

Relatório Trabalho de Grupo

TSIW | Tecnologias Web

Realizado por
Yathaarth Handa 40250254
Miguel Silva 40230223

Conteúdo

1.	Introdução	1
2.	Objetivos do Projeto	1
3.	Estrutura do Website	1
4.	Implementação Técnica	2
4.1.	Representação dos Dados com <i>Arrays</i> de Objetos	2
4.2.	Sistema de Filtros e Pesquisa	3
4.3.	Sistema de Criação de Conta de <i>Login</i>	3
4.4.	Catálogo de Filmes	4
4.5.	<i>Toasts</i>	4
5.	Conclusão	5

1. Introdução

Neste relatório vamos falar do desenvolvimento do projeto de grupo realizado no âmbito da unidade curricular de Tecnologias Web.

Este projeto teve como principal objetivo a consolidação dos conhecimentos adquiridos ao longo do semestre, através da criação de um *website* interativo usando apenas HTML, Bootstrap e JavaScript.

Nós optamos pelo tema “Filmes” e então decidimos fazer um *website* onde o utilizador pode criar uma conta, pode fazer *login*, pode ver um catálogo de filmes onde pode também avaliar os mesmos. Por fim também criámos a distinção de utilizadores normais e administradores, sendo dada a habilidade de inserir novos filmes no catálogo ao administrador.

2. Objetivos do Projeto

Com a realização deste projeto tínhamos em mente evoluir as nossas capacidades de Bootstrap: como usar as classes e estilos pré-definidos para o desenvolvimento de um design mais rápido e consistente e também o uso de componentes do Bootstrap como o *Carousel* e o *Toast*. Também tínhamos como objetivo evoluir na programação com JavaScript, nomeadamente a manipulação do DOM através do mesmo e o uso da *localStorage* para guardar dados localmente.

3. Estrutura do Website

O *website* ficou com a seguinte estrutura:
Pasta do projeto

- *img*
 - *backdrops*
 - *posters*
- *js*
 - *account.js*
 - *admin.js*
 - *carousel.js*
 - *login.js*
 - *navbar.js*
 - *session.js*
 - *signup.js*
 - *store.js*
 - *toasts.js*
- *20_movies.json*
- *account.html*
- *index.html*
- *login.html*
- *signup.html*
- *store.html*

A pasta “*img/backdrops*” contém as imagens usadas no *Carousel* da página inicial, onde ocupam a largura e altura toda da página (estilo inspirado no design do site da Netflix). A pasta “*img/posters*” contém as imagens usadas na página de catálogo para os cartões dos filmes.

A pasta “*js/*” contém o código todo utilizado no *website*.

A raiz do projeto contém os ficheiros de HTML das páginas disponíveis a serem acessadas e o *20_movies.json* sobre vários filmes e é utilizado como fonte de dados para o catálogo.

4. Implementação Técnica

4.1. Representação dos Dados com *Arrays de Objetos*

Os dados dos filmes (título, descrição, género, capa de imagem) estão todos definidos no ficheiro *20_movies.json* através de um *array* de objetos.

Abaixo está o exemplo de um dos objetos no ficheiro, sendo informação sobre o filme “*Inception*”.

```
{
  "Series_Title": "Inception",
  "Image": "inception.jpg",
  "Released_Year": 2010,
  "Certificate": "UA",
  "Runtime": "148 min",
  "Genre": "Action, Adventure, Sci-Fi",
  "IMDB_Rating": 8.8,
  "Overview": "A thief who steals corporate secrets through the use of dream-sharing technology is given the inverse task of planting an idea into the mind of a C.E.O.",
  "Meta_score": 74,
  "Director": "Christopher Nolan",
  "Star1": "Leonardo DiCaprio",
  "Star2": "Joseph Gordon-Levitt",
  "Star3": "Elliot Page",
  "Star4": "Ken Watanabe",
```

```

    "No_of_Votes": 2067042,
    "Gross": 292576195
},

```

Este ficheiro é carregado na página de catálogo através da função `fetch()` e após recebermos esta `array` é feito um *loop* pelos objetos da mesma e chamada uma função para criar um elemento HTML (o cartão) com os dados deste objeto.

Nesta página também guardamos uma `array` de objetos que guardam os filmes introduzidos pelos administradores, sendo esta `array` juntada à previamente mencionada antes do *loop*.

4.2. Sistema de Filtros e Pesquisa

Na página de catálogo é possível filtrar os filmes disponíveis por nome e género. Para isto ser possível, decidimos a seguinte implementação:

1. Guardamos o valor dos elementos HTML em variáveis
 - Uma variável guarda a *string* para a pesquisa de nome
 - Outra para a *string* para a pesquisa de género
2. No *loop* pelos filmes todos apagamos os cartões todos (se existirem), verificamos se a *string* do nome está no título do item do *loop* atual e verificamos a mesmo para o género.

Quando estas 2 variáveis são “”(vazias) não fazemos nenhuma filtragem.

```

inputSearch.addEventListener("input", (e) => {
  search = e.target.value;
  updateTableData();
});

select.addEventListener("input", (e) => {
  genre = e.target.value;
  updateTableData();
});

```

4.3. Sistema de Criação de Conta de *Login*

Este sistema foi implementado através de um *script* `session.js` que é um módulo (permite ser acessado por outros *scripts*) e aqui é guardada uma variável chamada `session`. Esta variável é um objeto que nos diz se o utilizador está *logged in* e se sim, mostra-nos também informação sobre o mesmo como o `username`, `email` e `password`.

Como este *script* expõe esta variável e também outras funções para facilitar o processo de fazer *login*, criar conta, fazer *logout* torna-se fazer o conteúdo de outras páginas dinâmicas apenas acessando esta informação.

As funções também usam o `localStorage`, ou seja caso a página seja fechada e aberta de novo a sessão continuará a mesma até o utilizador fazer *logout* ou apagar manualmente os dados no *Browser*.

```

export const session = {
  loggedIn: false,
  user: null,

  export function setUser(user) {
    localStorage.setItem("user", JSON.stringify(user));
    session.user = user;
}

```

```

        localStorage.setItem("loggedIn", true);
        session.loggedIn = true;
    }
};

```

4.4. Catálogo de Filmes

Como foi mencionado previamente, o catálogo de filmes é populado através de um ficheiro `.json` como cartões de Bootstrap onde se mostra cada cartão possui a imagem de capa do filme que tapa o cartão inteiro, na parte de cima do cartão possui os géneros do filme e em baixo um botão que abre o modal de informação.

```

function UpdateCardContainerInnerHTML(data) {
    cardContainer.innerHTML = data
        .map((movie) =>
            getCardFromMovieData(
                "./img/posters/" + movie.Image,
                movie.Series_Title,
                movie.Overview,
                movie.Genre,
            ),
        )
        .join(" ");
}

```

O modal de informação é um elemento único da página mas que quando é aberto o mesmo é populado com os dados do filme selecionado. Possui toda a informação sobre o filme, como o título, capa, descrição, géneros e também possui uma secção onde o utilizador pode introduzir e publicar comentários com avaliação sobre este filme selecionado.

```

function setModalData(image, title, overview, genre) {
    modalImage.src = image;
    modalTitle.innerHTML = title;
    modalDesc.innerHTML = overview;
    modalGenre.innerHTML = genre;
}

```

Uma dificuldade que encontrámos foi como separar os comentários por cada filme, mas conseguímos realizar isto com uma variável chamada `comments` que é um objeto que tem mais objetos dentro: os objetos diretamente dentro desta variável são a `string` do título do filme, e dentro deste objeto temos uma `array` dos comentários, sendo então possível acessar os comentários de um filme específico como `comments[NomeDoFilme]`.

4.5. Toasts

Também implementamos um sistema de `toasts` (notificações) com o uso do componente disponibilizado pelo bootstrap.

Essencialmente expõe uma função chamada `addToast` que primeiro verifica se existe um `container` para os toasts (é necessário para que eles apareçam verticalmente numa `stack`), se não houver adiciona este elemento. Após isto pegamos numa template (`string`) de um toast e adicionamos como filho deste `container` e também chamamos a função do Bootstrap para iniciar o `toast` e a respetiva animação.

```

toastContainer.insertAdjacentHTML("beforeend", toastTemplate);

const toastEl = document.getElementById(newToastUid);
const toast = bootstrap.Toast.getOrCreateInstance(toastEl, {
  delay: 2500,
  autohide: true,
});

toastEl.addEventListener("hidden.bs.toast", () => toastEl.remove());
toast.show();

```

5. Conclusão

Para concluir, este projeto permitiu-nos aplicar os conhecimentos adquiridos durante as aulas e sentimos que o *website* cumpre todos os requisitos definidos na proposta.

Para além dos conhecimentos técnicos também conseguimos desenvolver as nossas competências de trabalho em grupo e organização de tarefas.

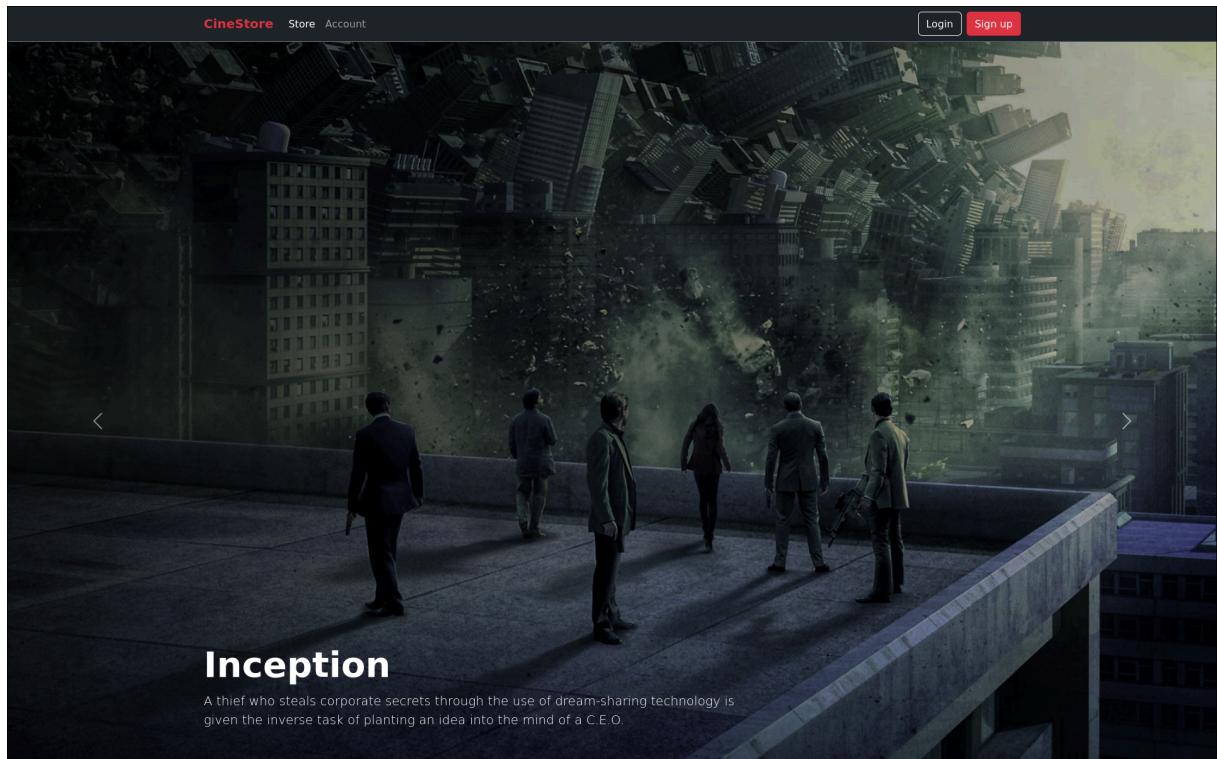


Figura 1: Página inicial do *website* final