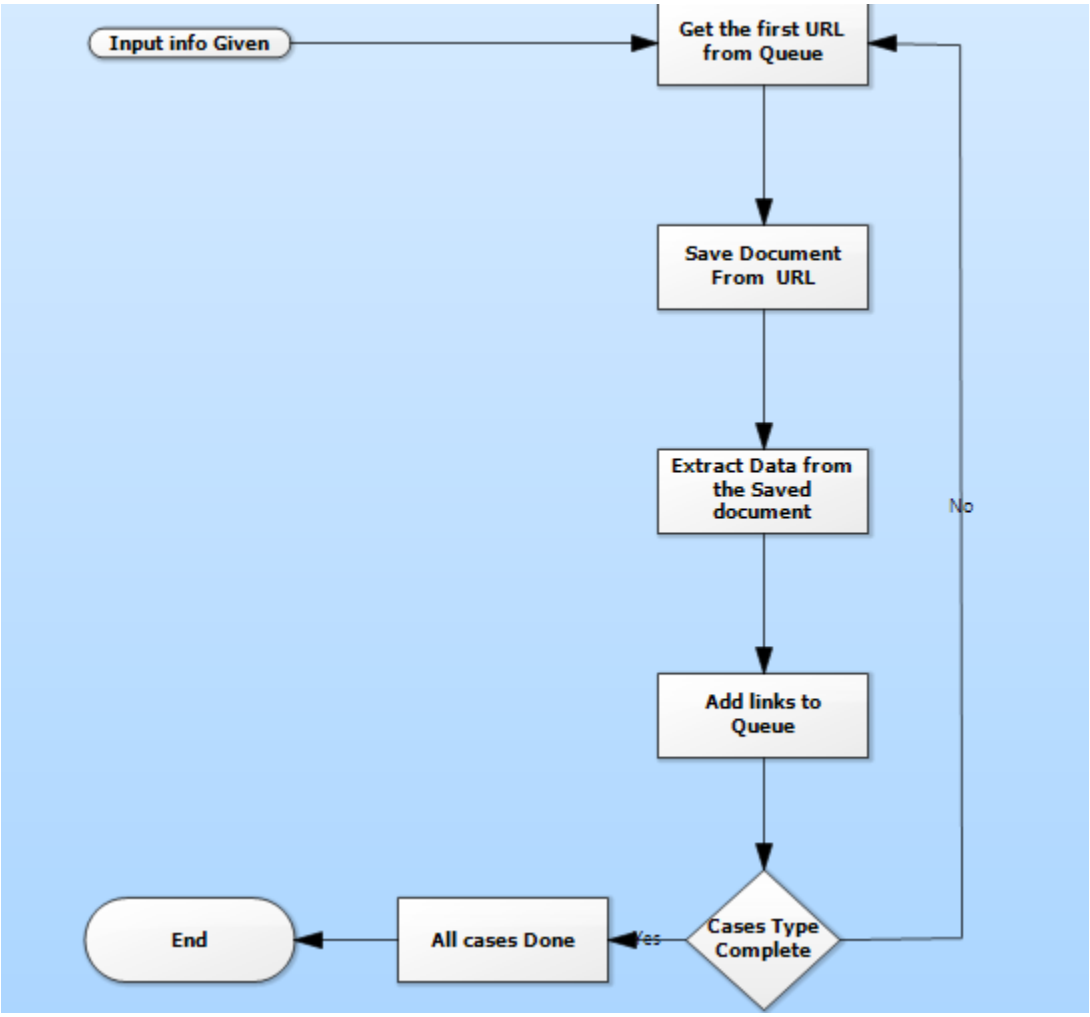


Figure 6.1.1 Court case Info crawler‘s flow chart

6.2 CONTACT INFO DATA CRAWLER

6.2.1 INTRODUCTION

This is a python crawler using Selenium library which doesn't crawl for data this time. Since the first crawler returns a lawyer suggestion list, a single name is picked up from the list and that name is searched in the justdial.com website.

6.2.2 WORKING

This is the working of the crawler :

- A name is picked up from the suggestion .csv file to find his contact info.
- For each entry in the lawyer database the lawyer's name is searched on the jusdial.com website using Selenium Web Driver.
- If it can be established that the output is that of a lawyer then we collect his/her contact info and update our database.
- Also the rating of that particular lawyer is modified according as a lawyer with contact info is more sought out for.

6.2.3 SELENIUM LIBRARY

This is a full fledged library of python which works in the following way :

- Firstly , it opens up the web browser and enters the customized search URL: it towards justdial.com
- Then it goes to the search list and matched it with the lawyer name.
- Then it forwards the data to python if it gets a match.

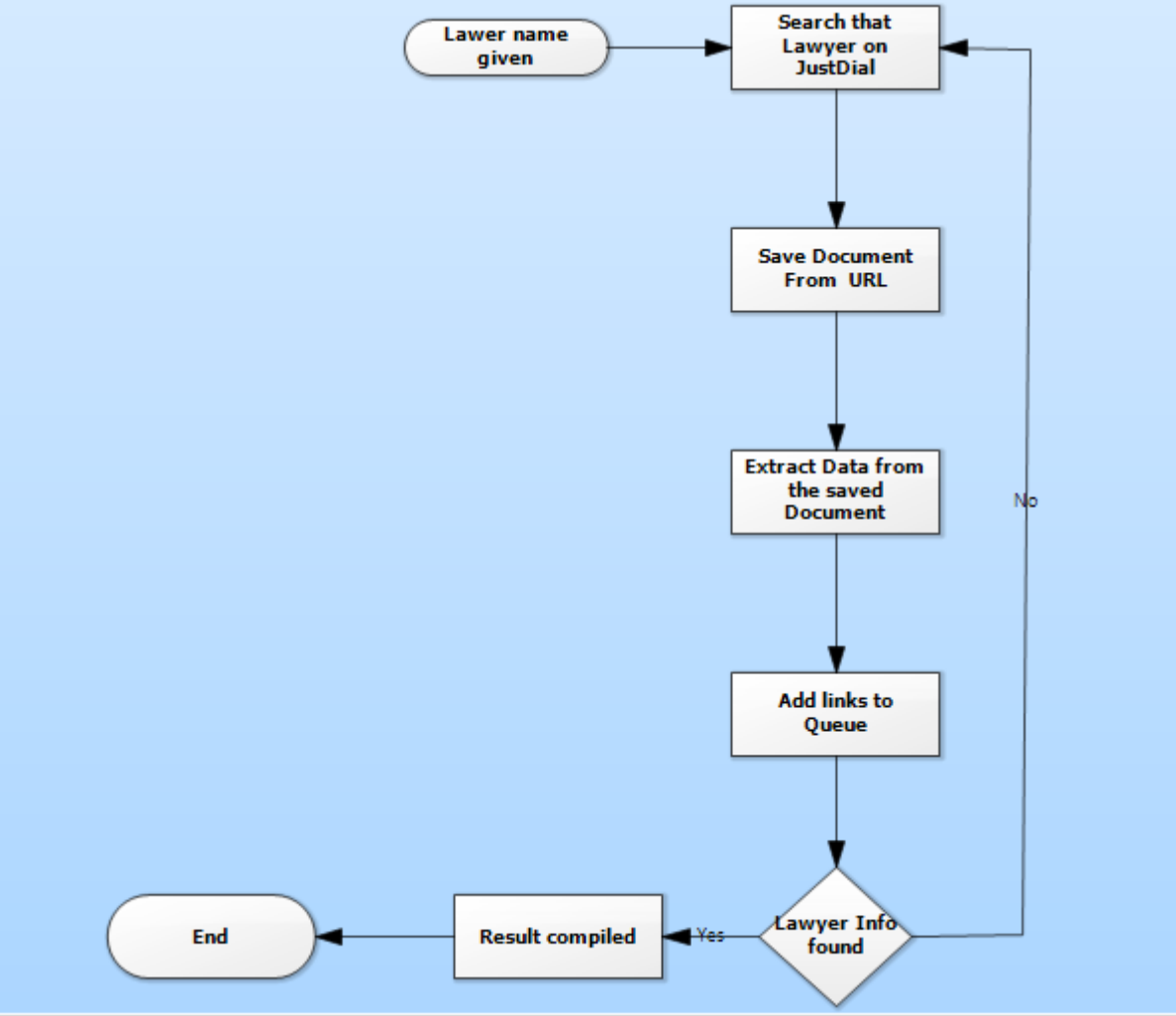


Figure 6.2.1 Contact info crawler's flow chart

6.3 RATING SYSTEM

We provide rating to the lawyers in accordance with the data crawled from the services.ecourt.gov.in/ecourtindia. This rating is then used to provide a final order of Ranking to them to be displayed when the final suggestion List of Lawyer.

Also the rating is modified by the second crawler too.

The second crawler provides the contact info on the lawyers and if we get contact info on them, then we increase the rating to move it up in the lawyer suggestion list as a lawyer whose contact info is available will be more sought out for than the other ones.

6.3.1 E-Court Services Data rating.

Here the rating is provided according to the following scheme:

- 10 points are awarded if the lawyer won the case straightaway.
- 4 points are awarded if the lawyer lost the case in that domain of "Case type".
- 8 points are awarded in case of in-between settlement among the two parties.

6.3.2 Contact info found

Here the ratings of the lawyers is modified if their contact info is found as by the second crawler as a lawyer whose contact info is found will be better than other available as approaching him/her would be easier increasing the usablility for the user.

If the info on that particular lawyer is found then it is returned back to the database and appropriate modifications are made in the database so as to accommodate contact info too.

6.4 WEB PORTAL

The web portal has been coded in HTML CSS. It provides an interactive interface to the user where he can put up the type of case for which he wants a lawyer . The interface is connected through Django at the backend.

SCREENSHOTS:
The figure 6.4.1 refers to the front end of our web portal.

The figure 6.4.2 refers to the different type of case domains available.

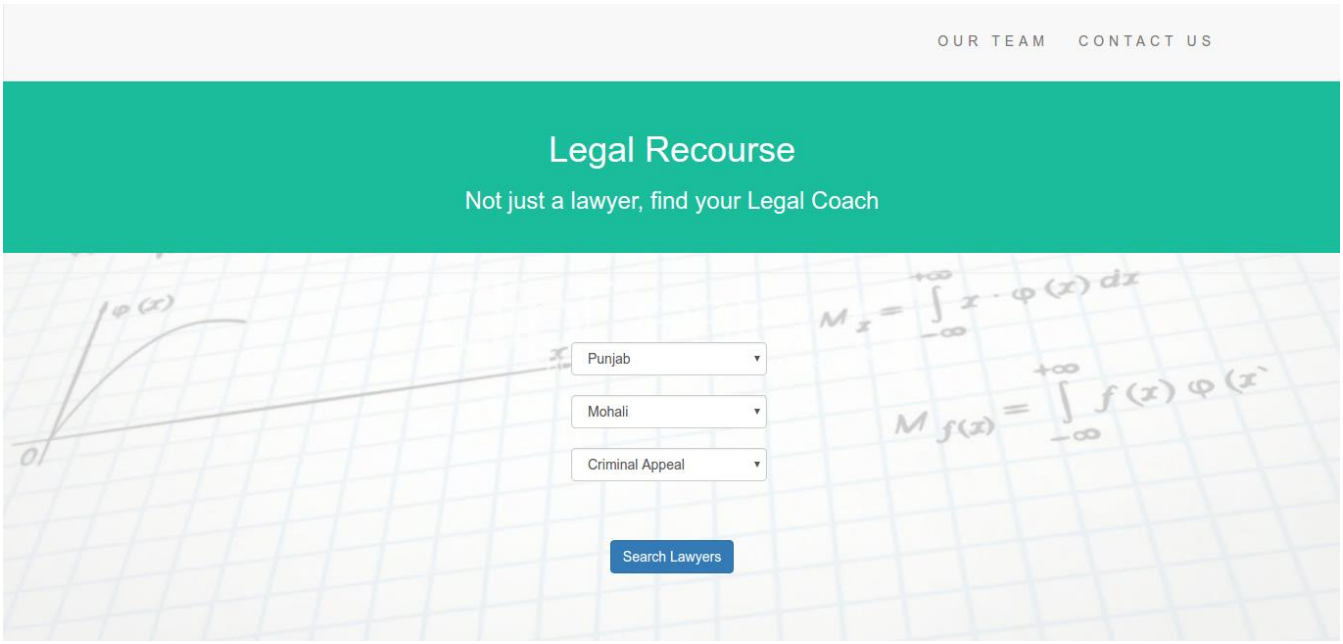


Figure 6.4.1 front end of our web portal

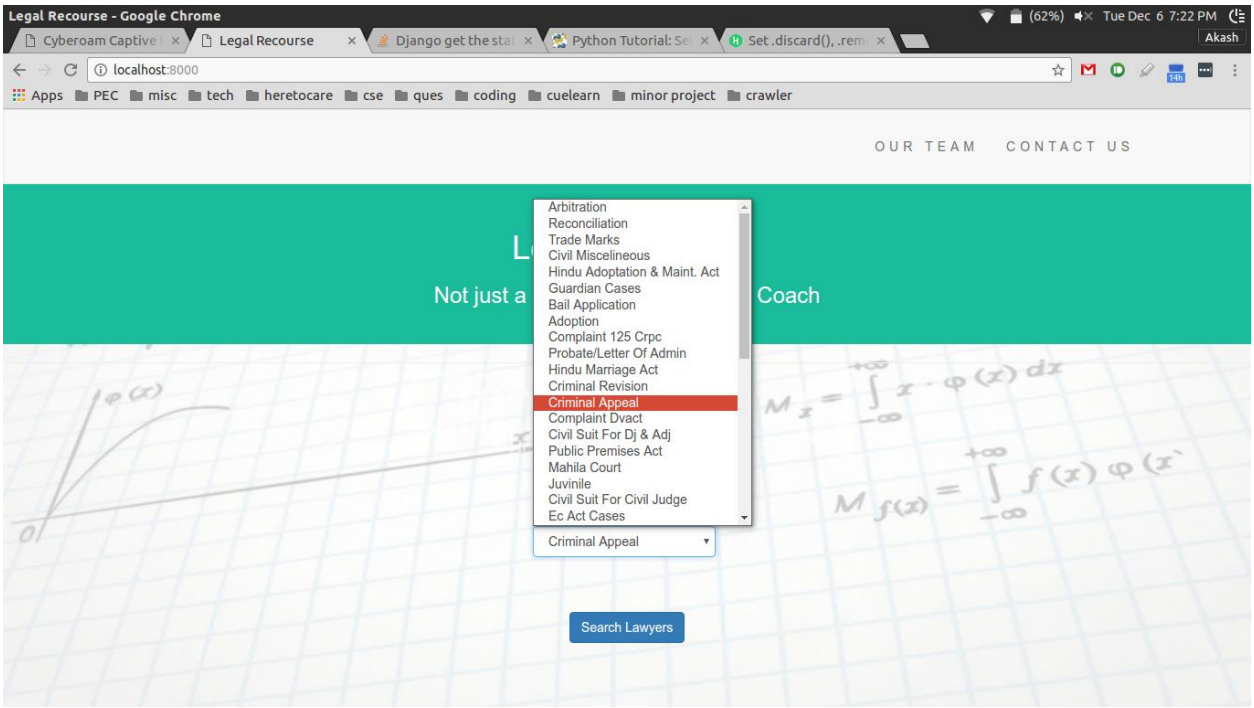


Figure 6.4.2 list of available type of case domains

7. RESULTS AND DISCUSSION

7.1 INPUT TO THE WEBSITES

The following screesnhots (figures 7.1.1 and 7.1.2 refer to input being given to our portal)

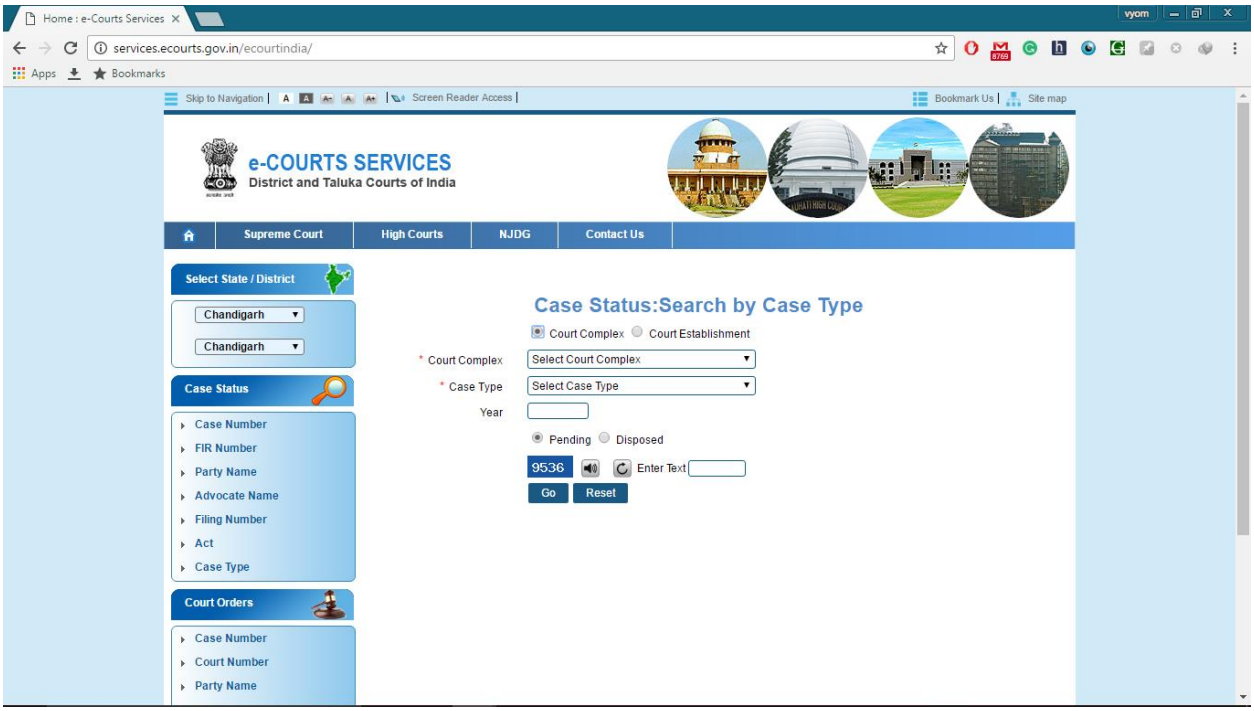


Figure7.1.1 Input being automated using selenium library

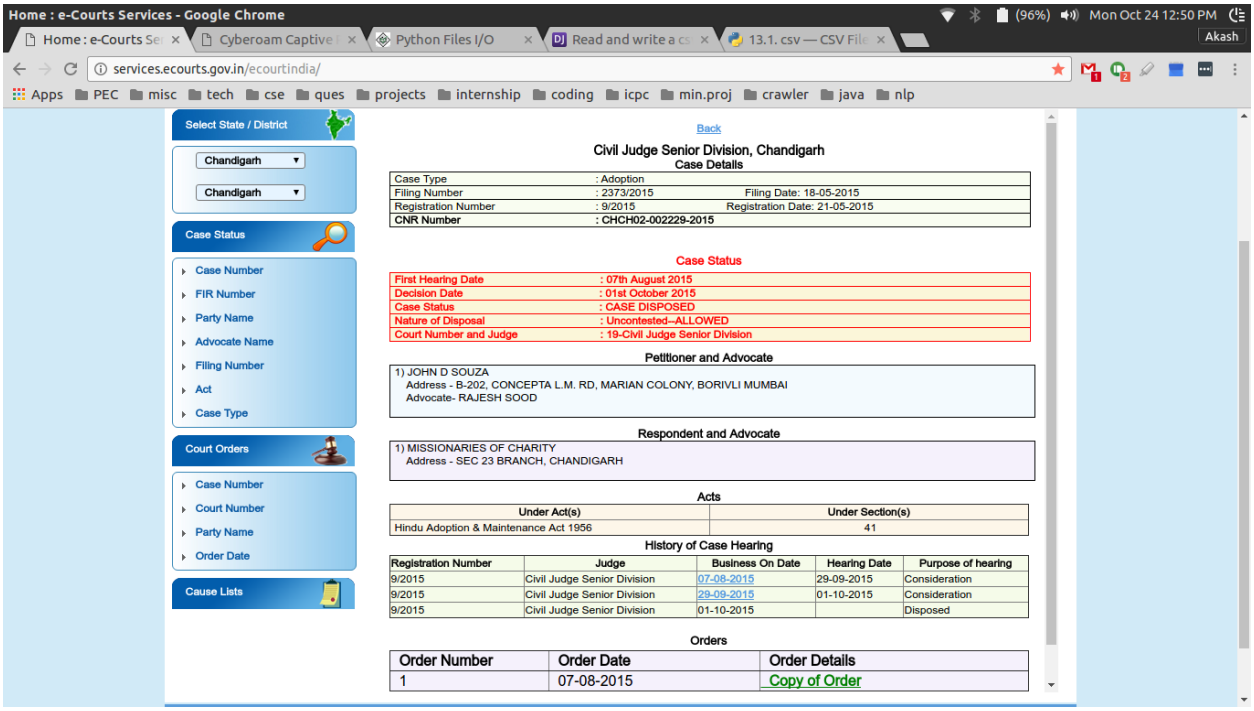


Figure 7.1.2 Selenium running the browser and collecting info on lawyers

7.2 DATABASE

The following screenshots (figure 7.2.1 and 7.2.1) refer to the screenshots of databases at various times.

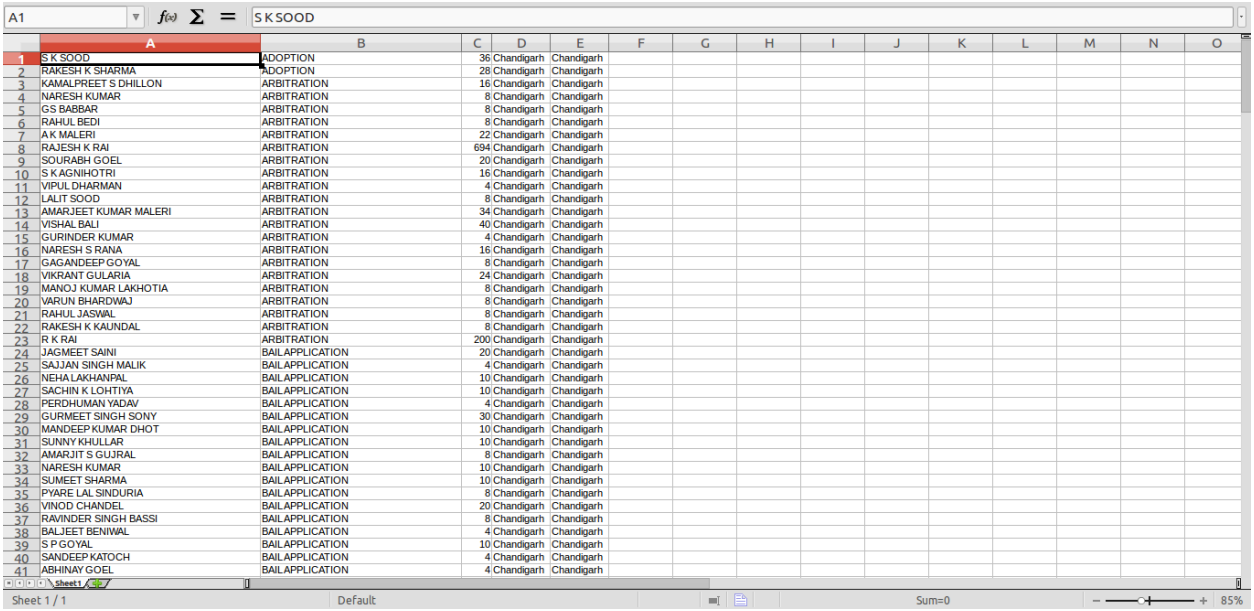
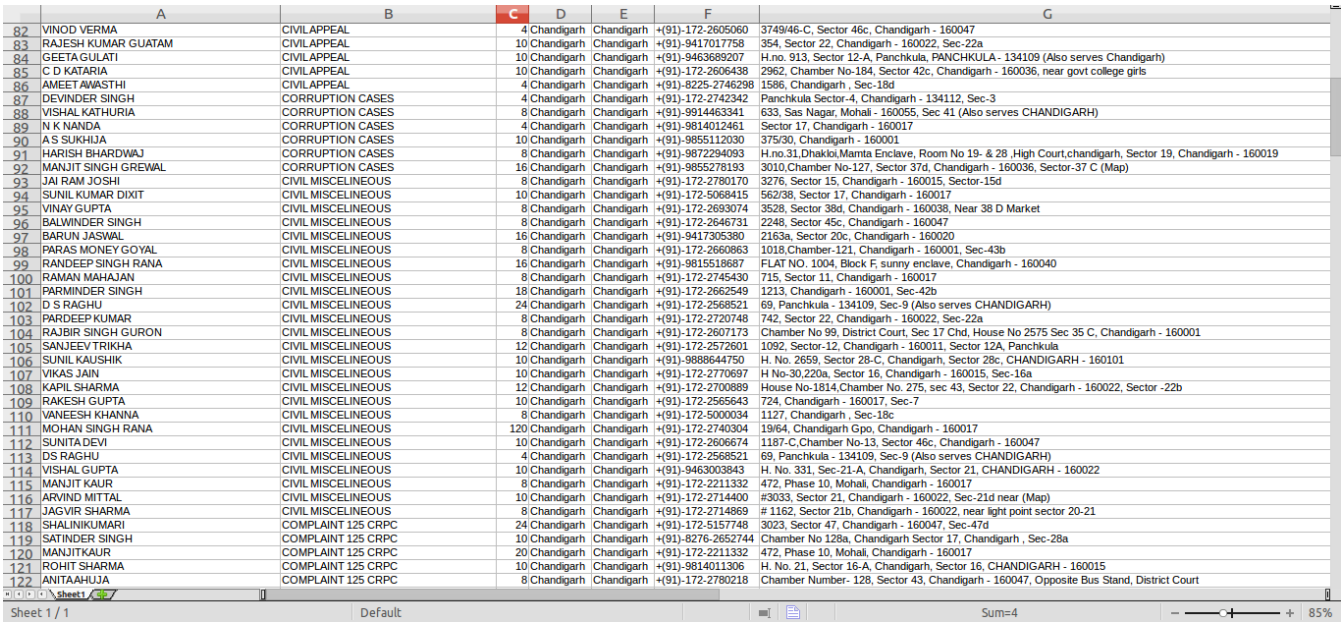


Figure 7.2.1 Info returned from court case website crawler



	A	B	C	D	E	F	G
82	VINOD VERMA	CIVILAPPEAL	4	Chandigarh	Chandigarh	+(91)-172-2605060	3749/46-C, Sector 46c, Chandigarh - 160047
83	RAJESH KUMAR GUATAM	CIVILAPPEAL	10	Chandigarh	Chandigarh	+(91)-9417017758	354, Sector 22, Chandigarh - 160022, Sec-22a
84	GEETA GULATI	CIVILAPPEAL	10	Chandigarh	Chandigarh	+(91)-9463689207	H.no. 913, Sector 12-A, Panchkula, PANCHKULA - 134109 (Also serves Chandigarh)
85	C D KATARIA	CIVILAPPEAL	10	Chandigarh	Chandigarh	+(91)-172-2606438	2962, Chamber No-184, Sector 42c, Chandigarh - 160036, near govt college girls
86	AMEET AWASTHI	CIVILAPPEAL	4	Chandigarh	Chandigarh	+(91)-8225-2746298	1586, Chandigarh , Sec-18d
87	DEVINDER SINGH	CORRUPTION CASES	4	Chandigarh	Chandigarh	+(91)-172-2742342	Panchkula Sector-4, Chandigarh - 134112, Sec-3
88	VISHAL KATHURIA	CORRUPTION CASES	8	Chandigarh	Chandigarh	+(91)-9914463341	633, Sas Nagar, Mohali - 160055, Sec 41 (Also serves CHANDIGARH)
89	N K NANDA	CORRUPTION CASES	4	Chandigarh	Chandigarh	+(91)-9814012461	Sector 17, Chandigarh - 160017
90	A S SUKHIJA	CORRUPTION CASES	10	Chandigarh	Chandigarh	+(91)-9855112030	375/30, Chandigarh - 160001
91	HARISH BHARDWAJ	CORRUPTION CASES	8	Chandigarh	Chandigarh	+(91)-9872294093	H.no.31,Dhakloi,Mamta Enclave, Room No 19- & 28 ,High Court,chandigarh, Sector 19, Chandigarh - 160019
92	MANJIT SINGH GREWAL	CORRUPTION CASES	16	Chandigarh	Chandigarh	+(91)-9855278193	3010,Chamber No-127, Sector 37d, Chandigarh - 160036, Sector-37 C (Map)
93	JAI RAM JOSHI	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2780170	3276, Sector 15, Chandigarh - 160015, Sector-15d
94	SUNIL KUMAR DIXIT	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-172-5068415	562/38, Sector 17, Chandigarh - 160017
95	VINAY GUPTA	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2693074	3528, Sector 38d, Chandigarh - 160038, Near 38 D Market
96	BALWINDER SINGH	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2646731	2248, Sector 45c, Chandigarh - 160047
97	BARUN JASWAL	CIVIL MISCELINEOUS	16	Chandigarh	Chandigarh	+(91)-9417305380	2163a, Sector 20c, Chandigarh - 160020
98	PARAS MONEY GOYAL	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2660863	1018,Chamber-121, Chandigarh - 160001, Sec-43b
99	RANDEEP SINGH RANA	CIVIL MISCELINEOUS	16	Chandigarh	Chandigarh	+(91)-9815518687	FLAT NO. 1004, Block F, sunny enclave, Chandigarh - 160040
100	RAMAN MAHAJAN	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2745430	715, Sector 11, Chandigarh - 160017
101	PARMINDER SINGH	CIVIL MISCELINEOUS	18	Chandigarh	Chandigarh	+(91)-172-2662549	1213, Chandigarh - 160001, Sec-42b
102	D S RAGHU	CIVIL MISCELINEOUS	24	Chandigarh	Chandigarh	+(91)-172-2568521	69, Panchkula - 134109, Sec-9 (Also serves CHANDIGARH)
103	PARDEEP KUMAR	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2720748	742, Sector 22, Chandigarh - 160022, Sec-22a
104	RAJBIR SINGH GURON	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2607173	Chamber No 99, District Court, Sec 17 Chd, House No 2575 Sec 35 C, Chandigarh - 160001
105	SANJEEV TRIKHA	CIVIL MISCELINEOUS	12	Chandigarh	Chandigarh	+(91)-172-2572601	1092, Sector-12, Chandigarh - 160011, Sector 12A, Panchkula
106	SUNIL KAUSHIK	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-9888644750	H. No. 2659, Sector 28-C, Chandigarh, Sector 28c, CHANDIGARH - 160101
107	VIKAS JAIN	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-172-2770697	H No-30,220a, Sector 16, Chandigarh - 160015, Sec-16a
108	KAPIL SHARMA	CIVIL MISCELINEOUS	12	Chandigarh	Chandigarh	+(91)-172-2700889	House No-1814,Chamber No. 275, sec 43, Sector 22, Chandigarh - 160022, Sector -22b
109	RAKESH GUPTA	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-172-2565643	724, Chandigarh - 160017, Sec-7
110	VANEESH KHANNA	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-5000034	1127, Chandigarh , Sec-18c
111	MOHAN SINGH RANA	CIVIL MISCELINEOUS	120	Chandigarh	Chandigarh	+(91)-172-2740304	19/64, Chandigarh Gpo, Chandigarh - 160017
112	SUNITA DEVI	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-172-2606674	1187-C,Chamber No.-13, Sector 46c, Chandigarh - 160047
113	DS RAGHU	CIVIL MISCELINEOUS	4	Chandigarh	Chandigarh	+(91)-172-2568521	69, Panchkula - 134109, Sec-9 (Also serves CHANDIGARH)
114	VISHAL GUPTA	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-9463003843	H. No. 331, Sec-21-A, Chandigarh, Sector 21, CHANDIGARH - 160022
115	MANJIT KAUR	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2211332	472, Phase 10, Mohali, Chandigarh - 160017
116	ARVIND MITTAL	CIVIL MISCELINEOUS	10	Chandigarh	Chandigarh	+(91)-172-2714400	#3033, Sector 21, Chandigarh - 160022, Sec-21d near (Map)
117	JAGVIR SHARMA	CIVIL MISCELINEOUS	8	Chandigarh	Chandigarh	+(91)-172-2714869	# 1162, Sector 21b, Chandigarh - 160022, near light point sector 20-21
118	SHALINIKUMARI	COMPLAINT 125 CRPC	24	Chandigarh	Chandigarh	+(91)-172-5157748	3023, Sector 47, Chandigarh - 160047, Sec-47d
119	SATINDER SINGH	COMPLAINT 125 CRPC	10	Chandigarh	Chandigarh	+(91)-8276-2652744	Chamber No 128a, Chandigarh Sector 17, Chandigarh , Sec-28a
120	MANJITKAUR	COMPLAINT 125 CRPC	20	Chandigarh	Chandigarh	+(91)-172-2211332	472, Phase 10, Mohali, Chandigarh - 160017
121	ROHIT SHARMA	COMPLAINT 125 CRPC	10	Chandigarh	Chandigarh	+(91)-9814011306	H. No. 21, Sector 16-A, Chandigarh, Sector 16, CHANDIGARH - 160015
122	ANITAAHUJA	COMPLAINT 125 CRPC	8	Chandigarh	Chandigarh	+(91)-172-2780218	Chamber Number- 128, Sector 43, Chandigarh - 160047, Opposite Bus Stand, District Court

Figure 7.2.2 Info in database after second crawler returns contact info

7.3 INFO CRAWLER

The info crawler crawls justdial website as shown in figure 7..3

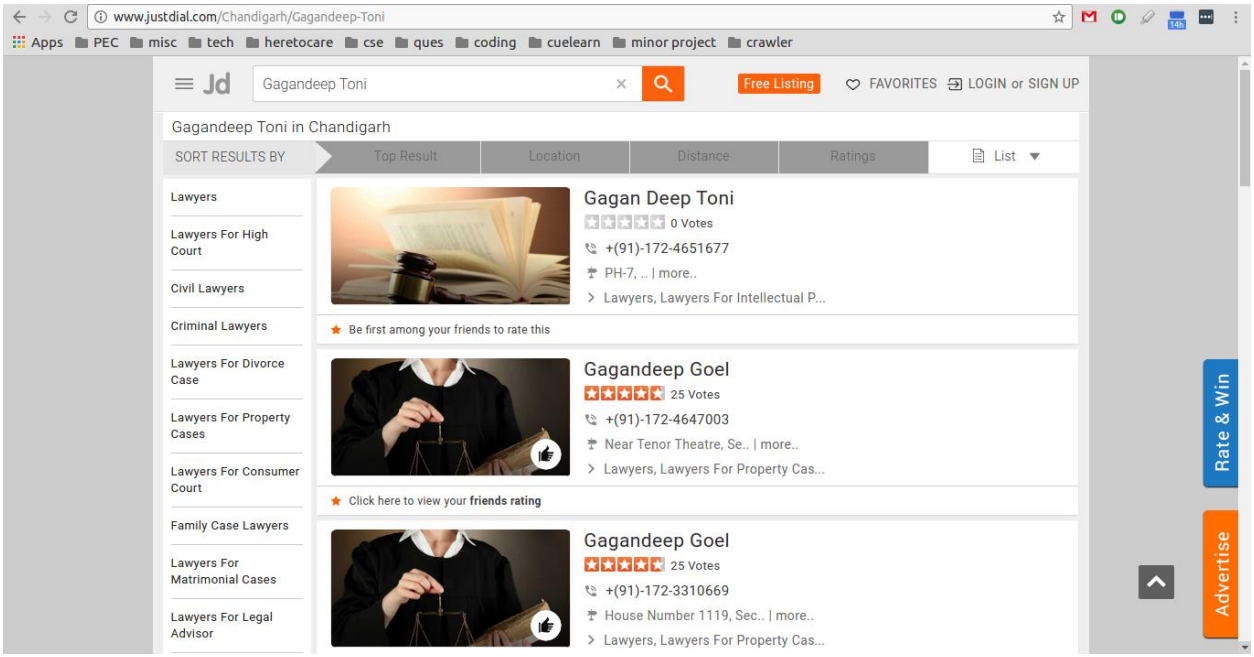


Figure7.3 Contact info crawling crawling the justdial website

7.4 RESULT BEING DISPLAYED

The following screenshot (figure7.4) refers to the result being displayed.

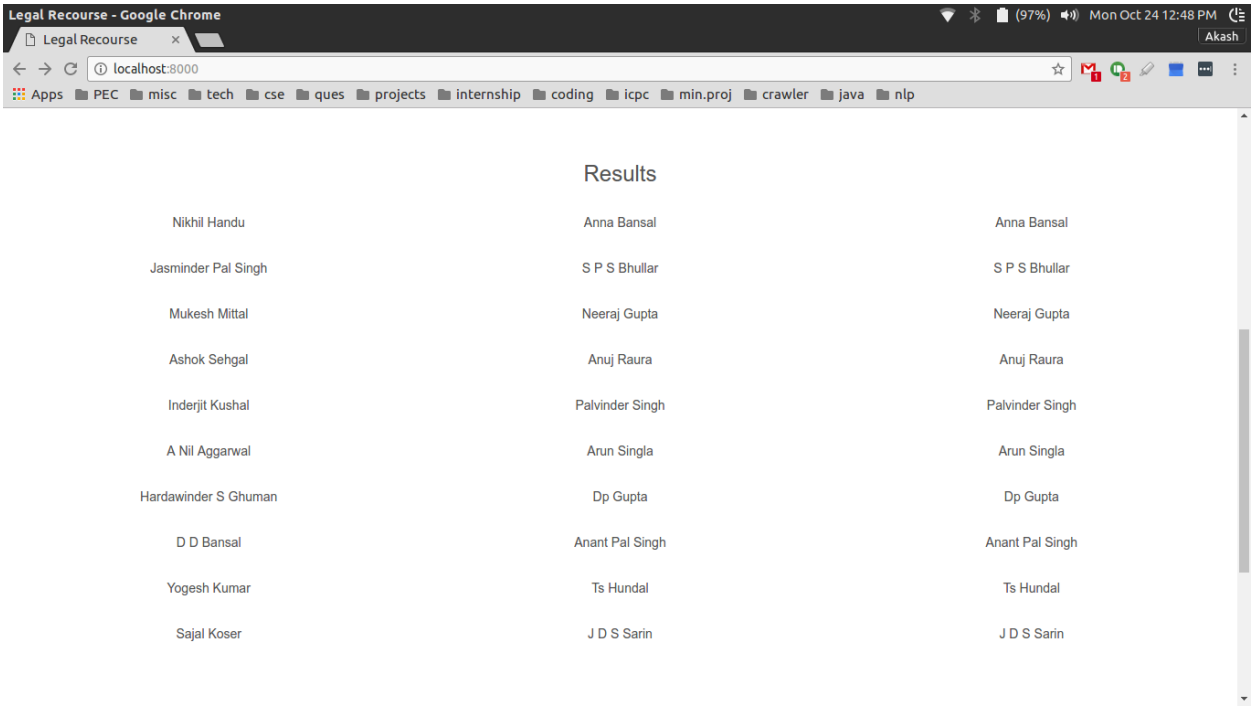


Figure7.4 list of recommended lawyers according to their ranking

8. APPLICATIONS

Lawyer Recommendation System

1. **AUTOMATED LAWYER SELECTION**: Lawyer selection is made easier for the general public and the whole process is automated from case input to lawyer suggestion list.

2. **CONTACT INFO ON LAWYER**: Along with just giving a Lawyer Suggestion List , the portal also provides with the contact info on the lawyer, to contact him easily and immediately.

3. **CASE SPECFIC SEARCH**: By grouping lawyers into their specific domain of case type we can achieve a case-specific which results in more specialized lawyer selection list resulting in higher accuracy in lawyer selection.

4. **ANALYSE PREVIOUS DATA**: Our portal can also be used to analyze the court case data along all these years to make an index of how well each lawyer has performed in a specific domain in the recent years.

5. **PERFORMANCE INDEX FOR LAWYERS**: Through this portal even a lawyer can get results on a particular case type and therefore compare his proficiency in that particular domain with that of his compatriots.

9. FUTURE SCOPE

1. **EXTEND TO MULTIPLE STATES**: By increasing the functionality of Selenium to extend it to crawl data from other states and U.T.s too, we will be able to widen the scope of our lawyer database.

2. **UPDATE AT REGULAR INTERVALS**: The data about court cases is ever changing with new data being added every single day, and therefore by performing regular updates and running our crawler at regular intervals (i.e. weekly).

3. **CRAWL MULTIPLE COURT CASE WEBSITES**: By crawling multiple websites we can get a more varied database of lawyers with more entries which would result in an extensive lawyer recommendation list.

4. **CRAWL MULTIPLE INFO WEBSITES**: By crawling multiple websites for contact info we can get contact info on more number of lawyers. More contact info on lawyers, means that an extensive list of (with contact name) lawyers can be achieved.

5. **MONETIZE THE WEBPORTAL**: The web portal can also be used as a source of income by it's developers and therefore be an extended source of income.

6. **AREA SPECIFIC LAWYERS**: The web portal can also be used to implement area specific search of lawyers and therefore we can get a suggestion list of domain specific lawyers which have performed well in court cases of a particular district.

10. CONCLUSION

This report proposed a Lawyer Recommendation System (LRS) which could provide free legal aid to it's users as it would provide a lawyer recommendation list based on data collected from previous court cases.

The web portal ,at its heart, has 2 webcrawlers. One is used to crawl the court case website and therefore fill up database with the names of the lawyers. There lawyers are subsequently given a rating based on their win-loss percentage. The next crawler is used to pick up lawyer names from the above database and therefore crawl justial.com on this name. If we get a match on justdial.com on a lawyer with the same name, then the contact info is returned back to the database and it is updated.

Finally a ranking list is prepared based on the rating of the various lawyers and returned as output to the user. The user gets to interact only with our friendly interface and gets result in an instant as all the websites are previously crawled the data is already stored in our database. The HTML/CSS Web portal only acts as an interface between the backend database containing all the info and the user.

The resultant suggestion list could be made more varied and diverse by including data for other UTs and states of india and also updating our database at a higher frequency (i.e weekly).

11. References

1. http://services.ecourts.gov.in/ecourtindia/
2. http://www.justdial.com/
3. https://github.com/tesseract-ocr/tesseract/wiki/Documentation
4. http://www.seleniumhq.org/docs/
5. https://en.wikipedia.org/wiki/Judiciary
6. http://lawrato.com/
7. http://selenium-python.readthedocs.io/navigating.html
8. https://en.wikipedia.org/wiki/Comma-separated_values
9. http://dev.tutorialspoint.com/selenium/index.htm

APPENDIX A

Code for configuring court case info crawler :

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
import selenium.webdriver.support.expected_conditions as EC
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.support.ui import Select
from selenium.webdriver.common.keys import Keys
from pytesseract import image_to_string
from PIL import Image
import time
import csv


path_to_chromedriver =
"/home/akash/projects/legal_recourse/chromedriver"
options = webdriver.ChromeOptions()
options.add_argument("--start-maximized")
browser = webdriver.Chrome(executable_path =
path_to_chromedriver, chrome_options = options)

url = "http://services.ecourts.gov.in/ecourtindia/"
browser.get(url)
sel_state = browser.find_element_by_id("sess_state_code")
all_states = [st for st in
sel_state.find_elements_by_tag_name("option")]
all_states = [state.get_attribute("text") for state in
all_states]
del all_states[0]
total=0


def refresh_page(state, dist, court):

     browser.switch_to_default_content()
     browser.execute_script("window.scrollTo(0, 0)")

     browser.get(url)
     state_path =
"//*[@id='sess_state_code']/option[contains(text(), '%s')]" %
state
     browser.find_element_by_xpath(state_path).click()
     browser.implicitly_wait(1)
```

```
     dist_path =
"//*[@id='sess_dist_code']/option[contains(text(), '%s')]" %
dist
     browser.find_element_by_xpath(dist_path).click()
     browser.implicitly_wait(1)

     browser.find_element_by_id("s_casetype.php").click()
     browser.switch_to_frame("ifr")

     court_path =
"//*[@id='court_complex_code']/option[contains(text(), '%s')]" %
court
     browser.find_element_by_xpath(court_path).click()
     browser.implicitly_wait(1)


def refresh_case_page(state, dist, court, case_type):

     browser.switch_to_default_content()
     browser.execute_script("window.scrollTo(0, 0)")

     refresh_page(state, dist, court)
     case_path = "//*[@id='case_type']/option[contains(text(),
'%s')]" % case_type
     browser.find_element_by_xpath(case_path).click()

     browser.implicitly_wait(1)
     browser.find_element_by_id("radD").click()
     browser.find_element_by_id("captcha").send_keys(get_captcha
())

     browser.find_element_by_xpath("//*[@id='caseNoDet']/div[8]/
span[3]/input[1]").click()
     time.sleep(10)


def get_valid_name(adv_name):

     adv = ""
     if len(adv_name)>1:

          adv_name = adv_name[1].split()
          adv = adv_name[0]
          del adv_name[0]
          for nm in adv_name:
               adv = adv + ' ' + nm
          adv= adv.replace('.',' ')
```

```
            adv= adv.replace('  ',' ')
            adv= adv.upper()
            adv= adv.split(",")[0]
            adv= adv.split(")")[0]
            if adv[-1].isdigit():
                 adv = adv[:-1]
            adv= adv.split("(")[0]
            adv= adv.rstrip()

            if "PUBLIC" in adv or "PROSEC" in adv or "PERSON" in
adv:
                 adv=""
            if ("PP" in adv or "GP" in adv) or (len(adv)<5):
                 adv=""

      else:
            adv=""

      return adv


def get_captcha():

      browser.save_screenshot("screenshot.png")
      img = Image.open("screenshot.png")
      left = 573
      top = 405
      right = 620
      bottom = 425
      img = img.crop((left, top, right, bottom))
      img.save("captcha.png")

      captcha_img = Image.open("captcha.png")
      captcha = image_to_string(captcha_img)
      img.close()
      captcha_img.close()

      return captcha


def get_case_details(state, dist):

      browser.find_element_by_id("s_casetype.php").click()
      browser.switch_to_frame("ifr")

      sel_court =
browser.find_element_by_id("court_complex_code")
```

```
     all_courts = [x for x in
sel_court.find_elements_by_tag_name("option")]
     all_courts = [court.get_attribute("text") for court in
all_courts]
     del all_courts[0]

     for court in all_courts :
          try:
               court_path =
"//*[@id='court_complex_code']/option[contains(text(), '%s')]" %
court
               browser.find_element_by_xpath(court_path).click()
          except:
               continue
          browser.implicitly_wait(1)

          sel_case = browser.find_element_by_id("case_type")
          all_cases = [x for x in
sel_case.find_elements_by_tag_name("option")]
          all_cases = [case.get_attribute("text") for case in
all_cases]
          del all_cases[:46]  #index 7,38

          for case_type in all_cases:

               nextloop = False

               hyphen = -1
               case_name = case_type
               case_name = case_name.replace('amp;amp;', '')
               case_id = ""
               hyphen = case_type.find('-')
               if hyphen != -1:
                    case_id = case_type[:hyphen-1]
                    case_id = case_id.upper()
                    case_name =
case_type[hyphen+2:].replace('amp;amp;', '')

               case_name = case_name.title()
               lawyer_list = {}
               r_status = 0
               p_status = 0


               while not nextloop:
                    nextloop = True
                    try:
```

```
                         case_path =
"//*[@id='case_type']/option[contains(text(), '%s')]" %
case_type

     browser.find_element_by_xpath(case_path).click()
                    except:
                         continue

                    browser.implicitly_wait(1)
                    browser.find_element_by_id("radD").click()

     browser.find_element_by_id("captcha").send_keys(get_captcha
())


     browser.find_element_by_xpath("//*[@id='caseNoDet']/div[8]/
span[3]/input[1]").click()
                    time.sleep(10)

                    try:
                         error_msg =
browser.find_element_by_id("txtmsg").get_attribute("title")
                         if error_msg == "Invalid Captcha":
                              nextloop = False
                              refresh_page(state, dist, court)
                              continue
                         elif error_msg == "Record Not Found":
                              refresh_page(state, dist, court)
                              continue

                    except AttributeError:
                         print cnr_no +', '+ case_name +', '+
state +', '+  dist +', '+  court

                         err_stream = open('errlog.csv', 'a')
               # log error into file
                         err_obj = csv.writer(err_stream)
                         err_obj.writerow([cnr_no, case_name,
state, dist, court])
                         err_obj.close()
                         continue

                    case_list =
browser.find_element_by_id("showList1").find_elements_by_tag_nam
e("tr")
                    index=0
```

```
                    for case in case_list:

                         index+=1
                         case_details =
case.find_elements_by_tag_name("td")
                         if len(case_details) < 2:
                              continue

                         try:
                              case_string =
case_details[2].text.replace("\n","")
                              case_ppl =
case_string.split("Versus")
                              petitioner = case_ppl[0]
                              defendant = case_ppl[1]

                              if petitioner=="test" or
defendant=="test":
                                   continue


     case_details[3].find_element_by_tag_name("a").click()

     #browser.find_element_by_id("back_top").location_once_scrol
led_into_view

                              #retrieving information (finally
:P)

                              p_adv =
browser.find_element_by_class_name("Petitioner_Advocate_table").
text
                              adv_name = p_adv.split("Advocate-
")
                              p_adv = get_valid_name(adv_name)

                              r_adv =
browser.find_element_by_class_name("Respondent_Advocate_table").
text
                              adv_name = r_adv.split("Advocate-
")
                              r_adv = get_valid_name(adv_name)

                              if (p_adv + r_adv) == "":
                                   # if advocate name not
present
```

```
     browser.find_element_by_id("back_top").find_element_by_tag_
name("a").click()
                                    continue

                                cnr_no =
browser.find_element_by_xpath('//*[@id="secondpage"]/div[2]/div[
1]/b/span').text
                                cnr_no = cnr_no.split(": ")[1]

                                status_str =
browser.find_element_by_xpath('//*[@id="secondpage"]/div[2]/div[
2]/span[4]/label/strong[2]').text
                                status_str = status_str[2:]
                                r_status=0
                                p_status=0

                                if "ALLOWED" in status_str or
"CONVICTED" in status_str:      # petitioner lawyer won
                                     p_status=5
                                elif "WITHDRAW" in status_str or
"COMPROMISE" in status_str:
                                     p_status=4
                                elif "DISMISSED" in status_str:
                                     p_status=2
                                else:
                                     p_status=4

                                if "WITHDRAW" in status_str or
"COMPROMISE" in status_str:
                                     r_status=4
                                elif "DISMISSED" in status_str:
                                     # respondant lawyer won
                                     r_status=5
                                elif "ALLOWED" in status_str or
"CONVICTED" in status_str:
                                     r_status=2
                                else:
                                     r_status=4

                                try:
                                     file_stream =
open('caseDB.csv', 'a')                  # write data into file
                                     file_obj =
csv.writer(file_stream)

                                     if p_adv=="":
```

```
     file_obj.writerow([cnr_no, case_name, r_status, r_adv,
state, dist, court])
                                        if
lawyer_list.has_key(r_adv):
                                             lawyer_list[r_adv]
= lawyer_list[r_adv] + r_status*2
                                        else:
                                             lawyer_list[r_adv]
= r_status*2

                                   elif r_adv=="":

     file_obj.writerow([cnr_no, case_name, p_status, p_adv,
state, dist, court])

                                        if
lawyer_list.has_key(p_adv):
                                             lawyer_list[p_adv]
= lawyer_list[p_adv] + p_status*2
                                        else:
                                             lawyer_list[p_adv]
= p_status*2

                                   else:

     file_obj.writerow([cnr_no, case_name, p_status, p_adv,
state, dist, court])

     file_obj.writerow([cnr_no, case_name, r_status, r_adv,
state, dist, court])

                                        if
lawyer_list.has_key(r_adv):
                                             lawyer_list[r_adv]
= lawyer_list[r_adv] + r_status*2
                                        else:
                                             lawyer_list[r_adv]
= r_status*2

                                        if
lawyer_list.has_key(p_adv):
                                             lawyer_list[p_adv]
= lawyer_list[p_adv] + p_status*2
                                        else:
                                             lawyer_list[p_adv]
= p_status*2
```

```
                                        file_stream.close()

                                   except:
                                        print cnr_no +', '+ case_name
+', '+ state +', '+  dist +', '+  court

                                        err_stream =
open('errlog.csv', 'a')                      # log error into file
                                        err_obj =
csv.writer(err_stream)
                                        err_obj.writerow([cnr_no,
case_name, state, dist, court])
                                        err_stream.close()


     browser.find_element_by_id("back_top").find_element_by_tag_
name("a").click()

                                   if len(lawyer_list)>50 or
index>200:
                                        break



                              except:
                                   refresh_case_page(state, dist,
court, case_type)


                    lawyer_file = open('lawyerDB.csv', 'a')
     # lawyer data into file
                    fil_obj = csv.writer(lawyer_file)

                    for lawyer in lawyer_list:
                         fil_obj.writerow([lawyer, case_name,
lawyer_list[lawyer], state, dist])

                    lawyer_file.close()
                    lawyer_list.clear()
                    browser.switch_to_default_content()
                    browser.execute_script("window.scrollTo(0, 0)")
                    browser.switch_to_frame("ifr")


def get_all_dist_details():
```

```
     for state in all_states:

          state_path =
"//*[@id='sess_state_code']/option[contains(text(), '%s')]" %
state     # using xpath for javascript dropdown click
          browser.find_element_by_xpath(state_path).click()
          browser.implicitly_wait(1)

          sel_dist =
browser.find_element_by_id("sess_dist_code")
          all_dist = [x for x in
sel_dist.find_elements_by_tag_name("option")]
          all_dist = [dist.get_attribute("text") for dist in
all_dist]
          del all_dist[0]

          for dist in all_dist:

               state_path =
"//*[@id='sess_state_code']/option[contains(text(), '%s')]" %
state
               browser.find_element_by_xpath(state_path).click()
               browser.implicitly_wait(1)

               dist_path =
"//*[@id='sess_dist_code']/option[contains(text(), '%s')]" %
dist
               browser.find_element_by_xpath(dist_path).click()
               browser.implicitly_wait(1)

               get_case_details(state, dist)


def get_dist_details(state, dist):

     state_path =
"//*[@id='sess_state_code']/option[contains(text(), '%s')]" %
state
     browser.find_element_by_xpath(state_path).click()
     browser.implicitly_wait(1)

     dist_path =
"//*[@id='sess_dist_code']/option[contains(text(), '%s')]" %
dist
     browser.find_element_by_xpath(dist_path).click()
     browser.implicitly_wait(1)
```

```
        get_case_details(state, dist)
        print "Added details for court cases in : "+state


get_dist_details("Chandigarh", "Chandigarh")
```

APPENDIX 2

Code for lawyer info crawler:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
import selenium.webdriver.support.expected_conditions as EC
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.support.ui import Select
from selenium.webdriver.common.keys import Keys
#from pytesseract import image_to_string
#from PIL import Image
import time
import csv


path_to_chromedriver =
"/home/akash/projects/legal_recourse/chromedriver"
options = webdriver.ChromeOptions()
options.add_argument("--start-maximized")
browser = webdriver.Chrome(executable_path = path_to_chromedriver,
chrome_options = options)

with open('lawyerDB.csv', 'r') as read_stream:
        with open('contactDB.csv', 'a') as write_contact:

                read_obj = csv.reader(read_stream)

                write_wo_contact = open('unknownDB.csv', 'a')
                write_obj1 = csv.writer(write_contact, lineterminator='\n')
```

```
          write_obj2 = csv.writer(write_wo_contact,
lineterminator='\n')

          test = 0
          for row in read_obj:
               test+=1
               if test>1082:
                    url = "http://www.justdial.com/" + row[4] + "/" +
row[0].replace(" ","-").title()
                    browser.get(url)

                    lawyer_list =
browser.find_elements_by_class_name("jcn")
                    del lawyer_list[5:]
                    lawyer_page = []

                    for lawyer in lawyer_list:
                         if lawyer.text.replace(" ","").upper() in
row[0].replace(" ","") or row[0].replace(" ","") in
lawyer.text.replace(" ","").upper():

     lawyer_page.append(lawyer.find_element_by_css_selector('a').get_a
ttribute('href'))

                    phone_list =
browser.find_elements_by_class_name("contact-info")
                    del phone_list[5:]
                    phone_list = [number.text for number in
phone_list]

                    index = -1
                    flag = 0
                    for page in lawyer_page:
                         index += 1
                         browser.get(page)
                         all_details =
browser.find_element_by_class_name("comp-
contact").find_elements_by_tag_name("li")

                              for info in all_details:
                                   if "Lawyer" in info.text:
                                        flag = 1
                                        break
                              if flag == 1:
                                   address =
browser.find_element_by_class_name("adrstxtr").text
                                   row.append(str(phone_list[index]))
                                   row.append(str(address))
                                   break

                    row[1] = row[1].upper()
```

```
            if(flag == 1):
                  write_obj1.writerow(row)
            else:
                  write_obj2.writerow(row)

write_wo_contact.close()
```