

Projeto realizado em HTML, CSS e Javascript

**Projetos: Museu e Formulário**

**Museu:**

**Uma imagem com texto

Descrição gerada automaticamente**

**Formulário:**



Índice:

[1. Introdução e objetivos 3](#_Toc126354110)

[2. Desenho e estrutura 5](#_Toc126354111)

[3. Implementação do Projeto Museu Informático 8](#_Toc126354112)

[4. Implementação do Projeto Formulário 12](#_Toc126354113)

[5. Conclusão 19](#_Toc126354114)

# Introdução e objetivos

Este trabalho foi realizado num contexto académico, da UFCD 5414 - Programação para a Web Cliente(cliente-side), e visa a criação de dois projetos.

O primeiro projeto, trata-se de um website fictício de um Museu da História da Informática, realizado unicamente com HTML e CSS. Para a realização do mesmo foi entregue uma imagem e um vídeo do que era pretendido alcançar.

De modo a alcançar o solicitado, os formandos Alexandre Janeiro e Anaïs Gilbert definiram os principais objetivos a serem conseguidos. Nomeadamente:

1. Criação de um design do que iria ser implementado.
2. Não é permitido usar “flex” nem “grid”, deve utilizar-se apenas “float” no css.
3. Não deve ser utilizado javacript.
4. Criação da estrutura HTML o mais simples possível.
5. Criação da estrutura CSS, sem a utilização de “media querys” como solicitado.

O segundo projeto, consiste na criação de uma página com um formulário á direita, título, texto e botões á esquerda. Foi ainda solicitado que o formulário fosse todo validado em javascript com critérios dados. Os principais objetivos para a sua implementação foram:

1. Criação do design que refletisse a estrutura a implementar.
2. Tem de ser utilizado flex nos estilos css.
3. Devem ser utilizados media querys nas dimensões das resoluções dadas.
4. Todos os valores passados pelo utilizador devem ser validados em javascript
   1. O campo “Nome completo”, é de preenchimento obrigatório, deverá ter pelo menos duas palavras alfabéticas, com pelo menos 2 letras cada.

Se o nome introduzido já existir na base de dados, o forms será substituído por uma mensagem de que o utilizador já existe.

* 1. O campo email, é de preenchimento obrigatório, deverá seguir o formato “alfanumérico@alfanumérico.alfabético”, a última parcela com pelo menos 2 letras.
  2. O campo telefone deverá ter nove dígitos ou doze dígitos precedidos de “+”.
  3. O campo data deverá seguir o formato “DD/MM/AAAA”, só pode aceitar dígitos, ter em conta o ano bissexto relativamente ao mês de Fevereiro (28 ou 29 dias) e validar também os meses com 30 e 31 dias.
  4. O campo palavra-passe, é um campo obrigatório, deverá ter pelo menos: 1 dígito,1 letra minúscula ,1 letra maiúscula, 1 símbolo (#$%&) e deverá ter no mínimo 6 caracteres e no máximo 10.
  5. O campo de seleção, é de preenchimento obrigatório e deverá ter as opções: “Iniciado”, “Intermédio”, “Avançado”, “Pré-profissional” e “Profissional”, sendo o campo “Iniciado” o valor *default*.
  6. O campo de “li e concordo com a política de privacidade”, é um campo de ativação obrigatória.

1. Quando os valores introduzidos não respeitarem os pré-requisitos de validação, deverá aparecer uma cruz vermelha do lado direito do input.
2. O botão submeter valida os dados e envia-os assincronamente para o servidor, substituindo o forms por uma mensagem de sucesso, de seguida apresenta novamente o forms com os *defaults. Se o utili*
3. O botão “Limpar” limpa todos os campos do formulário.

# Desenho e estrutura

Para a devida análise e implementação do projeto Museu de História da Tecnologia, o plano de desenvolvimento consistiu na criação do design baseado na imagem e vídeo fornecidos, antes de começar a escrever código.

O design será colocado de seguida, mas por questões de melhor visibilidade, encontrar-se-á também em anexo no diretório onde se encontra este documento

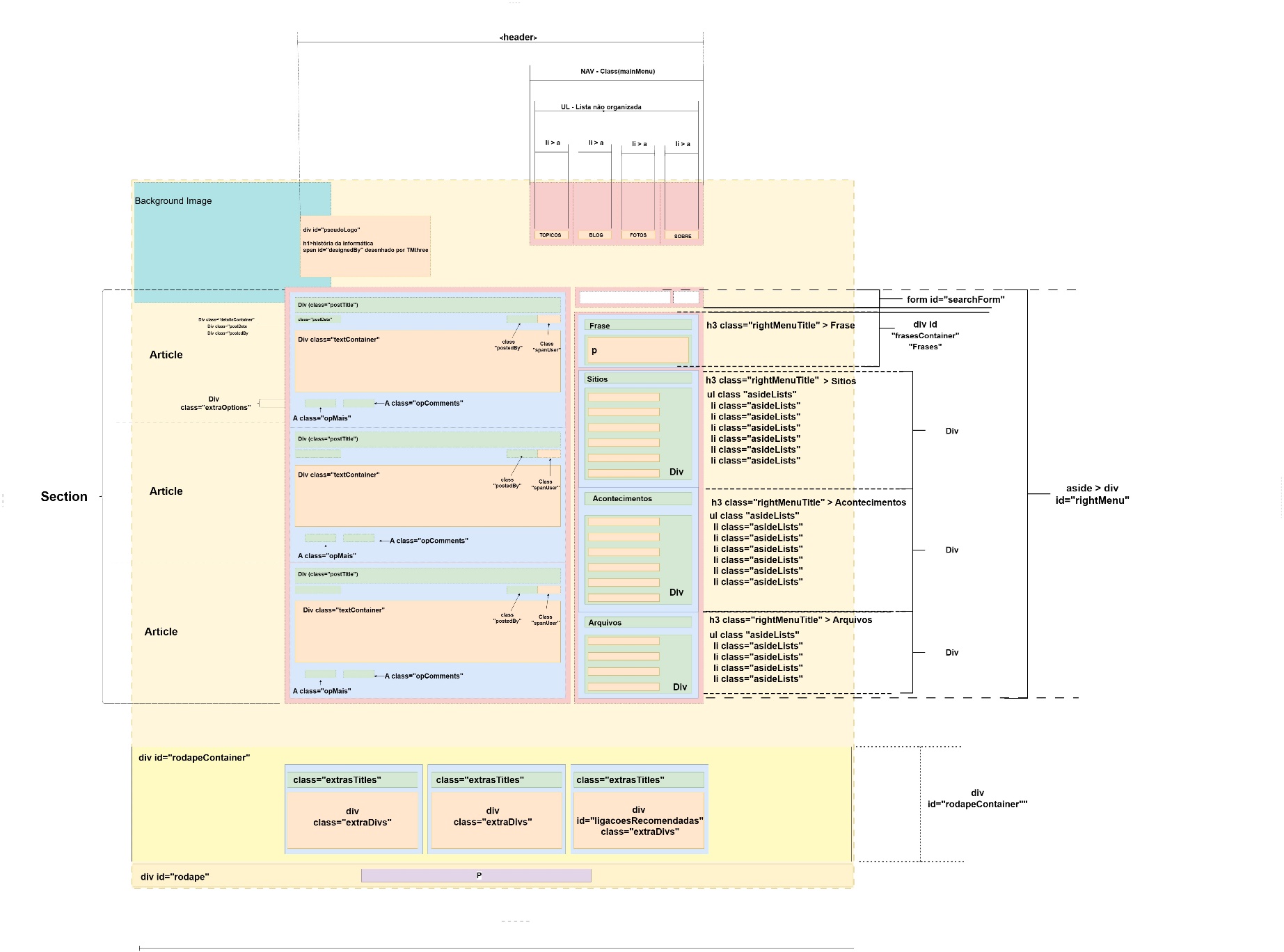


Ilustração - Design do projeto 1 - Museu História da Tecnologia

Para este projeto, o que ressaltou á vista em primeira instância foi a localização do “logo” com texto por trás situado mais à esquerda e um menu de navegação mais à direita. Tendo em consideração que estes dois “blocos” estavam situados no meio da página e os elementos que se seguem estavam alinhados a estes, considerámos que deveríamos incorporar na tag <header> do HTML como demonstrado no design. Por sua vez existiria um “container” (uma div) que iria albergar tanto o conjunto de elementos do lado esquerdo como por sua vez o conjunto de elementos do lado direito. Por fim, estaria o rodapé. Para este, decidimos usar “divs” ao invés da tag <footer> dado ser mais fácil manipular e organizar a estrutura que se pretendia sem a utilização de “flex” no css. Deste modo, pensou-se criar uma “div” que albergaria as 3 “divs” com conteúdo localizado ao centro, e por baixo, outra div a conter o texto relacionado com o copyright e com um fundo diferente.

Para o projeto 2 do formulário, aplicou-se o seguinte design de acordo com os elementos a replicar:

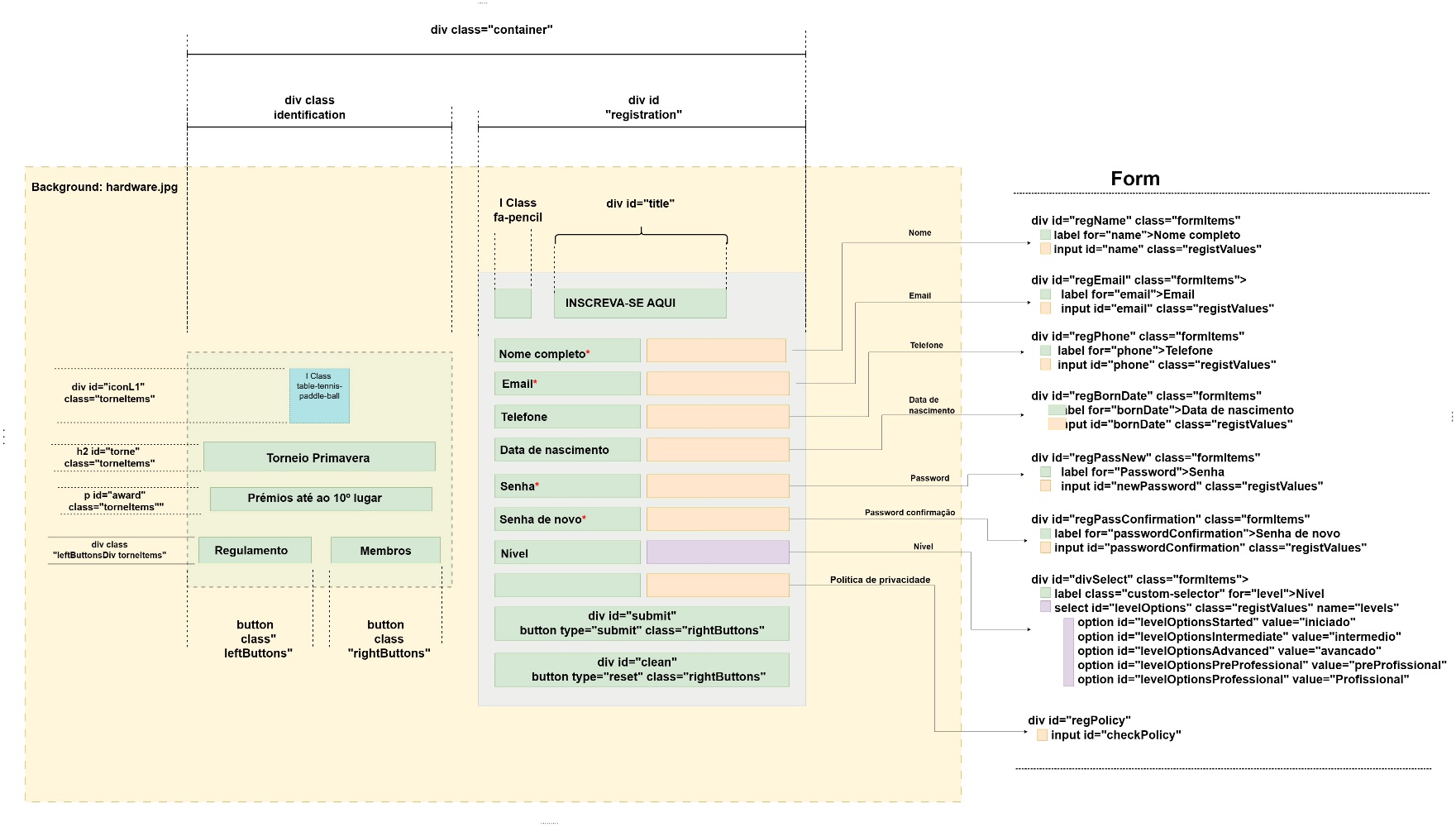


Ilustração - Design do Projeto Formulário

É de salientar novamente que este design também se encontra em anexo para melhor visualização.

Para este segundo projeto, os formandos consideraram importante criar uma “div container” que contivessem as duas divs demonstradas no design: uma div relacionada às informações do torneio, representada à esquerda do design acima e uma div com um formulário. Optou-se por esta escolha, tendo em consideração que deve ser utilizado “flex” neste projeto. Assim sendo, e como a posição da div esquerda muda consoante a resolução do ecrã. Quando o ecrã diminui, a div “informação” é integrada no menu da direita, ficando estas como se fossem uma só.

É de salientar ainda que foi utilizado as etiquetas/”tags” HTML pré-definidas para um formulário com os seus campos. Nomeadamente, as tags: <form>, <label>,<input>,<select> e <options>. Note-se que nestas etiquetas “input” não foi colocado o tipo correspondente, como deveria ser feito como boa prática por ter sido solicitado explicitamente que não se fizesse dado que a validação seria toda realizada manualmente no javascript.

Após pesquisa da documentação, esboço dos passos a serem realizados no projeto, realização e análise dos designs, o projeto poderá ser iniciado.

# Implementação do Projeto Museu Informático

Na implementação do projeto Museu, considerou-se relevante começar por escrever o código referente ao “cabeçalho”. Em relação a esta parte do cabeçalho, achámos pertinente destacar estas tags: <header> e <nav>.

A tag HTML <header> representa o conteúdo introdutório, por norma é composto por um grupo de auxílios introdutórios ou de navegação. Pode conter alguns elementos de título, um logotipo, um formulário de pesquisa, um nome de autor e outros elementos.

O elemento HTML <nav> representa uma seção cuja finalidade é fornecer links de navegação

Uma imagem com texto

Descrição gerada automaticamentePara que a parte superior estivesse centrada, optamos por dar

Ilustração - Tag HTML <header>

A propriedade CSS float, coloca um elemento no lado esquerdo ou direito do seu container, permitindo assim que o texto e os seus elementos o envolvam.

Uma imagem com texto

Descrição gerada automaticamentePara que a parte superior estivesse centrada, optamos por dar as propriedades em baixo descritas.

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamentePara a div com o id “pseudologo”, demos a propriedade float com o atributo left, para que esta div esteja flutuante à esquerda do seu “pai”, que é o header. Por sua vez colocámos aa propriedade float com o atributo right.

Criou-se uma div que servirá como container dos elementos do lado direito e do lado esquerdo, designado por mainContainer. De modo que os floats que pretendemos dar aos elementos que se seguem, não entrassem em conflito com os elementos do header, colocámos clear: both.

Uma imagem com texto

Descrição gerada automaticamente

Segue-se a implementação do lado direito. Aqui são de salientar as tags aside e a criação de uma barra de pesquisa. O elemento <aside> HTML representa uma parte de um documento cujo conteúdo está apenas indiretamente relacionado com o conteúdo principal do documento.

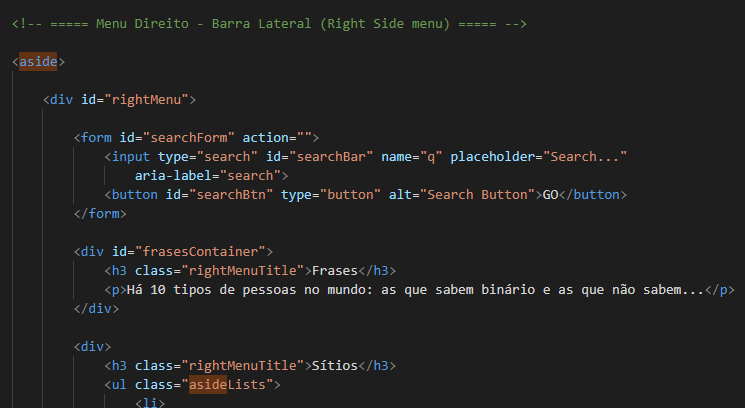


Ilustração - Right side menu <aside>

.

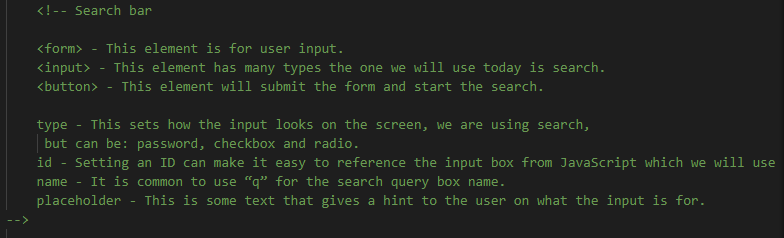


Ilustração - Search bar options

A tag <main> especifica o conteúdo principal de um documento. O conteúdo dentro do elemento <main> deve ser exclusivo do documento. Este não deve conter nenhum conteúdo repetido em documentos, como barras laterais, links de navegação, informações de direitos autorais, logotipos de sites e formulários de pesquisa.

Como foi mencionado anteriormente, o elemento <main> irá conter as tags <section> e <article>, conforme imagem abaixo, onde temos o exemplo de um dos artigos.

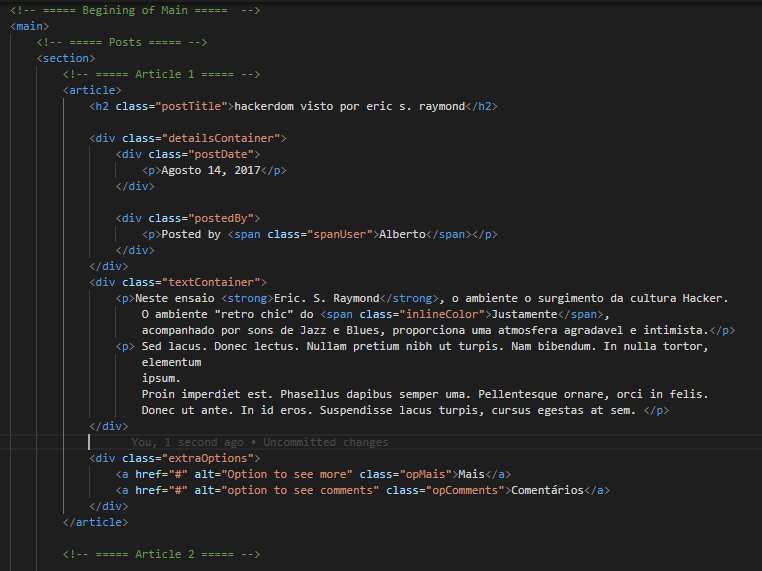


Ilustração - Main, section e um dos articles

Em relação ao footer, conforme imagem em baixo …

Uma imagem com texto

Descrição gerada automaticamente

… os formandos optaram por utilizar div’s ao invés da tag <footer> tendo em consideração a estrutura indicada para o rodapé, com a utilização dos floats.

O Rodapé é composto por uma div container, onde se encontram 3 divs centradas em função da div principal(container). Abaixo da div mencionada encontra-se a div relacionada com o copyright, optamos por uma div separada pois o estilo que lhe teria que ser aplicado é diferente ao da div que o antecede.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Footer, estilos relevantes

# Implementação do Projeto Formulário

Para a implementação do projeto Museu, utilizou-se a biblioteca externa do font-awesome para utilizar os dois ícones solicitados. Considerou-se relevante começar por adicionar uma div container que será constituída pelos dois blocos de código esquerdo (informação sobre o torneio) e direito (formulário), onde foi aplicada a propriedade flexbox do css. É ainda de salientar que se utilizou a propriedade “flex-wrap: wrap” dado que este conjunto permite definir a direção dos seus itens flex descentes, no fundo é como se ”embrulhasse” os seus descendentes para flexibilidade de manipulação.

Uma imagem com texto, placa, captura de ecrã

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

Neste projeto e no que concerne ao HTML, é de relevância salientar as tags <form>, <label> e <input>.

* Form: representa uma seção do documento que contem controlos interativos para envio de informações. No fundo, especifica que estará um formulário na página.
* Label: representa uma legenda para um item na interface do utilizador.
* Input:: é utilizado para a criação de controlos interativos com o objetivo de aceitar os dados introduzidos pelo utilizador. Existem diversos tipos de inputs. Para este projeto foram apenas utilizados os inputs de tipo texto e checkbox.

Os formandos decidiram seguir as boas práticas de acessibilidade e aplicaram “for” dentro de cada label e um id ao input, para que estes permaneçam associados, facilitando a leitura e compreensão de que dados devem ser introduzidos tanto para os utilizadores como para as “tecnologias de assistência”

Adicionalmente, foi utilizado sempre que possível as regras e atributos ARIA – “*Acessible Rich Internet Applications*” de modo a tornar os elementos mais acessíveis para as pessoas com incapacidades.

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Na parte que utiliza JavaScript, começou-se por inicializar as variáveis referentes aos caminhos que identificam os elementos form e de cada input no html.

let form = document.querySelector("form");

let fullName = document.getElementById("name");

let email = document.getElementById("email");

let phone = document.getElementById("phone");

let bornDate = document.getElementById("bornDate");

let newPassword = document.getElementById("newPassword");

let pwdConfirm = document.getElementById("passwordConfirmation");

let submitedMessage = document.getElementById("messagValidation");

Á posterior, criou-se uma escuta de evento para cada input através do método JavaScript “addEventLIstener” de modo que estes estejam “em constante monitorização” aquando um evento ocorra. Esta decisão foi tomada dado que foi solicitado aos formandos que adicionassem uma cruz vermelha, sempre que a validação estiver incorreta, logo é um evento constante e automático. Ao invés de uma escuta passiva ao formulário apenas quando é submetido.

Criou-se funções individuais para validar os respetivos padrões a serem seguidos em cada um dos inputs.

Para as validações, optou-se pela utilização de expressões regulares dado que facilitam a identificação de padrões, a sua substituição e testes.

Usaram-se as expressões em baixo para cada campo a validar:

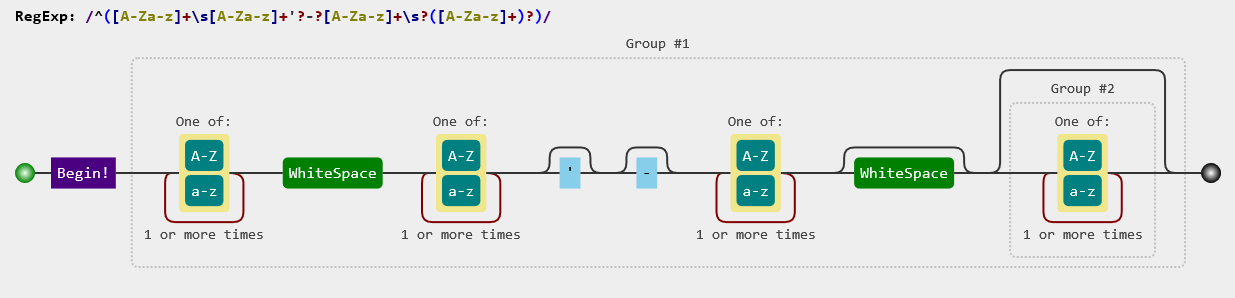


Ilustração – Expressão regular: “Nome completo”

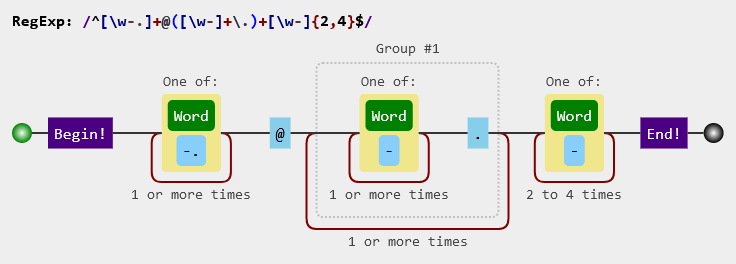
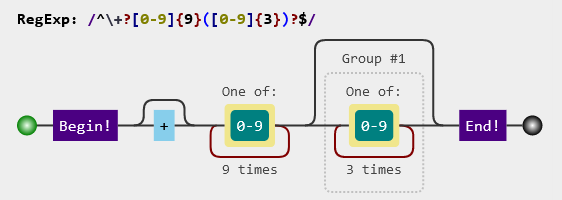
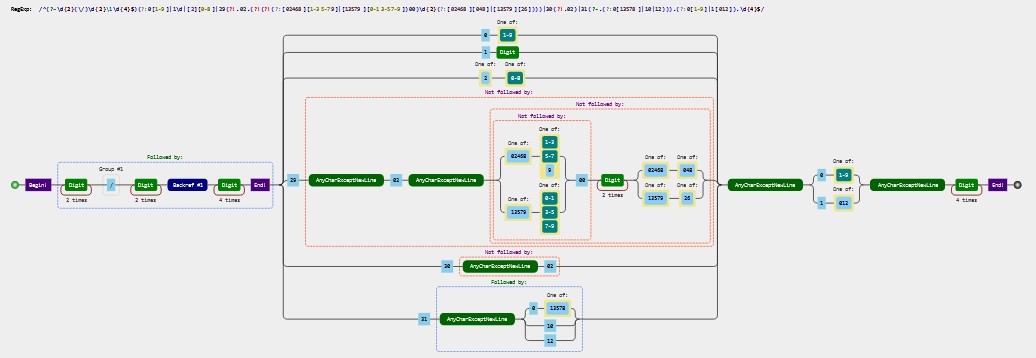


Ilustração 11 - Expressão regular: “Data de Nascimento”

Ilustração - Expressão regular: “Telefone”

Ilustração 9 - Expressão regular: “EMail”

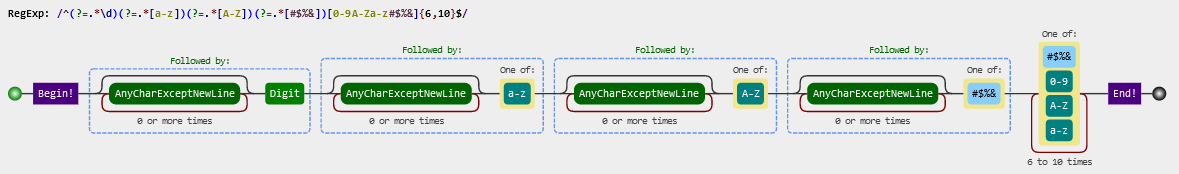
Para a verificação de, se a Confirmação da password é igual à Password, usou-se uma função “validatePwdConfirm()” com uma instrução simples de If.

Ilustração - Expressão regular: “Password”

No que concerne a ativação da aceitação da política de privacidade, colocou-se uma função para que esta apenas seja validada quando a sua “checkbox” esteja ativa.

Para a sua realização utilizou-se uma função como nos demais inputs.

Botão de Submeter:

Criou-se um event listener com uma função, que ao ser submetido o formulário, verifica se a função “master” que contem o conjunto de funções de validação foram bem sucedidas. Caso seja verdade, despoletará a respetiva mensagem de sucesso.

Para este, optou-se por adicionar um event listener ao form, ao invés de se adicionar diretamente ao botão submit. Esta escolha baseou-se na leitura da documentação “MDN Web Docs” que alerta que se um event listener for adicionado exclusivamente ao botão submit, o utilizador poderia enviar na mesma o formulário através de outros meios, no teclado, ferramentas, etc.

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto, interior, ecrã, captura de ecrã

Descrição gerada automaticamente

Botão Apagar:

Para este, criou-se um event listener com uma função para que quando o utilizador clicar no botão *“Clean”*, o método *reset* proveniente do javascript é ativado e o os campos serão todos eliminados.

# Conclusão

Estes projetos permitiram, recorrendo sempre que possível às boas práticas, consolidar as diversas tags HTML, modos de aplicar estilos CSS e explorar a criação de validações manuais através do Javascript.

No que diz respeito ao HTML, cita-se as tags com mais relevância: A tag <main> especifica o conteúdo principal de um documento, este deve ser exclusivo, não deve conter barras laterais, links de navegação, etc.; o form, representa uma seção do documento que contem controlos interativos para envio de informações; a label representa uma legenda para um elemento na interface do utilizador; input:: é utilizado para a criação de controlos interativos com o objetivo de aceitar os dados introduzidos pelo utilizador. Existem diversos tipos de inputs. Para este projeto foram apenas utilizados os inputs de tipo texto e checkbox.

A propriedade float é um recurso importante para posicionar elementos de uma página à direita ou à esquerda de um elemento “pai”. Considera-se uma prática importante de saber quando não é permitido usar o flexbox ou grid.

O FlexBox facilita a criação de uma estrutura de design com um layout responsivo sem utilizar "float" ou posicionamentos.

Estes projetos permitiram aprofundar os conhecimentos de HTML, CSS e JavaScript.

Despertaram o conhecimento dos formandos em aprofundar ainda mais estas matérias.