

Portfolio Webstatique

Une présentation “type” soutenance

1- Présentation

Prénom : Idriss

Passionné par le développement, les jeux vidéo et la réalité augmentée.

Objectif de formation :

- Etre capable de construire une application de A à Z.

- Etre sensible à l'accessibilité.

- Savoir collaborer sur des projets.

- Travailler dans le monde du jeux vidéo et/ou réalité augmentée.

Sommaire

- Présentation
- Besoins
- Solutions
- Environnement de travail
- Sitemap
- Structure HTML
 - Démonstration détaillé sur :
 - Head - Header - Contact - Réalisation
- Conception dynamique de la partie réalisations
- La validation du formulaire en javascript
- Charte Graphique
- Maquette
- Structure CSS
 - Démonstration détaillé sur :
 - Header - Réalisations
- Remerciement
- Questions ?

2- Besoins

- Faire un portfolio
 - Mobile first.
 - Pages obligatoires
 - Accueil, Réalisations, A propos, Contact
 - Utiliser un fichier XML
 - Formulaire de contact
 - Faire une validation du formulaire de contact
 - A rendre Lundi 9 Juin 2024.
-
- Thème en dehors de ma zone de confort
 - Faire les scripts en orienté objet.
 - Pas de framework
 - Pas utiliser d'IA

3- Solutions

Petit projet classique.

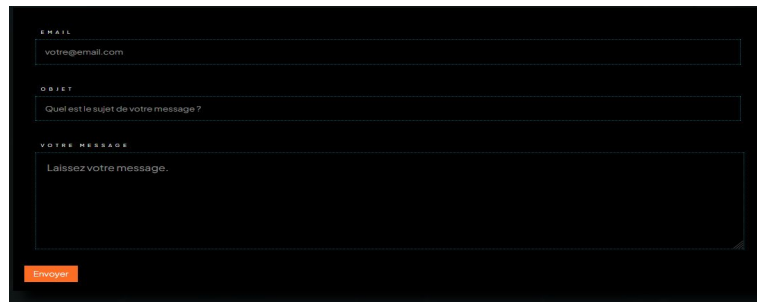
Afficher mes projets à l'aide du fichier xml.

Réutiliser le script fait en groupe.

Réutiliser formulaire de mon site.

Sémantique HTML pour la SEO.

Doit copié collé les footer et head/header



EMAIL

vous@email.com

OBJET

Quel est le sujet de votre message ?

VOTRE MESSAGE

Laissez votre message.

Envoyer

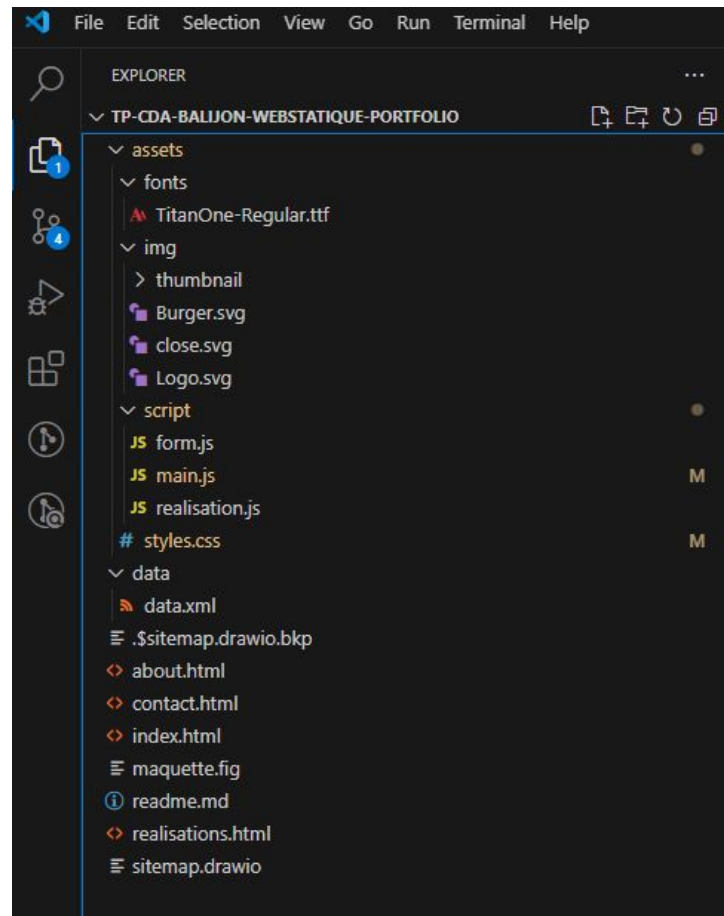
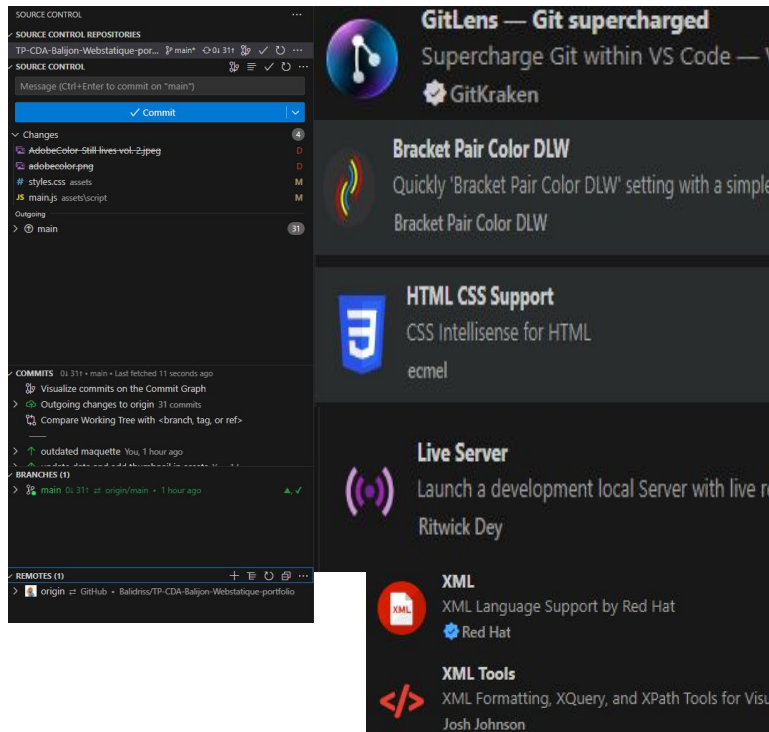
4- Environnement de travail

Vscode

Adobe color

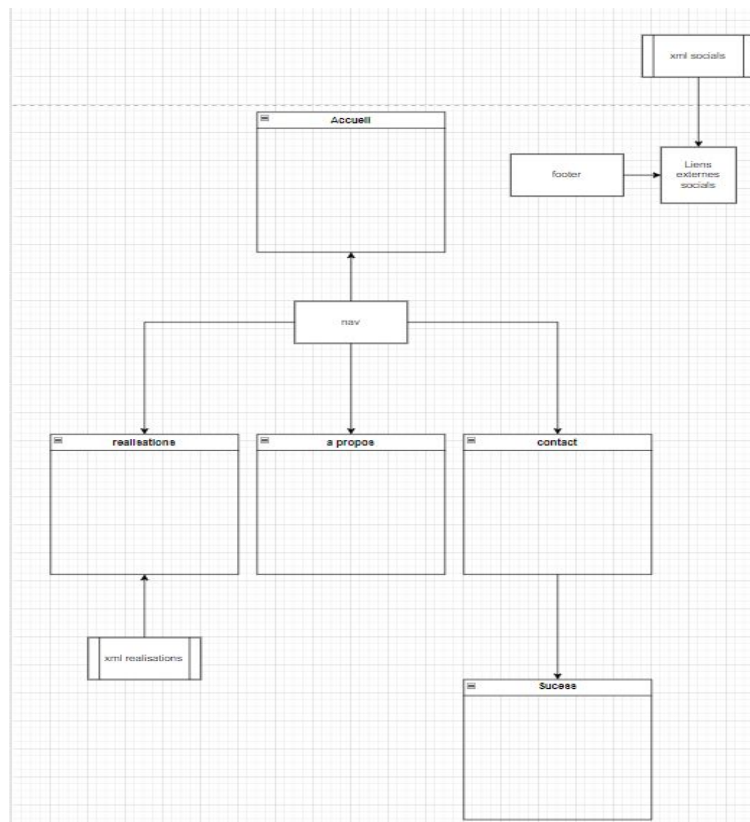
figma

draw.io



5- Sitemap

N'a pas été nécessaire



6- Structure HTML

Besoin d'un menu qui s'adapte à un large écran.

Head - Body

Body comprend Header Main Footer

Main comprend des sections

Les sections comprend des articles

Un seul H1 par page

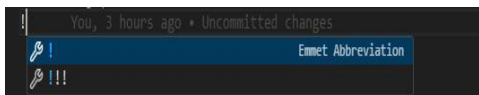
H2 accompagne un article

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="assets/script/main.js" defer></script>
  <link rel="stylesheet" href="assets/styles.css">
  <title>Idriss Balijon - TP CDA - Portfolio - Accueil</title>
</head>

<body>
  <header>
    <div class="logo">...
    </div>
    <nav>...
    </nav>
  </header>
  <main>
    <section id="accueil">
      <h1>Idriss Balijon </br>en Recherche de Stage</h1>
      <article class="bienvenue">...
      </article>
      <article class="consigne">...
      </article>
      <a class="cta" href="realisations.html">Mes réalisations ></a>
    </section>
  </main>
  <footer>...
  </footer>
</body>
</html>
```


6a- Head



Je charge les scripts
selon le besoin.

Title descriptif pour la
SEO

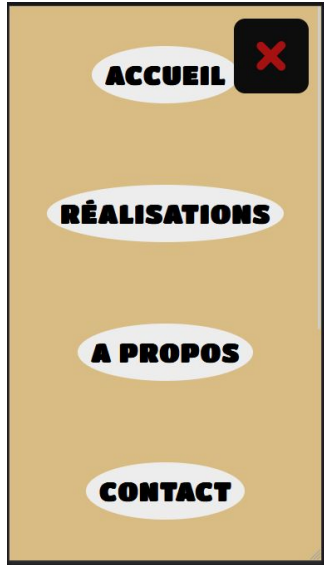
```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="assets/script/main.js" defer></script>
  <script src="assets/script/form.js" defer></script>
  <link rel="stylesheet" href="assets/styles.css">
  <title>Idriss Balijon - TP CDA - Portfolio - Contact</title>
</head>
```

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="assets/script/main.js" defer></script>
  <script src="assets/script/realisation.js" defer></script>
  <link rel="stylesheet" href="assets/styles.css">
  <title>Idriss Balijon - TP CDA - Portfolio - Réalisations</title>
</head>
```

6b- Header



```
<header>
  <div class="logo">
    <a href="index.html"></a>
  </div>
  <nav>
    <menu class="hide">
      <li id="close-menu-burger" class="hide"></li>
      <li><a href="index.html">Accueil</a></li>
      <li><a href="realisations.html">Réalisations</a></li>
      <li><a href="about.html">A propos</a></li>
      <li><a href="contact.html">Contact</a></li>
    </menu>
    <div id="menu-burger"></div>
  </nav>
</header>
```

Des class "hide" sont "toggle" dans le script.

Utilisation de Media query pour le responsive dans le css.

6c - Contact

method et action
vide en webstatique

Div englobe un
champ avec son
label et l'erreur

```
<section id="section-form">
  <h1>Contact</h1>
  <div class="container">
    <form method="POST" action="" id="contact-form">
      <div class="email-container">
        <label for="email" class="email-label">Email</label>
        <input id="email" name="email" value="" class="email-input" placeholder="votre@email.com">
        <p class="error" id="email-error"></p>
      </div>
      <div class="subject-container">
        <label for="subject" class="subject-label">Objet</label>
        <input type="text" id="subject" name="subject" value="" class="subject-input"
          placeholder="Quel est le sujet de votre message ?">
        <p class="error" id="subject-error"></p>
      </div>
      <div class="message-container">
        <label for="message" class="">Votre message</label>
        <textarea id="message" name="message" rows="6" class=""
          placeholder="Laissez votre message."></textarea>
        <p class="error" id="message-error"></p>
      </div>
      <button type="submit" class="">Envoyer</button>
    </form>
  </div>
</section>
```



The image shows a visual rendering of the HTML code provided. It is a contact form titled "Contact" in a red, stylized font. The form is enclosed in a light beige box with a red border. It contains three main sections: "EMAIL" with a text input field containing "votre@email.com", "OBJET" with a text input field containing "Quel est le sujet de votre message", and "VOTRE MESSAGE" with a larger text area containing "Laissez votre message.". Each section has a corresponding error message placeholder below it. At the bottom, there is a "Envoyer" button.

6d- Réalisations

Créations du DOM

“project” est une instance de la Class Project.

Contient les données qu'on a besoin.

Ordre d'exécution importante.

```
<main>
  <section id="projects">
    <h1>Réalisations</h1>
    <div class="realisations-container"></div>
  </section>
  <a class="cta" href="contact.html">Contactez moi ! ></a>
</main>
```



```
function displayProjects(xml) {

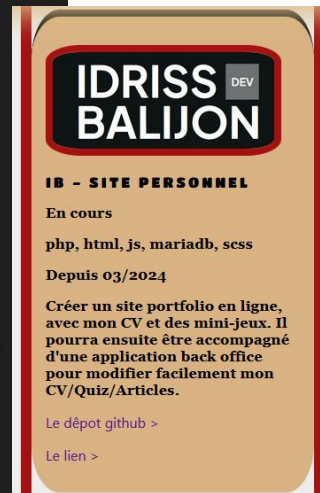
  const projects = createProjects(xml);
  projects.forEach(project => {

    project.element.classList.add(project.titre.replace(/ /g, ''));
    project.element.classList.add('projet');

    createThumbnailOnContainer(project.element, project.thumbnailPath);
    createElementOnContainer("h2", project.element, project.titre);
    createElementOnContainer("p", project.element, project.etat);
    createElementOnContainer("p", project.element, project.stacks);
    createElementOnContainer("p", project.element, project.date);
    createElementOnContainer("p", project.element, project.objectifs);

    const lienRepoElement = createElementOnContainer("a", project.element, 'Le dépôt github >');
    const lienElement = createElementOnContainer("a", project.element, 'Le lien >');
    lienRepoElement.setAttribute('href', project.repo);
    lienElement.setAttribute('href', project.link);

  });
}
```



7- Conception dynamique de la partie réalisations

La class project comprend exactement ce que contient le xml

Utilise une fonction pour récupérer la donné au moment de la création d'instance

A la création, son élément est créé dans le DOM et s'associe au container.

```
<realisations>
  <projet>
    <titre>TP-CDA Webstatique Portfolio</titre>
    <etat>Finis</etat>
    <stacks>html,js,css, xml</stacks>
    <date>06/2024</date>
    <objectifs>Créer un portfolio en webstatique
    <thumbnail>TPCDAPortfolio.jpg</thumbnail>
    <repo>https://github.com/Balidriss/TP-CDA-Balidriss</repo>
    <link>https://balidriss.github.io/TP-CDA-Balidriss</link>
  </projet>
</realisations>
```

```
class Project {
  static container = document.querySelector(".realisations-container");

  constructor(projects, index) {
    this.titre = getProjectcontent(projects, 0, index);
    this.etat = getProjectcontent(projects, 1, index);
    this.stacks = getProjectcontent(projects, 2, index);
    this.date = getProjectcontent(projects, 3, index);
    this.objectifs = getProjectcontent(projects, 4, index);
    this.thumbnailPath = assetsPath("thumbnail") + getProjectcontent(projects, 5, index);
    this.repo = getProjectcontent(projects, 6, index);
    this.link = getProjectcontent(projects, 7, index);
    this.element = createElementOnContainer('div', Project.container);
  }
}
```


7- Conception dynamique de la partie réalisations

```
this.titre = getProjectcontent(projects, 0, index);
```

La fonction cible dans le DOM du xml le contenu associé au paramètre et la retourne.

```
function getProjectcontent(projects, indexDom, index) {  
    return projects[index]['children'][indexDom].textContent;  
}
```

```
function createProjects(xml) {  
    const projects = xml.querySelectorAll("projet");  
    let projectArray = [projects.length];  
    for (let i = 0; i < projects.length; i++) {  
        projectArray[i] = new Project(projects, i);  
    }  
    return projectArray;  
}
```

You, 2 days ago • create project class and fonctions

```
// Fonction pour charger le fichier XML avec fetch  
async function loadXMLDoc(filename) {  
    try {  
        const response = await fetch(filename)  
        if (!response.ok) {  
            throw new error('Problème de chargement.')  
        }  
        const textXml = await response.text();  
        const parser = new DOMParser();  
        const doc = parser.parseFromString(textXml, "text/xml");  
        return doc;  
    } catch (error) {  
        console.error(error);  
    }  
}
```

You, 2 days ago • create project class and fonctions helper

Avec une boucle, je créer des instances de Project.

Créer un DOM à partir du XML est basé du code fait en groupe la veille.

8- La validation du formulaire en javascript

```

</nav>
</menu>
<div id="menu-burger"></div>
</nav>
header
in
<section id="section-form">
  <h1>Contact</h1>
  <div class="container">
    <form method="POST" action="" id="contact-form">
      <div class="email-container">
        <label for="email" class="email-label">Email</label>
        <input id="email" name="email" value="" class="email-input">
        <p class="error" id="email-error"></p>
      </div>
      <div class="subject-container">
        <label for="subject" class="subject-label">Objet</label>
        <input type="text" id="subject" name="subject" value="" class="subject-input">
        <p class="error" id="subject-error"></p>
      </div>
      <div class="message-container">
        <label for="message" class="">Votre message</label>
        <textarea id="message" name="body" rows="6" class="">
        <p class="error" id="message-error"></p>
      </div>
      <button type="submit" class="">Envoyer</button>
    </form>
  </div>

```

```

class FormHandler {
  constructor(formId) {
    this.form = document.getElementById(formId);
    this.fields = this.form.querySelectorAll('input, textarea');
    this.errors = this.form.querySelectorAll('.error');
    this.validators = this.setupValidators();
    this.setupSubmit();
  }

  setupValidators() {
    return {
      email: new Validator('email', { max: 254 }),
      object: new Validator('text', { min: 1, max: 254 }),
      message: new Validator('text', { min: 1, max: 5000 })
    };
  }

  setupSubmit() {
    this.form.addEventListener('submit', (event) => this.handleSubmit(event));
  }

  document.addEventListener('DOMContentLoaded', () => {
    new FormHandler('contact-form');
  });
}

```

La classe est plutôt rigide mais adapté au besoin.

La validation devra tester le nombre de caractère pour chaque input.

Le mail doit être vérifié avec le bon format.

`.setupValidators()` va créer un array d'objets avec les conditions de validation

8- La validation du formulaire en javascript

Possibilité de créer plus de type de validation

“Value” est l’input de l’utilisateur

validateEmail et validateText
ont leur propre façon de tester
la validation.

validate() retourne vraie ou
faux

```
You, 13 seconds ago | 1 author (You)
class Validator {
  constructor(type, conditions) {
    this.type = type;
    this.conditions = conditions;
  }
  validate(value) {
    switch (this.type) {
      case 'email':
        return this.validateEmail(value);
      case 'text':
        return this.validateText(value);
      default:
        return false;
    }
  }
}
```

You, 6 seconds ago • Uncommitt

8- La validation du formulaire en javascript

Remarquez que
validateEmail() appelle
validateText().

conditions est un objet
comportant un min et max
dans notre contexte.

```
validateEmail(value) {  
  if (this.isValidEmail(value)) {  
    return 'Le format de votre mail est incorrect exemple : votre@email.fr';  
  }  
  return this.validateText(value);  
}  
  
validateText(value) {  
  const min = this.conditions.min || 1;  
  const max = this.conditions.max || 254;  
  if (value.length < min) {  
    return `Doit avoir ${min} caractères minimum.`;  
  }  
  if (value.length > max) {  
    return `Doit avoir ${max} caractères maximum.`;  
  }  
  return '';  
}  
  
isValidEmail(email) {  
  
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;  
  return !email.match(emailRegex)  
}
```

8- La validation du formulaire en javascript

```

<form id="section-form">
  <h1>Contact</h1>
  <div class="container">
    <form method="POST" action="" id="contact-form">
      <div class="email-container">
        <label for="email" class="email-label">Email</label>
        <input id="email" name="email" value="" class="email-input" placeholder="votre@email">
        <p class="error" id="email-error"></p>
      </div>
      <div class="subject-container">
        <label for="subject" class="subject-label">Objet</label>
        <input type="text" id="subject" name="subject" value="" class="subject-input"
          placeholder="Quel est le sujet de votre message ?">
        <p class="error" id="subject-error"></p>
      </div>
      <div class="message-container">
        <label for="message" class="">Votre message</label>
        <textarea id="message" name="body" rows="6" class=""
          placeholder="Laissez votre message."></textarea>
        <p class="error" id="message-error"></p>
      </div>
      <button type="submit" class="">Envoyer</button>
    </form>
  </div>
</form>

```

```

46
47
48
49
50   email: new Validator('email', { max: 254 }),
51   subject: new Validator('text', { min: 1, max: 254 }),
52   message: new Validator('text', { min: 1, max: 5000 })
53 };
54
55
56   setupSubmit() {
57     this.form.addEventListener('submit', (event) => this.handleSubmit(event));
58   }
59   handleSubmit(e) {
60     e.preventDefault();
61
62     this.fields.forEach((field) => {
63       const fieldName = field.getAttribute('name');
64       const validator = this.validators[fieldName];
65       const errorElement = this.form.querySelector(`#${fieldName}-error`);
66     });
67   }
68
69

```

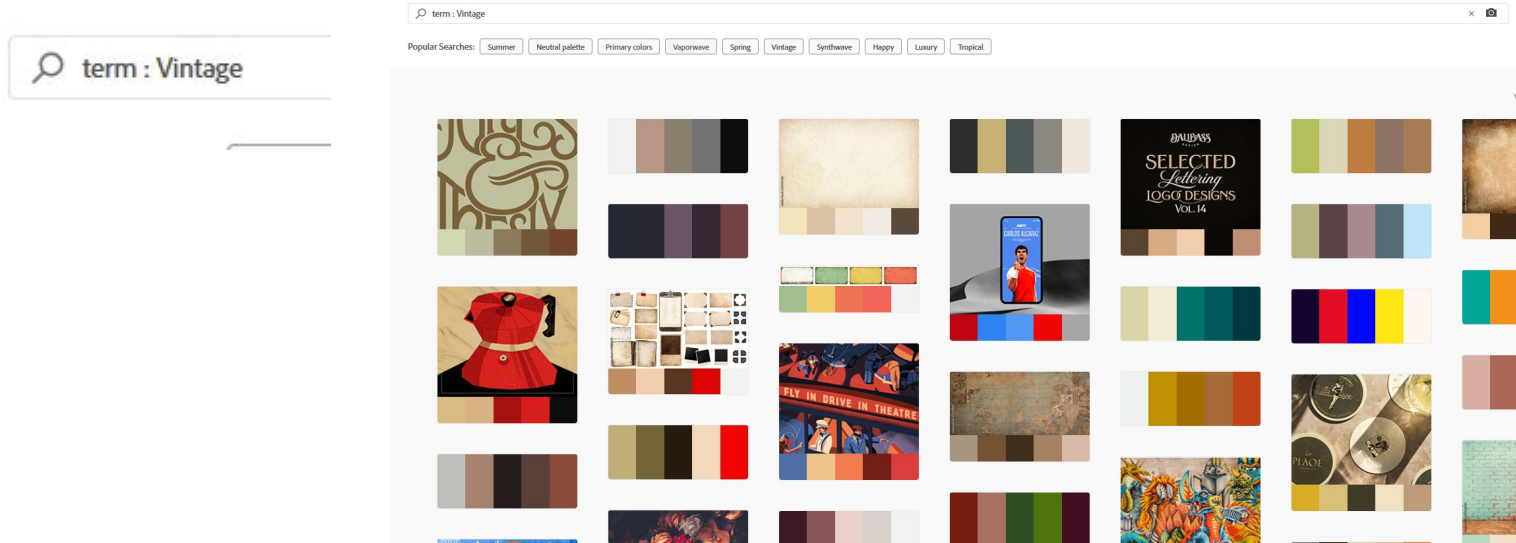
Fields dans notre context comprend input et textarea
On fait attention au nommage pour le fieldName.

8- La validation du formulaire en javascript

Si validator (mail, objet ou message) existe on récupère le texte d'erreur associé.

```
handleSubmit(e) {  
  e.preventDefault();  
  this.fields.forEach((field) => {  
  
    const fieldName = field.getAttribute('name');  
    const validator = this.validators[fieldName];  
    const errorElement = this.form.querySelector(`#${fieldName}-error`);  
  
    if (validator) {  
      const errorMessage = validator.validate(field.value);  
      if (errorMessage) {  
        textOnElement(errorElement, errorMessage);  
      } else {  
        textOnElement(errorElement, '');  
      }  
    }  
  });  
}
```

9- Charte graphique



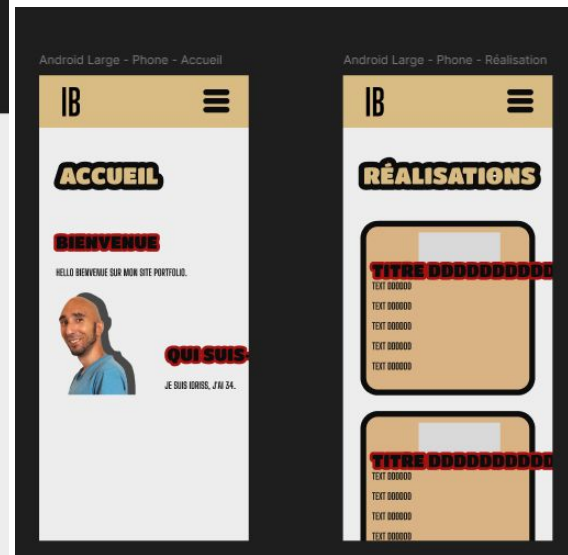
Il était décidé de partir sur du vintage.

10- Maquette

Commencé très tôt dans le projet.

Avoir une taille de police et avoir un visuel sur la partie réalisation

Le client a changé d'avis au dernier moment.



11- Structure CSS

Variable =

La charte graphique.

```
@font-face {  
  font-family: "Titan One";  
  src: url("fonts/TitanOne-Regular.ttf") format("truetype");  
}
```

```
:root {  
  --primaryBeige: #D9BB84;  
  --secondaryBeige: #D9B384;  
  --primaryRed: #A61212;  
  --secondaryRed: #D91E1E;  
  --primaryDark: #0D0D0D;  
  --primaryLight: #EDEDED;  
  --primaryFont: "Titan One";  
}
```

```
li {  
  list-style: none;  
}  
  
body {  
  background-color: var(--primaryLight);  
}
```

```
.cta {  
  display: block;  
  margin-top: -0.5em;  
  padding: 1em;  
  background-color: var(--secondaryRed);  
  border: var(--primaryRed) solid 0.5em;  
  width: fit-content;  
  border-radius: 0 0 0.5em 0.5em;  
  font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;  
  text-decoration: none;  
  color: inherit;  
}
```

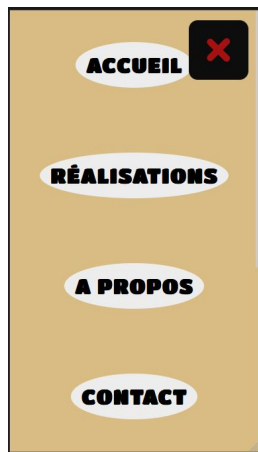
You, 6 hours ago • add cta

```
.cta:hover {  
  background-color: var(--primaryLight);  
}
```

```
* {  
  font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen,  
  font-size: 1em;  
  margin: 0;  
  padding: 0;  
}
```

You, 2 days ago • update styling

11a- Header



```
header {  
  position: fixed;  
  top: 0;  
  left: 0;  
  right: 0;  
  
  display: flex;  
  justify-content: space-between;  
  
  background-color: var(--primaryBeige);  
  
  border-bottom: solid 0.25em var(--primaryDark);  
  box-shadow: 0em 0em 2em 2em var(--primaryBeige);  
}
```

```
@media (min-width: 768px) {  
  .realisations-container {  
    flex-direction: row;  
  }  
  
  #menu-burger {  
    display: none;  
  }  
  
  menu.hide {  
    display: flex;  
    flex-direction: row;  
    position: unset;  
    height: unset;  
    font-size: 1em;  
    text-align: center;  
    width: 100%;  
  }  
  
  #close-menu-burger {  
    display: none;  
  }  
  
  nav {  
    margin-left: auto;  
    align-items: center;  
    width: 60%;  
  }  
  
  menu li {  
    min-width: 200px;  
  }  
}
```

Toggle de “hide” avec du js
pour le menu burger

```
.hide {  
  display: none;  
}
```

```
menu {  
  
  position: absolute;  
  top: 0;  
  left: 0;  
  right: 0;  
  display: flex;  
  flex-direction: column;  
  justify-content: space-around;  
  background-color: var(--primaryBeige);  
  height: 100vh;  
  font-size: 2em;  
  text-align: center;  
}
```

La majorité du responsive s’est
fait dans le header



11b- Réalisations

```
@media (min-width: 768px) {  
  .realisations-container {  
    flex-direction: row;  
  }  
}
```

Display : flex facilite grandement le responsive.



```
.realisations-container {  
  margin: auto;  
  display: flex;  
  flex-wrap: wrap;  
  flex-direction: column;  
  border: 0.75em solid var(--primaryRed);  
  border-radius: 0.5em;  
  background-color: var(--primaryLight);  
}  
  
.projet {  
  border-top: var(--primaryDark) 0.5em solid;  
  /* border-bottom: var(--primaryDark) 0.5em solid; */  
  border-radius: 10%;  
  background-color: var(--secondaryBeige);  
  margin: 1em auto;  
  padding: 1em;  
  max-width: 400px;  
  height: fit-content;  
}  
  
.projet>* {  
  margin: 1em auto;  
}  
  
.projet a {  
  text-decoration: none;  
  display: block;  
}
```


12- Remerciement

Fouziya K. pour l'accompagnement, le cour et d'avoir trouvé l'erreur de @font-face.

Thibault P. pour sa compagnie et son aide.

Human Booster pour la mise en place de la formation

La région pour le financement de la formation.

Merci de votre attention.

Liens

Preview : <https://balidriss.github.io/TP-CDA-Balijon-Webstatique-portfolio/>

Dépôt : <https://github.com/Balidriss/TP-CDA-Balijon-Webstatique-portfolio>

Maquette Figma obsolete :

<https://www.figma.com/proto/WQXHXftw725nzeZ8gaAA/Maquette?node-id=1-2&t=Gna35C9BkHMqip3n-0&scaling=scale-down&page-id=0%3A1&starting-point-node-id=1%3A2>

13- Pedro ?



13- Avez-vous des questions ?