Virginia General District and Circuit Court Data Scraping Project

Technical Report
Presented to the Faculty of the
School of Engineering and Applied Science
University of Virginia

By

David Stern

May 1, 2021

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

Signed: _____

Approved: _____ Date _____

Jack W. Davidson, Professor, Department of Engineering and Computer Science

Introduction

This document describes the State Court Trial and Appellate Database (SCOUTAPP-DB). It will be primarily comprised of two sections. The first section aims to explain the functionality of SCOUTAPP-DB, as well as its purpose and potential uses. The second section will elaborate of what I in particular have contributed to the project. This document is intended to be understandable to those unaffiliated with the project, as well as those with minimal or no programming experience. Thus, the first section should provide the necessary background information to comprehend the second section.

Functionality and Purpose

SCOUTAPP-DB is a project adapted from work initially completed independently by Ben Schoenfeld. Ben's initial court scraper publicized data from Virginia Court System records, which are not under any copyright protection. This data was then made available for bulk download on Ben's website at http://virginiacourtdata.org/. This downloadable data consisted of large .csv files that were organized by year and type of court. SCOUTAPP-DB is an expansion on Ben's scraper and website with some additional improvements added.

Circuit Criminal Court Cases

Year of most recent hearing	Cases	Download Size	Uncompressed Size	
2000	136,480	4.5 MB	41.4 MB	Download
2001	145,396	4.9 MB	44.3 MB	Download
2002	149,555	5.0 MB	45.6 MB	Download
2003	150,647	5.0 MB	46.0 MB	Download
2004	158,770	5.3 MB	48.5 MB	Download
2005	158,165	5.3 MB	48.6 MB	Download
2006	170,525	5.7 MB	52.6 MB	Download
2007	177,355	6.0 MB	54.9 MB	Download
2008	178,093	6.1 MB	55.3 MB	Download

Figure 1: Screenshot from Ben Schoenfeld's Website

Functionality

The SCOUTAPP-DB project consists of two major components: a data scraper and a website. The data retrieved by the scraper is used to fill a mysql database that allows the website to run, which is also where the data is made downloadable. I will first describe the scraper in greater detail before moving onto the website.

A data scraper is a program which extracts data from a source into some form of human-readable output. Our data scraper extracts court case information from Virginia general district and circuit courts. Examples of the kinds of information scraped includes hearing date, information about the plaintiffs and defendants, and judgment. The full list of kinds of information scraped is outlined in our data dictionaries, of which there are four (district civil, district criminal, circuit civil, and circuit criminal). Our scraper extracts data from two different sites. Data for the district courts is extracted from eapps.courts.state.va.us, while data for the circuit courts is extracted from ewsocis1.courts.state.va.us. The scraper creates directories for each individual court (these directories will be created wherever the scraper code is located), which each contain subdirectories for each year data was scraped for. Each of these subdirectories will then contain two .json files, one for civil data and one for criminal data. These

files are where the case data is actually stored. If these files already exist when the scraper is run, then the scraper will merely append the newly scraped data to them rather than writing to new ones or overwriting the current ones. It should also be noted that the circuit court scraper and the general district court scraper must be run separately.

VIRGINIA COURT DATA
CODEBOOK - Circuit Civil
(2020-03-23)

Highlighted entries are incomplete.

AppealedDate

Description:	Date of appeal (from district court?) – a case can be appealed from a district court to a circuit court, or from a circuit court to either the Court of Appeals or to the Supreme Court of Virginia. A case can be appealed multiple times.
Variable Type:	Date (YYYY-MM-DD)

Bond

Description:	Bond amount in dollars.
Variable Type:	String. Data includes "$" but otherwise is a float.
Values:	NaN : No entry. $0 - $2,600,000

Figure 2: Excerpt from Circuit Civil Data Dictionary

{"case_number":"GV20000006-00","filed_date":"01/03/2020","case_type":"Administrative License Suspension","debt_type":"","judgment":"Other","costs":"","attorney_fees":"","principal_amount":"","other_amount":"","interest_award":"","possession":"","writ_of_eviction_issued_date":"","writ_of_fieri_facias_issued_date":"","homestead_exemption_waived":"","is_judgment_satisfied":"","date_satisfaction_filed":"","other_awarded":"","further_case_information":"","alert":"failed","appeal_date":"","appealed_by":"","plaintiff_name_1":"COMMONWEALTH OF VIRGINIA/ ADMIN O/L SUPEN","plaintiff_dba/ta_1":"","plaintiff_address_1":"","plaintiff_judgment_1":"","plaintiff_attorney_1":"NONE","defendant_name_1":"MOORE, MELODY FAITH","defendant_dba/ta_1":"","defendant_address_1":"HAMPTON, VA 23664","defendant_judgment_1":"","defendant_attorney_1":"NONE","hearing_date_1":"01/02/2020","hearing_time_1":"08:00 AM","hearing_result_1":"Other","hearing_hearing_type_1":"","hearing_courtroom_1":"","hearing_date":"1/2/2020","case_court":"York General District Court"}

{"case_number":"GV19003388-00","filed_date":"12/20/2019","case_type":"Impoundment","debt_type":"","judgment":"Other","costs":"","attorney_fees":"","principal_amount":"","other_amount":"","interest_award":"","possession":"","writ_of_eviction_issued_date":"","writ_of_fieri_facias_issued_date":"","homestead_exemption_waived":"","is_judgment_satisfied":"","date_satisfaction_filed":"","other_awarded":"","further_case_information":"","alert":"failed","appeal_date":"","appealed_by":"","plaintiff_name_1":"COMMONWEALTH OF VIRIGNIA/ IMPOUNDMENT","plaintiff_dba/ta_1":"","plaintiff_address_1":"","plaintiff_judgment_1":"","plaintiff_attorney_1":"NONE","defendant_name_1":"RIGGINS, FREDERICK HUNTER","defendant_dba/ta_1":"","defendant_address_1":"POQUOSON, VA 23662","defendant_judgment_1":"","defendant_attorney_1":"NONE","hearing_date_1":"01/03/2020","hearing_time_1":"08:00 AM","hearing_result_1":"Other","hearing_hearing_type_1":"","hearing_courtroom_1":"","hearing_date":"1/3/2020","case_court":"York General District Court"}

Figure 3: Example of Data Scraped (District Civil)

Our scraper code is written in javascript using Node Package Manager (npm) and Puppeteer. As the websites we are scraping from use Captcha for human verification, a bypass

Captcha key is needed to run the scraper. Once all environment setup is completed, our scraper can be run using the command “npm start”, which will have the scraper run across all courts and years until a timeout inevitably occurs. We also have a shell script that can run more specific commands such as scraping only specific courts and years, or even running multiple instances of the scraper in parallel (though this uses a lot of CPU).

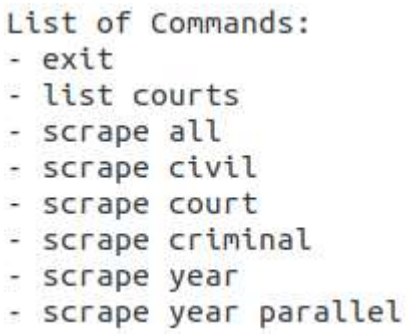


Figure 4: List of Commands for Shell Script

The second part of the SCOUTAPP-DB project is the website. Our website queries from a mysql database in which we have stored all of our scraped data. This means that the website will look at parts of the data contained in the database and format it into a .csv file based on the specifications of the user. The website currently features a simplistic user interface (UI). Though it is not yet finished (as this project will continue on past this semester), the goal is to refine the UI but still keep it simplistic. Users will be able to select which courts and/or years they want to retrieve data from, and the data meeting these specifications will be outputted to them in a .csv file format. It is also possible that other file formats will be supported in the future. However, most of the back-end (code) for this project is dependent on the scraper, which is why the website will remain relatively simple.

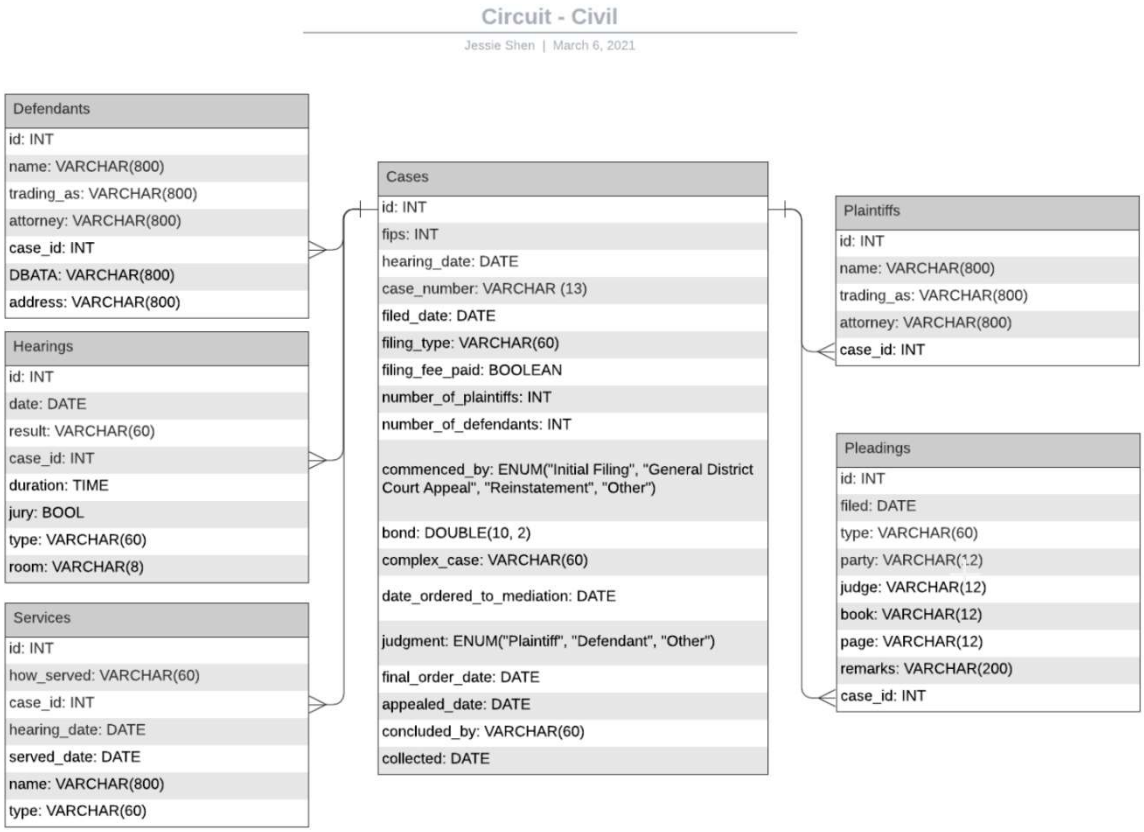


Figure 5: Database Schema for Circuit Civil Courts

Purpose

The primary goal of SCOUTAPP-DB is to provide easy access to Virginia court case data in bulk to the public. Though this data is already publicly available at the websites listed earlier (the ones we are scraping from), they are not available in a table or database format, making large-scale analysis extremely difficult. Moreover, the process of retrieving data from these websites is cumbersome, as there are many intermediary web pages to go through, as well as the Captchas mentioned previously. Thus, a data scraper is necessary to collect this data into a systematic, analyzable format. SCOUTAPP-DB provides this, and it makes the collected data easily accessible and downloadable to anyone who needs it. This provides many more research

opportunities into the Virginia court system than would previously be available. With this bulk data, researchers would be able to identify trends and correlations across the data. Further analysis could potentially expose biases within the court system. The ability to filter by data and court also allows researchers to identify specific courts that are especially prevalent in these trends, or how trends have changed over time (perhaps due to a change in judge for example). Essentially, this project was established on the idea that having more data available to the public could allow us to identify and alleviate problems that exist within systems currently in place.

Of course, we considered that having certain data available to the public is not always a good thing. This is why we prioritized anonymity by removing defendant names from our scraped data. This is because we wanted the data from our scraper to be used to find larger-scale problems with the court system, rather than having it be used against specific people. For example, we did not want someone to be denied a loan because of data retrieved from our scraper.

Future Prospects

Though the Virginia scraper is now completed and a simple website has been put in place, we have talked about possible advancements that could be made in the future. For example, our scraper may one day advance beyond just Virginia to other states as well. Also, as mentioned previously, advancing the UI of the website is a future goal that we have in mind, as well as potentially supporting other file formats besides .csv files.

Contributions

Much as the structure of SCOUTAPP-DB itself, my own work on the project can be divided into two halves. In the first semester of working on the project, I spent most of my time

getting things set up and then working on the website. During the second semester, I worked on the scraper instead, as well as some intermediary work and documentation that supported the overall project. In this section, I will detail the specifics of my contributions to the project during the past year.

Data Dictionaries

My first major assignment after getting my virtual environment working was to upload the data dictionaries to the website. The data dictionaries detail all of the fields that are outputted when downloading datasets from the website. For each field, the following components are given: the name of the field, a description of what the field means, the variable type of the field (integer, boolean, etc.), and possible values the field can take on if applicable to that field. Another project member and I worked to get these data dictionaries (given to us as four separate Word documents) onto their own separate pages on the website. We decided to split up the work between district and circuit courts, and I was in charge of adding the two circuit court data dictionaries (circuit civil and circuit criminal).

It should be noted that we collaboratively worked on both the scraper and the website using Github repositories. This allowed members of the project to work on parts of it simultaneously without needing to rewrite someone else's code. The website was created using Django, which is a framework for building web applications. It uses a model-view-controller (MVC) design pattern, where the data is stored in the "model" which is represented to the user as a "view," and the "controller" processes input from the user and updates the model and view accordingly. Since the data dictionaries are individual web pages that do not react much to any user input, I mainly only had to edit the view portion to add them to the website. To do this, I added two html files (known as templates in Django) to the repository: one for the circuit civil

data dictionary and one for the circuit criminal data dictionary. I then had to translate each data dictionary into html code. Since I was told to keep the UI simple, as we planned to do more formatting to the UI of the website once everything else was completed, I kept the format of the dictionaries similar to how they appeared in the Word documents (shown in Figure 2). Since each field was stored in its own table, I ended up creating many html tables for each page. As the Word documents were fairly long, the html templates ended up being quite lengthy as well. After that, I simply had to update the controller to prompt a switch to the correct data dictionary web page when the user indicates to do so (note that in Django, the controller is confusingly named "views.py").

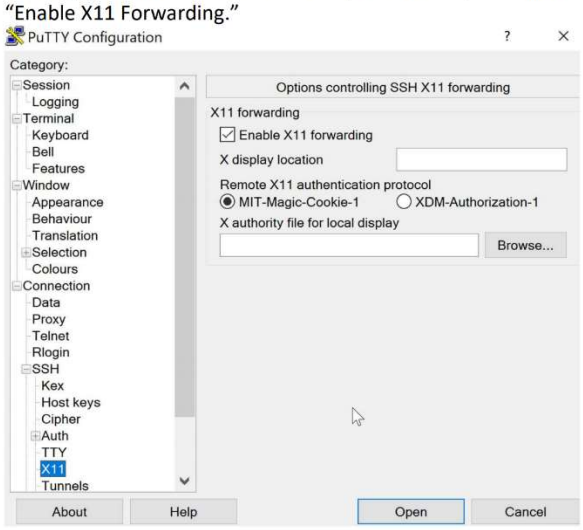
Various Documentation

Throughout both semesters, I wrote a few documents that helped contribute to the project in ways other than just code. As the setup process for this project is fairly complicated, and we have new contributors coming and going quite commonly, I was tasked with writing setup instructions for both the website and the scraper individually. While not entirely comprehensive, as different contributors experience different issues with the setup process depending on their computer and work environment, these instructions were meant to help catch new member up to speed more easily than simply going through the trials and tribulations of repeating the setup process for each new member.

First, I created setup instructions for the website as I was figuring it out myself. We initially used a virtual machine located on a dedicated courts server so as to not have to install too much stuff locally. We were given accounts to connect to the courts server machine. From there we had to launch vmware, a piece of software for running virtual machines, and create a new virtual machine. As a display server was necessary for launching vmware, I downloaded

XMing. This virtual machine is where we would download the website source code and launch the site. My instructions detail any bugs and fixes I had with the setup process, recommended software such as XMing and Putty, necessary packages to be installed on the virtual machine such as mysql and Chromium, and commands to run the website.

1. Download XMing from https://sourceforge.net/projects/xming/files/Xming/6.9.0.31/Xming-6-9-0-31-setup.exe/download and download Putty from http://www.chiark.greenend.org.uk/~sgtatham/putty/
2. Follow the instructions on https://virginia.service-now.com/its/?id=itsweb_kb_article&sys_id=f24e5cdfdb3acb804f32fb671d9619d0 to download and setup the UVA VPN.
3. Connect to the VPN. Then launch XMing followed by Putty.
4. In Putty, go to the SSH tab followed by the X11 tab, and click the checkmark next to "Enable X11 Forwarding."



Then type courts.cs.virginia.edu into the textbox in the session tab and hit open.

Figure 6: Excerpt from Website Setup Instructions

Instructions for the scraper were far less complicated. As the virtual environment used to launch the website proved to be too slow for running the scraper for several project members, we found alternative methods for running the scraper. All that was necessary was a Linux environment, so I used a virtual machine I had locally on my computer, but others solutions, such as the Windows Subsystem for Linux, work as well. As I did not have to detail setting up a new virtual environment for the scraper instructions, they ended up being much shorter. All I had

to explain for the most part was the process of installing npm and Puppeteer, as well as setting up the bypass Captcha key. I also once again noted any bugs and fixes I had for the whole process.

Virginia Courts Scraper Setup Instructions

1. Clone the Virginia scraper github repository using `git clone https://github.com/SCOUTAPP-DB/virginia-scraper`
2. Install npm using `sudo apt install npm`
3. Install puppeteer using `npm install puppeteer`. This should also automatically install chromium to your environment. If you are using the terminal interface vm on the courts machine, you may need to restart your vm before you can access the gui interface.
4. Download the secrets.js file into the virginia-scraper directory. This file is pinned to the #updates channel of the slack workspace (though this is subject to change, and you may need to ask someone where to find this file if that is the case).
5. Run the scraper from inside the virginia-scraper directory using `npm start`. After running the scraper, you will be directed to the General District Court Online Case Information System and be presented with a series of Captchas. Do not click on the Captchas and just let the script bypass them or you are likely to receive an error saying UnhandledPromiseRejectionWarning.
6. If you plan on editing the scraper code, first fork the repository, and when you want to commit your changes to the main branch, submit a pull request.

*Note: Everyone tends to run into different kinds of problems when running the scraper for the first time, especially since different people are using different environments to run it. So if you encounter a problem that is not listed in these instructions, then feel free to continue adding to this document.

Figure 7: Scraper Setup Instructions

Finally, one other piece of documentation I helped with was a comparison between our scraped data and that of Ben Schoenfeld's scraper. We did this in order to ensure robustness of our scraper (meaning we did not want to miss any important fields). The original document was written by another project member. However, after the document was created, and updated version of the scraper was then posted to Github. I then updated the document to be consistent with the newest version of the scraper.

District – Civil	
US (what we have)	**BEN (what ben has)**
hearing_date_#***	Fips**
alert*	writ_issue_date
court name**	garnishee
defendant_judgment_#	answer_date
	checks_received
	collected
Terminology Changes	
principal_amount	principle_amount
possession	possession_date
writ_of_eviction_issued_date	writ_of_eviction_issue_date
writ_of_fieri_facias_issued_date	writ_of_fieri_facias_issue_date
is_judgement_satisfied	judgement_satisfied
date_satisfaction_filed	satisfaction_filed_date

Figure 8: Excerpt from Comparison to Ben's Scraped Data

Database Population

As mentioned previously, the website makes queries from a database in order to provide downloadable csv data to the user. In order to do this, a database containing all of our scraped data needed to be created. I was part of a three-person team working on this goal, where one member created the database itself, I worked to populate the database with the data from the scraper, and another member worked to query from the populated database. In order to accomplish my task, I wrote a Python program which takes in a json file as input. It then converts that file into a json object in Python, which it is then able to insert into a mysql database.