So in my mind, the workflow was going to be a simple matter of:

1. Search for the film using a URL like above
2. Scrape the search results page for a link to the film
3. Scrape the film page for the rating and synopsis

Unfortunately, the Rotten Tomatoes web site is set up, deliberately or otherwise, so that you cannot do this programmatically.
While indeed you can try to make such a search query from R, for example:

```
rt_lookup <- function(film) {
  query <- URLencode(film)
  url <- sprintf('https://www.rottentomatoes.com/search?search=%s', query)
  httr::GET(url)
}
request <- rt_lookup('The Good, the Bad and the Ugly')
```

the resulting `request` does not contain any film results, as on the web site they seem to be generated by JavaScript code that only runs if you are accessing the page in a real web browser (i.e. not a robot).
There is probably a way around this, by simulating a browser session using **RSelenium** or similar, but it seemed too complicated, for me.

Instead, I looked for other sources of film ratings.
My first thought was the web site Metacritic, but their web site is even more robot-unfriendly than Rotten Tomatoes.
I tried to query it with very similar code to before:

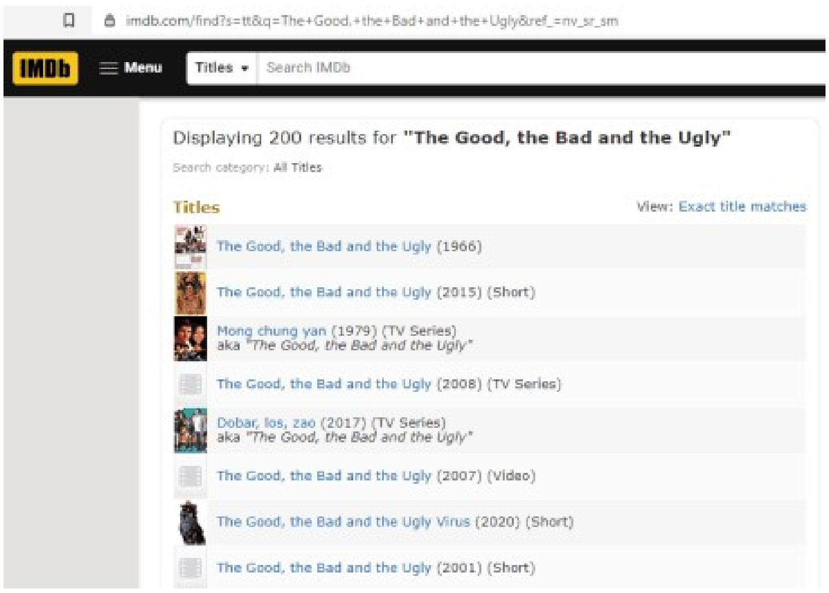
```
mc_lookup <- function(film) {
  string <- URLencode(film)
  url <- sprintf('https://www.metacritic.com/search/all/%s/results', string)
  httr::GET(url)
}
request <- mc_lookup('The Good, the Bad and the Ugly')
```

and not only do we not get any useful results, but the server blocks us completely with a 403 (access forbidden) error.
It is as if the people who make these web sites would prefer us to access the web site in a browser, presumably so they can serve us ads or other services, and make a living.

Undeterred, I turned to the stalwart of film information, the Internet Movie Database (IMDb).
Maybe it is an older web site or maybe they just don't care about robots, but this one much more accessible.
We can make an IMDb search query using

```
imdb_lookup <- function(film) {
  string <- gsub(' ', '+', film)
  url <- sprintf('https://www.imdb.com/find?s=tt&q=%s&ref_=nv_sr_sm', string)
  httr::GET(url)
}
request <- mc_lookup('The Good, the Bad and the Ugly')
```

which programmatically retrieves the content of the following page.
Notice there are no ratings or summaries displayed on the results page, so we will need to follow the link to each film's page to get that information.


imdb.com/find?s=tt&q=The+Good,+the+Bad+and+the+Ugly&ref_=nv_sr_sm

Displaying 200 results for "**The Good, the Bad and the Ugly**"

Search category: All Titles

Titles — View: Exact title matches

The Good, the Bad and the Ugly (1966)
The Good, the Bad and the Ugly (2015) (Short)
Mong chung yan (1979) (TV Series)
aka *"The Good, the Bad and the Ugly"*
The Good, the Bad and the Ugly (2008) (TV Series)
Dobar, los, zao (2017) (TV Series)
aka *"The Good, the Bad and the Ugly"*
The Good, the Bad and the Ugly (2007) (Video)
The Good, the Bad and the Ugly Virus (2020) (Short)
The Good, the Bad and the Ugly (2001) (Short)

## Scraping results

We are only interested in one film per query, not the 200 that are returned.
A pragmatic decision is just to pick the top result, which usually corresponds to the most popular film with that title.
(If we are after a less popular edition, we can always include the year in the film title, increasing the likelihood of the desired version being the top result.)

On IMDb, view the search results page in your browser's element inspector and you will see it has an HTML structure like this:

The Good, the Bad and the Ugly (1966)

...

Parsing this table is easy with the **rvest** package.
The following function finds and extracts the first hyperlink with class `result_text` in the `findList` table, and requests the corresponding page.

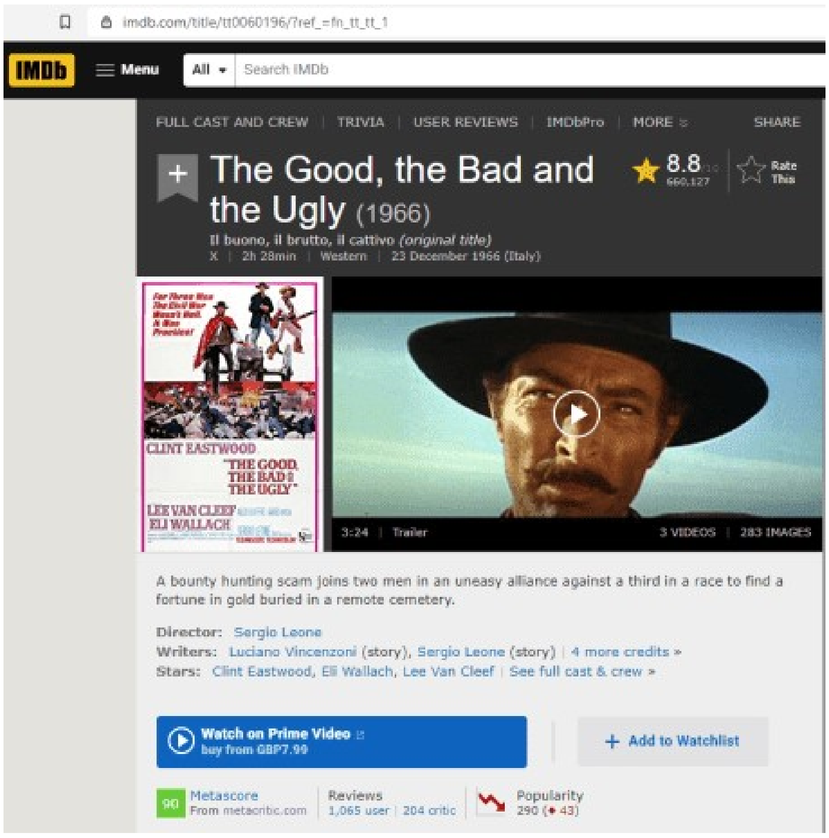
```
library(rvest)
get_first_imdb_result <- function(imdb_results) {
  if (httr::http_error(imdb_results)) return(NA)
  film_path <- imdb_results %>%
    read_html() %>%
    html_node('table.findList .result_text a') %>%
    html_attr('href')
  if (is.na(film_path)) return(NA)
  url <- paste0('https://imdb.com', film_path)
  httr::GET(url)
}
```

If we wanted to extract *all* links in the table, we would use the function `html_nodes`, rather than `html_node`, which just returns the first.

We cannot, in general, assume that earlier our call to `imdb_lookup` will always return results, however, because we could have passed it an invalid search query, or the IMDb server could become wise to our robotic behaviour (which is surely violating that site's terms of use) and block the request (as Metacritic does).
Thus we include return `NA` if the search page was not found.
Also, the page could be found, but contain no results, in which case we also return `NA`.
This is to avoid errors when we process lots of films at a time.

It is something you should always consider when querying data on a server or network outside your control.

The above function requests the IMDb film page linked in the search results.
For example:



The Good, the Bad and the Ugly (1966)

8.8 660,127

Il buono, il brutto, il cattivo *(original title)*
X | 2h 28min | Western | 23 December 1966 (Italy)

A bounty hunting scam joins two men in an uneasy alliance against a third in a race to find a fortune in gold buried in a remote cemetery.

Director: Sergio Leone
Writers: Luciano Vincenzoni (story), Sergio Leone (story) | 4 more credits »
Stars: Clint Eastwood, Eli Wallach, Lee Van Cleef | See full cast & crew »

90 Metascore From metacritic.com | Reviews 1,065 user | 204 critic | Popularity 290 (▾ 43)

Almost there.
Now we just need to scrape the key information from this page:

- Title (which may be different from our search query)
- IMDb rating
- Short description of the film
- Metascore

Interestingly we have ended up obtaining Metacritic scores indirectly by scraping IMDb, even if it is not possible to scrape them from Metacritic's own web site.

Peek at the page with an element inspector to find the HTML elements you need.
For the above four pieces of information, we can select them using

- `.title_wrapper h1`
- `.imdbRating .ratingValue span`
- `.summary_text`
- `.metacriticScore span`

I wrote the following function to extract the same.
The base R function `trimws` is especially useful for getting rid of any nuisance whitespace at the start and end of the page content.

```
extract_imdb_info <- function(request) {
  page <- title <- metascore <- rating <- synopsis <- NA
  if (!is.na(request[[1]]))
    if (!httr::http_error(request))
      page <- xml2::read_html(request)
  if (!is.na(page)) {
    title <- html_node(page, '.title_wrapper h1') %>% html_text %>% trimws
    metascore <- html_node(page, '.metacriticScore span') %>% html_text
    rating <- html_node(page, '.imdbRating .ratingValue span') %>% html_text
    synopsis <- html_node(page, '.summary_text') %>% html_text %>% trimws
  }
  c(title = title,
    metascore = metascore,
    imdb_rating = rating,
    synopsis = synopsis)
}
```

Once again, we can't assume that the previous two functions passed us valid web pages, so `NA` is returned if the `request` is either `NA` or is an HTTP error.
The final information is returned as a named vector.
We now have everything we need to parse a list of many films.

## Putting it all together

This is my final script, running on an example list of films (including one that I would expect to return an error at some point, had I not designed the code to handle such cases).

```
films <- c('The Matrix', 'Citizen Kane', 'Zootropolis',
           'The Good, The Bad And The Ugly', 'Her', 'Inception', 'Bladerunner',
           'How to train your dragon', 'A Film That Does Not Really Exist')

search_queries <- lapply(films, imdb_lookup)
film_pages <- lapply(search_queries, get_first_imdb_result)
film_info <- lapply(film_pages, extract_imdb_info)

results <- tibble::tibble(film = films, info = film_info)
results <- tidyr::unnest_wider(results, info)
```

And the result?
A tidy data frame of films, titles, summaries and ratings.
And any films that could not be found in the movie database simply show `NA`.

```
# A tibble: 9 x 5
  film               title              metascore imdb_rating synopsis

1 The Matrix         The Matrix (1999)  73        8.7         A computer hacker
learns from mysterious rebels ~
2 Citizen Kane       Citizen Kane (194~ 100       8.3         Following the death
of publishing tycoon Charles~
3 Zootropolis        Zootropolis (2016) 78        8.0         In a city of
anthropomorphic animals, a rookie b~
4 The Good, The Ba~ The Good, the Bad~ 90        8.8         A bounty hunting
scam joins two men in an uneasy~
5 Her                Her (2013)         90        8.0         In a near future, a
lonely writer develops an un~
6 Inception          Inception (2010)   74        8.8         A thief who steals
corporate secrets through the~
7 Bladerunner        Blade Runner (198~ 84        8.1         A blade runner must
pursue and terminate four re~
8 How to train you~ How to Train Your~ 75        8.1         A hapless young
Viking who aspires to hunt drago~
9 A Film That Does~ NA                 NA        NA          NA
```

The rating scores are initially character vectors, but you can convert these to numeric.
Export it to a csv file and you can search and sort it to your heart's content.

Hope this helps!