

# Justificatif SAÉ 1.05 : Produire un site Web

**AC14.01 | Exploiter de manière autonome un environnement de développement efficace et productif**

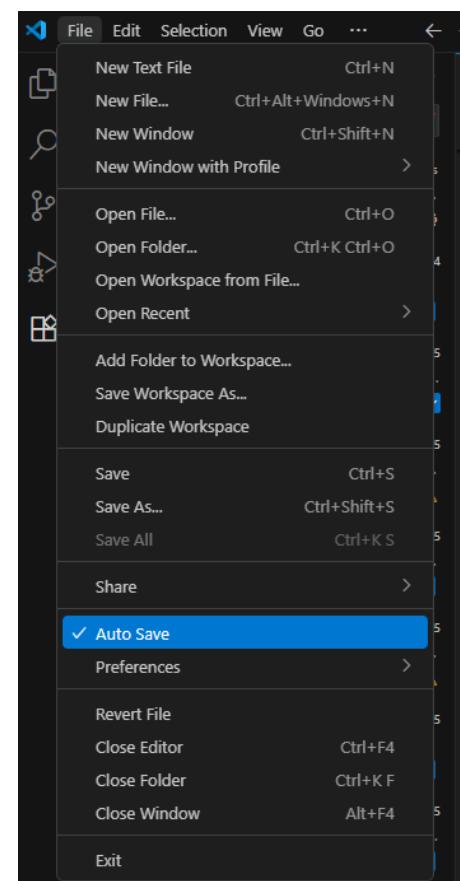
## 1.1 Utiliser des outils de débogage

Le logiciel Visual Studio Code permet d'optimiser le temps de développement grâce à:

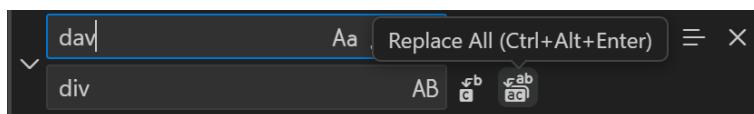


- L'Extension Live Server qui permet d'héberger localement depuis son PC et donc combiner à l'Auto-Save de voir en temps réel les changements de la page en fonction du code modifié.

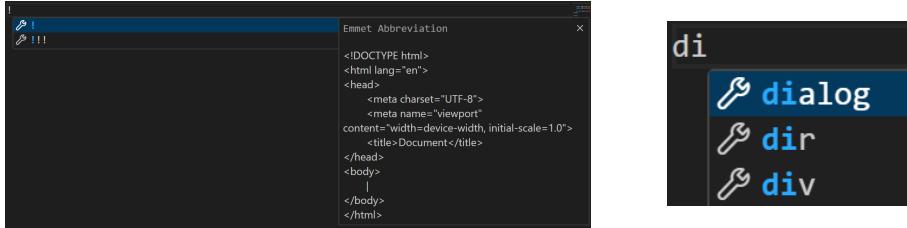
A screenshot of the Visual Studio Code interface. On the left, there's a code editor window displaying CSS code for a file named '#styleMain&SAE.css'. The code includes styles for an h1 element and a #GeoStar element. On the right, there's a 'Live Server' panel with a purple icon, the text 'Launch a development local Server ...', and the name 'Ritwick Dey'. At the bottom of the screen, there's a status bar showing 'In 65, Col 5 Spaces:4 UTF-8 CRLF () CSS' and a 'Go Live' button with a circular arrow icon, which is circled in red.



- (Ctrl+F) La recherche de caractères et la possibilité de les remplacer. Évitant la tâche fastidieuse de remplacer du code mal syntaxer.



- Suggestion automatique des balises et génération complète d'un template HTML via “!”



- Raccourcis appliquer une indenter automatiquement (Shift+Alt+F) et des commentaires en fonction du format de fichier (Ctrl+/).

## AC14.02 | Produire des pages Web fluides incluant un balisage sémantique efficace et des interactions simples

### 2.1 - Employer les bonnes balises

Les balises servent à hiérarchiser les textes, introduire des titres, des images, des vidéos, des liens, ainsi que tout autre élément servant à la mise en forme du texte. Dès lors, il est important de bien les employer afin de faciliter la lecture du code. Cela fait partie des bonnes pratiques.

Par exemple, le `<main>` sert pour le contenu principal tandis que le `<footer>` sert plutôt pour des infos ou options externes.

```
<main id="principal"> ...</main>
```

```
<Footer>
  <!-- OPTION MUSIC -->
  <select name="Audio" id="Audio" onchange="audio()">
    <option value="false">None Music</option>
  </select>

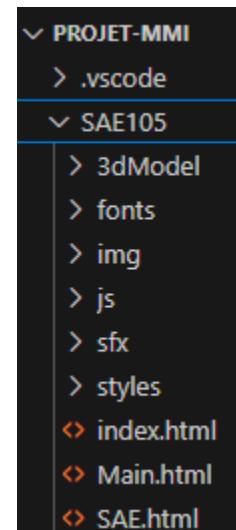
  <!-- OPTION SEMESTRE -->
  <select name="Semestre" id="Semestre" onchange="switchSemester()">
    <option value="S1">SEMESTRE 1</option>
    <option value="S2">SEMESTRE 2</option>
    <option value="S3">SEMESTRE 3</option>
    <option value="S4">SEMESTRE 4</option>
    <option value="S5">SEMESTRE 5</option>
    <option value="S6">SEMESTRE 6</option>
  </select>
</Footer>
```

### 2.2 - Utiliser la bonne arborescence

Il est important de bien structurer ses dossiers afin de ne pas se perdre. En effet, sur un site web, le nombre de fichiers peut varier d'une dizaine (encore gérable), à des milliers (un cauchemar).

Pour faire le tri hiérarchiquement, on les classe en fonction de leurs rôles. Les fichiers HTML (page web) sont à la racine comme base. Tous les autres composants qui servent à les compléter de façon interactive, esthétique et de n'importe quelle façon sont rangés dans des dossiers.

*Voir exemple ci-contre*



## 2.3 - Mettre en place une bonne indentation et des commentaires pertinents

Une bonne indentation sert à hiérarchiser des éléments, cela est très utile par exemple pour voir comment agencer et aligner des éléments d'une page. Dans l'exemple ci-dessous, on voit que les div sont emboîtés de façon à aligner sur 2 lignes et 3 colonnes 6 éléments.

```
<div id="flexSAE">
  <div class="RowSAE">
    <div class="SAE" onclick="SAEClick('SAE1.01')">SAE1.01</div>
    <div class="SAE" onclick="SAEClick('SAE1.02')">SAE1.02</div>
    <div class="SAE" onclick="SAEClick('SAE1.03')">SAE1.03</div>
  </div>
  <div class="RowSAE">
    <div class="SAE" onclick="SAEClick('SAE1.04')">SAE1.04</div>
    <div class="SAE" onclick="SAEClick('SAE1.05')">SAE1.05</div>
    <div class="SAE" onclick="SAEClick('SAE1.06')">SAE1.06</div>
  </div>
</div>
```

Par ailleurs, les commentaires sont tout aussi importants, ils permettent de se repérer, se souvenir, expliquer et séparer des zones dans des centaines de lignes de codes.

```
<!-- ----- SECTION META ----->

<div id="annoying-dog-undertale">
  
  <audio id="dogSFX" src="sfx/snd_catsalad.wav"></audio>
</div>

<div id="ListeAudio">
  <audio id="audio0" src="sfx/Music/Its_Raining_Somewhere_Else-Toby_Fox.mp3"></audio>
</div>
```

```
<!-- ----- FENETRE POP-UP ----->

<section id="error">
  <header>
    <div>Erreur</div>
    <div>X</div>
  </header>
  <div class="insideBox">
    <div>:(</div>
  </div>
</section>
```

```
<!-- ----- LOGIN ----->

<section id="intro">
  <div id="heure">24:00</div>
  <div id="date">Mercredi 07 janvier</div>
  
</section>

<div id="loginBtn">
  Login
</div>
</section>
```

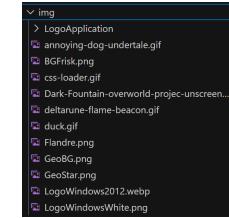
## 2.4 - Choisir de manière judicieuse l'architecture du site et son UX

La structure des pages s'organisent de la façon suivante :  
arborescence : Accueil → Menu de sélection → SAE

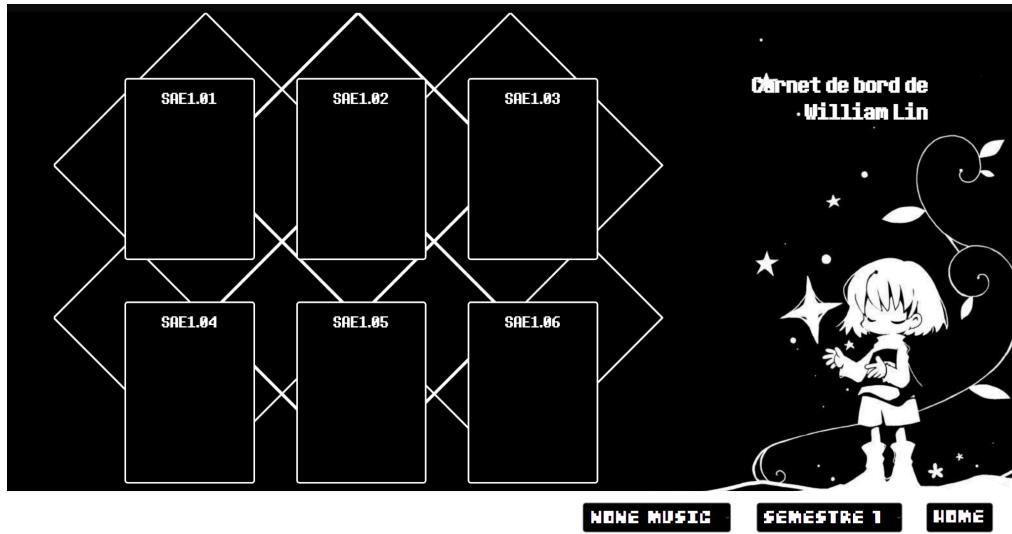
```
<-- index.html  
<-- Main.html  
<-- SAE.html
```

La navigation est composée de liens relatifs à des fichiers. Aucun lien externe n'est présent afin d'assurer le fonctionnement du site, même hors connexion.

La hiérarchie de l'information met en avant le contenu dans la page. Tout ce qui est essentiel y est présent. Le footer est l'endroit où toutes les options sont présentes.



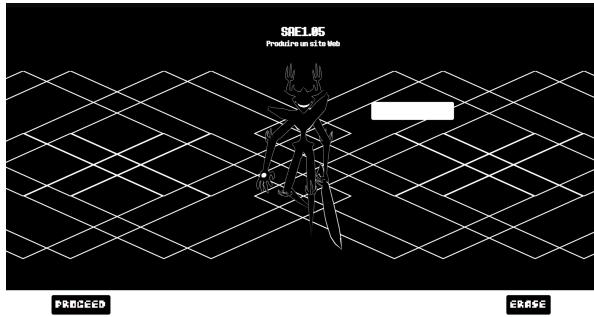
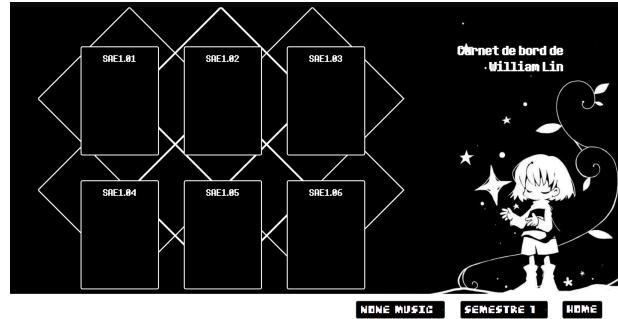
La logique de circulation se base simplement sur une navigation guidée de l'accueil au menu. Puis par une simple sélection par clique de l'information voulue. Les options sont limitées par l'interface qui est assez simple afin de réduire les efforts cognitifs.



En termes d'ergonomie, c'est assez simple et intuitif à utiliser. On comprend clairement le but du site (à partir du menu). Le site ne comporte pas de chargement long car peu lourd (excepté les transitions qui prennent du temps mais volontaire afin de ne pas perturber les micro chargements). Le site est accessible de n'importe quel navigateur. Le design fonctionnel fait en sorte que la recherche est agréable. Le principal est dans la page tandis que les options sont dans le footer. Rien d'autre. Les interactions sont multiples. Il y a des sélecteurs avec transition et boutons fonctionnels qui rendent le site très interactif.

## 2.5 - UI

L'aspect esthétique est majoritairement subjectif.



## AC14.03 | Générer des pages Web à partir de données structurées

### 3.1 - Utiliser une structure JSON pour remplir les templates

Une structure JSON adaptée en JS fonctionne comme un objet. C'est-à-dire que pour accéder à une **valeur** spécifique, il faut une **clé**.

```
const SAE_Data = {
  "SAE1.01": {
    "titre": "Auditer une communication numerique",
    "competences": ["Comprendre"],
    "description": "Cette SAE doit amener les etudiants a utiliser des outils d'audit objectifs",
    "AC": {
      "AC11.01": "Presenter une organisation, ses activites et son environnement (economique, so",
      "AC11.02": "evaluer un site web, un produit multimedia ou un dispositif interactif existan",
      "AC11.03": "Produire des analyses statistiques descriptives et les interpreter pour evaluer",
      "AC11.04": "Analyser des formes mediatiques et leur semiotique",
      "AC11.05": "Identifier les cibles (critères socio-economiques, demographiques, geographiqu"
    },
    "ressources": {
      "R1.03": "Ergonomie et Accessibilite",
      "R1.04": "Culture numerique",
      "R1.05": "Strategies de communication et marketing",
      "R1.09": "Culture artistique",
      "R1.14": "Representation et traitement de l'information",
      "R1.16": "economie, gestion et droit du numerique"
    },
    "semestre": 1
  }
},
```

Donc dans le cas d'une SAE, il faut récupérer un paramètre X lorsqu'on clique sur une SAE, c'est la clé pour accéder aux données de la SAE (titre, compétences, description, AC, ressources, semestre) et l'envoyer via l'URL sur une autre page afin de pouvoir utiliser cette clé pour compléter un template de façon dynamique de cette façon :

```
function SAEClick(X) {
    window.open("SAE.html?msg="+X, "_blank")
}
```

Une fois envoyé sur la nouvelle page, il faut enregistrer dans une variable cette clé (N° SAE) afin de pouvoir l'utiliser comme une clé d'objet.

```
const params = new URLSearchParams(window.location.search);
const msg = params.get("msg");
```

Cependant, on peut déjà remplir les éléments de la page via un `document.querySelector(...).innerText/HTML` mais pas pour les AC et ressources. Ce sont toujours des clés à utiliser pour accéder à des valeurs. Donc pour résoudre ce problème, il faut stocker les différentes clés dans des variables AC et ressource puisqu'elles ne sont pas les mêmes que dans les autres SAE et donc pas réutilisables. Par exemple, la clé titre est constante alors que AC11.01 est unique.

A l'aide d'une boucle forEach, on stocke les valeurs des AC et ressource dans des variables pour les ajouter en HTML plus tard.

```
var AC = Object.keys(SAE_Data[msg]["AC"])
var ressource = Object.keys(SAE_Data[msg]["ressources"]);
ListeAC = "<div>Apprentissage critique</div>";
ListeRessource = "<div>Ressource</div>"

AC.forEach(function(i,j){
    ListeAC += "<div>" + i + " : " + SAE_Data[msg]["AC"][i] + "</div>"
})
ressource.forEach(function(i,j){
    ListeRessource += "<div>" + i + " : " + SAE_Data[msg]["ressources"][i] + "</div>"
})
```

Enfin, dans une div qui sert de réceptacle, avec un gabarit on ajoute chaque éléments sur la page pour enfin finaliser le template par rapport à la clé initial qui a ouvert cette page.

```
const stats = document.getElementById("stats");
var statistique = `<div>Description : ${SAE_Data[msg]["description"]}<br></div>
<div>Competence(s) : ${SAE_Data[msg]["competences"]}</div>
${ListeAC}
${ListeRessource}
`;
```

```

title.innerText = msg;
subTitle.innerText = SAE_Data[msg]["titre"];
stats.innerHTML = statistique;

```

### 3.2 - Utiliser une méthode pertinente pour afficher les AC

A partir du template, on cherche à afficher les AC d'une SAE via un fichier PDF dans le navigateur. Pour cela, par rapport au code précédent en JS, on va ajouter la méthode onclick pour activer une fonction qui servira à ouvrir le justificatif de la SAE incluant toutes les AC.

```

AC.forEach(function(i,j){
    ListeAC += "<div id='ListeAC' onclick='justificatif()' " + i + " : " + SAE_Data[msg]["AC"][i]
})

```

Donc lorsqu'on va cliquer sur une AC, cela va ouvrir le PDF en utilisant la variable de l'URL contenant le numéro de la SAE. Les paramètres de la commande sont le chemin d'accès et la propriété "\_blank" qui indique que c'est dans une nouvelle page qu'il faut ouvrir le PDF.

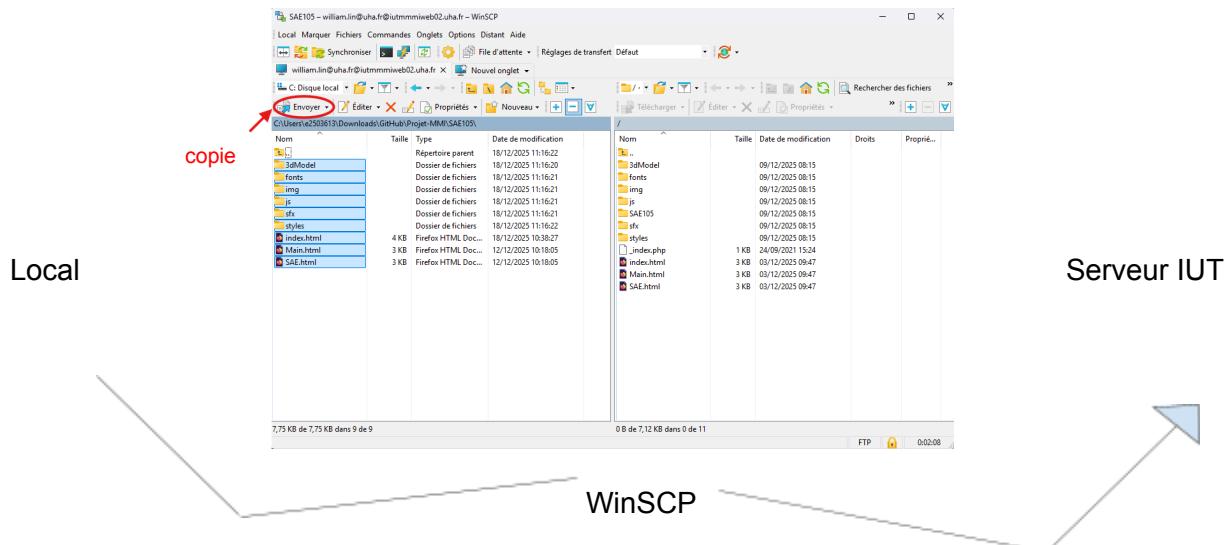
```

function justificatif() {
    window.open("justificatif/" + msg + ".pdf", "_blank")
}

```

## AC14.04 | Mettre en ligne une application Web en utilisant une solution d'hébergement standard

### 4.1 - Transférer vers un serveur distant



Le logiciel WinSCP fait office de passerelle. Il copie les dossiers et fichiers d'un appareil local et les envoie sur les serveurs de l'IUT dans un emplacement dédié. Ensuite, il suffit juste d'entrer le nom de domaine de cet emplacement sur un ordinateur connecté aux serveurs de l'IUT pour y accéder. Ce nom de domaine fait référence à l'adresse IP du HostID. Pour résumer, sur 32 bits (4 octets), en fonction du masque de sous-réseau, une adresse IP est liée à un appareil. Donc l'accès à cette adresse permet l'accès aux données d'un appareil.