

# PÀGINA

# CLUB ESCACS

PEC-2

Eines HTML i CSS II

Alumne: Jordi Sánchez Torras

<https://github.com/jorditosa/escacs-pec2>

[gambitodedama.netlify.app](https://gambitodedama.netlify.app)

---

## Contingut

Briefing y template inicial .....	2
Dependències.....	4
CSS Grid.....	6
Components Bootstrap.....	7
Exemples.....	10
Personalització de la llibreria.....	11
Utilització de Sass.....	13
@Mixins.....	13
@Supports.....	13
Built in.....	13
Bibliografia.....	15

# Briefing y template inicial

L'objectiu principal d'aquest projecte és proporcionar als membres del club i als visitants de la pàgina una plataforma en línia on puguin accedir a informació actualitzada sobre l'esport d'escacs i millorar les seves habilitats i estratègies en el joc.

Per a la creació d'aquesta pàgina web, s'ha utilitzat la llibreria de Bootstrap, una eina popular i amplament utilitzada en el disseny de pàgines web. S'ha implementat una interfície fàcil d'utilitzar i navegar, amb una estètica atractiva i moderna.

A més, s'ha creat una pàgina de creació lliure que conté una guia interactiva de jugades d'escacs, amb l'objectiu de proporcionar als usuaris una experiència més entretinguda i informativa.

## Guia d'estils bàsics

Pel web s'ha utilitzat una paleta de **colors** extreta de <https://colorhunt.co/>. Hi ha infinitat de paletes predefinides, majoria d'elles amb 4 colors que ja ens son més que suficients pel projecte que volem crear.

## Design System

Un design system és un conjunt de principis, estàndards, guies i components de disseny que s'utilitzen per crear i mantenir una experiència d'usuari coherent en un producte digital o servei.

Com a elecció pel projecte, s'utilitzarà el patró fet per Estonia.

<https://adele.uxpin.com/estonia-brand-estonia>

I per la **tipografia**, s'ha seleccionat la font Aino<sup>1</sup>. És de pagament, però ja la tenia d'un projecte personal i l'he aprofitat.

---

<sup>1</sup> <https://www.atipofoundry.com/fonts/borna>

# Dependències

En primer lloc prepararem la configuració inicial amb les dependències que controlaran els nostres estils.

→ Primer configurarem la llibreria **Bootstrap**, versió 5, i l'adaptarem per poder treballar-la amb Sass.

```
npm i --save bootstrap @popperjs/core
```

I l'importem al arxiu principal de scss

```
@import "bootstrap/scss/bootstrap";
```

→ Seguidamente prepararem la configuració del linter, **Stylelint**.

```
npm install --save-dev stylelint
```

I configurarem l'arxiu amb les següents regles:

```
.stylelintrc.json > ...
1  {
2    "plugins": [
3      "stylelint-scss"
4    ],
5    "rules": {
6      "at-rule-no-unknown": null,
7      "scss/at-rule-no-unknown": true,
8      "scss/dollar-variable-pattern": "^foo",
9      "scss/selector-no-redundant-nesting-selector": true,
10     "selector-class-pattern": "^[a-z]([a-z0-9-]+)?(__([a-z0-9-]+)?)(--[a-z0-9-]+)?{0,2}$"
11   }
12 }
```

Comentaris:

→ **MD. Markdown** és un llenguatge molt conegut que actualment s'utiliza molt per a documentar i mostrar-ho a les pàgines html. És robust pel que genera un mínim marge d'error en la seva escriptura i compilació. Utilitzarem la llibreria **Marked<sup>2</sup>** per renderitzar markdown.

```
npm install marked
```

→ **I18next** és una llibreria per a treballar amb traduccions. Amb una configuració no molt difícil, podem crear un botó que apliqui els textos del idioma que seleccionem. Pel treball en qüestió, utilitzarem de base de dades arxius javascript amb un objecte principal amb els textos i continguts oficials del web. Tindrem només català i castellà.

En aquest cas, vist que he tingut alguns problemes amb la web deployed a producció, utilitzaré carga mitjançant CDN.

Tota la importació s'ha fet al arxiu main.js.

Com no és l'objectiu del projecte, no entrarem en detall de com s'estructura el JavaScript, però si es deixa indicat el codi i els 3 passos bàsics que s'han de fer:

### 1r → Importar els arxius amb les traduccions i inicialitzar la llibreria

```
import i18next from 'i18next';
import { caLocale } from '../locales/ca';
import { esLocale } from '../locales/es';

/*****
 * Translations
 *****/
// render translation to html
i18next.init({
  lng: 'es',
  debug: true,
  resources: {
    es: {
      translation: caLocale
    },
    ca: {
      translation: esLocale
    }
  },
}, function(err, t) {
  updateContent();
});
```

---

<sup>2</sup> <https://marked.js.org/>

## 2n → Afegir el botó de canvi d'idioma

```
// Agrega un controlador de eventos al botón para cambiar el idioma
document.querySelector('#switch-lang').addEventListener('click', function() {
  // Obtiene el idioma actual
  const currentLanguage = i18next.language;

  const newLanguage = currentLanguage === 'es' ? 'ca' : 'es';

  // Cambia el contenido del idioma
  document.querySelector('#switch-lang').textContent = currentLanguage.toUpperCase();

  // Actualiza el idioma activo en i18next y vuelve a renderizar las traducciones en el idioma correspondiente
  i18next.changeLanguage(newLanguage, function(err, t) {
    if (err) return console.log('Something went wrong loading the translation:', err);
    updateContent();
  });
});
```

## 3r → Renderitzar els textos de les traduccions al html

```
// Función para actualizar el contenido de la página con las traducciones
function updateContent() {
  const elements = document.querySelectorAll('[data-i18n-target]');
  elements.forEach(element => {
    const key = element.getAttribute('data-i18n');
    const target = element.getAttribute('data-i18n-target');
    element[target] = i18next.t(key);
  });
}
```

→ **@zkreations/imagehover**<sup>3</sup>, es una llibreria molt fàcil d'utilitzar, on un cop importat l'arxiu scss al nostre projecte, simplement afegint classes a les imatges provoca l'efecte hover corresponent. La norma és crear una figure:

```
<figure class="img-fade">
  
  <figcaption>Text here</figcaption>
</figure>
```

---

3 <https://www.cssscript.com/smooth-image-hover-effects/>

# CSS Grid

CSS Grid és una característica de CSS que permet crear dissenys d'una manera molt més flexible i sofisticada que amb altres mètodes com Flexbox o Floats. Permet crear una graella (grid) de files i columnes, i definir com s'han de distribuir els elements dins d'aquesta graella. Amb CSS Grid, es pot controlar el posicionament, la mida i l'espaiat dels elements, així com l'ordre en què apareixen en la pàgina.

S'estableix una disposició en graella per a l'element amb la classe `.layout`. La graella té una fila per al capçalera i tres columnes per als enllaços. També s'utilitza `display: grid` per establir la disposició en graella. Aquesta graella s'estableix utilitzant la propietat `grid-template-areas` per definir els noms de les àrees de la graella. Els selectors `&__heading` i `&__links` són pseudo-elements que s'utilitzen per referir-se a elements interns de `.layout`.

El layout de la pàgina principal, que abarca Grid, és el següent:

Heading		
Link 1	Link 2	Link 3

# Components Bootstrap

Els components de Bootstrap són un conjunt de blocs de construcció o elements predefinits que es poden utilitzar per crear interfícies d'usuari en pàgines web. Aquests components inclouen una àmplia gamma d'elements com ara botons, icones, menús de navegació, taules, formularis, caixes de diàleg, alertes, carruseles i molt més.

Aquí hi ha un recull dels components utilitzats al projecte:

## Navbar

L'element "navbar" és un component de Bootstrap que s'utilitza per crear menús de navegació responsius i adaptatius. Aquest component és molt útil perquè permet als usuaris accedir fàcilment a diferents seccions o pàgines del lloc web, independentment del dispositiu que estiguin utilitzant.

La "navbar" de Bootstrap és una barra horitzontal que normalment es col·loca a la part superior de la pàgina. Aquesta barra pot contenir diferents elements, com ara enllaços a pàgines, botons d'acció, formularis de cerca o menús desplegable. A més, la "navbar" de Bootstrap inclou opcions per personalitzar el seu aspecte, com ara la seva posició, el seu color i la seva animació.



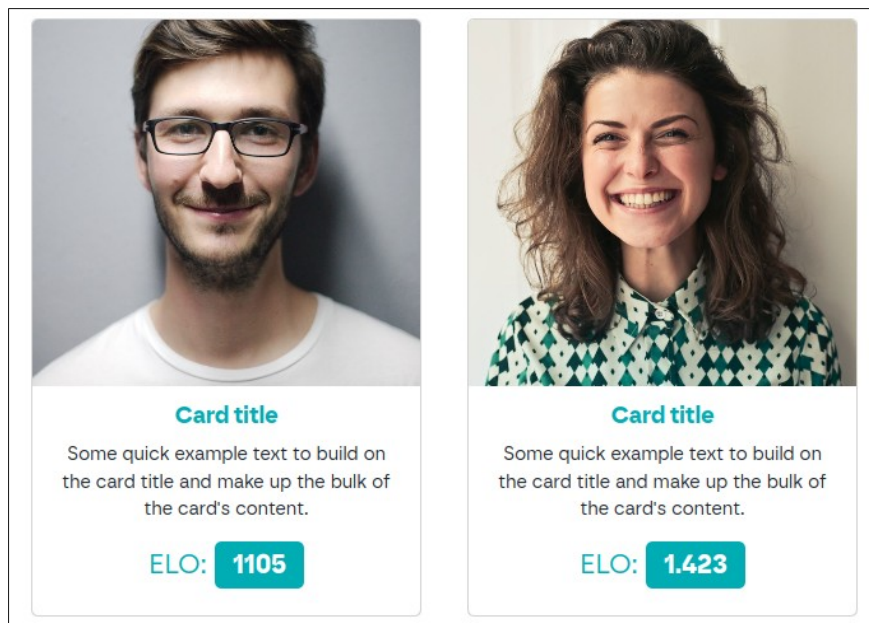
Members Blog Practice

**UTILITZACIÓ** → En el nostre projecte s'ha utilitzat per construir el menú de navegació principal a tot el web, el que ocuparà tots els headers de cadascuna de les pàgines.

## Card

El component "Card" de Bootstrap és un element visual que s'utilitza per mostrar informació d'una manera estructurada i atractiva. Les "Cards" són caixes rectangulars amb diferents seccions i poden contenir text, imatges, botons o altres components interactius.

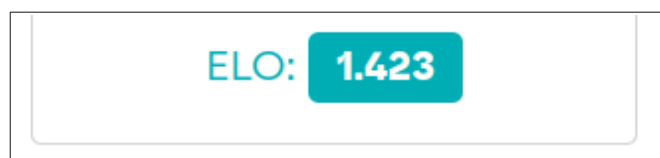
Amb aquesta funcionalitat, Bootstrap facilita la creació de dissenys responsius i adaptatius a diferents dispositius, com ara tauletes o telèfons mòbils.



**UTILITZACIÓ** → En el nostre projecte s'han utilitzat per montar, juntament amb les propietats de Grid Bootstrap, una graella amb els team member a la pàgina about.

## Badge

El component "badge" de Bootstrap és una etiqueta que s'utilitza per ressaltar informació important o destacada dins d'una pàgina web. Aquesta etiqueta es pot afegir a diferents elements HTML, com ara botons, enllaços o elements de llista, i es pot personalitzar per mostrar diferents colors, tipus de lletra i fons.



**UTILITZACIÓ** → S'han introduït els badges a la pàgina about, per mostrar les puntuacions ELO de cada persona del club.

## Breadcrumb

L'element "breadcrumb" de Bootstrap és un component de navegació que ajuda a l'usuari a entendre la seva ubicació dins d'un lloc web o aplicació. Consisteix en una sèrie de línies de text separades per símbols que representen les pàgines o seccions anteriors que l'usuari ha visitat per arribar a la pàgina actual.



L'element "breadcrumb" ajuda a l'usuari a orientar-se dins del lloc web o aplicació, ja que mostra la jerarquia de les pàgines i ajuda a comprendre la relació entre les diferents seccions. Això pot ser especialment útil en llocs web amb moltes pàgines o en aplicacions amb diverses capes i submenús.



**UTILITZACIÓ** → En totes les pàgines interiors, és a dir, totes excepte la home, s'ha introduït el breadcrumb per facilitar més la interacció i navegació del usuari.

## Formulari

El components del formulari de Bootstrap són fàcils d'utilitzar i personalitzar. Per exemple, podeu afegir validació de camps i personalitzar el disseny dels formularis per adaptar-se a les vostres necessitats.

A screenshot of a contact form on a dark grey background. At the top, the title 'Envian's el teu missatge i opinió!' is written in a bold, white, sans-serif font. Below the title, there are three main sections: 1. 'Email address' with a white input field containing the placeholder text 'El teu email'. 2. 'Message textarea' with a white text area containing the placeholder text 'El teu missatge'. 3. A checkbox labeled 'Estic d'acord amb la política de privacitat' in white text. Below the checkbox is a wide, rounded rectangular button with a blue border and the text 'Som-hi!' in blue. The entire form is centered on the page.

En el nostre cas només utilitzarem un input pel email, un textarea pel missatge, un checkbox de validació i el boto per enviar el missatge.

# Exemples

Bootstrap incorpora a la seva documentació exemple pre creats que poden ser incorporats ràpidament als projectes,

En el nostre cas, s'ha afegit una petita secció de boxes de pricing, en referència a les dues tipologies de quota que ofereix al club pels seus socis nous i actuals.

El recurs ha estat el següent:

**<https://getbootstrap.com/docs/5.0/examples/pricing/>**

S'ha adaptat el contingut a només dues caixes, amb una propietat molt interessant, que permet especificar concretament quantes columnes volem mostrar, que és “`row-cols-1 row-cols-md-2`”.

I el resultat final:



# Personalització de la llibreria

Per tal de seguir amb una correcta estructura i agrupament del codi, hem definit l'arxiu “\_root.scss” per fer un reset del css i agrupar les variables de Bootstrap que s’han personalitzat.

A root queden personalitzades totes les variables que no siguin d’estil bàsic, com els colors i tipografies. Aquests estils bàsics queden personalitzats dins l’arxiu “\_variables.scss”.

## Cards predefinits de Bootstrap

En aquest exemple, on modifiquem un aspecte del component Card e bootstrap, s'utilitzen dos mètodes:

1. Redefinim una variable pròpia i la col·loquem dins la variable de bootstrap que volem personalitzar.
2. De la classe pròpia de la llibreria ( card-body ) afegim més propietats que complementaran les que ja incorpora Bootstrap.

```
// Rooting CARD component
$card-height: 600px;

.card {
  --bs-card-height: $card-height;
}

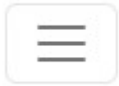

.card-body {
  display: flex;
  flex-direction: column;
  justify-content: space-evenly;
}
```

## Menú navegació

I com un segon exemple, el que s’ha volgut es modificar l’estil per defecte del botó per a la navegació per defecte de bootstrap.

Per fer-ho, hem anat un altre cop al arxiu \_variables.scss i em redifinit les variables que porta per defecte el botó de navegació.

La part és important és que, principalment, treballarem a l’arxiu **\_navbar.scss de bootstrap**, per detectar quines variable si noms s’utilitzen i poder treballar posteriorment en redifnir aquestes propietats.

Versió Inicial	Versió Final
	

L'exemple del canvi en el codi bàsicament ha estat:

```
// Boton de navegacion
$navbar-light-toggler-border-color: transparent;
$navbar-toggler-focus-width: 0;
```

# Utilització de Sass

<https://sass-lang.com/>

## @Mixins

Un `@mixin` és una funció reutilitzable en Sass que permet aplicar un conjunt de propietats CSS a diferents seleccions de l'estil.

```
// Heading color
@mixin color-heading($color, $shadow) {
  color: $color;
  text-shadow: 2px 2px 5px $shadow;
}
```

## @Supports

La directiva `@supports` en Sass s'utilitza per aplicar regles d'estil CSS basades en la capacitat de suport del navegador per a una determinada característica o propietat de CSS.

L'exemple en el nostre projecte, aplica la propietat CSS Grid si el navegador ho suporta. Si no, aplica un flex, que ho preparem amb una nova classe.

```
// Utilitzem la directiva @supports per comprovar si el navegador suporta CSS Grid
@supports (display: grid) {
  // Si el navegador suporta CSS Grid, apliquem l'estil de graella amb CSS Grid
  .layout {
    @extend .layout;
  }
}
// Si el navegador no suporta CSS Grid, apliquem l'estil de graella amb Flexbox
@supports not (display: grid) {
  .layout {
    display: none;
  }
  .layout-flex {
    @extend .layout;
  }
}
```

## Built in Modules

Un "built-in module" de Sass és un mòdul que forma part de la sintaxi del llenguatge Sass i que proporciona funcions i mixins predefinits. Aquests mòduls estan inclosos

dins el propi Sass, per la qual cosa no és necessari instal·lar cap llibreria addicional per a utilitzar-los.

Pel nostre projecte hem utilitzat un de molt bàsic, el **Sass:Color**.

Aquest és un mòdul integrat a la sintaxi de Sass que proporciona funcions i mixins per manipular colors. Això inclou funcions per canviar la lluminositat, la saturació, el to i l'opacitat d'un color, així com per mesclar colors, crear gradients, etc.

Pel projecte hem ampliat el mixin inicial que teniem que redefinir el color:

```
// Heading color
@mixin color-heading($color, $shadow, $darken: 10%) {
  color: darken($color: $color, $amount: $darken);
  text-shadow: 2px 2px 5px $shadow;
}
```

# **Bibliografia**

## **Recursos gràfics ( imatges )**

<https://www.pexels.com/es-es/>

## **Recursos gràfics ( videos )**

<https://www.youtube.com/watch?v=8IlJ3v8I4Z8>

<https://www.youtube.com/watch?v=goW1cFHaxtU>

<https://www.youtube.com/watch?v=MRNZE990cgc>

<https://www.youtube.com/watch?v=UWDraLFMHm0>

## **Traduccions**

<https://chat.openai.com/>