Uso de APIs

Para el trabajo de la sesión se procede a hacer uso de la API de la OMDB, lo que permitirá consultar información de diferentes películas y series, así como consultar su poster.

Con la información obtenida se procede a realizar una visualización sencilla, haciendo uso de los componentes de bootstrap para darle un estilo simple que permita su correcta visualización.

El primer paso es obtener la propia API Key de cada persona, ya que es necesaria para realizar las consultas, para ello se deben dirigir al sitio web OMDb API - The Open Movie Database. Se puede ver un poco de la documentación, ya que en realidad no cuenta con mucha más, y en el menú superior se encuentra el enlace de la API Key, se selecciona la opción gratis y se diligencia el formulario, a continuación, debe llegar al correo indicado en el formulario un mensaje con dos enlaces, el primero una ruta de ejemplo para uso de la API y segundo un enlace de validación y activación de la API, primero se debe dar clic sobre el segundo enlace para posteriormente probar el enlace de ejemplo.

Desde aquí se debe regresar a la documentación para revisar lo que allí indica del procedimiento y otras opciones y filtros que se pueden aplicar sobre las ruta para las peticiones de la API.

La primera tabla nos indica como hacer peticiones individuales y sus filtros y opciones disponibles

By ID or Title

Parameter	Required	Valid Options	Default Value	Description
i	Optional*		<empty>	A valid IMDb ID (e.g. tt1285016)
t	Optional*		<empty>	Movie title to search for.
type	No	movie, series, episode	<empty>	Type of result to return.
y	No		<empty>	Year of release.
plot	No	short, full	short	Return short or full plot.
r	No	json, xml	json	The data type to return.
callback	No		<empty>	JSONP callback name.
v	No		1	API version (reserved for future use).

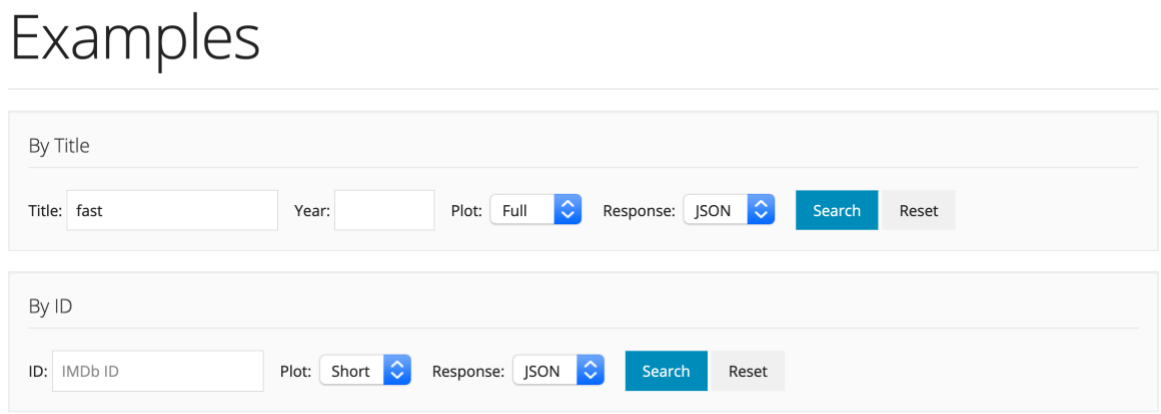
*Please note while both "i" and "t" are optional at least one argument is required.

La segunda tabla permite la búsqueda de más de 1 resultado, si existen.

By Search

Parameter	Required	Valid options	Default Value	Description
s	Yes		<empty>	Movie title to search for.
type	No	movie, series, episode	<empty>	Type of result to return.
y	No		<empty>	Year of release.
r	No	json, xml	json	The data type to return.
page New!	No	1-100	1	Page number to return.
callback	No		<empty>	JSONP callback name.
v	No		1	API version (reserved for future use).

Y en la parte inferior se encuentra una pequeña herramienta de ejemplo para hacer pruebas y conocer la respuesta



Para el proyecto a desarrollar se utilizarán dos archivos muy sencillo un index.html para representar el front, que como se indicó, va acompañado de bootstrap y un app.js, en donde se tendrá la lógica de consulta de la API y la lógica para manipular el DOM y presentar la información.

La primera versión tendrá un código como el siguiente:

```
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">
<head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-
scale=1.0">
      <meta http-equiv="X-UA-Compatible" content="ie=edge">
      <title>OMDB API</title>
      <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.mi
n.css" integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"
```

```
crossorigin="anonymous">
</head>
<body>

      <div class="container pt-5">

      <div class="row">

            <div class="col-8 ml-auto mr-auto">


                  <form>
                  <div class="form-group">
                        <label for="movieSearchTitle">Search
Movie</label>
                        <input minlength=2 type="text" class="form-
control" id="movieSearchTitle" placeholder="Enter the movie title, at
least 2 characters">
                  </div>
                  <div class="form-group w-100">
                              <button id="submit" type="submit"
class="btn btn-info w-100">Search</button>
                  </div>

                  </form>
            </div>
      </div>
      </div>

      <div class="container pt-5">
      <div class="row">
            <div class="col-4 ml-auto mr-auto mb-5">
                  <div id="cardMovie" class="card">

                  </div>
            </div>
      </div>
      </div>

      <script src="./app.js"></script>
      <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
```

```
crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bund
le.min.js" integrity="sha384-
ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx"
crossorigin="anonymous"></script>
</body>
</html>
```

```
// app.js
const getSearchText = () => {

   const submit = document.getElementById("submit");

   submit.addEventListener('click', async (e) => {
       e.preventDefault();

       let promise = new Promise((resolve, reject) => {

           const movieSearchTitle = document.getElementById("movieSearchTitle").value ?
document.getElementById("movieSearchTitle") : false;

           if(movieSearchTitle){

               resolve()
           } else{
               console.log("reject");
               reject();
           }
       });
       promise.then(
           await (testAPI(movieSearchTitle.value)),
           error => console.log(error),
       );

   })

}

getSearchText();

const testAPI = async (movieSearch) => {

   let urlText = `http://www.omdbapi.com/?t=${movieSearch}&type=movie&apikey=23daade9`;

   const resultText = await fetch(urlText, {
       method: 'GET',
   });

   const dataText = await resultText.json();
```

```
    let urlPoster =
`http://img.omdbapi.com/?i=${dataText.imdbID}&type=movie&apikey=23daade9`;
    const resultPoster = await fetch(urlPoster, {
        method: 'GET',
    });
    const dataPoster = await resultPoster.url;

    printMovies(dataText, dataPoster);

    return dataText;
};


const printMovies = async (dataText, dataPoster) => {

    let cardMovie = document.getElementById('cardMovie');

    cardMovie.innerHTML= "";

    const moviePoster = document.createElement('img');
    moviePoster.className = 'card-img-top';
    moviePoster.src = dataPoster;
    moviePoster.alt = dataText.Title;

    const movieUl = document.createElement('ul');
    movieUl.className = "list-group list-group-flush";

    let array = [];

    for(let key in dataText)
    {
        array.push(key + ': ' + dataText[key]);
    //    console.log(key + ': ' + dataText[key]);
    }

    array.map( (value, index) => {
        if([1,2,3,4,5,6,7,8,9,10,11,12,15,16].includes(index))
        {
            let dataLi = document.createElement('li');
            dataLi.className = 'list-group-item';
            dataLi.textContent = value;

            movieUl.appendChild(dataLi);
        }
    });

    const movieTitle = document.createElement('h4');
    movieTitle.className = 'card-title pt-4 pl-2';
    movieTitle.textContent = dataText.Title.toUpperCase();

    const movieLink = document.createElement('a');
    movieLink.className = "btn btn-dark";
```

```
    movieLink.href = `https://www.imdb.com/title/${dataText.imdbID}/`;
    movieLink.textContent = 'IMDB';
    movieLink.target = '_blank';

    cardMovie.appendChild(moviePoster);
    cardMovie.appendChild(movieTitle);
    cardMovie.appendChild(movieLink);
    cardMovie.appendChild(movieUl);
}
```