Netflix API

Here we have Neflix API which can be used to search Movies or TV Shows from Netflix dataset. This uses REST API and ElasticSearch in the background.

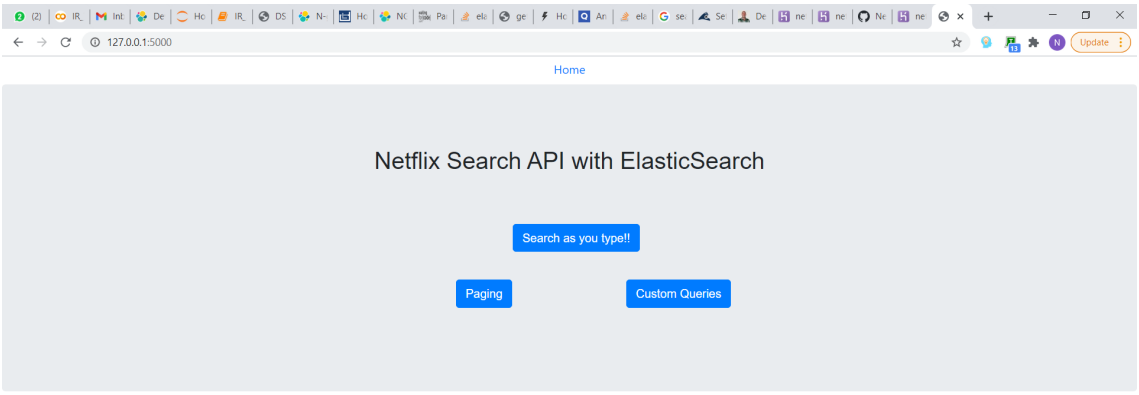

Figure 1: Home Page

The search engine has different functionalities with different End-points as given:

i. **Auto Completion Endpoints**: The data from the *input* field is piped (using */pipe* route) to port 4000 running the API along with the status of *checkbox* status. Based on these inputs query is made to **elasticsearch**.

   (a) *Endpoint for Adults*: This mode does not prevent any R, NC or PG rated movies/TV Shows from being suggested. The **GET** query for elasticsearch is:

```
{
"_source": [],
"size": 0,
"min_score": 0.5,
"query": {
    "bool": {
"must": [
{
"match_phrase_prefix": {
"title": {
"query": "{}".format(<input>)
```

```
}
}
}
],
"filter": [],
"should": [],
"must_not": []
}
},
"aggs": {
"auto_complete": {
"terms": {
"field": "title.keyword",
"order": {
"_count": "desc"
},
"size": 10
}
}
}
}
```

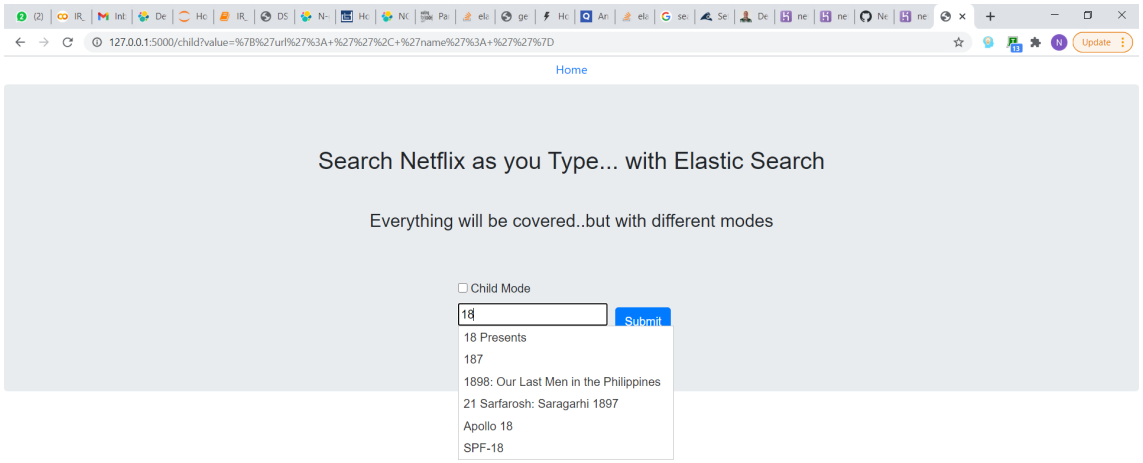

Figure 2: Adult queries too!

(b) *Child Proof Endpoint*: This mode prevents any R, NC or PG rated movies/TV Shows from being suggested. The **GET** query for elasticsearch is:

```
{
"_source": [],
"size": 0,
"min_score": 0.5,
"query": {
"bool": {
"must": [
{
"match_phrase_prefix": {
"title": {
"query": "{}".format(self.query)
}
}
}
],
"filter": [],
"should": [],
"must_not": [
{
"bool": {
"should": [
{"match": {
"rating": "R"
}},
{"match": {
"rating": "NC"
}},
{"match": {
"rating": "PG"
}}
]
}
}
]
}
},
"aggs": {
"auto_complete": {
"terms": {
"field": "title.keyword",
"order": {
"_count": "desc"
},
```

```
"size": 10
}
}
}
}
```

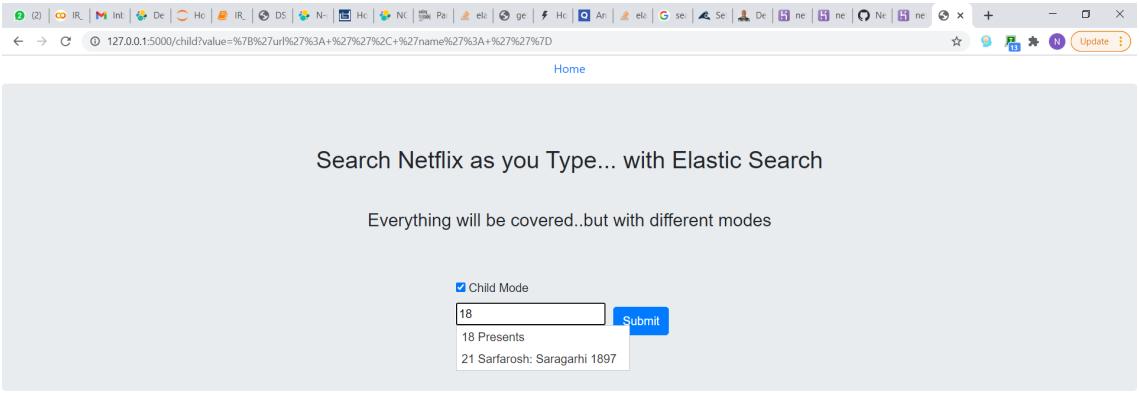

Figure 3: Child proof

From both Fig. 3 and 2, we can understand the queries which are filtered.

ii. **Pagination (Sorted by release_year) Endpoint**: This Endpoint builds the queries based on the number of page assigned to them for a given page size and the type of video it is (either *Movie* or *TV show*). The query used for this is

```
{
"sort": [
{"release_year": {"order": "desc"}},
"_score"
],
"size": 10000,
"query": {
"match": {
"type": typeP
}
}
}
```

where *typeP* denotes the type of video being queried.

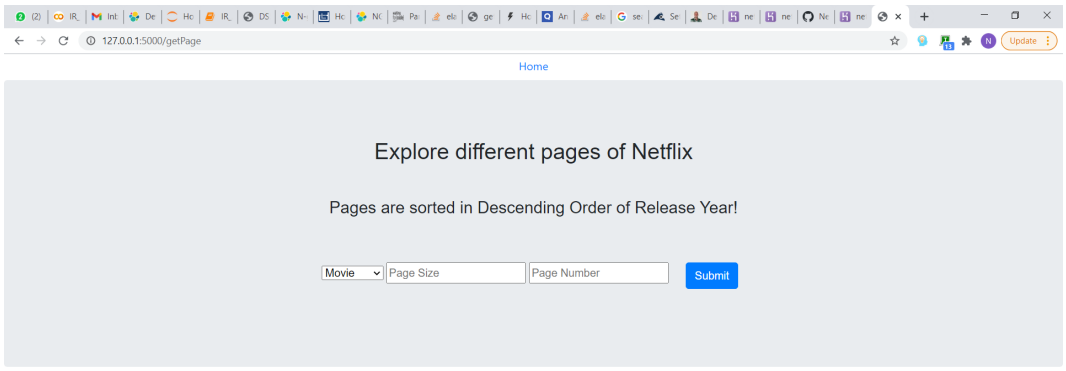

Figure 4: Pagination before

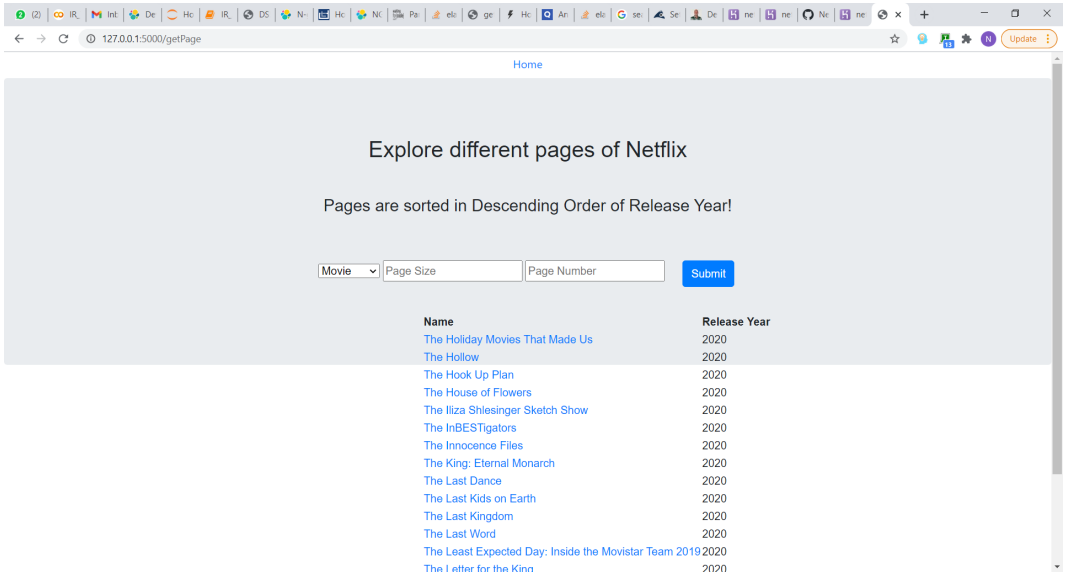

Figure 5: After a query of Movie and Page size:20 and Page number:20

iii. **Custom Queries**: This endpoint is for customized queries.

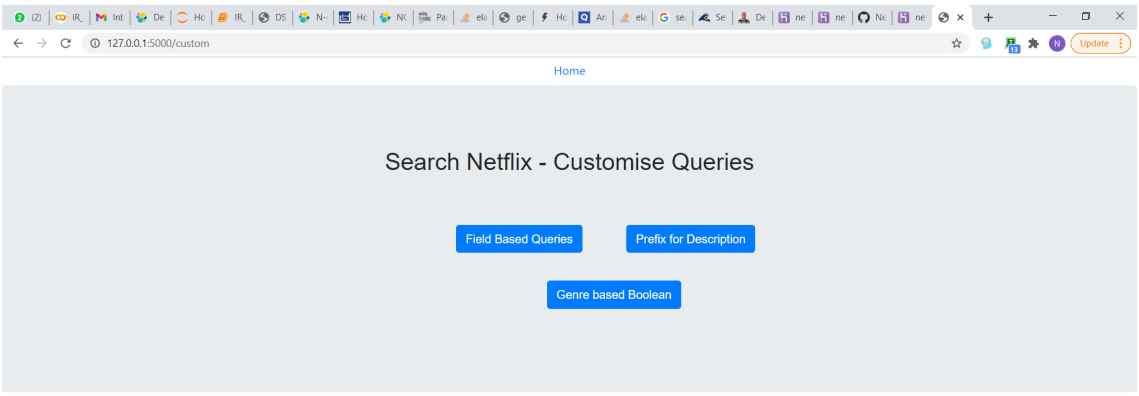

Figure 6: Homepage of Custom Queries

(a) *Exact Match Endpoint*: Here we can match any field with an exact match. The query used for this is

```
{
"size": 10000,
"query": {
"match": {
option: {
"query": value,
"operator": "and"
}
}
}
}
```

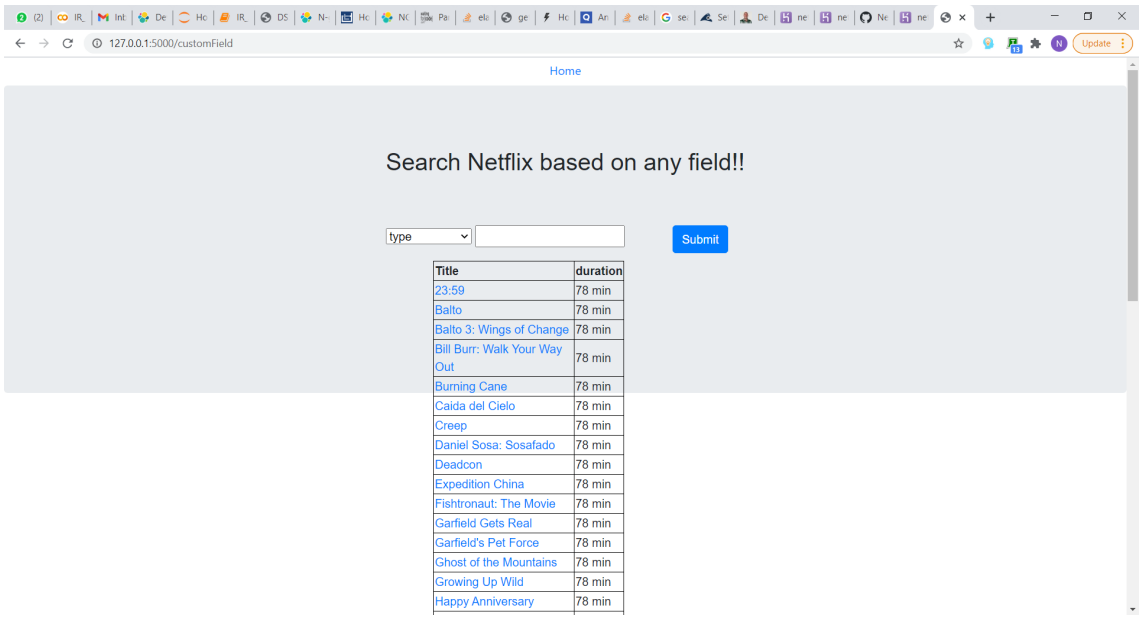

Figure 7: Query for Duration:78 min

(b) *Prefix Match Endpoint*: Here the prefix of a word in **Description** of a video is matched and results are returned with the request to elasticsearch

```
{
"size": 10000,
"query": {
"match_phrase_prefix": {
"description": {
"query": descr
}
}
}
}
```

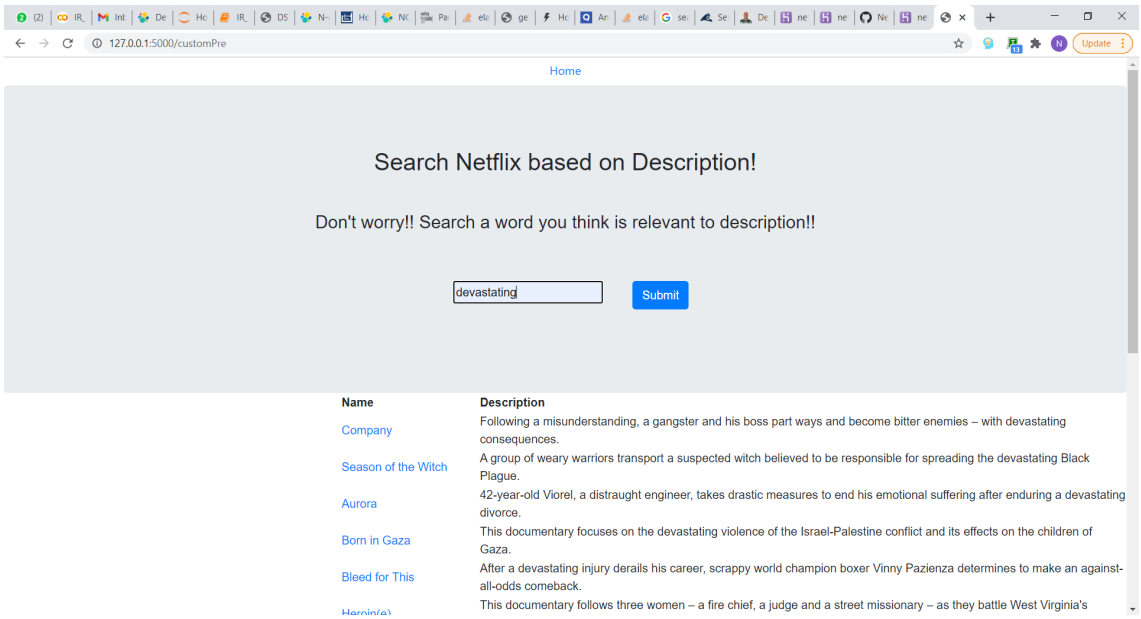

Figure 8: Query based on Description: devastating

(c) *Genre Match Endpoint*: This is the most interesting part of all the endpoints and the requests here are built dynamically with the provided input using `luqum` module. The given *boolean query* containing AND's and OR's is made into a tree using parser from `luqum` module. Then the request is generated as

```
{
"size":10000,
"query": boolQuery
}
```

where `boolQuery` is the parsed output of a `luqum` function. For Ex: the query **(children and adventure) or comedies** generates the request:

```
{"size":10000,
"query":
{"bool":
{"should":
[{"bool":
{"must":
[{"match_phrase_prefix": {"listed_in": {"query": "children "}}},
{"match_phrase_prefix": {"listed_in": {"query": "adventure "}}}
]}},
{"match_phrase_prefix": {"listed_in": {"query": "comedies "}}}]}}}
```

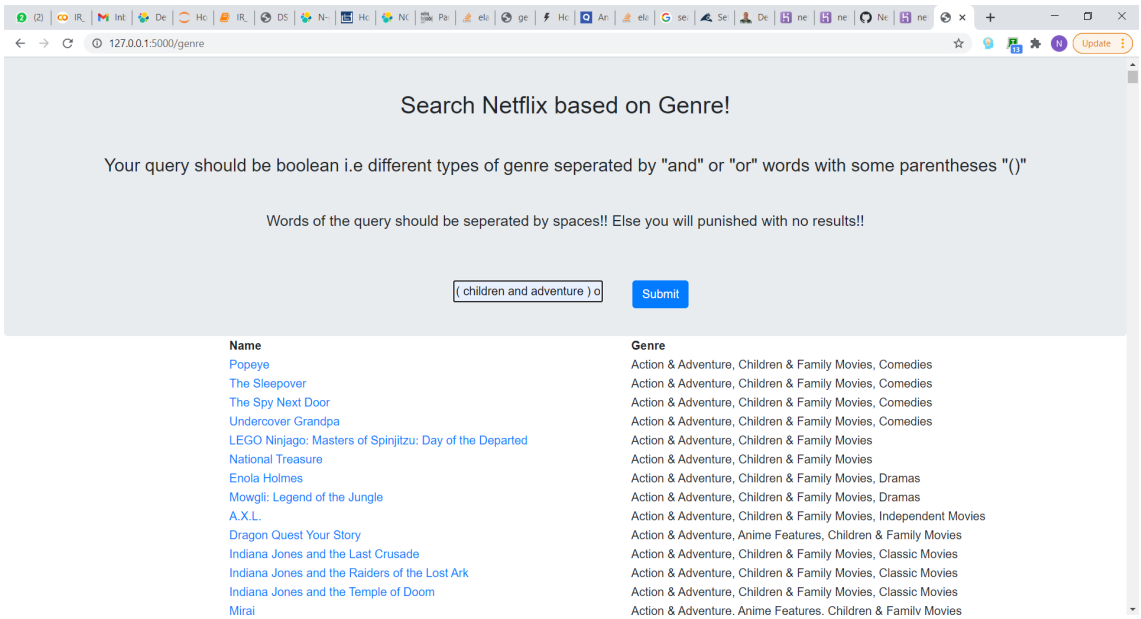

Figure 9: Query for genre:(children and adventure) or comedies