

Manual Test Plan for sp21-CS242-assignment 2.2

Table of Contents

- **Environment Setup**
- **Test - Scraping/Database Handling**
- **Test - API Requests**
- **Test - UI**

Environment Setup

- **Python 3.0**
- **unittest**
- **Javascript**
- **HTML**
- **D3 and svg**
- **Eclipse - 4.20.0**
- **Windows 10**

Test : Scraping/Database Handling

Test 1.1 Basic Program Run

```
1 from scraper import Scraper
2 import time
3 from database import get_key
4 from database import database_handler
5
6 """
7 Implements the scraping of data of all the champions from op.gg
8 each champions data is based off their current meta role
9 e.g aatrox meta role is top lane so we scrape data based of his top lane stats
10 """
11 def main():
12     s = Scraper()
13     arr = s.scrape_champion_links()
14     c_counter = 0
15     N = len(arr)
16     while(c_counter < N):
17         retArr = s.scrape_champion_page(arr[c_counter])
18         if retArr is None:
19             continue
20         c_counter+=1;
21         time.sleep(10)
22         database_handler(retArr)
23
24 if __name__ == "__main__":
25     main()
```

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with a '+ Create Database' button and a search bar for 'NAMESPACES'. Below that, a tree view shows 'Collection' expanded, with 'Champions' selected. The main panel displays the 'Collection.Champions' details, including 'COLLECTION SIZE: 35.55KB', 'TOTAL DOCUMENTS: 158', and 'INDEXES TOTAL SIZE: 36KB'. There are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', 'Aggregation', and 'Search Indexes'. The 'Find' tab is active, showing a filter bar with '(field: 'value')' and buttons for 'FILTER', 'OPTIONS', 'Apply', and 'INSERT DOCU'. Below the filter bar, it says 'QUERY RESULTS 1-20 OF MANY'. Two document snippets are shown:

```
{
  "_id": ObjectId("617b2a410541a6022d2c7a58"),
  "name": "Aatrox",
  "win_rate": "49.37%",
  "pick_rate": "9.73%",
  "champ_tier": "Tier 2",
  "counter_champs": Array,
  "strong_against": Array
}
```

```
{
  "_id": ObjectId("617b2a9a0541a6022d2cc098"),
  "name": "Ahri",
  "win_rate": "51.31%",
  "pick_rate": "2.94%",
  "champ_tier": "Tier 4",
  "counter_champs": Array,
  "strong_against": Array
}
```

@Test 1.1

- Users can run the python module 'scraper_exe' to execute the Scraper class functionality and push all champions listed on the website op.gg to the MongoDB database.

Test - API Requests

Test 1.2 - Get Request

http://127.0.0.1:5000/champion?name=Aatrox

GET http://127.0.0.1:5000/champion?name=Aatrox Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Co

Query Params

	KEY	VALUE	DESCRIPTION	...	Bi
<input checked="" type="checkbox"/>	name	Aatrox			
	Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 200 OK Time: 4.68 s Size: 294 B Save Respo

Pretty Raw Preview Visualize JSON

```
1 {
2   "counter_champs": [
3     "Zac",
4     "Malphite",
5     "Kled"
6   ],
7   "name": "Aatrox",
8   "pick_rate": "9.73%",
9   "strong_against": [
10    "Ryze",
11    "Sylas",
12    "Renekton"
13  ],
14   "win_rate": "49.37%"
15 }
16
17 }
```

@Test 1.2

- Users can use the GET API Request by first entering the command 'set FLASK_APP=src/api' and then flask run to start the local server.
- Specify the Request to be GET and then enter a champion's name to start the request. Upon a valid name being given, the json object should contain all the fields that are in the MongoDB database.

Test 1.3 - Put Request

The screenshot shows a REST client interface with the following details:

- URL:** `http://127.0.0.1:5000/champion?name=Aatrox`
- Method:** PUT
- Body (JSON):**

```
1 { "name": "Aatrox",
2   "pick_rate": "5.7%",
3   "win_rate": "79%",
4   "champ_tier": "Tier 0",
5   "counter_champs": ["Alistar", "Akali", "Zed"],
6   "strong_against": ["Yasuo", "Yone", "Riven"] }
```
- Response:** Status: 200 OK, Time: 4.58 s, Size: 183 B
- Response Body (Pretty):**

```
1 updated champion entry: Aatrox
```

@Test 1.3

- Users can use the PUT API Request by first entering the command 'set FLASK_APP=src/api' and then flask run to start the local server.
- Specify the Request to be PUT and then enter a champion's name to start the request. Upon a valid name being given along with a valid json object that updates any or all fields, the server will return a message depending on if the request was successful.

Test 1.4 - Post Request

The screenshot displays a REST client interface with the following components:

- URL Bar:** `http://127.0.0.1:5000/champion`
- Method:** `POST`
- Body:** A JSON object representing a champion's stats:

```
1 {
2   "name": "Lex",
3   "pick_rate": "5.7%",
4   "win_rate": "79%",
5   "champ_tier": "Tier 0",
6   "counter_champs": ["Alistar", "Akali", "Zed"],
7   "strong_against": ["Yasuo", "Yone", "Riven"]
8 }
```
- Response:** Status `200 OK`, Time: `4.59 s`, Size: `315 B`. The response body is a JSON object:

```
1 {
2   "result": {
3     "champ_tier": "Tier 0",
4     "counter_champs": [
5       "Alistar",
6       "Akali",
7       "Zed"
8     ],
9     "name": "Lex",
10    "pick_rate": "5.7%",
11    "strong_against": [
12      "Yasuo",
13      "Yone",
14      "Riven"
15    ],
16    "win_rate": "79%"
17  }
18 }
```

@Test 1.4

- Users can use the POST API Request by first entering the command `'set FLASK_APP=src/api'` and then `flask run` to start the local server.
- Specify the Request to be `DELETE` and then enter a champion's name to start the request. Upon a valid name being given, the

Test 1.4 - Delete Request

The screenshot shows a REST client interface with the following details:

- URL:** `http://127.0.0.1:5000/champion?name=Lex`
- Method:** `DELETE`
- Send Button:** A blue button labeled "Send" is visible.
- Params Tab:** The "Params" tab is active, showing a table of query parameters.
- Query Params Table:**

	KEY	VALUE	DESCRIPTION		Bulk Edit
<input checked="" type="checkbox"/>	name	Lex			
	Key	Value	Description		
- Body Tab:** The "Body" tab is active, showing the response message: `1 deleted champion with name Lex`.
- Status Bar:** The status bar at the bottom indicates: `Status: 200 OK Time: 4.51 s Size: 183 B`.

@Test 1.4

- Users can use the DELETE API Request by first entering the command 'set FLASK_APP=src/api' and then flask run to start the local server.
- Specify the Request to be DELETE and then enter a champion's name to start the request. Upon a valid name being given, the MongoDB database will delete the specified entry by name and return a message depending on if the request was successful.

Test - UI

Test 1.5 - Home Page

A League of Legends Statistics Website

[Top Champions by Pick Rate](#) | [Top Champions by Win Rate](#) | [CRUD operations](#)

Champion Name:

Get Champ

Query (Ex : champion.win_rate: > 55):

Get Query

Clear Results

Results Below

Results Below

name	pick_rate	win_rate	champ_tier	counter_champs	strong_against
Anivia	3.03%	52.94%	Tier 2	Aurelion Sol,Corki,Diana	Ryze,Lucian,Nunu & Willump
Annie	1.09%	52.98%	Tier 3	Neeko,Singed,Diana	Renekton,Garen,Sett
Aurelion Sol	0.67%	54.41%	Tier 3	Fizz,Malzahar,Katarina	Jayce,Azir,Ryze
Blitzcrank	14.65%	52.49%	Tier 1	Leona,Shaco,Taric	Vex,Jarvan IV,Yuumi
Camille	14.22%	52.52%	Tier 1	Teemo,Vayne,Shen	Ryze,Jayce,Yasuo
Graves	18.07%	52.06%	Tier 1	Poppy,Warwick,Sejuani	Qiyana,Zed,Rengar
Karthus	3.29%	52.46%	Tier 2	Nunu & Willump,Rengar,Graves	Udyr,Olaf,Qiyana
Kled	1.86%	52.94%	Tier 2	Singed,Quinn,Urgot	Pantheon,Viktor,Sylas
Maokai	6.95%	52.52%	Tier 1	Taliyah,Rell,Braum	Jarvan IV,Vex,Pantheon
Nasus	1.96%	52.04%	Tier 3	Lillia,Garen,Camille	Pantheon,Zac,Viktor
Nunu & Willump	3.29%	52.13%	Tier 2	Vi,Zac,Warwick	Udyr,Qiyana,Rumble
Quinn	1.14%	52.66%	Tier 2	Maokai,Sion,Singed	Jayce,Trundle,Kled
Rek'Sai	3.8%	52.12%	Tier 2	Shyvana,Karthus,Zac	Ivern,Rengar,Qiyana
Rell	2.02%	52.34%	Tier 2	Veigar,Swain,Soraka	Jarvan IV,Sett,Gragas
Rengar	1.59%	52.04%	Tier 3	Maokai,Pantheon,Zac	Ryze,Jayce,Vayne
Shen	3.34%	52.6%	Tier 2	Viktor,Quinn,Vayne	Cassiopeia,Gragas,Pantheon
Tahm Kench	4.9%	52.81%	Tier 1	Cho'Gath,Kled,Sion	Ryze,Jayce,Gragas
Taric	0.79%	52.66%	Tier 4	Brand,Shaco,Braum	Seraphine,Bard,Karma
Twitch	1.5%	52.05%	Tier 4	Swain,Yasuo,Xerath	Kog'Maw,Aphelios,Sivir
Vayne	8.54%	52.11%	Tier 2	Twitch,Swain,Kog'Maw	Yasuo,Ezreal,Aphelios
Ziggs	5.3%	52.55%	Tier 2	Twitch,Xerath,Sivir	Aphelios,Lucian,Kog'Maw
Zyra	2.03%	52.94%	Tier 3	Shen,Shaco,Blitzcrank	Anivia,Seraphine,Zac

@Test 1.5

- Home page with default route- User can enter champion by Name or enter a specific query to return a list of champions. The output is in a readable table format. Users can also clear any previous entries with the Clear Results button if needed.

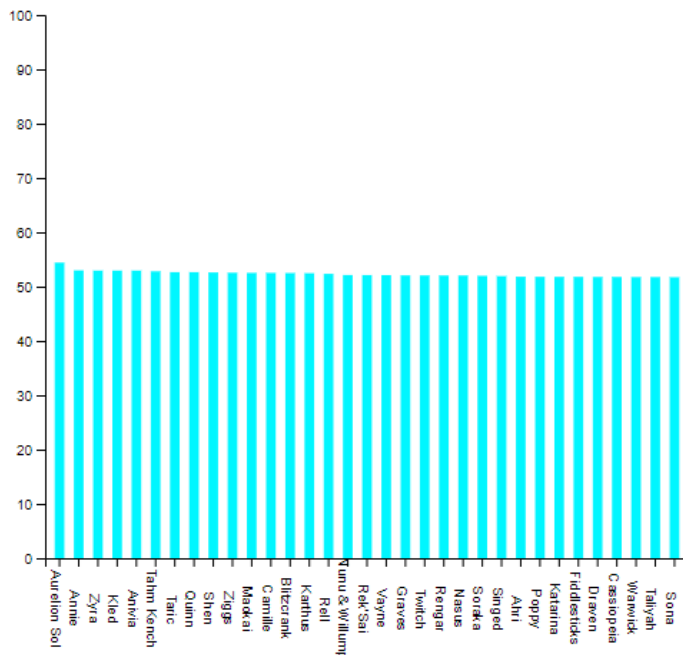
Test 1.6 - Visualization Pages

Top k Champions by Win Rate

[homepage](#)

Pick k below

k:



@Test 1.6

- Visualization pages prompt the user to enter a number k (less than 150) to generate a graph of k champions by win rate or pick rate)

Test 1.7 - CRUD Handler

[homepage](#)

CRUD

Champion Name:

Win Rate:

Pick Rate:

Champion Tier:

Counter Champs(e.g Maokai, Yasuo, Zed):

Strong Against Champs(e.g Yone, Yasuo, Malphite):

UPDATE

CREATE

DELETE

SCRAPE

Results

@Test 17

- Users can specify fields to update, create, or delete. After processing the request, the results will notify whether the request was successful or not.