ext_aeid4666-EsportsProSettingsAsk_v11

Analysis & Design Document

Date – 26 Oct 2022

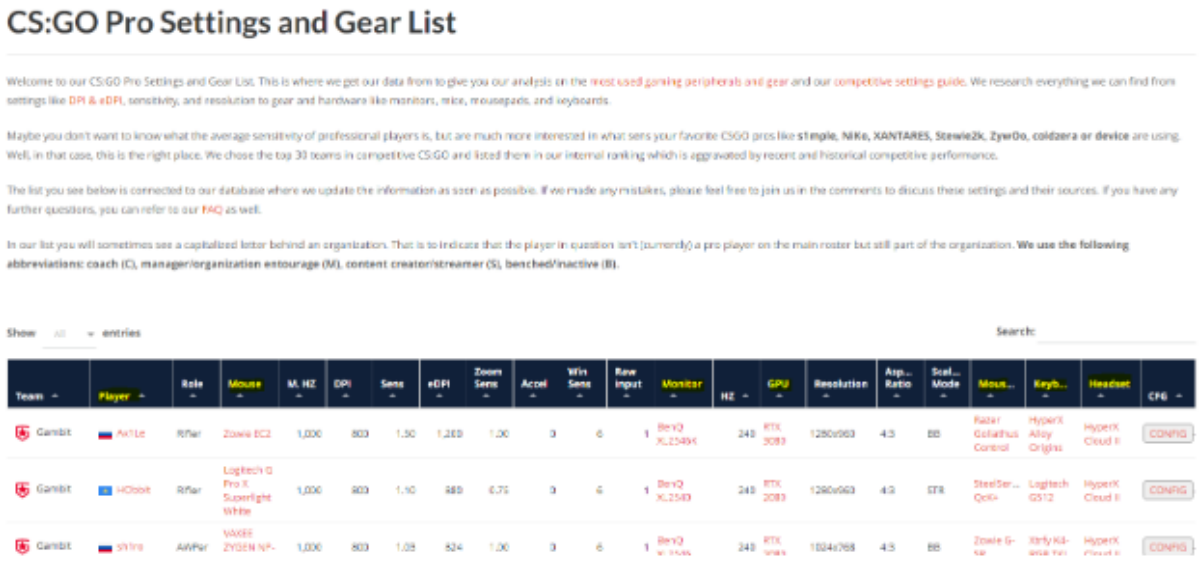
Table of Contents

Topic **Page No.**

1. Scope of work

Scrap the below data from SITE: https://prosettings.net/cs-go-pro-settings-gear-list/

1. Link of the page we are collecting (there is one for each game)
2. Link of the player
3. Mouse Name
4. Mouse Link
5. Monitor Name
6. Monitor Link
7. GPU Name
8. GPU
9. Mousepad Name
10. Mousepad Link
11. Keyboard Name
12. Keyboard Link
13. Headset Name
14. Headset Link



CS:GO Pro Settings and Gear List

Welcome to our CS:GO Pro Settings and Gear List. This is where we get our data from to give you our analysis on the most used gaming peripherals and gear and our competitive settings guide. We research everything we can find from settings like DPI & eDPI, sensitivity, and resolution to gear and hardware like monitors, mice, mousepads, and keyboards.

Maybe you don't want to know what the average sensitivity of professional players is, but are much more interested in what sens your favorite CSGO pros like **stingle, NiKo, XANTARES, Stewie2k, ZywOo, coldzera or device** are using. Well, in that case, this is the right place. We chose the top 30 teams in competitive CS:GO and listed there in our internal ranking which is aggravated by recent and historical competitive performance.

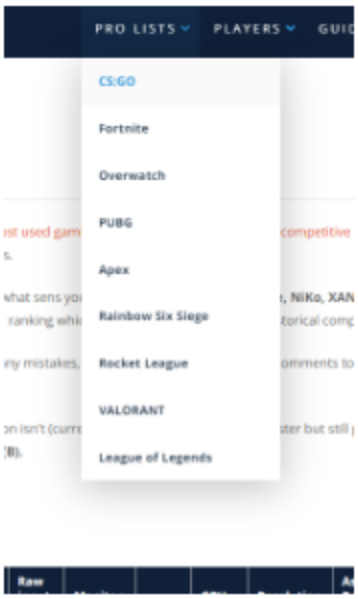
The list you see below is connected to our database where we update the information as soon as possible. If we made any mistakes, please feel free to join us in the comments to discuss these settings and their sources. If you have any further questions, you can refer to our FAQ as well.

In our list you will sometimes see a capitalized letter behind an organization. That is to indicate that the player in question isn't (currently) a pro player on the main roster but still part of the organization. **We use the following abbreviations: coach (C), manager/organization entourage (M), content creator/streamer (S), benched/inactive (B).**

Show All entries Search:

Team	Player	Role	Mouse	M. HZ	DPI	Sens	eDPI	Zoom Sens	Accel	Win Sens	Raw Input	Monitor	HZ	GPU	Resolution	Asp... Ratio	Scal... Mode	Mous...	Keyb...	Headset	CFG
Gambit	Ax1Le	Rifler	Zowie EC2	1,000	800	1.50	1,200	1.00	0	6	1	BenQ XL2546K	240	RTX 3080	1280x960	4:3	BB	Razer Goliathus Control	HyperX Alloy Origins	HyperX Cloud II	CONFIG
Gambit	HObbit	Rifler	Logitech G Pro X Superlight White	1,000	800	1.10	880	0.75	0	6	1	BenQ XL2540	240	RTX 2080	1280x960	4:3	STR	SteelSer... QcK+	Logitech G512	HyperX Cloud II	CONFIG
Gambit	Viine	AWPer	VAXEE ZYGEN NP-	1,000	800	1.08	824	1.00	0	6	1	BenQ	240	RTX	1024x768	4:3	BB	Zowie G-	Varty K6-	HyperX	CONFIG

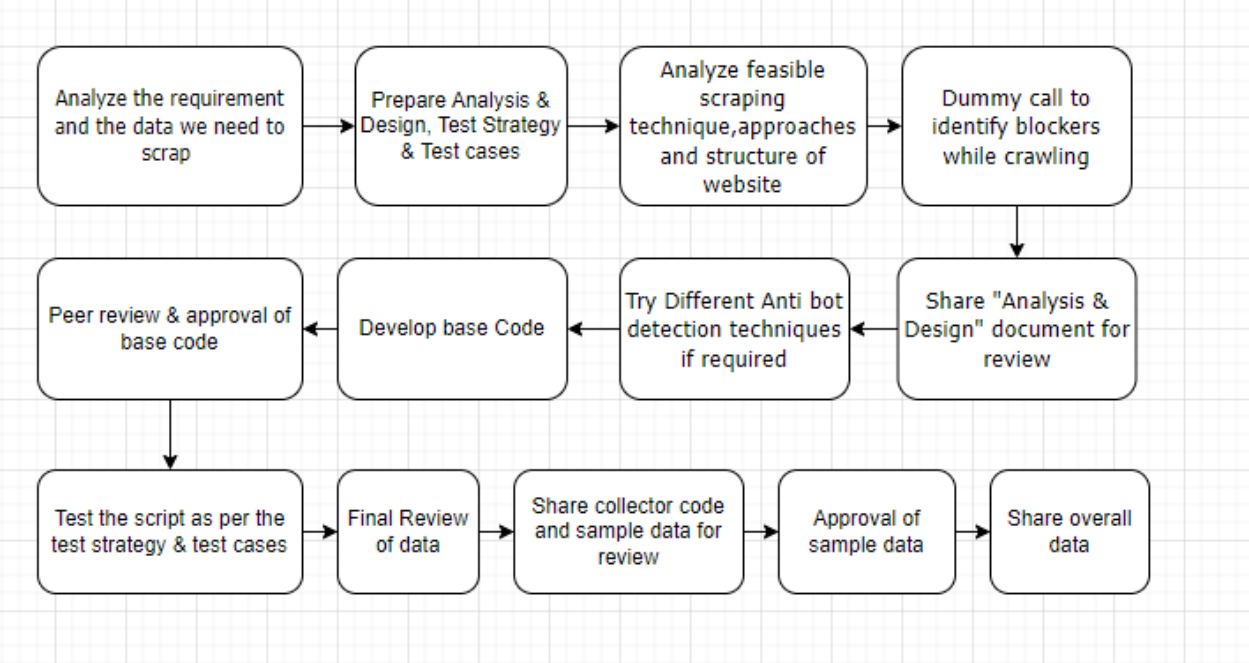
2. Collect the columns highlighted in yellow
3. Each of those values is both a name and hyperlinked URL. Please collect both.
4. Go to the next game by hovering over “Pro Lists”



2. Solution Approach

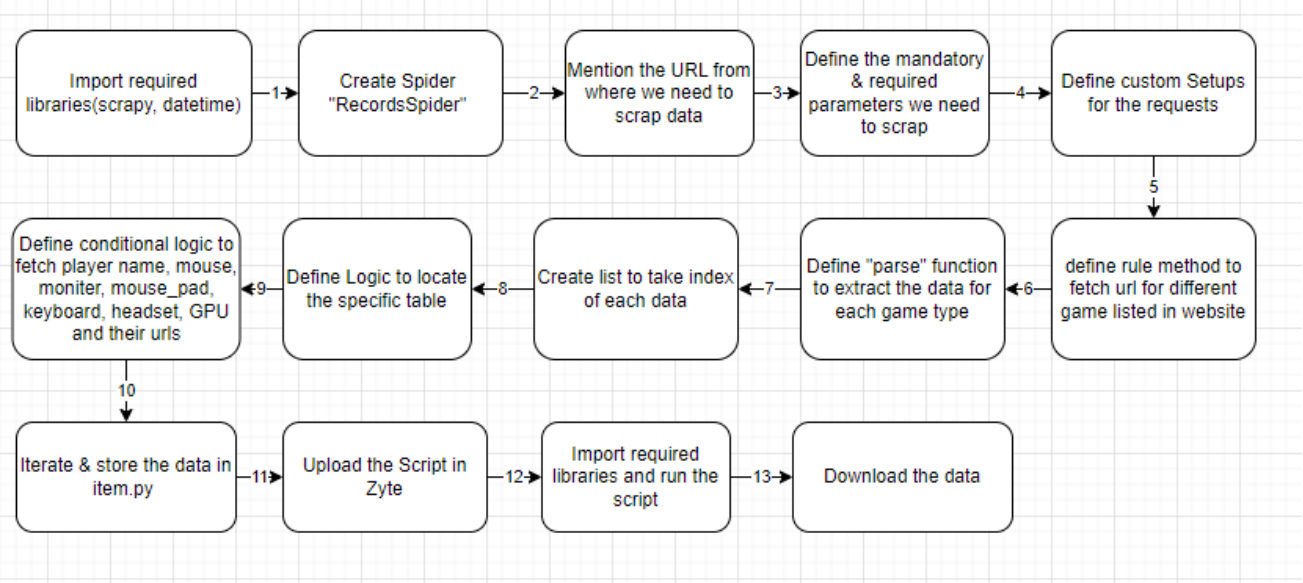
We are following the below steps to develop the script as per the requirement

- The website is **global**, hence only one collector code is needed.
- We are fetching the required details for each product.
- Checked the javascript data (the data we get from AJAX calls) with the help of view page source.
- The data is rendered in a tabular format, so we are scrapping the data from this table.


Analyze the requirement and the data we need to scrap
Prepare Analysis & Design, Test Strategy & Test cases
Analyze feasible scraping technique,approaches and structure of website
Dummy call to identify blockers while crawling
Share "Analysis & Design" document for review
Try Different Anti bot detection techniques if required
Develop base Code
Peer review & approval of base code
Test the script as per the test strategy & test cases
Final Review of data
Share collector code and sample data for review
Approval of sample data
Share overall data

3. Script Development Flow

Below steps are followed to create spider



4. Technology Considerations

Custom signup - Not required

Programming Language - Python

Framework - Scrapy

Tool - Zyte

Functions & Libraries used - datetime, scrapy-user-agents

Storage (Database) - Zyte Cloud

Deployment Requirements

- Install all the required libraries in Zyte Cloud

Logging considerations

- No logging is required
- No CAPCTHA authentication required

Proxy Details

- We are using user agent to avoid getting blocked, this is present in settings.py file.

5. Base Collector Code

File name - records.py

Here we are scraping the data as per the requirements

Step 1 - Importing required libraries

```
import scrapy
import datetime

from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor
from ..items import ProsettingsItem
```

Step 2 - Here a spider named "RecordsSpider" is created & allowed domain and start url of the website are defined that we are crawling

```
class RecordsSpider(CrawlSpider):
    name = 'records'
    start_urls = ['https://prosettings.net/cs-go-pro-settings-gear-list/']
```

Step 3 - Here we are defining the mandatory data

```
    # AEID_project_id = ''
    site = 'prosettings.net'
    source_country = 'Global'
    context_identifier = ''
    record_created_by = ""
    execution_id = ""  # This will be taken automatically from zyte
    feed_code = "AEID-4666"
    type = ""
```

Step 4 - Here we are defining the custom settings needed for Crawling

```
    custom_settings = {
        'ROBOTSTXT_OBEY': False,
        'CONCURRENT_REQUESTS': 20,
        'COOKIES_ENABLED': False,
        'COOKIES_DEBUG': False,
        'CONCURRENT_REQUESTS_PER_DOMAIN': 500,
        'DOWNLOAD_DELAY': 0,
        'AUTOTHROTTLE_ENABLED': False,
        'DOWNLOAD_TIMEOUT': 20,
```

```
        'DUPEFILTER_DEBUG': True,
    }
```

Step 5 - Rule method to fetch url for different game listed in website

```
    rules = (
        Rule(LinkExtractor(
            restrict_css='li.menu-item.menu-item-type-custom.menu-item-object-custom.current-menu-ancestor.current-menu-parent.menu-item-has-children.menu-item-241 a', ),
             callback='parse_games'),
    )
```

Step 6 - Here we are defining parse function. Inside this function we are writing code for crawling the data to process data for each game

```
 def parse_games(self, response):

        item = ProsettingsItem()  # object to store data in "items.py"
        # list to take index of the data in the table
        list = [[2,4,13,15,19,20,21], [3,4,10,12,14,15,16], [2,4,11,13,16,17,18], [2,3,15,17,20,21,22], [2,4,13,15,18,19,20], [3,8,12,'a',9,10,11], [2,6,10,'a',7,8,9], [2,3,9,11,13,14,15]]
        # loop to identify table number which carries data within website
        for i in range(50, 70):
```

Step 7 - conditional logic to fetch player name, mouse, monitor, mouse_pad, keyboard, headset, GPU and their urls

```
    if '</a></td>' in response.css(row_no_n).extract()[listed[0]]:
                            str1, str2 = '"_blank">', '</a></td>'
                            idx1, idx2 = response.css(row_no_n).extract()[listed[0]].index(str1), response.css(row_no_n).extract()[listed[0]].index(str2)
                            item["Player_Name"] = response.css(row_no_n).extract()[listed[0]][idx1 + len(str1): idx2]
                            idx1, idx2 = response.css(row_no_n).extract()[listed[0]].index(str3), response.css(row_no_n).extract()[listed[0]].index(str4)
                            item["Player_Link"] = response.css(row_no_n).extract()[listed[0]][idx1 + len(str3): idx2]
```

Step 15 - yielding all items here

```
            yield item
```

6. Template Parameters & Description

The template contains the data that is scraped as per the ranking of newly listed products.

For the parameters where **mandatory** is mentioned, this is mandatory parameters as per the required template.

For the parameters where **Required** is mentioned, this is parameters needed as per the requirement document.

Below are the parameters that we are scraping and their description

1. **key -** Zyte by default add this as an identifier.
2. **Context_identifier (Mandatory) -** Currently we don't have any breadcrumbs for the website so hardcoded this.
3. **Execution_id (Mandatory) -** Execution id will be taken automatically from zyte.
4. **Feed_code (Mandatory) -** This is hardcoded as project name.
5. **GPU_Link (Required) -** This we are getting from website.
6. **GPU_Name (Required) -** This we are getting from website.
7. **GPU_Link (Required) -** This we are getting from website.
8. **Headset_Link (Required) -** This we are getting from website.
9. **Headset_Name (Required) -** This we are getting from website.
10. **Keyboard_Link (Required) -** This we are getting from website.
11. **Keyboard_Name (Required) -** This we are getting from website.
12. **Player_Name (Required) -** This we are getting from website.
13. **Player_Link (Required) -** This we are getting from website.
14. **Monitor_Link (Required) -** This we are getting from website.
15. **Monitor_Name (Required) -** This we are getting from website.
16. **Mouse_Link (Required) -** This we are getting from website.
17. **Mouse_Name (Required) -** This we are getting from website.
18. **Mousepad_Link (Required) -** This we are getting from website.
19. **Mousepad_Name (Required) -** This we are getting from website.
20. **Record_create_by (Mandatory) -** This is hardcoded with spider name
21. **Record_create_dt (Mandatory) -** This is the timestamp for capturing the data.
22. **Site (Mandatory)-** This is hardcoded.
23. **Source_country (Mandatory) -**This is hardcoded as per the specific country.
24. **Type (Mandatory) -** This is hardcoded.

7. Risks and Dependencies

Below are the identified risks and their possible solutions:

Risk	Mitigation
Risk of getting blacklisted/blocked/IP restrictions due to security/network policies on the web server.	we need to control the concurrency & use different proxy methods.
If the semantic code/markup of the website changes, the script will have a possibility of failure.	Identify the changes in the semantic code/markup of the website and modify the script accordingly.