|  |
| --- |
| Mohamed AL-AZAWI  23/10/2023 |

**Réalisation d’un site web type CV**



****

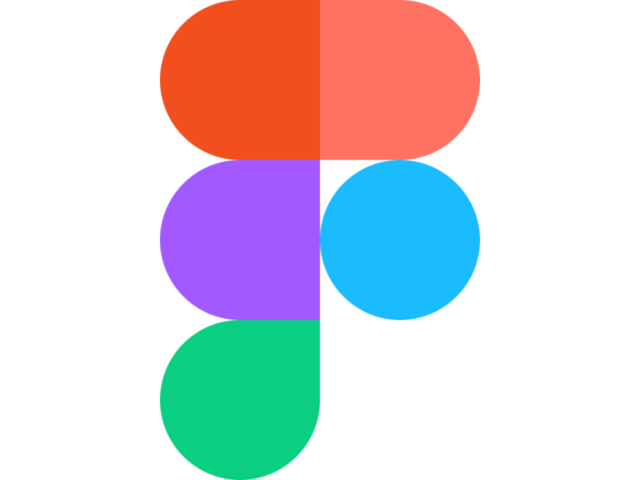


Table des matières

[Introduction 3](#_Toc151389746)

[FIGMA 4](#_Toc151389747)

[Page d'accueil 5](#_Toc151389748)

[Éléments clés : 5](#_Toc151389749)

[Page de Présentation 5](#_Toc151389750)

[Page Expériences 6](#_Toc151389751)

[Page Loisirs 6](#_Toc151389752)

[Page de Contact 7](#_Toc151389753)

[<>HTML 7](#_Toc151389754)

[En-tête (Head) : 7](#_Toc151389755)

[En-tête (Header) : 8](#_Toc151389756)

[Contenu Principal (Content) : 8](#_Toc151389757)

[Pied de page (Footer): 9](#_Toc151389758)

[Page expériences pro 9](#_Toc151389759)

[Page contact 10](#_Toc151389760)

[# CSS 11](#_Toc151389761)

[Le CSS de mon site : 11](#_Toc151389762)

[Général 11](#_Toc151389763)

[Header 12](#_Toc151389764)

[Faire appel à différents class en un coup 13](#_Toc151389765)

[Responsive 15](#_Toc151389766)

[JavaScript 16](#_Toc151389767)

[Conclusion 18](#_Toc151389768)

# Introduction

Cette documentation se portera sur la réalisation d'un site web responsive. Dans un premier temps je vais implémenter une maquette du site sur Figma, suivie de l'implémentation avec HTML et CSS. Le site sera rendu responsive sur téléphone mobile également.

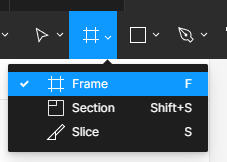
Le processus débute avec Figma, où je vais construire les wireframes, prototypes interactifs et maquettes haute-fidélité. Cet outil me permettra de travailler sur le design en tenant compte des différents points de rupture et de l'expérience utilisateur (UX). J'exploiterai les fonctionnalités de Figma pour créer des composants réutilisables, assurant ainsi une cohérence visuelle et fonctionnelle sur l'ensemble du site.

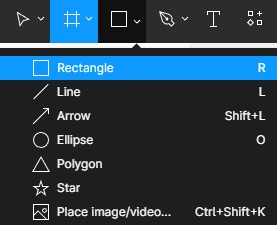
Une fois le design approuvé, je passerai à la transcription du design en code. HTML sera utilisé pour marquer la structure sémantique du site, tandis que CSS gérera la mise en page, l'esthétique et les média queries nécessaires pour le responsive design. Vue que je commence par créer un CSS pour le desktop, le responsive pour le mobile ne sera pas idéal mais plutôt un atout en plus.

L'aspect responsive sera pris en compte tout au long du développement. Les média queries en CSS permettront d'ajuster la mise en page en fonction de la largeur de l'écran. Je ferai également usage des unités relatives comme les pourcentages, 'em', et 'vw/vh' pour une flexibilité maximale. Les tests de fonctionnalité sur le dispositif mobile seront systématiques, utilisant des outils d'émulation comme le DevTools.

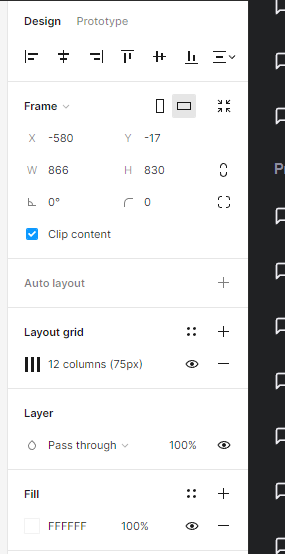
# FIGMA

Figma est un outil de design d'interface utilisateur et d'expérience utilisateur basé sur le cloud qui permet aux équipes de collaborer en temps réel. C'est une plateforme polyvalente utilisée pour la création de wireframes, de prototypes interactifs et de designs de haute-fidélité pour des applications web et mobiles. Un aspect puissant de Figma est son système de grille ("grid system"), qui permet aux designers de créer des mises en page structurées et cohérentes.

Pour notre projet nous allons créer un frame pour desktop :

Ensuite nous allons créer des rectangles pour illustrer le footer et barre de navigation :

Dans le menu en haut à gauche nous avons à disposition tout ce que dont nous avons besoin pour créer nos pages web.



Pour ajouter des grids aux pages : il faut cliquer sur le nom de la page en suite dans le menu de droite ajouter un layout grid. J’ai paramétré mes pages à 12 colonnes de 75 px.

Pour les éléments que nous avons créés sur la page telle que la barre de navigation, footer, photo, menu, logo et autres qui apparaissent sur plusieurs pages. Je les ai transformés en « component » pour les réutiliser.

## Page d'accueil

La page d'accueil est le point d'entrée du site et contient un bref accueil et une introduction à Mohamed, ainsi que des liens vers ses profils GitHub et LinkedIn.

### Éléments clés :

Le logo avec le mot-clé "Developer" entouré de balises HTML, à gauche dans la barre de navigation.

Un message de bienvenue suivi du nom "Mohamed, apprenti développeur".

Un emplacement pour une citation philosophique.

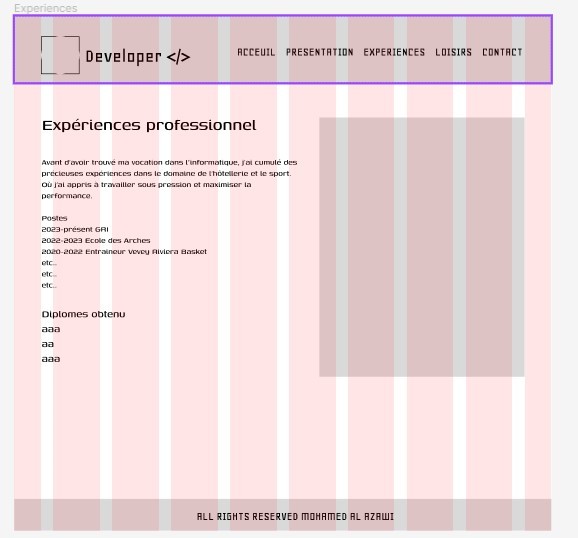
Un bouton invitant à en apprendre plus sur moi.

## Page de Présentation

Cette page offre un aperçu plus détaillé de la personnalité et des compétences.

Un paragraphe introductif sur mon rôle et ambitions, un bouton menant à ses expériences professionnelles.

## Page Expériences

Ici, je détaille mon parcours professionnel et académique.

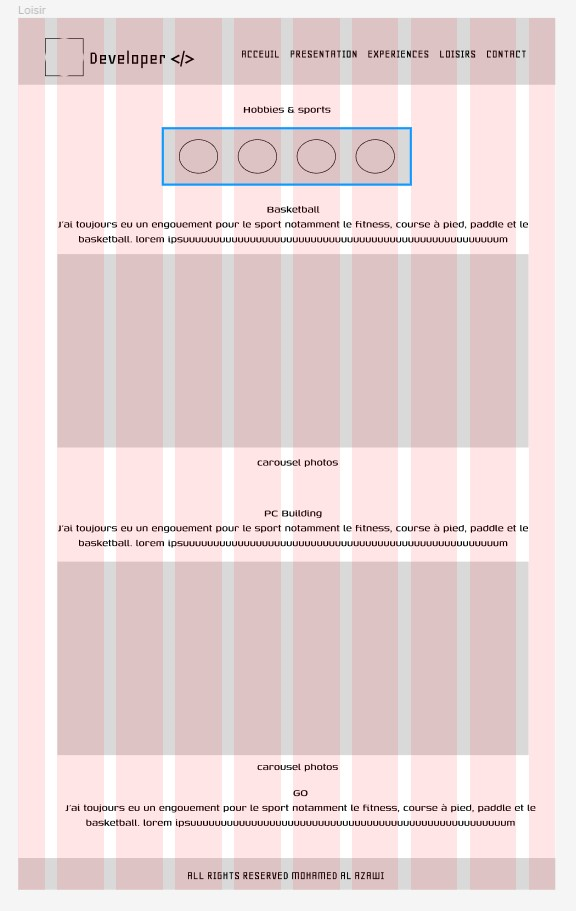
Éléments clés :

Un aperçu de mes expériences antérieures dans l'hôtellerie et le sport.

Une liste de postes précédents, avec des dates correspondantes.

Une section séparée pour les diplômes obtenus.

## Page Loisirs

Cette page illustre les intérêts personnels et les hobbies, soulignant mon engagement pour le sport.

Éléments clés :

Des catégories de hobbies comme le basketball, la construction de PC, et le Go.

Les carrousels nécessitent une implémentation de glissement/défilement pour afficher différentes images.

## Page de Contact



Cette page illustre comporte un formulaire de contact

Eléments général :

Les boutons doivent rediriger l'utilisateur vers la section correspondante ou extérieure (GitHub/LinkedIn). Ces maquettes Figma pour mon portfolio fournissent une structure de base pour mon site type CV. Le design vise à être simple mais engageant, mettant en avant mes compétences et mes passions de de manière professionnelle et personnelle.

# <>HTML

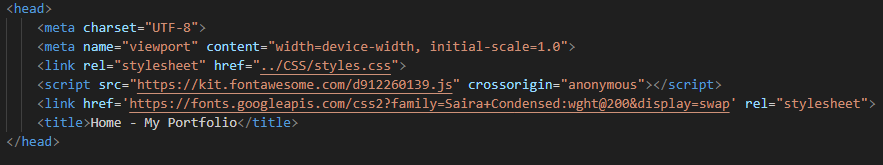
HTML, acronyme de HyperText Markup Language, est le langage de balisage standard utilisé pour créer des pages web. Il décrit et définit le contenu d'une page web en utilisant une série d'éléments, appelés balises, pour structurer le texte, les liens, les images et d'autres contenus. Les navigateurs web lisent les documents HTML et les affichent sous forme de pages web.

Doctype et Langue :

Le document commence par la déclaration <!DOCTYPE html>, qui indique au navigateur qu'il s'agit d'un document HTML5.

La balise <html lang="en"> définit la langue principale du document comme étant l'anglais.

## En-tête (Head) :



La balise <head> contient des métadonnées et des liens vers des ressources externes comme fontawesome pour les icones et une fonte Google appelé « Saira ».

<meta charset="UTF-8"> : spécifie l'encodage des caractères pour le contenu de la page.

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Permet d'assurer que la mise en page est responsive et s'affiche correctement sur les appareils mobiles.

<link rel="stylesheet" href="../CSS/styles.css"> Le lien vers la page de style CSS.

Corps (Body) :

L'élément <body> englobe tout le contenu affiché à l'utilisateur.

## En-tête (Header) :

La section <header> contient un logo et un menu de navigation, avec des liens vers différentes pages de votre site (Accueil, Présentation, Expériences, Loisirs, Contact).

La balise img src dans le code HTML sert à incorporer une image dans une page web, et l'attribut src spécifie le chemin de l'image à afficher ; ici, le chemin ../Images/logo2.png indique que l'image se trouve dans le dossier "Images" qui est un niveau au-dessus du dossier courant où se trouve le fichier HTML.

La balise href (hypertext reference) est utilisée dans les éléments d'ancrage a en HTML pour définir la destination (URL ou chemin de fichier) où le navigateur doit aller lorsque l'utilisateur clique sur le lien.

## Contenu Principal (Content) :

Dans la section <div class="content">, j’ai divisé la zone principale en deux sections, droite et gauche, en utilisant des divs avec les classes right et left.

    <div class="content">

      <div class="right">

        <img src="../Images/photo.jpg" alt="my Image" class="home-image">

        <div class="social-icons">

          <a href="https://www.linkedin.com/in/mohamed-al-azawi-63162b74/">

            <i class="fa-brands fa-linkedin"></i>

          </a>

          <a href="https://github.com/MohAIzawi/NewRepo">

            <i class="fa-brands fa-square-github"></i>

          </a>

        </div>

      </div>

      <div class="left">

        <h2>BONJOUR !</h2><br>

        <p>je m'appelle</p><br>

        <h1>Mohamed</h1><br>

        <h1>apprenti développeur</h1><br><br>

        <a href="Presentation.html">

        <button class="button">en savoir plus...</button>

        </a>

      </div>

    </div>

La partie droite affiche une image personnelle et des icônes de réseaux sociaux qui sont des liens vers vos profils LinkedIn et GitHub.

Dans ce code HTML, la balise <i> est utilisée pour ajouter des icônes (ici, des icônes de réseaux sociaux LinkedIn et GitHub) avec une classe de la bibliothèque Font Awesome, tandis que la balise « button » est utilisée pour créer un bouton cliquable, souvent pour soumettre des données ou déclencher une action en JavaScript.

La partie gauche contient une introduction personnelle avec des éléments <h2> et <h1> pour mettre en avant le nom et l’apprentissage de développeur.

## Pied de page (Footer):

<footer>

      <div class="footer">

        &copy; 2023 by Mohamed Al Azawi. All Rights Reserved.

      </div>

    </footer>

La balise <footer> en HTML est utilisée pour définir le pied de page d'un document ou d'une section, généralement contenant des informations d'auteur, des droits d'auteur, des liens de navigation, des contacts ou d'autres informations générales liées au contenu principal.

## Page expériences pro

La structure HTML de la section "Expériences professionnelles" est organisée pour présenter les antécédents professionnels de manière chronologique, avec chaque poste listé accompagné des dates correspondantes et une description des responsabilités ou des compétences acquises.

<span class="job-title">Apprenti développeur</span>

**Balise span**: Chaque titre de poste est entouré d'une balise span avec une classe job-title, ce qui permet d'appliquer un style spécifique aux titres de poste pour les faire ressortir du reste du texte, facilitant ainsi la lecture et la distinction des différentes positions tenues.

<ul>

     <li>Royal Plaza | Montreux<br>Reservations, check-in check-</li>

     <li>Hôtel Ibis | Crissier<br>Clôture journalière sur système</li>

     <li>Grand Hôtel Suisse-Majestic | Montreux<br>Contrôle des </li>

</ul>

**Balises <ul> et <li>** : Elles sont utilisées pour créer une liste non ordonnée, avec chaque élément li représentant une expérience de travail différente au sein de la section "Réceptionniste | Night Auditor". Ces balises aident à structurer l'information de manière claire et hiérarchisée, rendant la liste des tâches et des responsabilités faciles à suivre pour le lecteur.

## Page contact

La balise <form> est utilisée pour créer un formulaire HTML sur une page web. Les attributs de cette balise déterminent le comportement du formulaire lors de son envoi (soumission) :

<form action="mailto:mohamed.alazawi@gri.ch" method="get"

action="mailto:mohamed.alazawi@gri.ch" : Cet attribut indique que lorsque le formulaire sera soumis, l'application cliente de messagerie par défaut de l'utilisateur tentera d'ouvrir une nouvelle fenêtre de message avec l'adresse e-mail "mohamed.alazawi@gri.ch" déjà remplie dans le champ destinataire.

method="get" : Cet attribut spécifie la méthode HTTP à utiliser lors de l'envoi des données du formulaire. Ici, "get" indique que les données du formulaire seront ajoutées à l'URL de la requête sous forme de chaîne de requête.

<div class="input-group">

<label for="firstname">Prénom:</label>

<input type="text" id="firstname" name="firstname" placeholder="votre prénom" required>

</div>

**Balise <label> :** Cette balise définit une étiquette pour l'élément de saisie du formulaire (<input>). L'attribut for est utilisé pour associer l'étiquette à l'élément de saisie spécifique ayant l'identifiant (attribut id) correspondant, ce qui est "firstname" dans ce cas. Cela signifie que si vous cliquez sur le texte de l'étiquette "Prénom :", le curseur sera placé dans la zone de saisie associée, ce qui améliore l'accessibilité et l'expérience utilisateur.

**Balise <input> :** Cet élément est utilisé pour créer un champ de saisie où l'utilisateur peut entrer du texte. L'attribut type="text" indique que l'entrée est destinée à du texte simple. L'id est utilisé pour identifier l'élément de manière unique dans le DOM et est également utilisé pour l'associer avec sa <label> correspondante via l'attribut for. L'attribut « name » donne le nom du champ de données, qui est utilisé pour identifier les données du formulaire lors de la soumission. L'attribut « **placeholder »** fournit un texte d'exemple qui sera affiché dans le champ de saisie jusqu'à ce que l'utilisateur commence à taper. Enfin, l'attribut « **required »** rend le champ obligatoire, ce qui signifie que le formulaire ne peut pas être soumis si l'utilisateur ne remplit pas ce champ.

<input type="radio" name="sex" id="male" value="male" required>

**La balise <input type="radio">** crée un bouton d'option unique au sein d'un groupe portant le même attribut « name ».

<input type="checkbox" id="confirmation" name="confirmation" required>

**La balise <input type="checkbox">** crée une case à cocher que l'utilisateur peut sélectionner ou désélectionner ;

Les pages **présentation** et **loisirs** ne figure pas sur la documentation car elles n’apportent aucun élément HTML supplémentaire.

# # CSS

Le CSS, ou Cascading Style Sheets, est un langage de feuille de style utilisé pour décrire la présentation d'un document écrit en HTML. Il permet de contrôler le layout, les couleurs, les polices et le style de l'ensemble d'un site web de manière centralisée, améliorant ainsi l'accessibilité et la maintenance du design.

## Le CSS de mon site :

### Général

\*{

  box-sizing: border-box;

  margin: 0; /\* enlever les paramètres par défault du navigateur \*/

  padding: 0; /\* enlever les paramètres par défault du navigateur \*/

}

CSS commun qui réinitialise la marge et le rembourrage de tous les éléments à 0 et configure le modèle de boîte pour que les dimensions des éléments (largeur et hauteur) incluent le contenu, la bordure et le rembourrage, éliminant les différences par défaut entre les navigateurs.

html {

  scroll-behavior: smooth;

}

Cette règle CSS applique un défilement doux ("smooth scrolling") à toute la page, permettant une transition en douceur lors du défilement vers des ancres sur la même page.

a {

  text-decoration: none;

}

Cette règle CSS supprime les soulignements par défaut des liens hypertextes dans le document, pour un style de texte plus épuré.

body {

  height: auto;

  min-height: 100vh;

  width: 100vw;

}

**La propriété vh (viewport height)** est une unité de mesure en CSS qui correspond à un pourcentage de la hauteur de la fenêtre d'affichage (viewport). 100vh est donc égal à 100% de la hauteur actuelle de la fenêtre d'affichage.

**La propriété vw (viewport width)** fonctionne de la même manière, mais se rapporte à la largeur de la fenêtre d'affichage. 100vw signifie 100% de la largeur de la fenêtre d'affichage.

Lorsque height est réglé sur auto, la hauteur de l'élément est déterminée par son contenu : il prendra la hauteur nécessaire pour afficher son contenu intérieur.

Dans l'ensemble, cette règle CSS garantit que le <body> s'étend au moins à la hauteur complète de l'écran (viewport), tout en prenant la largeur complète, indépendamment de la quantité de contenu qu'il contient.

.content p {

  text-align: justify;

  width: 50%;

}

Cette règle CSS définit que les paragraphes (<p>) à l'intérieur d'éléments avec la classe « .content » seront justifiés, c'est-à-dire que le texte sera aligné uniformément le long des marges gauche et droite, et que ces paragraphes occuperont 50% de la largeur de leur conteneur parent.

### Header

.header {

  display: flex;

  justify-content: space-between;

align-items: center;

  padding: 1%;

  opacity: 80%;

  background-color: black;

  color: white;

}

**« .header » :**

display: flex active Flexbox, permettant un layout flexible des éléments enfants.

justify-content: space-between répartit les éléments enfants uniformément dans le conteneur, avec le premier élément au début et le dernier à la fin.

align-items: « center » aligne verticalement les éléments enfants au centre du conteneur.

padding: 1% ajoute un peu d'espace à l'intérieur du bord du conteneur, de toutes parts.

opacity: 80% rend le conteneur légèrement transparent.

background-color: black donne au conteneur un fond noir.

color: white définit la couleur du texte à blanc pour un contraste élevé avec l'arrière-plan noir.

.header a {

  padding: 0.7em 1em;

}

Ajoute un rembourrage intérieur autour du texte des liens, 0.7em en haut et en bas et 1em sur les côtés.

En CSS, **em** est une unité relative de mesure qui est basée sur la taille de la police courante. Si utilisée pour spécifier une font-size, 1em équivaut à la taille de la police du parent direct. Pour d'autres propriétés (comme width, margin, padding), 1em correspond à la taille de la police de l'élément lui-même. L'utilisation d'em permet une conception flexible et accessible, car la mise en page s'adapte selon la taille de la police de l'utilisateur ou du parent.

.header a:hover {

  text-decoration: underline;

}

Souligne les liens au survol, ce qui est un indicateur visuel commun pour les éléments du menu.

.footer {

  position: fixed;

  bottom: 0;

  width: 100%;

}

position : fixed; permet de positionner l'élément de manière fixe par rapport à la fenêtre du navigateur. Cela signifie que l'élément restera à sa place même si l'utilisateur fait défiler la page. Il ne se déplacera pas avec le contenu lors du défilement.

bottom: 0; travaille en conjonction avec position: fixed; pour ancrer l'élément au bas de la fenêtre du navigateur. Une valeur de 0 signifie que le bas de l'élément sera exactement au bord inférieur de la fenêtre du navigateur, sans aucun espace entre les deux.

width: 100%; définit la largeur de l'élément pour qu'il s'étende et occupe 100% de la largeur de l'élément conteneur, qui est, dans le cas de position: fixed;, la fenêtre du navigateur. Ainsi, le pied de page s'étendra sur toute la largeur de l'écran, de gauche à droite.

### Faire appel à différents class en un coup

.carousel img.fit,

.carousel img.basket,

.carousel img.run1,

.carousel img.run2 {

  object-position: top;  /\* recentrer ces photos au milieu du cover \*/

}

.home-image {

  object-fit: cover;

  box-shadow: 0 -10px 20px 20px rgb(0, 217, 240);

}

box-shadow: 0 -10px 20px 20px rgb(0, 217, 240); box-shadow est utilisée pour ajouter une ombre autour de l'élément HTML. Les valeurs spécifient respectivement le décalage horizontal (0), le décalage vertical (-10px), le flou (20px), l'étendue de l'ombre (20px), et la couleur de l'ombre (rgb(0, 217, 240)), qui dans ce cas est un bleu turquoise lumineux. Object-fit: cover; assure que le contenu visuel s'adapte à son conteneur sans distorsion.

.button {

  transition: transform 0.5s ease; /\* transition et vitesse \*/

}

.button:hover {

  transform: scale(1.3); /\* Scale the entire button on hover \*/

}

**.button:**

transition: transform 0.5s ease; applique un effet de transition à la propriété transform sur une période de 0.5 seconde avec un effet d'accélération et de ralentissement (ease).

**.button:hover:**

Cette règle s'applique lorsque l'utilisateur passe la souris sur le bouton.

transform: scale(1.3); agrandit le bouton à 130 % de sa taille d'origine, donnant un effet de zoom pour une rétroaction visuelle au survol.

« transition »:

Cette propriété CSS permet aux changements de propriétés de se produire doucement sur une période de temps donnée. Dans votre cas, transform est la propriété animée, et 0.5s ease est la durée et le type de timing pour la transition. ease signifie que la transition commence lentement, devient plus rapide au milieu, puis ralentit à nouveau à la fin.

« transform »:

transform applique une transformation à un élément. Dans le cas de transform: scale(1.3);, la transformation est une mise à l'échelle, ce qui signifie que le bouton sera agrandi à 130 % de sa taille lorsqu'il est survolé.

.social-icons {

  text-align: center; /\* Centrer les icones horizentalement \*/

}

.social-icons i {

  margin: 0.2em;

  color: #fff;

  transition: transform 0.2s ease;

}

.social-icons i:hover {

  color: #fff;

  transform: scale(1.3);

}

Le même effet a été implémenter sur les icones loisirs et réseaux sociaux avec l’ajout de l’élément. <i> Cette règle cible les éléments <i> à l'intérieur des éléments avec la classe social-icons. Les éléments <i> sont souvent utilisés pour les icônes de police.

J’ai utilisé ces 3 propriétés à diffèrent endroit se sont des propriétés souvent utilisé en CSS pour une mise en page facile et responsive friendly.

* display: flex ;

Cette propriété est utilisée pour définir un conteneur flex. Lorsque vous appliquez display: flex; à un élément, cet élément devient un conteneur flex, et ses enfants directs deviennent des éléments flexibles (flex items). Cela signifie que les enfants seront placés dans un contexte de flexibilité, ce qui permet de les aligner et de les distribuer facilement dans l'espace disponible.

* justify-content ;

Cette propriété définit comment les éléments flexibles sont alignés et répartis le long de l'axe principal (horizontalement dans le cas d'une flex-direction par défaut de row). Elle permet de contrôler l'espacement entre les éléments et autour d'eux. Les valeurs courantes comprennent flex-start, flex-end, center, space-between, space-around, et space-evenly. Par exemple, justify-content:

center; alignera tous les éléments flexibles au centre du conteneur flex sur l'axe principal.

* align-items ;

Cette propriété est similaire à justify-content, mais elle agit sur l'axe perpendiculaire (l'axe transversal, verticalement dans le cas d'une flex-direction par défaut de row). Elle détermine comment les éléments flexibles sont alignés et répartis le long de cet axe transversal. Les valeurs courantes incluent flex-start, flex-end, center, baseline, et stretch. Par exemple, align-items: center; alignera tous les éléments flexibles au centre du conteneur flex sur l'axe transversal.

# Responsive

@media screen and (max-width: 600px)

Cette ligne CSS cible les appareils avec un écran d'une largeur maximale de 600 pixels et applique les styles contenus dans les accolades qui suivent, uniquement lorsque cette condition est remplie.

Pour le responsive, j’ai simplement changé les directions de flex, les valeurs de margin, de padding et l'alignement. La seule petite complication était de créer une nouvelle classe pour les images en bas de la page Loisirs, afin de pouvoir les décaler sans chambouler les autres éléments de la page. Car il se glissait légèrement sous le footer.

# JavaScript

JavaScript est un langage de programmation essentiel dans le développement web, utilisé pour ajouter interactivité et fonctionnalités aux sites web. Initialement conçu pour le front-end, il est également utilisé côté serveur grâce à Node.js. Sa popularité repose sur sa polyvalence, sa facilité d'apprentissage, et son large écosystème. JavaScript est un outil incontournable pour les développeurs web modernes.

En JavaScript, let et const sont deux mots-clés utilisés pour déclarer des variables :

* let : Permet de déclarer des variables avec une portée limitée au bloc de code où elles sont définies. Ces variables peuvent être réaffectées après leur déclaration.
* const : Utilisé pour déclarer des variables dont la valeur ne peut pas être réaffectée après leur initialisation. Comme let, const a une portée de bloc.

Mon code :

      <div class="extra">

        <h2>Number Guessing Game</h2>

        <p>Enter the minimum and maximum values:</p>

        <input type="number" id="Min" placeholder="Min">

        <input type="number" id="Max" placeholder="Max">

        <button onclick="startGame()">Start Game</button>

        <div id="gameArea" style="display: none;">

          <p>Guess a number between &nbsp<span id="valMin"></span>&nbsp and &nbsp<span id="valMax"></span></p>

            <input type="number" id="guess">

            <button onclick="submitGuess()">Submit Guess</button>

            <p id="feedback"></p>

          </p>

        </div>

      </div>

Section principale : Le code est enveloppé dans une <div> avec la classe "extra". À l'intérieur, un titre <h2> annonce qu'il s'agit d'un "Number Guessing Game" (jeu de devinette de nombres).

Entrée des valeurs minimale et maximale : Deux champs de saisie de type "number" permettent à l'utilisateur d'entrer les valeurs minimales (Min) et maximales (Max) pour le jeu. Ces champs sont accompagnés de textes explicatifs.

Bouton de démarrage du jeu : Un bouton est présent pour lancer le jeu (Start Game). Il est lié à un événement onclick qui appelle la fonction startGame().

Zone de jeu : La <div> avec l'ID "gameArea" est initialement cachée (display : none). Elle contient les instructions pour deviner un nombre, un champ de saisie pour la supposition du joueur, un bouton pour soumettre la supposition (lié à la fonction submitGuess()) et un paragraphe pour les retours (feedback).

Dynamisme avec JavaScript : L'interaction avec le jeu (comme démarrer le jeu et soumettre des suppositions) est gérée via des fonctions JavaScript (startGame() et submitGuess()), suggérant que le jeu est interactif et dynamique.

Le code crée donc une interface utilisateur pour un jeu simple où les joueurs devinent un nombre dans une plage définie.

Le code du jeu en JavaScript :

let valMax;

let valMin;

let randomNumber;

let attempts= 0;

let guessHistory= [];

function startGame () {

    valMin = parseInt(document.getElementById('Min').value)

    valMax = parseInt(document.getElementById('Max').value)

    if (isNaN(valMin) || isNaN(valMax) || valMin >= valMax) {

    alert('Please enter valid min and max values.');

    return;

}

    guessHistory = []; /\* réinitialliser au démaragge du jeux \*/

    randomNumber = Math.floor(Math.random()\*(valMax-valMin+1)) + valMin;

    attempts = 0;

    document.getElementById('valMin').textContent = valMin; /\* contenu qu'on va étblir dans la foction submit number \*/

    document.getElementById('valMax').textContent = valMax;

    document.getElementById('gameArea').style.display = 'block';

    document.getElementById('feedback').textContent;

    document.getElementById('guessHistory').textContent = '';

}

function submitGuess () {

    const userGuess = parseInt(document.getElementById('guess').value); /\* valeur de variable ne doit pas etre changer \*/

    attempts++;

    guessHistory.push(userGuess); // ajoute le # deviner à la liste

    if (userGuess === randomNumber) {

        document.getElementById('gameArea').style.display = 'block'

        document.getElementById('feedback').textContent = `Congradulations! the right number was ${randomNumber} ! you guessed it in ${attempts} attempts.`

    }

    else if (userGuess < randomNumber){

        document.getElementById('feedback').textContent = `Try again. The number is greater than ${userGuess}`

    }

    else {

        document.getElementById('feedback').textContent = `Try again. The number is less than ${userGuess}`

    }

    document.getElementById('guessHistory').textContent = `Guess History: ${guessHistory.join(', ')}`; // affiche l'historique

}

**Fonction startGame :**

Validation des entrées : Vérifie si les valeurs minimale (valMin) et maximale (valMax) saisies sont valides. Si les valeurs ne sont pas des nombres (isNaN) ou si valMin est supérieur ou égal à valMax, une alerte est affichée et la fonction s'arrête.

Génération du nombre aléatoire : Utilise Math.random() pour générer un nombre aléatoire entre valMin et valMax. Math.floor() est utilisé pour s'assurer que ce nombre est entier.

Affichage de la zone de jeu : Modifie le DOM pour afficher les valeurs minimales et maximales dans valMin et valMax et rend visible la zone de jeu (gameArea).

**Fonction submitGuess :**

Récupération et traitement de la supposition : Récupère la supposition de l'utilisateur (userGuess) et incrémente le compteur de tentatives (attempts).

Logique de comparaison : Compare userGuess avec le randomNumber :

Si égal, affiche un message de félicitations et révèle le nombre de tentatives.

Si userGuess est inférieur, indique que le nombre mystère est plus grand.

Si userGuess est supérieur, indique que le nombre mystère est plus petit.

Dans chaque cas, un feedback est affiché dans l'élément avec l'ID feedback.

Ces fonctions manipulent le DOM pour interagir avec l'utilisateur et gérer la logique du jeu, en affichant des messages pertinents selon les actions de l'utilisateur.

# Conclusion

En conclusion, cette documentation représente un parcours complet à travers la création d'un site web. Depuis la conception initiale sur Figma jusqu'à la mise en œuvre avec HTML et CSS, en passant par l'optimisation pour le responsive design, chaque étape du processus est expliquée en détail.

En suivant ce guide, j'ai pu planifier efficacement la structure de mon site, créer des maquettes fonctionnelles, et les traduire en code HTML et CSS. J'ai appris comment structurer mon contenu de manière sémantique en utilisant des balises HTML appropriées, et comment styler mon site pour qu'il soit esthétiquement agréable.

Chaque page, de la page d'accueil aux pages de présentation, d'expériences professionnelles, de loisirs et de contact, a été examinée, fournissant des conseils sur la manière de présenter mes compétences et mon parcours de manière professionnelle et engageante.

Le CSS a été utilisé pour créer une mise en page flexible et réactive, et les média queries ont permis de garantir une expérience utilisateur optimale sur différents appareils.

En fin de compte, cette documentation m'a aidé à créer mon propre site web CV de manière efficace, en mettant en avant mes compétences et ma personnalité de manière professionnelle. Elle constitue une ressource précieuse pour quiconque souhaite entreprendre un projet similaire.