11/06/2023

Hacettepe University

Computer Engineering Department

BBM479/480 End of Project Report

Project Details

Title	**Table Catcher : Extracting Data Tables from Websites**
Supervisor	Assoc. Prof. BURKAY GENÇ

Group Members

	Full Name	**Student ID**
1	Eray Dindaş	21946041
2	Enes Yavuz	21989712
3	Mehmet Giray Nacakcı	21989009

Table Catcher Logo

Documentation Website

`https://tablecatcher.azurewebsites.net/`

Demo Video

`https://www.youtube.com/watch?v=jWpMPikg6FM`

GitHub: Install and discover Table Catcher

`https://github.com/b21989009/Table_Catcher_public`

Abstract of the Project (/ 10 Points)

Explain the whole project shortly including the introduction of the field, the problem statement, your proposed solution and the methods you applied, your results and their discussion, expected impact and possible future directions.

There is a vast amount of data in the world of the internet. We may want to just copy-paste some tabular data from a website, yet it is hard to keep the structure and format of the cells and data when the table is copy-pasted as text. We aimed to efficiently extract these tables in a standard, editable format. For this purpose, we developed a Chrome browser extension called "Table Catcher".

Even though similar tools exist, our aims included: to gain insight into development of such tools and best-practices ourselves, experiment on what can be improved, and showcase our development and design skills gained in Hacettepe.

We believed that the easiest way to “catch” a table on a web page as it is, is a browser extension. Because of the popularity and rich developer documentation, we chose Chrome browser to be our pilot. We set up our system as a Client-Server architecture, with Django on server side, to be able to flexibly work with Python language.

We have provided two methods for recognizing tables. First method scans the current HTML content of the webpage and catches standard HTML tables. Second method uses Computer Vision and handles other non-standard formats. In both methods, we achieved high accuracy in the outputted CSV and Excel files in terms of table layout and cell data.

Overall, we combined some methods and provided yet another free tool that can facilitate data collection work of Data Scientists or anyone who needs it. Table Catcher is open-source and open to evolution; thus, along with our existing ideas for future development, anyone can contribute.

Introduction, Problem Definition & Literature Review (/ 20 Points)

Introduce the field of your project, define your problem (as clearly as possible), review the literature (cite the papers) by explaining the proposed solutions to this problem together with limitations of these problems, lastly write your hypothesis (or research question) and summarize your proposed solution in a paragraph. Please use a scientific language (you may assume the style from the studies you cited in your literature review). You may borrow parts from your previous reports but update them with the information you obtained during the course of the project.

With the rise of Machine Learning, Data Science and Data Mining has gained great importance over the recent years. It is one of the most open-wide techniques to explore the reality of today’s World. And researchers need more data than ever, in less time to process it. And we aimed to facilitate (automate/optimize) their initial (exhausting) work of data collection.

Large amounts of data is publicly available on the web in tabular form, and we see a need for efficiently extracting data in easily manipulatable formats. Extracting data from web sources and structuring it into convenient formats is referred to as “Web Scraping”. Table Catcher is an automatic web table extractor tool that returns parsed data in standard formats. In data science and in the industry, Excel is widely used for visual purposes, and CSV for directly parsing DataFrames into code. Files in these formats are the output of Table Catcher.

Related Studies and Existing Tools in Industry

- Table Capture browser extension copies tabular data to clipboard, gives ability to preview captured table in a new tab, and modify it. Unfortunately, there is no access to the source code, to get an idea of the underlying algorithm. It also has “workshop” mode, in which the user can text-select repeating items such as rows, and thus capture “div-based” tables. [1]

  For comparison, in Table Catcher’s Vision method, we capture div-tables directly by choosing the whole table.

- Bright Data: a more general data collection SaaS (paid service). Capture specific data from websites with Scraper APIs, recipes, and so on. [2]

- Microsoft Lens mobile app (since 2021) uses Computer Vision to extract tables from images, in PDF and Excel formats [3]. Research on table extraction using vision methods with feature extraction dates back to the early 2000s, while convolutional network based approaches gained prominence in the 2010s.

  Inspired by this approach, in the Vision method of Table Catcher; we too, employ vision to catch tables. We get help from github.com/xavctn/img2table library, and Azure Cognitive Services OCR API.

  For non-standard tables, we were compelled towards Vision methods also because we could not come across any articles or tools that proposed a complete, reliable, generalized algorithm to capture contents of web tables which does not follow standard-html-table format, without using Computer Vision. Furthermore, our own experiments did not reach success either.

- Since HTML has been around since 1993, parsing HTML tables in various ways dates back to the 90s.

  Oogane and Asawaka (1998), created an alternative non-visual representation of HTML tables for accessing via screen readers (non-visual interface devices which are used by visually impaired people). [4]

  Lim (1999), converted HTML tables to pseudo-table representations by retrieving the content hierarchy, and creating a content tree, which is simpler than DOM tree. Then, can be converted to XML. [5]

  Since the 2010s, there are Python libraries such as Pandas, that can use their own object representations, and functions to manipulate data and convert to various formats such as CSV, Excel, SQL, XML, JSON, Markdown, Latex, and so on.

- Python web scraping libraries such as Scrapy and Beautiful Soup can read HTML tables from URLs. Microsoft Excel and Google Sheets can import (only static content) tables by their webpage URL.

Their limitation is that, they cannot read (client-side-rendered) content that is dynamically created by webpages executing JavaScript scripts.
On the other hand, our approach in Table Catcher is to send the most recent client-side-rendered content to our Backend for processing.

- Matthew Harris, gets help from GPT-3 (OpenAI's Natural Language Processing model), to reformat tables. He trains the API with prompts that can help him restructure the table, manipulate and clean-up cell data. [6]

  Although we planned an automated "simplicity mode" of Table Catcher which would get help from Statistics and Machine Learning, that can do complex clean-up, normalization, numerical representation features; our time did not allow us to take-on this task.
  Currently, in Table Catcher, we only do simple "unnamed" columns, all "NaN" row and column removal.

HYPOTHESIS

In the Methodology section, in the "Tables on the Internet" sub-title, we define and classify web tables as 3 types: A. image , B. standard html <table> , C. div-based or arbitrary.

We claimed that, when we run Table Catcher, we will be able to identify and process all three types of tables (except some conditions that will be mentioned), and output CSV and Excel files containing these tables, with acceptable accuracy and in a few seconds time. We suggested we may use an ensemble of different methods to maximize successful coverage of different table formats in a page.

Methodology (/ 25 Points)

Explain the methodology you followed throughout the project in technical terms including datasets, data pre-processing and featurization (if relevant), computational models/algorithms you used or developed, system training/testing (if relevant), principles of model evaluation (not the results). Using equations, flow charts, etc. are encouraged. Use sub-headings for each topic. Please use a scientific language. You may borrow parts from your previous reports but update them with the information you obtained during the course of the project.

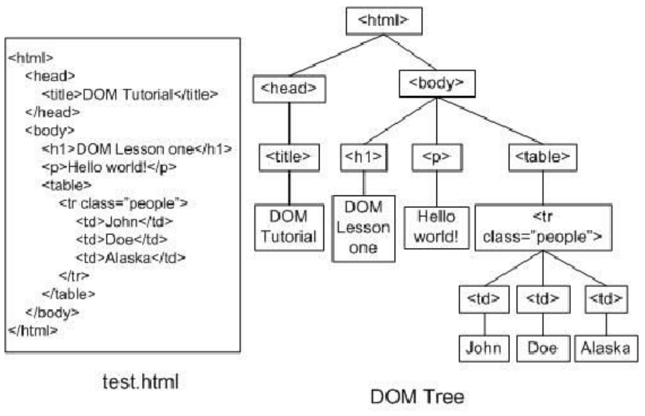
Tables on the Internet

What do we define as a table? Generally, a table is a list with multiple columns, where each list entry has a value associated with each column. Visually, on a rectangular grid. In web development, we classify tables as 3 Types:

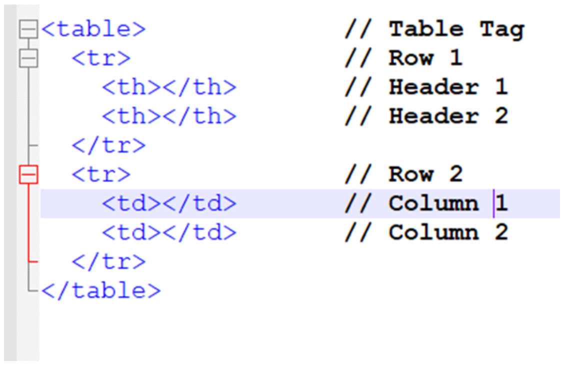
Type A. presented visually as image etc. files. Mostly embedded in pages with <img> tag.

Type B. programmed with HTML using standard <table> tag structures.

The most common type, and a reliable standard. How it looks can be tweaked with CSS, yet follows a flexible but predefined HTML syntax. Such as <tr> tag defining a row, <th> a column header, <td> a data cell.



```
<html>
  <head>
    <title>DOM Tutorial</title>
  </head>
  <body>
    <h1>DOM Lesson one</h1>
    <p>Hello world!</p>
    <table>
      <tr class="people">
        <td>John</td>
        <td>Doe</td>
        <td>Alaska</td>
      </tr>
    </table>
  </body>
</html>
```

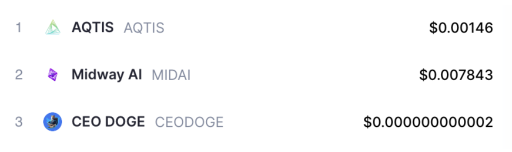


```
<table>                 // Table Tag
  <tr>                  // Row 1
    <th></th>           // Header 1
    <th></th>           // Header 2
  </tr>
  <tr>                  // Row 2
    <td></td>           // Column 1
    <td></td>           // Column 2
  </tr>
</table>
```

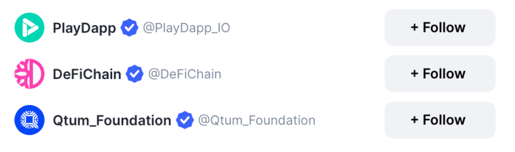
[7] [8]

Type C. programmed with HTML arbitrarily and visually arranged with CSS styling.

Since there is no standard layout, the definition and scope of such tables is arbitrary, thus we consider a limited number of pilot websites to demonstrate table recognition.



1	AQTIS AQTIS	$0.00146
2	Midway AI MIDAI	$0.007843
3	CEO DOGE CEODOGE	$0.000000000002

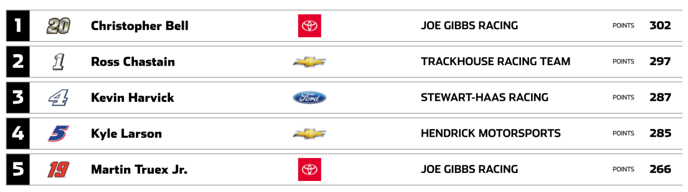


PlayDapp @PlayDapp_IO + Follow

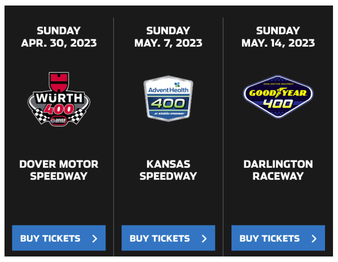
DeFiChain @DeFiChain + Follow

Qtum_Foundation @Qtum_Foundation + Follow

Table Not Table



1	20	Christopher Bell		JOE GIBBS RACING	POINTS	302
2	1	Ross Chastain		TRACKHOUSE RACING TEAM	POINTS	297
3	4	Kevin Harvick		STEWART-HAAS RACING	POINTS	287
4	5	Kyle Larson		HENDRICK MOTORSPORTS	POINTS	285
5	19	Martin Truex Jr.		JOE GIBBS RACING	POINTS	266



SUNDAY APR. 30, 2023 — DOVER MOTOR SPEEDWAY — BUY TICKETS

SUNDAY MAY. 7, 2023 — KANSAS SPEEDWAY — BUY TICKETS

SUNDAY MAY. 14, 2023 — DARLINGTON RACEWAY — BUY TICKETS

Table Not Table

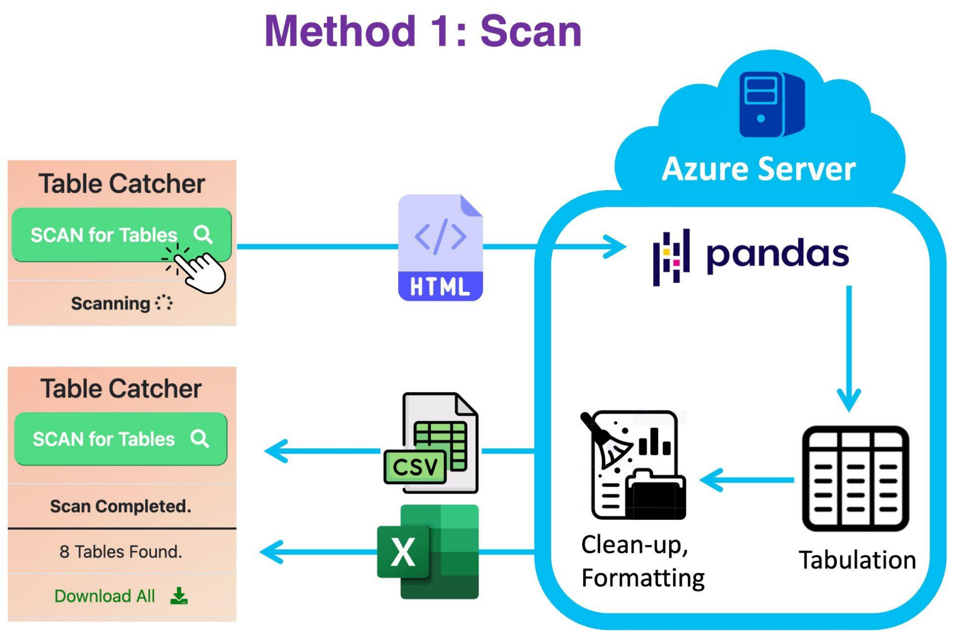
Architecture

Table Catcher is a Chrome Browser Extension; with a Django Backend deployed, tested, running on Azure server. Users only need to install the extension from our GitHub page.

Triggered by user, Client side decides what to send to Server; Server processes the input and returns responses as table files.

Table Recognition Methods

Method 1: Scan method



User clicks the “Scan for Tables” button in the extension pop-up, active tab’s current HTML content is sent to the Backend. Spinner animation indicates that the system is working, until response comes back in a few seconds.

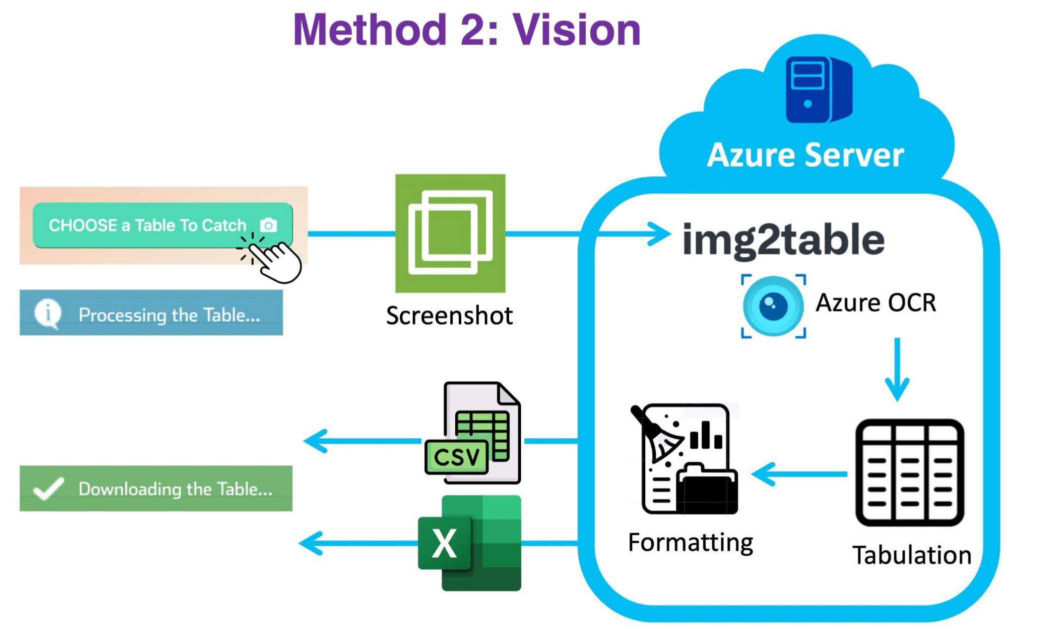
Pandas.read_html() parses the HTML into multiple DataFrames (tables). Internally, it identifies tag structures and creates parse trees. It can only identify <u>Type B (standard html <table>)</u> tables.

<u>Data Clean-up sub-process:</u> Extracted tables include unwanted clutter inserted by Pandas; as well as columns and rows consisting of all NaN values, “unnamed” columns, web metadata. Also, some tables are false-positives which are empty, or did not exist on the webpage. We remove all this clutter.

For each table, a CSV file is created, an Excel file is created. A Zip archive is created to contain CSV and Excel of all tables, user can download in one click.

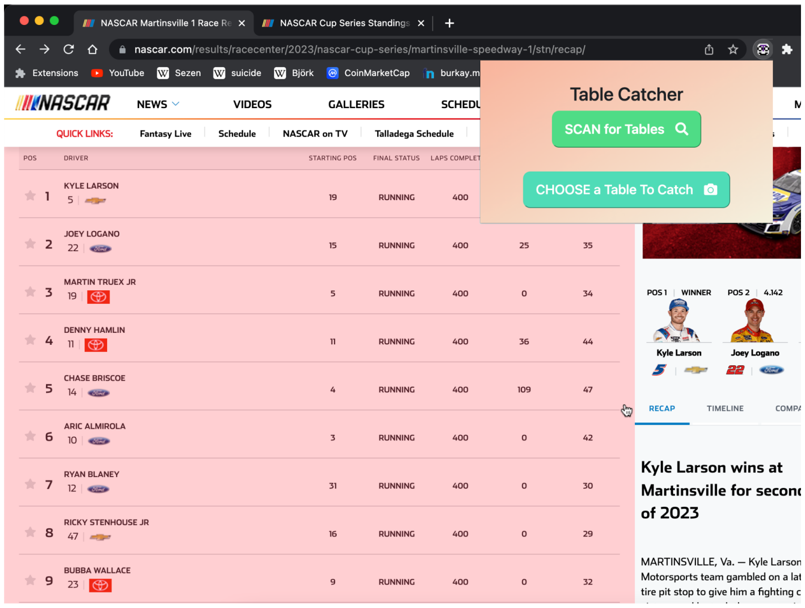
When download links are returned to the Chrome Extension, Extension modifies the webpage to display each download button next to the tables. Retrieved table count and “Download all” option is displayed inside the pop-up.

Method 2: Vision method



We developed this method mainly developed for Type C tables, yet, since we use Vision regardless of the underlying structure, this method is able to capture all **Type C, Type B, Type-A** tables. (Type-A has an exception of having to reside in the same domain as webpage, not cross-origin (CORS).

In our extension popup, user clicks “choose a table to catch”, then hover and click on an html element. html2canvas library captures a screenshot of the element, sends it to Backend.
Even if the element does not fit the viewport, screenshot contains the whole of it.



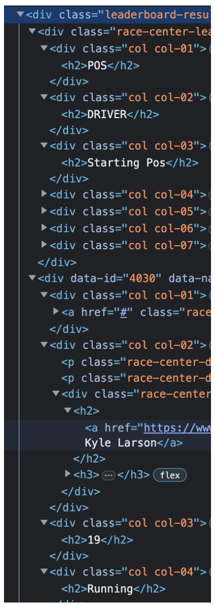
POS	DRIVER	STARTING POS	FINAL STATUS	LAPS COMPLET		
1	KYLE LARSON 5	19	RUNNING	400		
2	JOEY LOGANO 22	15	RUNNING	400	25	35
3	MARTIN TRUEX JR 19	5	RUNNING	400	0	34
4	DENNY HAMLIN 11	11	RUNNING	400	36	44
5	CHASE BRISCOE 14	4	RUNNING	400	109	47
6	ARIC ALMIROLA 10	3	RUNNING	400	0	42
7	RYAN BLANEY 12	31	RUNNING	400	0	30
8	RICKY STENHOUSE JR 47	16	RUNNING	400	0	29
9	BUBBA WALLACE 23	9	RUNNING	400	0	32

POS 1 | WINNER POS 2 | 4.142

Kyle Larson 5 Joey Logano 22

Kyle Larson wins at Martinsville for secon
of 2023

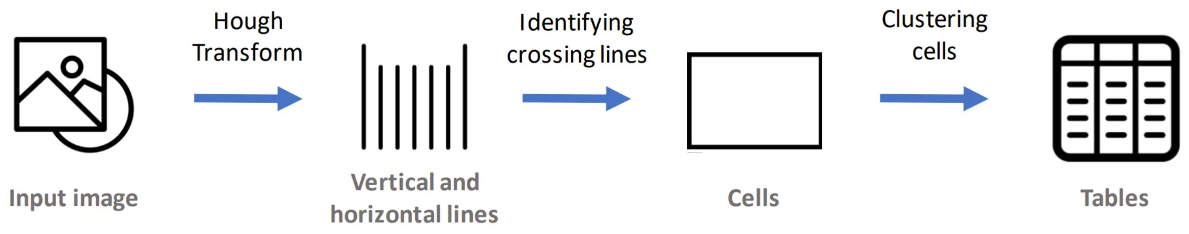
MARTINSVILLE, Va. — Kyle Larson
Motorsports team gambled on a lat
tire pit stop to give him a fighting c



```
▼<div class="leaderboard-resu
  ▼<div class="race-center-lea
    ▼<div class="col col-01">
       <h2>POS</h2>
     </div>
    ▼<div class="col col-02">
       <h2>DRIVER</h2>
     </div>
    ▼<div class="col col-03">
       <h2>Starting Pos</h2>
     </div>
    ▶<div class="col col-04">
    ▶<div class="col col-05">
    ▶<div class="col col-06">
    ▶<div class="col col-07">
   </div>
  ▼<div data-id="4030" data-na
    ▼<div class="col col-01">
      ▶<a href="#" class="race
     </div>
    ▼<div class="col col-02">
       <p class="race-center-d
       <p class="race-center-d
      ▼<div class="race-center
        ▼<h2>
           <a href="https://ww
           Kyle Larson</a>
         </h2>
        ▶<h3>…</h3> flex
       </div>
     </div>
    ▼<div class="col col-03">
       <h2>19</h2>
     </div>
    ▼<div class="col col-04">
       <h2>Running</h2>
```

https://www.nascar.com/ => "schedule" => "race results" tab

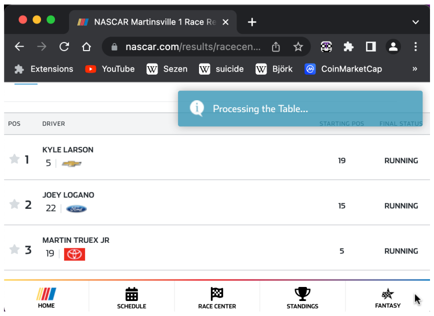
In the backend, github.com/xavctn/**img2table** python library is capable of inferring table cell structure from image input, and returning textual data of cells with use of Optical Character Recognition. This library is able to work with a variety of OCR APIs. Unlike the Tesseract OCR we experimented with before; Azure Cognitive Services Vision provides high detection performance, speed (just few seconds), and accuracy, and integrates better with our Azure Backend.

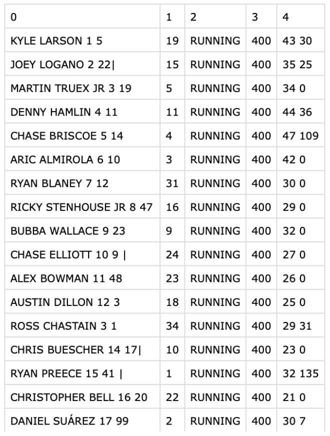


img2table uses Hough Transform to detect bordered tables, and uses alignment of word bounding boxes to detect borderless tables.

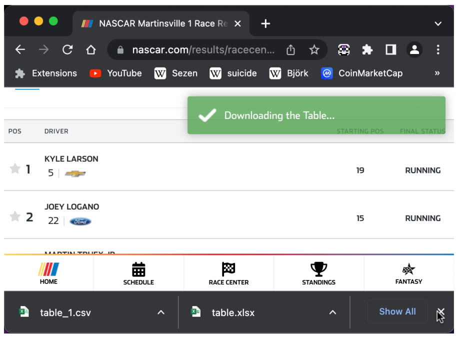
Since **img2table** is able to not only recognize the content inside tables but also detect tables, if the user chooses an area on the page larger than the table itself, or accidentally chooses multiple tables (not always reliable), it still works. Differently, If a table is detected as (two or more) separate pieces (of same column count), we concatenate those pieces.

For each (usually one) table, a CSV and an Excel file is created. When download links are returned to the Chrome Extension, download is done right-away automatically.





0	1	2	3	4
KYLE LARSON 1 5	19	RUNNING	400	43 30
JOEY LOGANO 2 22|	15	RUNNING	400	35 25
MARTIN TRUEX JR 3 19	5	RUNNING	400	34 0
DENNY HAMLIN 4 11	11	RUNNING	400	44 36
CHASE BRISCOE 5 14	4	RUNNING	400	47 109
ARIC ALMIROLA 6 10	3	RUNNING	400	42 0
RYAN BLANEY 7 12	31	RUNNING	400	30 0
RICKY STENHOUSE JR 8 47	16	RUNNING	400	29 0
BUBBA WALLACE 9 23	9	RUNNING	400	32 0
CHASE ELLIOTT 10 9 |	24	RUNNING	400	27 0
ALEX BOWMAN 11 48	23	RUNNING	400	26 0
AUSTIN DILLON 12 3	18	RUNNING	400	25 0
ROSS CHASTAIN 3 1	34	RUNNING	400	29 31
CHRIS BUESCHER 14 17|	10	RUNNING	400	23 0
RYAN PREECE 15 41 |	1	RUNNING	400	32 135
CHRISTOPHER BELL 16 20	22	RUNNING	400	21 0
DANIEL SUÁREZ 17 99	2	RUNNING	400	30 7



Note: During the creation of Excel files in Backend, Excel outputs of both Scan and Vision methods are stylized with minimal formatting for readability: auto-width for columns to fit content, cell borders, bold headers.

– – – end of section: Table Recognition Methods – – –

Discussion

We did not reinvent anything; but, as planned, used existing tools/libraries/algorithms whenever we could; in order to be focused on the overall quality of the outcome, integration of assembled parts, and timely delivery of our proposed system.

Scalability & Concurrency & Privacy

Django Backend Framework handles concurrency by isolating requests and processing them independently. Yet, still, in our previous implementations, we were saving saved table files in a single static folder with the same names (such as "table1.csv"), thus users overwriting each others' tables, or accessing other users' tables with the same download link. This interference hurts security and privacy of a user in a scenario where they scan tables in a webpage that requires log-in authentication; and also renders our app useless since another user's request could just overwrite caught table files in a few milliseconds.

With the new implementation, each run (not exactly session) of the extension creates table files in a new folder identified by a random 4 digit alphanumeric character sequence. Thus, the chance of interference drops to 1/600,000. Thus, it is also hard for any user to guess the download links of any other user's tables.

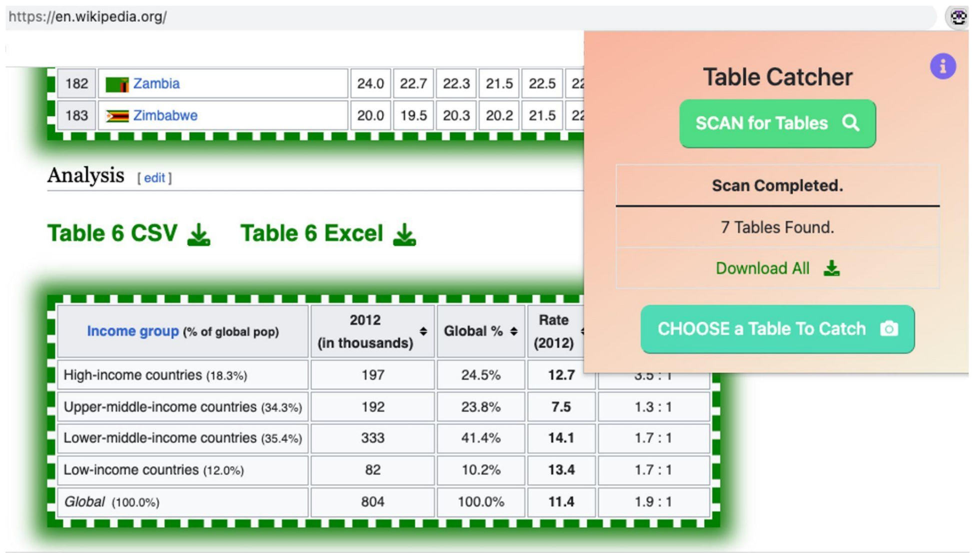
Our user might want to keep the page open for maybe hours, and then download the files. We got them covered. We delete the table files only when our set storage limit is surpassed.

In our pilot websites, a directory of table files created for a screenshot_table method takes up around 10 KB each, and scan_tables is around 50-250 KB when there are 10-ish many tables on the page. Thus, our 10GB disk storage on Azure is more than enough.

There is no need to define a scheduled background task and increase the complexity of our backend, current clean-up process is already quick and robust and responsive enough, does not block the flow, clean up lasts less than 5 seconds, scan takes a few milliseconds; and thus no need to run it independent from user requests (async, background), it can be done during a request.

Results & Discussion (/ 30 Points)

Explain your results in detail including system/model train/validation/optimization analysis, performance evaluation and comparison with the state-of-the-art (if relevant), ablation study (if relevant), a use-case analysis or the demo of the product (if relevant), and additional points related to your project. Also include the discussion of each piece of result (i.e., what would be the reason behind obtaining this outcome, what is the meaning of this result, etc.). Include figures and tables to summarize quantitative results. Use sub-headings for each topic.



182	Zambia	24.0	22.7	22.3	21.5	22.5	22
183	Zimbabwe	20.0	19.5	20.3	20.2	21.5	22

Income group (% of global pop)	2012 (in thousands)	Global %	Rate (2012)	
High-income countries (18.3%)	197	24.5%	**12.7**	3.5 : 1
Upper-middle-income countries (34.3%)	192	23.8%	**7.5**	1.3 : 1
Lower-middle-income countries (35.4%)	333	41.4%	**14.1**	1.7 : 1
Low-income countries (12.0%)	82	10.2%	**13.4**	1.7 : 1
Global (100.0%)	804	100.0%	**11.4**	1.9 : 1

User Interface in Wikipedia

"i" (info) navigates to our Documentation website.

Pilot websites:

(Websites on which Table Catcher is tested and working correctly. These websites include multiple tables with different sizes of data, numerical and alphabetical values and so on.)

<u>Scan Method:</u>

- https://tr.wikipedia.org/wiki/Sezen_Aksu_diskografisi

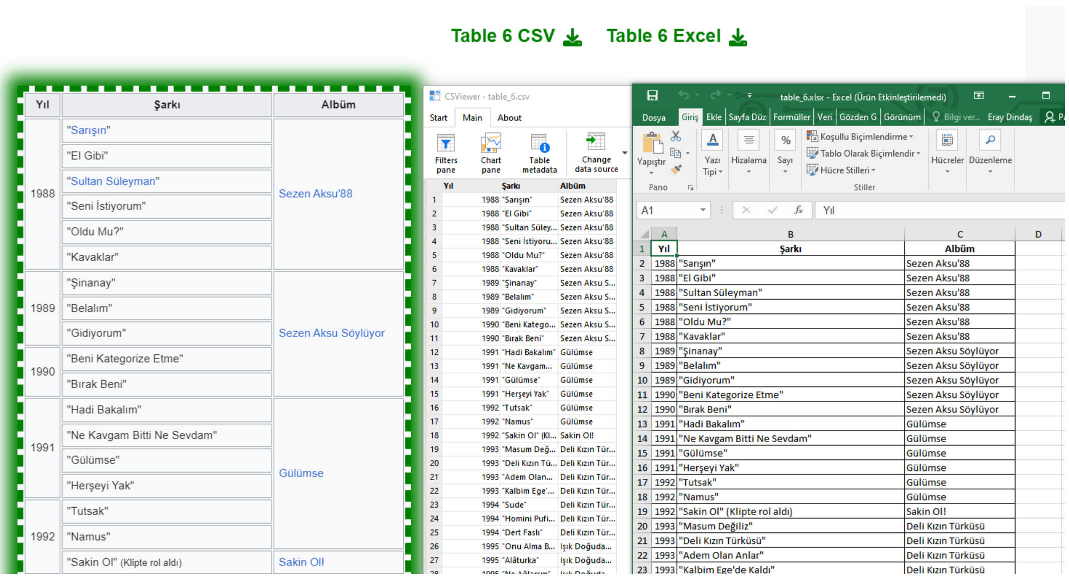


Table 6 CSV   Table 6 Excel

Yıl	Şarkı	Albüm
1988	"Sarışın"	Sezen Aksu'88
	"El Gibi"	
	"Sultan Süleyman"	
	"Seni İstiyorum"	
	"Oldu Mu?"	
	"Kavaklar"	
1989	"Şinanay"	Sezen Aksu Söylüyor
	"Belalım"	
	"Gidiyorum"	
1990	"Beni Kategorize Etme"	
	"Bırak Beni"	
1991	"Hadi Bakalım"	Gülümse
	"Ne Kavgam Bitti Ne Sevdam"	
	"Gülümse"	
	"Herşeyi Yak"	
1992	"Tutsak"	
	"Namus"	
	"Sakin Ol" (Klipte rol aldı)	Sakin Ol!

CSViewer - table_6.csv

	Yıl	Şarkı	Albüm
1	1988	"Sarışın"	Sezen Aksu'88
2	1988	"El Gibi"	Sezen Aksu'88
3	1988	"Sultan Süley...	Sezen Aksu'88
4	1988	"Seni İstiyoru...	Sezen Aksu'88
5	1988	"Oldu Mu?"	Sezen Aksu'88
6	1988	"Kavaklar"	Sezen Aksu'88
7	1989	"Şinanay"	Sezen Aksu S...
8	1989	"Belalım"	Sezen Aksu S...
9	1989	"Gidiyorum"	Sezen Aksu S...
10	1990	"Beni Katego...	Sezen Aksu S...
11	1990	"Bırak Beni"	Sezen Aksu S...
12	1991	"Hadi Bakalım"	Gülümse
13	1991	"Ne Kavgam...	Gülümse
14	1991	"Gülümse"	Gülümse
15	1991	"Herşeyi Yak"	Gülümse
16	1992	"Tutsak"	Gülümse
17	1992	"Namus"	Gülümse
18	1992	"Sakin Ol" (Kl...	Sakin Ol!
19	1993	"Masum Değ...	Deli Kızın Tür...
20	1993	"Deli Kızın Tü...	Deli Kızın Tür...
21	1993	"Adem Olan...	Deli Kızın Tür...
22	1993	"Kalbim Ege'...	Deli Kızın Tür...
23	1994	"Sude"	Deli Kızın Tür...
24	1994	"Homini Pufi...	Deli Kızın Tür...
25	1994	"Dert Faslı"	Deli Kızın Tür...
26	1995	"Onu Alma B...	Işık Doğuda...
27	1995	"Alâturka"	Işık Doğuda...

table_6.xlsx - Excel (Ürün Etkinleştirilemedi)

A1 Yıl

	A	B	C
1	Yıl	Şarkı	Albüm
2	1988	"Sarışın"	Sezen Aksu'88
3	1988	"El Gibi"	Sezen Aksu'88
4	1988	"Sultan Süleyman"	Sezen Aksu'88
5	1988	"Seni İstiyorum"	Sezen Aksu'88
6	1988	"Oldu Mu?"	Sezen Aksu'88
7	1988	"Kavaklar"	Sezen Aksu'88
8	1989	"Şinanay"	Sezen Aksu Söylüyor
9	1989	"Belalım"	Sezen Aksu Söylüyor
10	1989	"Gidiyorum"	Sezen Aksu Söylüyor
11	1990	"Beni Kategorize Etme"	Sezen Aksu Söylüyor
12	1990	"Bırak Beni"	Sezen Aksu Söylüyor
13	1991	"Hadi Bakalım"	Gülümse
14	1991	"Ne Kavgam Bitti Ne Sevdam"	Gülümse
15	1991	"Gülümse"	Gülümse
16	1991	"Herşeyi Yak"	Gülümse
17	1992	"Tutsak"	Gülümse
18	1992	"Namus"	Gülümse
19	1992	"Sakin Ol" (Klipte rol aldı)	Sakin Ol!
20	1993	"Masum Değiliz"	Deli Kızın Türküsü
21	1993	"Deli Kızın Türküsü"	Deli Kızın Türküsü
22	1993	"Adem Olan Anlar"	Deli Kızın Türküsü
23	1993	"Kalbim Ege'de Kaldı"	Deli Kızın Türküsü

- www.investing.com/stock-screener/

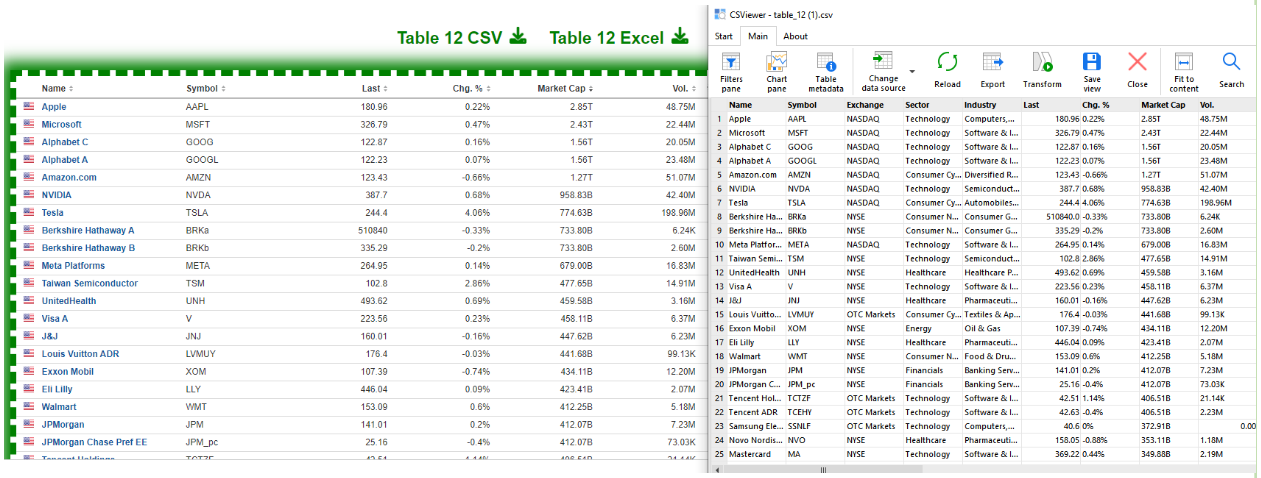


Table 12 CSV   Table 12 Excel

Name	Symbol	Last	Chg. %	Market Cap	Vol.
Apple	AAPL	180.96	0.22%	2.85T	48.75M
Microsoft	MSFT	326.79	0.47%	2.43T	22.44M
Alphabet C	GOOG	122.87	0.16%	1.56T	20.05M
Alphabet A	GOOGL	122.23	0.07%	1.56T	23.48M
Amazon.com	AMZN	123.43	-0.66%	1.27T	51.07M
NVIDIA	NVDA	387.7	0.68%	958.83B	42.40M
Tesla	TSLA	244.4	4.06%	774.63B	198.96M
Berkshire Hathaway A	BRKa	510840	-0.33%	733.80B	6.24K
Berkshire Hathaway B	BRKb	335.29	-0.2%	733.80B	2.60M
Meta Platforms	META	264.95	0.14%	679.00B	16.83M
Taiwan Semiconductor	TSM	102.8	2.86%	477.65B	14.91M
UnitedHealth	UNH	493.62	0.69%	459.58B	3.16M
Visa A	V	223.56	0.23%	458.11B	6.37M
J&J	JNJ	160.01	-0.16%	447.62B	6.23M
Louis Vuitton ADR	LVMUY	176.4	-0.03%	441.68B	99.13K
Exxon Mobil	XOM	107.39	-0.74%	434.11B	12.20M
Eli Lilly	LLY	446.04	0.09%	423.41B	2.07M
Walmart	WMT	153.09	0.6%	412.25B	5.18M
JPMorgan	JPM	141.01	0.2%	412.07B	7.23M
JPMorgan Chase Pref EE	JPM_pc	25.16	-0.4%	412.07B	73.03K

CSViewer - table_12 (1).csv

	Name	Symbol	Exchange	Sector	Industry	Last	Chg. %	Market Cap	Vol.
1	Apple	AAPL	NASDAQ	Technology	Computers,...	180.96	0.22%	2.85T	48.75M
2	Microsoft	MSFT	NASDAQ	Technology	Software & I...	326.79	0.47%	2.43T	22.44M
3	Alphabet C	GOOG	NASDAQ	Technology	Software & I...	122.87	0.16%	1.56T	20.05M
4	Alphabet A	GOOGL	NASDAQ	Technology	Software & I...	122.23	0.07%	1.56T	23.48M
5	Amazon.com	AMZN	NASDAQ	Consumer Cy...	Diversified R...	123.43	-0.66%	1.27T	51.07M
6	NVIDIA	NVDA	NASDAQ	Technology	Semiconduct...	387.7	0.68%	958.83B	42.40M
7	Tesla	TSLA	NASDAQ	Consumer Cy...	Automobiles...	244.4	4.06%	774.63B	198.96M
8	Berkshire Ha...	BRKa	NYSE	Consumer N...	Consumer G...	510840.0	-0.33%	733.80B	6.24K
9	Berkshire Ha...	BRKb	NYSE	Consumer N...	Consumer G...	335.29	-0.2%	733.80B	2.60M
10	Meta Platfor...	META	NASDAQ	Technology	Software & I...	264.95	0.14%	679.00B	16.83M
11	Taiwan Semi...	TSM	NYSE	Technology	Semiconduct...	102.8	2.86%	477.65B	14.91M
12	UnitedHealth	UNH	NYSE	Healthcare	Healthcare P...	493.62	0.69%	459.58B	3.16M
13	Visa A	V	NYSE	Technology	Software & I...	223.56	0.23%	458.11B	6.37M
14	J&J	JNJ	NYSE	Healthcare	Pharmaceuti...	160.01	-0.16%	447.62B	6.23M
15	Louis Vuitto...	LVMUY	OTC Markets	Consumer Cy...	Textiles & Ap...	176.4	-0.03%	441.68B	99.13K
16	Exxon Mobil	XOM	NYSE	Energy	Oil & Gas	107.39	-0.74%	434.11B	12.20M
17	Eli Lilly	LLY	NYSE	Healthcare	Pharmaceuti...	446.04	0.09%	423.41B	2.07M
18	Walmart	WMT	NYSE	Consumer N...	Food & Dru...	153.09	0.6%	412.25B	5.18M
19	JPMorgan	JPM	NYSE	Financials	Banking Serv...	141.01	0.2%	412.07B	7.23M
20	JPMorgan C...	JPM_pc	NYSE	Financials	Banking Serv...	25.16	-0.4%	412.07B	73.03K
21	Tencent Hol...	TCTZF	OTC Markets	Technology	Software & I...	42.51	1.14%	406.51B	21.14K
22	Tencent ADR	TCEHY	OTC Markets	Technology	Software & I...	42.63	-0.4%	406.51B	2.23M
23	Samsung Ele...	SSNLF	OTC Markets	Technology	Computers,...	40.6	0%	372.91B	0.00
24	Novo Nordis...	NVO	NYSE	Healthcare	Pharmaceuti...	158.05	-0.88%	353.11B	1.18M
25	Mastercard	MA	NYSE	Technology	Software & I...	369.22	0.44%	349.88B	2.19M

- https://coinmarketcap.com/
- https://www.oecd-ilibrary.org/economics/economics-key-tables-from-oecd_2074384x
- https://www.worldometers.info/gdp/gdp-by-country/
- https://en.wikipedia.org/wiki/Bj%C3%B6rk_discography
- https://en.wikipedia.org/wiki/List_of_countries_by_suicide_rate
- https://finance.yahoo.com/quote/TSLA/profile?p=TSLA

- https://www.oecd-ilibrary.org/economics/inflation-rate-2014-6_inflation-table-2014-6-en

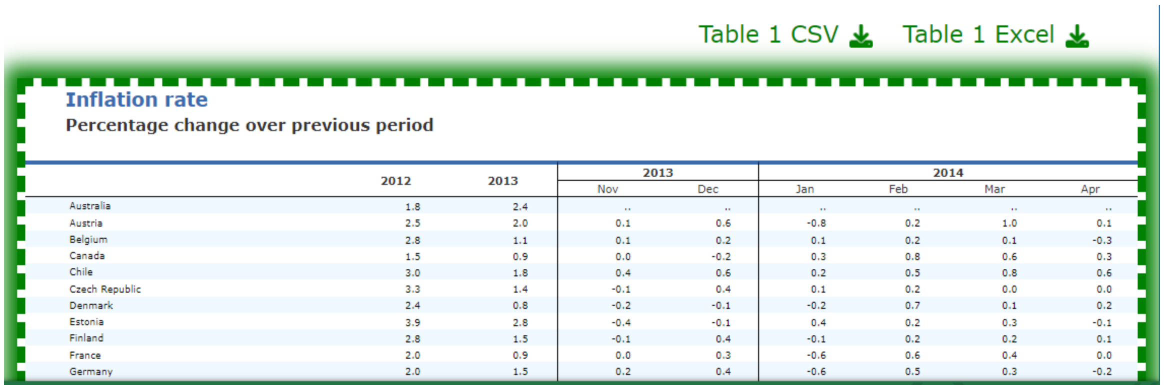


Table 1 CSV   Table 1 Excel

Inflation rate

Percentage change over previous period

	2012	2013	2013		2014			
			Nov	Dec	Jan	Feb	Mar	Apr
Australia	1.8	2.4	..	..	..	..	..	..
Austria	2.5	2.0	0.1	0.6	-0.8	0.2	1.0	0.1
Belgium	2.8	1.1	0.1	0.2	0.1	0.2	0.1	-0.3
Canada	1.5	0.9	0.0	-0.2	0.3	0.8	0.6	0.3
Chile	3.0	1.8	0.4	0.6	0.2	0.5	0.8	0.6
Czech Republic	3.3	1.4	-0.1	0.4	0.1	0.2	0.0	0.0
Denmark	2.4	0.8	-0.2	-0.1	-0.2	0.7	0.1	0.2
Estonia	3.9	2.8	-0.4	-0.1	0.4	0.2	0.3	-0.1
Finland	2.8	1.5	-0.1	0.4	-0.1	0.2	0.2	0.1
France	2.0	0.9	0.0	0.3	-0.6	0.6	0.4	0.0
Germany	2.0	1.5	0.2	0.4	-0.6	0.5	0.3	-0.2

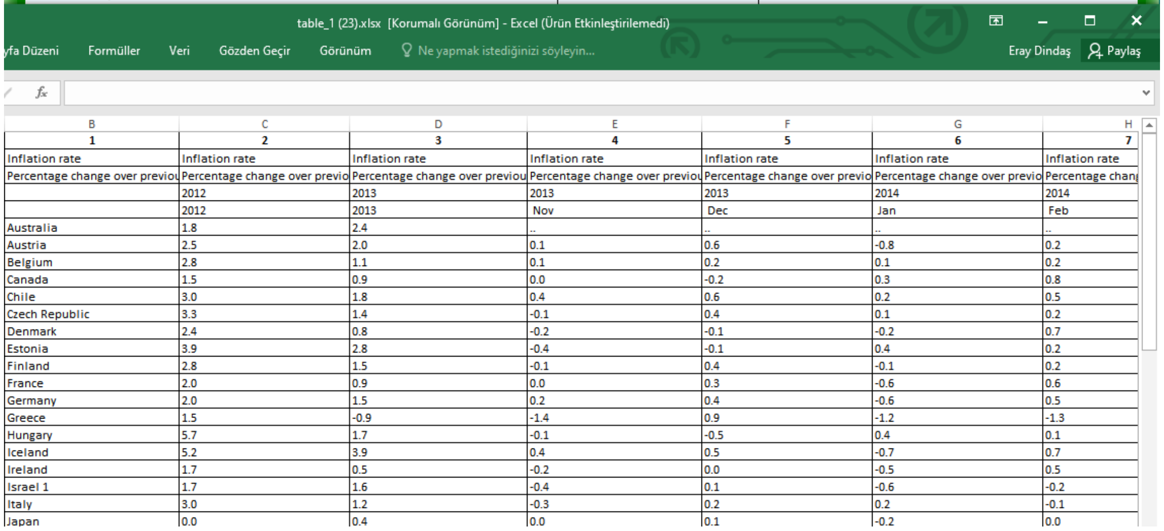


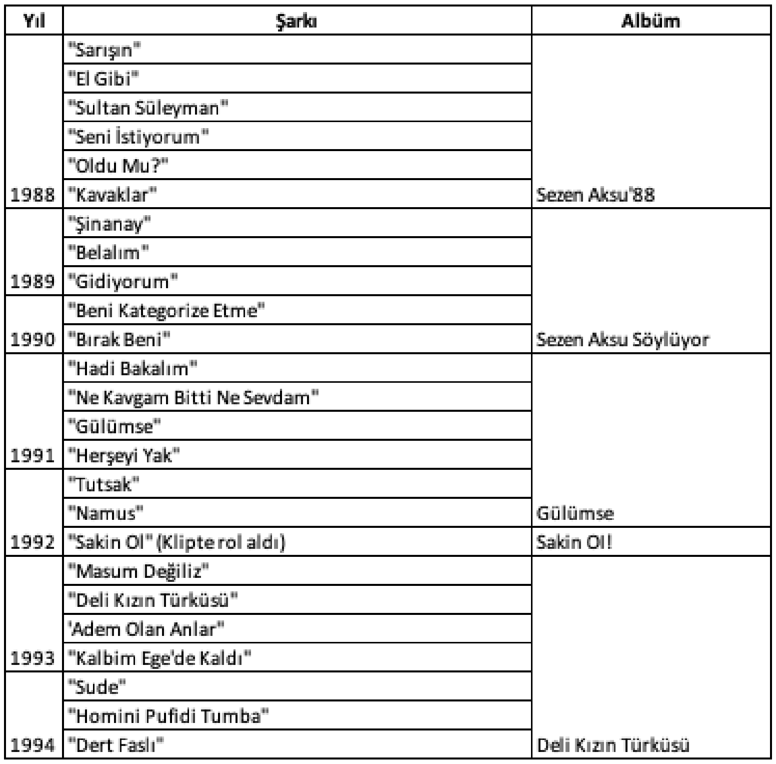
table_1 (23).xlsx [Korumalı Görünüm] - Excel (Ürün Etkinleştirilemedi)

B	C	D	E	F	G	H
1	2	3	4	5	6	7
Inflation rate	Inflation rate	Inflation rate	Inflation rate	Inflation rate	Inflation rate	Inflation rate
Percentage change over previou	Percentage change over previo	Percentage change over previou	Percentage change over previou	Percentage change over previo	Percentage change over previo	Percentage chan
	2012	2013	2013	2013	2014	2014
	2012	2013	Nov	Dec	Jan	Feb
Australia	1.8	2.4	..	..	..	..
Austria	2.5	2.0	0.1	0.6	-0.8	0.2
Belgium	2.8	1.1	0.1	0.2	0.1	0.2
Canada	1.5	0.9	0.0	-0.2	0.3	0.8
Chile	3.0	1.8	0.4	0.6	0.2	0.5
Czech Republic	3.3	1.4	-0.1	0.4	0.1	0.2
Denmark	2.4	0.8	-0.2	-0.1	-0.2	0.7
Estonia	3.9	2.8	-0.4	-0.1	0.4	0.2
Finland	2.8	1.5	-0.1	0.4	-0.1	0.2
France	2.0	0.9	0.0	0.3	-0.6	0.6
Germany	2.0	1.5	0.2	0.4	-0.6	0.5
Greece	1.5	-0.9	-1.4	0.9	-1.2	-1.3
Hungary	5.7	1.7	-0.1	-0.5	0.4	0.1
Iceland	5.2	3.9	0.4	0.5	-0.7	0.7
Ireland	1.7	0.5	-0.2	0.0	-0.5	0.5
Israel 1	1.7	1.6	-0.4	0.1	-0.6	-0.2
Italy	3.0	1.2	-0.3	0.2	0.2	-0.1
Japan	0.0	0.4	0.0	0.1	-0.2	0.0

Vision Method:

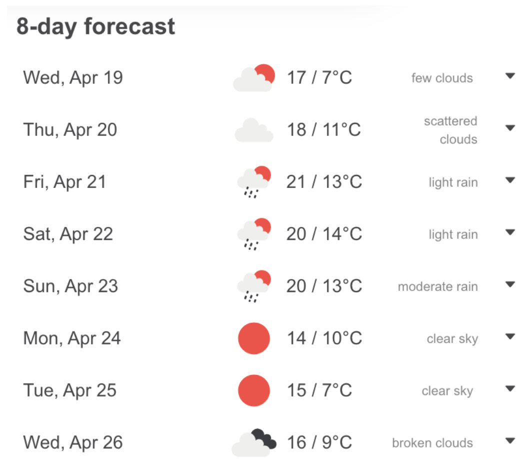
- Vision method can also work with the Scan method websites above.

- https://tr.wikipedia.org/wiki/Sezen_Aksu_diskografisi
  (In excel output, merged cells stay merged.)



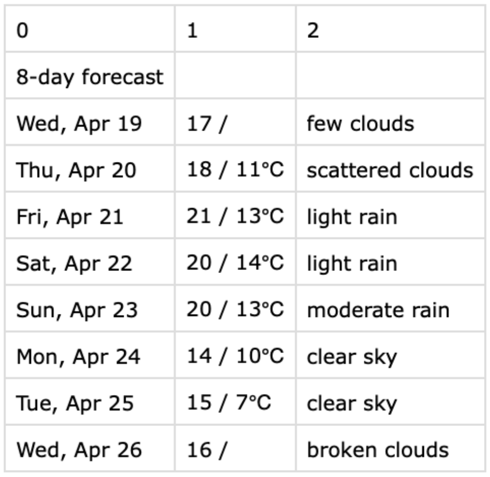
Yıl	Şarkı	Albüm
	"Sarışın"	
	"El Gibi"	
	"Sultan Süleyman"	
	"Seni İstiyorum"	
	"Oldu Mu?"	
1988	"Kavaklar"	Sezen Aksu'88
	"Şinanay"	
	"Belalım"	
1989	"Gidiyorum"	
	"Beni Kategorize Etme"	
1990	"Bırak Beni"	Sezen Aksu Söylüyor
	"Hadi Bakalım"	
	"Ne Kavgam Bitti Ne Sevdam"	
	"Gülümse"	
1991	"Herşeyi Yak"	
	"Tutsak"	
	"Namus"	Gülümse
1992	"Sakin Ol" (Klipte rol aldı)	Sakin Ol!
	"Masum Değiliz"	
	"Deli Kızın Türküsü"	
	'Adem Olan Anlar"	
1993	"Kalbim Ege'de Kaldı"	
	"Sude"	
	"Homini Pufidi Tumba"	
1994	"Dert Faslı"	Deli Kızın Türküsü

- https://openweathermap.org/city/5128581



8-day forecast

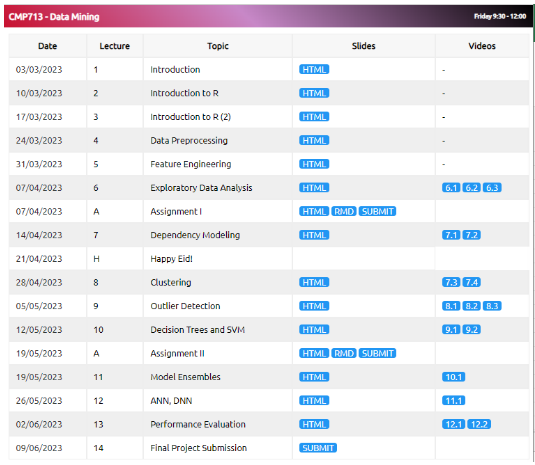
Wed, Apr 19	17 / 7°C	few clouds
Thu, Apr 20	18 / 11°C	scattered clouds
Fri, Apr 21	21 / 13°C	light rain
Sat, Apr 22	20 / 14°C	light rain
Sun, Apr 23	20 / 13°C	moderate rain
Mon, Apr 24	14 / 10°C	clear sky
Tue, Apr 25	15 / 7°C	clear sky
Wed, Apr 26	16 / 9°C	broken clouds



0	1	2
8-day forecast		
Wed, Apr 19	17 /	few clouds
Thu, Apr 20	18 / 11°C	scattered clouds
Fri, Apr 21	21 / 13°C	light rain
Sat, Apr 22	20 / 14°C	light rain
Sun, Apr 23	20 / 13°C	moderate rain
Mon, Apr 24	14 / 10°C	clear sky
Tue, Apr 25	15 / 7°C	clear sky
Wed, Apr 26	16 /	broken clouds

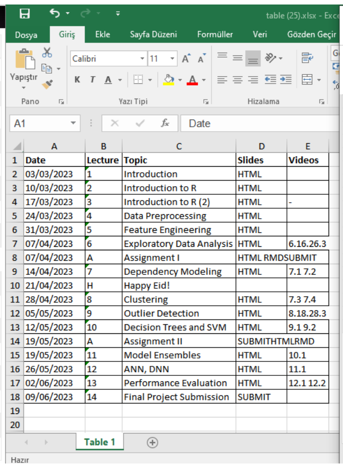
- https://www.nascar.com/ => “schedule” => “race results” tab
- https://www.worldometers.info/

- http://burkaygenc.ml

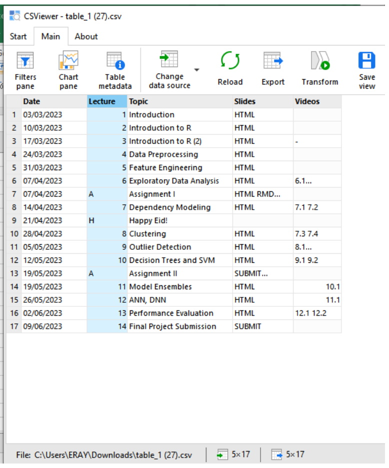


CMP713 - Data Mining — Friday 9:30 - 12:00

Date	Lecture	Topic	Slides	Videos
03/03/2023	1	Introduction	HTML	-
10/03/2023	2	Introduction to R	HTML	-
17/03/2023	3	Introduction to R (2)	HTML	-
24/03/2023	4	Data Preprocessing	HTML	-
31/03/2023	5	Feature Engineering	HTML	-
07/04/2023	6	Exploratory Data Analysis	HTML	6.1 6.2 6.3
07/04/2023	A	Assignment I	HTML RMD SUBMIT	
14/04/2023	7	Dependency Modeling	HTML	7.1 7.2
21/04/2023	H	Happy Eid!		
28/04/2023	8	Clustering	HTML	7.3 7.4
05/05/2023	9	Outlier Detection	HTML	8.1 8.2 8.3
12/05/2023	10	Decision Trees and SVM	HTML	9.1 9.2
19/05/2023	A	Assignment II	HTML RMD SUBMIT	
19/05/2023	11	Model Ensembles	HTML	10.1
26/05/2023	12	ANN, DNN	HTML	11.1
02/06/2023	13	Performance Evaluation	HTML	12.1 12.2
09/06/2023	14	Final Project Submission	SUBMIT	



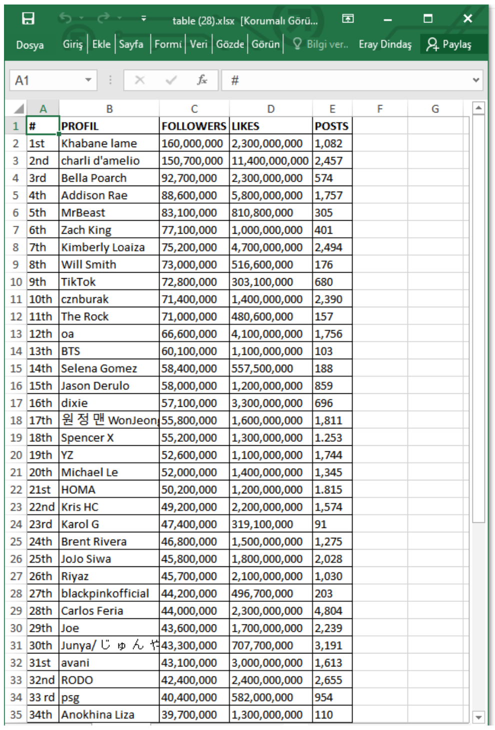
	A	B	C	D	E
1	Date	Lecture	Topic	Slides	Videos
2	03/03/2023	1	Introduction	HTML	
3	10/03/2023	2	Introduction to R	HTML	
4	17/03/2023	3	Introduction to R (2)	HTML	-
5	24/03/2023	4	Data Preprocessing	HTML	
6	31/03/2023	5	Feature Engineering	HTML	
7	07/04/2023	6	Exploratory Data Analysis	HTML	6.16.26.3
8	07/04/2023	A	Assignment I	HTML RMDSUBMIT	
9	14/04/2023	7	Dependency Modeling	HTML	7.1 7.2
10	21/04/2023	H	Happy Eid!		
11	28/04/2023	8	Clustering	HTML	7.3 7.4
12	05/05/2023	9	Outlier Detection	HTML	8.18.28.3
13	12/05/2023	10	Decision Trees and SVM	HTML	9.1 9.2
14	19/05/2023	A	Assignment II	SUBMITHTMLRMD	
15	19/05/2023	11	Model Ensembles	HTML	10.1
16	26/05/2023	12	ANN, DNN	HTML	11.1
17	02/06/2023	13	Performance Evaluation	HTML	12.1 12.2
18	09/06/2023	14	Final Project Submission	SUBMIT	
19					
20					



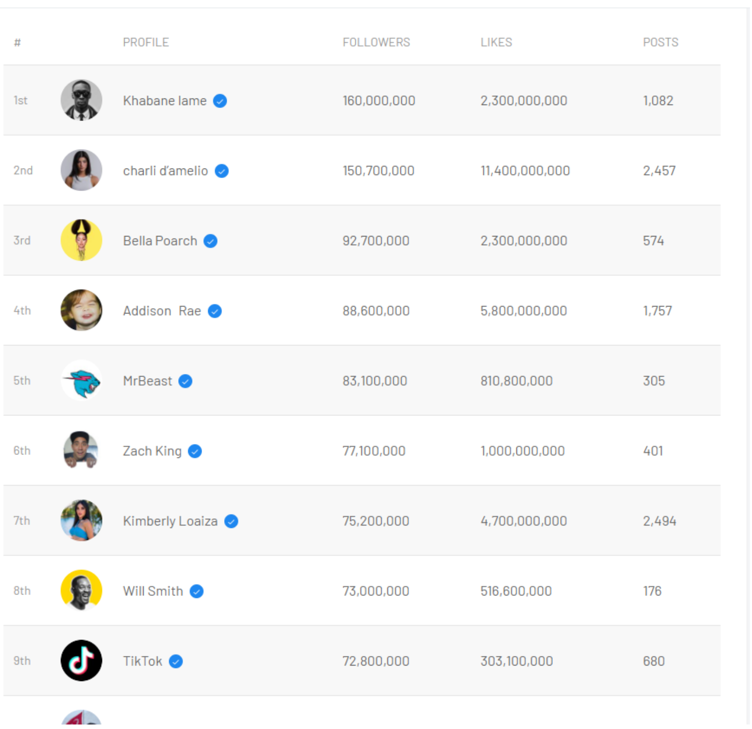
	Date	Lecture	Topic	Slides	Videos
1	03/03/2023	1	Introduction	HTML	
2	10/03/2023	2	Introduction to R	HTML	
3	17/03/2023	3	Introduction to R (2)	HTML	-
4	24/03/2023	4	Data Preprocessing	HTML	
5	31/03/2023	5	Feature Engineering	HTML	
6	07/04/2023	6	Exploratory Data Analysis	HTML	6.1...
7	07/04/2023	A	Assignment I	HTML RMD...	
8	14/04/2023	7	Dependency Modeling	HTML	7.1 7.2
9	21/04/2023	H	Happy Eid!		
10	28/04/2023	8	Clustering	HTML	7.3 7.4
11	05/05/2023	9	Outlier Detection	HTML	8.1...
12	12/05/2023	10	Decision Trees and SVM	HTML	9.1 9.2
13	19/05/2023	A	Assignment II	SUBMIT...	
14	19/05/2023	11	Model Ensembles	HTML	10.1
15	26/05/2023	12	ANN, DNN	HTML	11.1
16	02/06/2023	13	Performance Evaluation	HTML	12.1 12.2
17	09/06/2023	14	Final Project Submission	SUBMIT	

File: C:\Users\ERAY\Downloads\table_1 (27).csv | 5×17 | 5×17

- www.socialtracker.io/toplists/top-50-tiktok-users-by-followers/

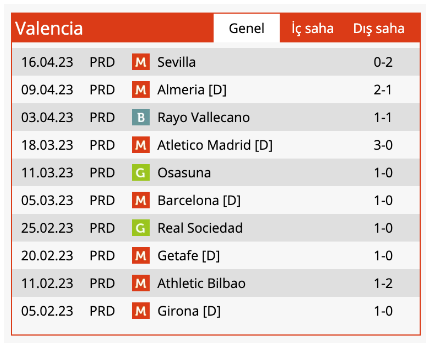


	A	B	C	D	E	F	G
1	#	PROFIL	FOLLOWERS	LIKES	POSTS		
2	1st	Khabane lame	160,000,000	2,300,000,000	1,082		
3	2nd	charli d'amelio	150,700,000	11,400,000,000	2,457		
4	3rd	Bella Poarch	92,700,000	2,300,000,000	574		
5	4th	Addison Rae	88,600,000	5,800,000,000	1,757		
6	5th	MrBeast	83,100,000	810,800,000	305		
7	6th	Zach King	77,100,000	1,000,000,000	401		
8	7th	Kimberly Loaiza	75,200,000	4,700,000,000	2,494		
9	8th	Will Smith	73,000,000	516,600,000	176		
10	9th	TikTok	72,800,000	303,100,000	680		
11	10th	cznburak	71,400,000	1,400,000,000	2,390		
12	11th	The Rock	71,000,000	480,600,000	157		
13	12th	oa	66,600,000	4,100,000,000	1,756		
14	13th	BTS	60,100,000	1,100,000,000	103		
15	14th	Selena Gomez	58,400,000	557,500,000	188		
16	15th	Jason Derulo	58,000,000	1,200,000,000	859		
17	16th	dixie	57,100,000	3,300,000,000	696		
18	17th	원 정 맨 WonJeon	55,800,000	1,600,000,000	1,811		
19	18th	Spencer X	55,200,000	1,300,000,000	1.253		
20	19th	YZ	52,600,000	1,100,000,000	1,744		
21	20th	Michael Le	52,000,000	1,400,000,000	1,345		
22	21st	HOMA	50,200,000	1,200,000,000	1.815		
23	22nd	Kris HC	49,200,000	2,200,000,000	1,574		
24	23rd	Karol G	47,400,000	319,100,000	91		
25	24th	Brent Rivera	46,800,000	1,500,000,000	1,275		
26	25th	JoJo Siwa	45,800,000	1,800,000,000	2,028		
27	26th	Riyaz	45,700,000	2,100,000,000	1,030		
28	27th	blackpinkofficial	44,200,000	496,700,000	203		
29	28th	Carlos Feria	44,000,000	2,300,000,000	4,804		
30	29th	Joe	43,600,000	1,700,000,000	2,239		
31	30th	Junya/じゅんや	43,300,000	707,700,000	3,191		
32	31st	avani	43,100,000	3,000,000,000	1,613		
33	32nd	RODO	42,400,000	2,400,000,000	2,655		
34	33 rd	psg	40,400,000	582,000,000	954		
35	34th	Anokhina Liza	39,700,000	1,300,000,000	110		

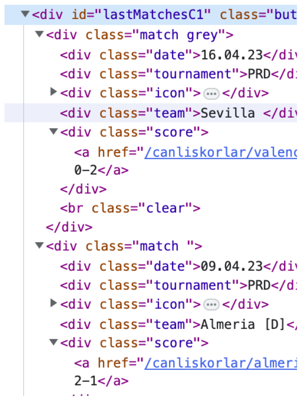


#	PROFILE	FOLLOWERS	LIKES	POSTS
1st	Khabane lame	160,000,000	2,300,000,000	1,082
2nd	charli d'amelio	150,700,000	11,400,000,000	2,457
3rd	Bella Poarch	92,700,000	2,300,000,000	574
4th	Addison Rae	88,600,000	5,800,000,000	1,757
5th	MrBeast	83,100,000	810,800,000	305
6th	Zach King	77,100,000	1,000,000,000	401
7th	Kimberly Loaiza	75,200,000	4,700,000,000	2,494
8th	Will Smith	73,000,000	516,600,000	176
9th	TikTok	72,800,000	303,100,000	680

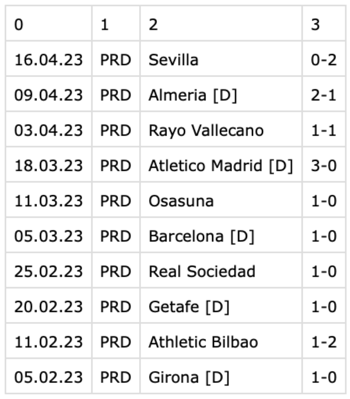
- www.sporx.com/valencia-rayo-vallecano-maci-canli-3918207



Valencia			Genel	İç saha	Dış saha
16.04.23	PRD	M	Sevilla		0-2
09.04.23	PRD	M	Almeria [D]		2-1
03.04.23	PRD	B	Rayo Vallecano		1-1
18.03.23	PRD	M	Atletico Madrid [D]		3-0
11.03.23	PRD	G	Osasuna		1-0
05.03.23	PRD	M	Barcelona [D]		1-0
25.02.23	PRD	G	Real Sociedad		1-0
20.02.23	PRD	M	Getafe [D]		1-0
11.02.23	PRD	M	Athletic Bilbao		1-2
05.02.23	PRD	M	Girona [D]		1-0



```
▼<div id="lastMatchesC1" class="but
  ▼<div class="match grey">
     <div class="date">16.04.23</di
     <div class="tournament">PRD</d
    ▶<div class="icon">⋯</div>
     <div class="team">Sevilla </di
    ▼<div class="score">
       <a href="/canliskorlar/valen
       0–2</a>
     </div>
     <br class="clear">
   </div>
  ▼<div class="match ">
     <div class="date">09.04.23</di
     <div class="tournament">PRD</d
    ▶<div class="icon">⋯</div>
     <div class="team">Almeria [D]<
    ▼<div class="score">
       <a href="/canliskorlar/almer
       2–1</a>
```



0	1	2	3
16.04.23	PRD	Sevilla	0-2
09.04.23	PRD	Almeria [D]	2-1
03.04.23	PRD	Rayo Vallecano	1-1
18.03.23	PRD	Atletico Madrid [D]	3-0
11.03.23	PRD	Osasuna	1-0
05.03.23	PRD	Barcelona [D]	1-0
25.02.23	PRD	Real Sociedad	1-0
20.02.23	PRD	Getafe [D]	1-0
11.02.23	PRD	Athletic Bilbao	1-2
05.02.23	PRD	Girona [D]	1-0

Evaluation

Overall, our outputs have almost exactly the same column and row structures as how they are displayed on websites. And the data inside cells are highly accurate too.

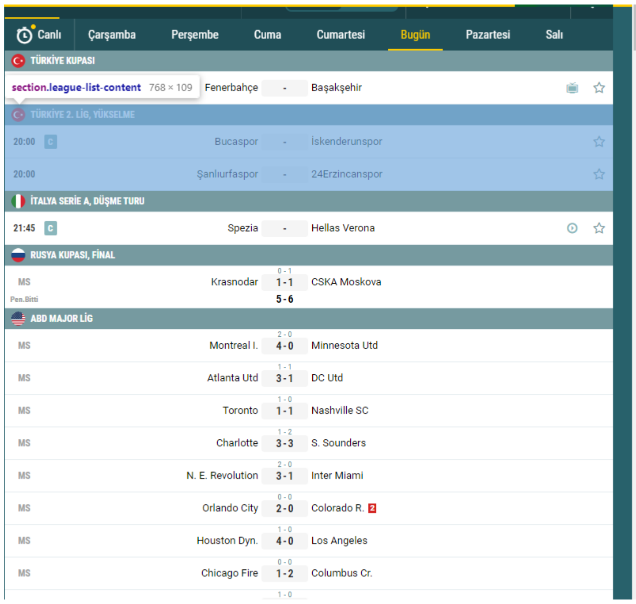
There is no method that can achieve 100% accuracy in the diversity of online tabular representations, yet, our system greatly reduces the manual labor needed to copy-paste-check data from the web. Our editable output formats give our users the ability to quickly fix any incorrect part of the output.

Non-Released Method attempt: 3.0
Catching Type C (non-standard) Tables. without Vision ?

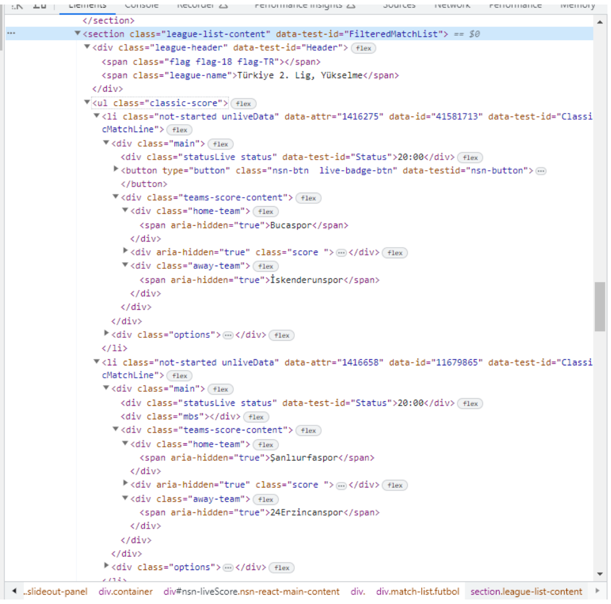
One of the major limitations we encountered was the inability to recognize the **Type-C** tables (created using <div> or any other than <table> tags). Standard <table> tags give a predictable structure, and a tree-like hierarchy to the HTML code, inherently. However, in some circumstances, developers may prefer the flexibility of <div> tags in terms of creating custom layouts and styling choices. However, in these cases, the lack of general criterion also constrains us.

We investigated the usage of libraries like Lxml and BeautifulSoup to create pattern recognition scripts in an effort to overcome these difficulties. Our trials, however, showed that the hardcoding required to get good accuracy, limited the applicability of this kind of approach. The scripts we developed proved to be too specific and lacked the generalization required to accommodate the diversity of table structures. As a result, we decided that these scripts weren't appropriate to integrate into our tool. The inclusion of <div>-based table recognition scripts, an experimental feature that is not yet made usable, although desirable, would have compromised the overall usability of our tool. And after we have successfully integrated the method 2: Vision method, all our use-cases have been already covered.

Let us now consider the example below. If we compare these two tables based on their HTML elements, it appears that different non-universal patterns are being used, and a dynamic standard tool cannot extract the table from here. Additionally, non-tabular elements are mixed in throughout the content, further complicating the process of extracting the table; even if the user specifically selects the desired tabular element.

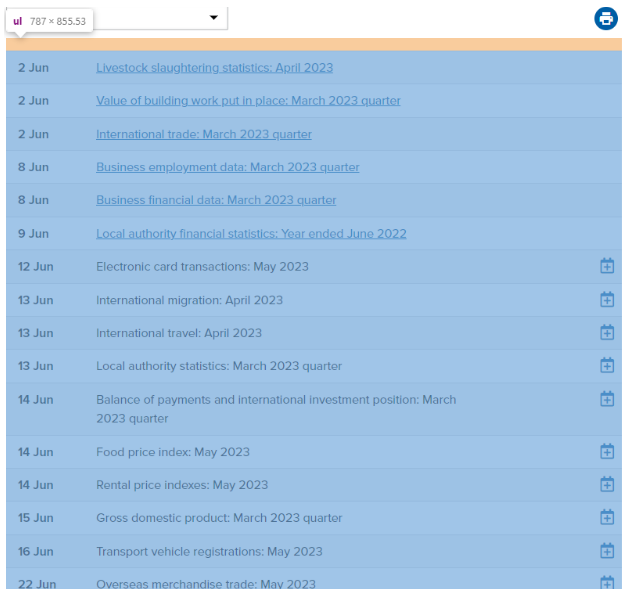


TÜRKİYE KUPASI				
	Fenerbahçe	-	Başakşehir	
TÜRKİYE 2. LİG, YÜKSELME				
20:00	Bucaspor	-	İskenderunspor	
20:00	Şanlıurfaspor	-	24Erzincanspor	
İTALYA SERİE A, DÜŞME TURU				
21:45	Spezia	-	Hellas Verona	
RUSYA KUPASI, FİNAL				
MS Pen.Bitti	Krasnodar	0 - 1 1 - 1 5 - 6	CSKA Moskova	
ABD MAJOR LİG				
MS	Montreal I.	2 - 0 4 - 0	Minnesota Utd	
MS	Atlanta Utd	1 - 1 3 - 1	DC Utd	
MS	Toronto	1 - 0 1 - 1	Nashville SC	
MS	Charlotte	1 - 2 3 - 3	S. Sounders	
MS	N. E. Revolution	2 - 0 3 - 1	Inter Miami	
MS	Orlando City	0 - 0 2 - 0	Colorado R.	2
MS	Houston Dyn.	1 - 0 4 - 0	Los Angeles	
MS	Chicago Fire	0 - 0 1 - 2	Columbus Cr.	
		1 - 0		

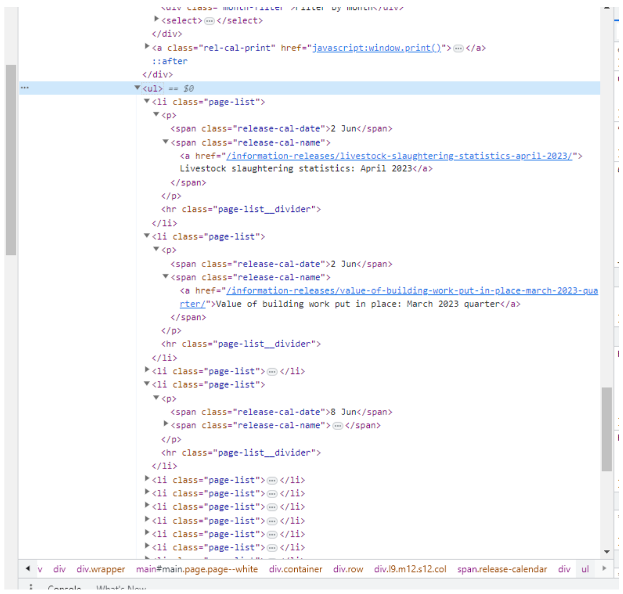


```
</section>
▼<section class="league-list-content" data-test-id="FilteredMatchList"> == $0
  ▼<div class="league-header" data-test-id="Header"> flex
     <span class="flag flag-18 flag-TR"></span>
     <span class="league-name">Türkiye 2. Lig, Yükselme</span>
   </div>
  ▼<ul class="classic-score"> flex
    ▼<li class="not-started unliveData" data-attr="1416275" data-id="41581713" data-test-id="Classi
     cMatchLine"> flex
      ▼<div class="main"> flex
         <div class="statusLive status" data-test-id="Status">20:00</div> flex
        ▶<button type="button" class="nsn-btn  live-badge-btn" data-testid="nsn-button"> ⋯
         </button>
        ▼<div class="teams-score-content"> flex
          ▼<div class="home-team"> flex
             <span aria-hidden="true">Bucaspor</span>
           </div>
          ▶<div aria-hidden="true" class="score "> ⋯ </div> flex
          ▼<div class="away-team"> flex
             <span aria-hidden="true">İskenderunspor</span>
           </div>
         </div>
       </div>
      ▶<div class="options"> ⋯ </div> flex
     </li>
    ▼<li class="not-started unliveData" data-attr="1416658" data-id="11679865" data-test-id="Classi
     cMatchLine"> flex
      ▼<div class="main"> flex
         <div class="statusLive status" data-test-id="Status">20:00</div> flex
         <div class="mbs"></div> flex
        ▼<div class="teams-score-content"> flex
          ▼<div class="home-team"> flex
             <span aria-hidden="true">Şanlıurfaspor</span>
           </div>
          ▶<div aria-hidden="true" class="score "> ⋯ </div> flex
          ▼<div class="away-team"> flex
             <span aria-hidden="true">24Erzincanspor</span>
           </div>
         </div>
       </div>
      ▶<div class="options"> ⋯ </div> flex
```

..slideout-panel div.container div#nsn-liveScore.nsn-react-main-content div. div.match-list.futbol section.league-list-content



2 Jun	Livestock slaughtering statistics: April 2023
2 Jun	Value of building work put in place: March 2023 quarter
2 Jun	International trade: March 2023 quarter
8 Jun	Business employment data: March 2023 quarter
8 Jun	Business financial data: March 2023 quarter
9 Jun	Local authority financial statistics: Year ended June 2022
12 Jun	Electronic card transactions: May 2023
13 Jun	International migration: April 2023
13 Jun	International travel: April 2023
13 Jun	Local authority statistics: March 2023 quarter
14 Jun	Balance of payments and international investment position: March 2023 quarter
14 Jun	Food price index: May 2023
14 Jun	Rental price indexes: May 2023
15 Jun	Gross domestic product: March 2023 quarter
16 Jun	Transport vehicle registrations: May 2023
22 Jun	Overseas merchandise trade: May 2023



```
      ▶<select> ⋯ </select>
     </div>
    ▶<a class="rel-cal-print" href="javascript:window.print()"> ⋯ </a>
     ::after
   </div>
  ▼<ul> == $0
    ▼<li class="page-list">
      ▼<p>
         <span class="release-cal-date">2 Jun</span>
        ▼<span class="release-cal-name">
           <a href="/information-releases/livestock-slaughtering-statistics-april-2023/">
           Livestock slaughtering statistics: April 2023</a>
         </span>
       </p>
       <hr class="page-list__divider">
     </li>
    ▼<li class="page-list">
      ▼<p>
         <span class="release-cal-date">2 Jun</span>
        ▼<span class="release-cal-name">
           <a href="/information-releases/value-of-building-work-put-in-place-march-2023-qua
           rter/">Value of building work put in place: March 2023 quarter</a>
         </span>
       </p>
       <hr class="page-list__divider">
     </li>
    ▶<li class="page-list"> ⋯ </li>
    ▼<li class="page-list">
      ▼<p>
         <span class="release-cal-date">8 Jun</span>
        ▶<span class="release-cal-name"> ⋯ </span>
       </p>
       <hr class="page-list__divider">
     </li>
    ▶<li class="page-list"> ⋯ </li>
    ▶<li class="page-list"> ⋯ </li>
    ▶<li class="page-list"> ⋯ </li>
    ▶<li class="page-list"> ⋯ </li>
    ▶<li class="page-list"> ⋯ </li>
    ▶<li class="page-list"> ⋯ </li>
```

v div div.wrapper main#main.page.page--white div.container div.row div.l9.m12.s12.col span.release-calendar div ul

About more of the methodology and reasoning, please refer to [9] our Progress Report 3.

Furthermore, we investigated that usage of non-standard tables on the web are relatively lower than standard ones, and discouraged among the web development community; making them less imperative for us to prioritize. Nevertheless, we luckily found a trained plug-and-play Computer Vision API and configured to complete our system.

A rare case: Content inside externally loaded content <frame>

Sometimes websites embed the contents of other websites (same origin or cross origin) in the same window. JavaScript is able to parse such content as follows, yet, only if it is the same origin/domain.

```
for (let i = 0; i < window.frames.length; i++) {
  console.log(window.frames[i].document.getElementsByTagName('html')[0].innerHTML)
}
```

Yet, most of the time, this content is cross-origin, and thus access/modification is blocked by browsers as a security feature, which we cannot overcome. Also, it is very rare for websites to serve Tabular content in frames; we have encountered only a single example since the beginning of our project, which is Dr. Genç's http://burkaygenc.ml website. In such cases, as mentioned [9] in our Progress_Report_3, our Vision method is able to handle the content that is visible.

Release & Quality Concerns

Table Catcher is not tested for more than 3 simultaneous users. Large scale concurrency might require server reconfiguration, if Table Catcher becomes popular.

We use Django's and Azure's default security settings, which we did not test for for example denial-of-service-attacks, yet we have a 2 minutes timeout limit for a request.

In terms of exception handling, we stop processing and display simple "table not recognized" messages.

Even though we utilized tested/trusted/popular tools/libraries/resources, we did not employ a strategy for formal/numerical evaluation of system integration, other than manual testing.

Discussion

Throughout the two semesters, we continuously extended Table Catcher's features and usability. Table Catcher is currently a working software. Aim of our development was to overcome limitations, extend features, and reach as good as it can towards the goal; in our limited development time. We were always open to revisions.

Overall, we have proudly proved our proposal and created an end-to-end working product.
We have achieved our primary objectives, yet mostly did not deliver our "further optional features" mentioned [9] in the "BM480 Term Study Plan".

We have not faced the technology risk of not being able to find the right tools or algorithms. We have overcome the time management and social environment risks related to the recent events, and stayed as a team till the end.

In general, we followed the design philosophies we have been taught, throughout our Hacettepe CS education; such as modularity, maintainability, and low-coupling. We know there are similar tools in the industry with many features; we preferred a robust and minimal solution, took care not to overbear users and also us developers with shiny features.

The Impact and Future Directions (/ 15 Points)

Explain the potential (or current if exist) impacts of your outcome in terms of how the methods and results will be used in real life, how it will change an existing process, or where it will be published, etc. Also, explain what would be the next step if the project is continued in the future, what kind of qualitative and/or quantitative updates can be made, shorty, where this project can go from here?

Overall, we combined some methods and provided yet another free tool that can facilitate data collection work of Data Scientists or anyone who needs it. Table Catcher is open-source and open to evolution; thus, along with our existing ideas for future development, anyone can contribute.

We have licensed our source code with MIT license; which states that we do give warranty and we cannot be held liable of possible damages arising in usage. This decision is on point, since we do not control by whom, or for what purposes or websites Table Catcher is used for, and currently cannot avoid a user with malicious intents retrieving or leaking data illegally.

MIT License also provides copyright, yet, our code can be [used / modified / published / distributed / commercially used as closed-source] without restriction.

Since we have extensively documented each phase, change, decision driver and decision; it will be easy to pick up on further development for any future contributors. Our efforts could also be seen as an online tutorial for building a system.

Some of our Future Development ideas Table catcher could be equipped with are:

- A data manipulation mode, similar to Matthew Harris's experiments [6] mentioned in the Literature Review section.
- Horizontal scaling of Backend, in case of popularity.
- Cross-browser support. With minimum changes in code, reaching maximum users. Publishing for free in extension markets for increasing user trust and reachability, and for even easier installation.

- Storing previously created tables in user accounts. However, this needs consideration regarding allowed server disk space, database, security.
- Encouraging Improvements to the Vision algorithm of `img2table` library, or independently developing our own vision algorithm.
- A user-helped mode, where the possible structure of the detected Table is shown to user as a preview on the browser, and user is able to move around and rearrange columns, rows, text, to obtain their own desired result. This could especially help when algorithms are not certain or confident about the structure, and give our user the final choice.

REFERENCES

[1] https://georgemike.com/tablecapture/

[2] https://brightdata.com/

[3] https://apps.apple.com/tr/app/microsoft-lens-pdf-scanner/id975925059

[4] An interactive method for accessing tables in HTML.
Toshiya Oogane, Chieko Asakawa, 1998. https://dl.acm.org/doi/abs/10.1145/274497.274521

[5] An automated approach for retrieving hierarchical data from HTML tables.
Seungjin Lim, Yiu Kai Dennis Ng, 1999. https://dl.acm.org/doi/abs/10.1145/319950.320052

[6] Harris, 2023.
https://towardsdatascience.com/parsing-irregular-spreadsheet-tables-in-humanitarian-datasets-with-some-help-from-gpt-3-57efb3d80d45

[7] figure. DOM tree. https://shorturl.at/ruTVY

[8] figure. HTML table. https://shorturl.at/oswMV

[9] Table Catcher Documentation
https://drive.google.com/drive/folders/1xB9fLiRFME2lGquD1h1NbtHFX_4bdvfv?usp=sharing