Lucas Alazary

Introduction
1.1 Objectif du rapport
1.2 Contexte de l'entreprise

Conception du site web
2.1 Choix des langages (HTML, CSS, Java)

Structure du code
3.1 Explication des fichiers HTML
3.2 Explication des fichiers CSS
3.3 Explication des fichiers JavaScript

Annexes
4.1 Capture d'écran du site web

Introduction

1.1 Objectif du rapport :
Le présent rapport vise à fournir une analyse détaillée du code du site web de l'entreprise Galaxy Swiss Bourdin (GSB) et à expliquer sa conception, ses fonctionnalités et son implémentation. L'objectif est de comprendre en profondeur le travail réalisé et d'évaluer l'efficacité du site web dans le contexte de l'entreprise.

1.2 Contexte de l'entreprise :
Le laboratoire Galaxy Swiss Bourdin (GSB) est le fruit d'une fusion entre le géant américain Galaxy, spécialisé dans le traitement des maladies virales telles que le SIDA et les hépatites, et le conglomérat européen Swiss Bourdin, qui se concentre sur les médicaments plus conventionnels. Cette fusion, survenue en 2009, a créé un leader dans l'industrie pharmaceutique.

L'industrie pharmaceutique est un secteur extrêmement lucratif, caractérisé par un mouvement intensif de fusion-acquisition. Au fil des années, de nombreuses entreprises pharmaceutiques se sont regroupées, donnant naissance à des entités gigantesques. Cependant, certaines controverses récentes autour de médicaments ou de molécules ayant entraîné des complications médicales ont mis en lumière des pratiques douteuses, notamment dans le domaine de la visite médicale.

GSB a entrepris une réorganisation pour optimiser ses activités, réaliser des économies d'échelle dans la production et la distribution des médicaments, et tirer parti des meilleures pratiques des deux laboratoires fusionnés. Cette réorganisation a nécessité une restructuration interne et une vague de licenciements. L'entreprise compte actuellement 480 visiteurs médicaux en France métropolitaine, ainsi que 60 visiteurs médicaux dans les départements et territoires d'outre-mer. Ces territoires sont répartis en six secteurs géographiques.

Conception du site web

2.1 Choix des langages (HTML, CSS, JavaScript) :
Pour la conception du site web de Galaxy Swiss Bourdin (GSB), les langages principalement utilisés sont les suivants :

HTML (HyperText Markup Language) : HTML est le langage de balisage utilisé pour structurer et organiser le contenu du site web. Il permet de définir la structure des pages, les titres, les paragraphes, les liens, les images, etc.

CSS (Cascading Style Sheets) : CSS est utilisé pour définir la présentation et le style du site web. Il permet de contrôler l'apparence des éléments HTML tels que les couleurs, les polices, les marges, les bordures, etc. CSS permet d'appliquer une mise en page cohérente à l'ensemble du site.

JavaScript : JavaScript est un langage de programmation utilisé pour ajouter des fonctionnalités interactives et dynamiques au site web. Il permet d'effectuer des actions côté client, telles que la validation de formulaires, les animations, les interactions avec l'utilisateur, etc. JavaScript est également utilisé pour interagir avec des services web et récupérer des données en temps réel.

Ces trois langages (HTML, CSS, JavaScript) sont couramment utilisés ensemble pour concevoir des sites web interactifs et attrayants.

Structure du code

3.1 Explication des fichiers HTML :

```
  <div class="chat-wrapper">
    <div class="user-tabs">
<ul>
  <li class="active" onclick="switchUser('Tous')" data-user="Tous"><img src="Image/groupe.png">Tous</li>
  <li onclick="switchUser('Damien')" data-user="Damien">
    <img src="Image/user1.png" alt="Utilisateur 1">
    <span>Damien</span>
  </li>
  <li onclick="switchUser('Julien')" data-user="Julien">
    <img src="Image/user2.png" alt="Utilisateur 2">
    <span>Julien</span>
  </li>
  <li onclick="switchUser('Marion')" data-user="Marion">
    <img src="Image/user3.png" alt="Utilisateur 3">
    <span>Marion</span>

  </li>
</ul>
</div>
```

Ce code représente la partie de l'application de chat qui concerne les onglets d'utilisateurs :

 : Cette balise div avec la classe "chat-wrapper" englobe tout le contenu lié à l'application de chat.
 : Cette balise div avec la classe "user-tabs" représente la section des onglets d'utilisateurs.
 : La balise ul crée une liste non ordonnée pour les onglets d'utilisateurs.
 : Les balises li représentent les éléments de la liste, qui correspondent à chaque onglet d'utilisateur.
class="active" : Cette classe est ajoutée au premier élément li pour indiquer qu'il s'agit de l'onglet actif par défaut.

onclick="switchUser('Tous')" : Lorsque l'onglet est cliqué, cette fonction JavaScript "switchUser" est appelée avec le paramètre 'Tous'. Cela permet de changer d'utilisateur.

data-user="Tous" : Cet attribut data est utilisé pour stocker la valeur de l'utilisateur associé à l'onglet. Dans cet exemple, 'Tous' est l'utilisateur associé à l'onglet.

<img src="Image/groupe.png"> : Cette balise img affiche une image représentant un groupe. Dans cet exemple, l'image est appelée "groupe.png" et est située dans le répertoire "Image".

Tous : Ce texte "Tous" est affiché à côté de l'image de l'onglet.

onclick="switchUser('Damien')" : Lorsque l'onglet est cliqué, la fonction "switchUser" est appelée avec le paramètre 'Damien'.

data-user="Damien" : Cet attribut data stocke la valeur 'Damien' en tant qu'utilisateur associé à l'onglet.

<img src="Image/user1.png" alt="Utilisateur 1"> : Cette balise img affiche une image représentant le premier utilisateur. L'image est appelée "user1.png" et est située dans le répertoire "Image". L'attribut alt fournit un texte alternatif pour l'image, qui est affiché si l'image ne peut pas être chargée.

Damien : Cette balise span contient le nom de l'utilisateur Damien.

```
        <div class="chat-container">
            <div class="chat-header">
                <h2>Tchat</h2>
            </div>
            <div class="chat-messages" id="chat-messages">
            </div>
            <div class="chat-input">
                <input type="text" id="message-input" placeholder="Tapez votre message...">
                <button onclick="sendMessage()">Envoyer</button>
            </div>
        </div>
    </div>

    <script src="script.js"></script>
```

3.2 Explication des fichiers CSS :

```css
body, html {
    font-family: 'Kantumruy Pro', sans-serif;
    font-family: 'Spline Sans Mono', monospace;
    height: 100%;
    margin: 0;
    padding: 0;
    background-image: url('Image/Fond.jpg');
    background-size: cover;
}

.chat-wrapper {
    display: flex;
    align-items: center;
    justify-content: center;
    height: 100%;
    padding: 20px;
}

.user-tabs {
    width: 600px;
    height: 500px;
    background-color: #f0f0f0a0;
    border: 1px solid #1b1313;
    border-radius: 50px;
    padding: 10px;
    margin-right: 50px;
}

.user-tabs ul {
    list-style: none;
    padding: 0;
    margin: 0;
}

.user-tabs li {
    align-items: center;
    padding: 10px 10px;
    cursor: pointer;
    margin-bottom: 5px;
}

.user-tabs li.active {
    background-color: #616161;
    border: 1px solid #616161;
    border-radius: 50px;

}
```

Ce code représente un ensemble de règles de style CSS qui est utilisé pour styliser notre chat :

body, html: Cette règle de style s'applique à la balise body et html de la page HTML et définit plusieurs propriétés :

font-family: Définit la police de caractères utilisée pour le texte à l'intérieur des balises body et html.
height: Définit la hauteur de la balise body et html à 100% de la hauteur de l'écran.
margin et padding: Définissent les marges et les espacements internes de la balise body et html.
background-image: Définit une image de fond pour la page en utilisant l'URL "Image/Fond.jpg".
background-size: Définit la taille de l'image de fond pour couvrir toute la surface de la balise body et html.
.chat-wrapper: Cette règle de style s'applique à la classe "chat-wrapper" et définit plusieurs propriétés :

display: flex: Définit le type d'affichage de la balise avec la classe "chat-wrapper" en tant que conteneur flexible.
align-items: center: Centre les éléments horizontalement à l'intérieur du conteneur flexible.
justify-content: center: Centre les éléments verticalement à l'intérieur du conteneur flexible.
height: Définit la hauteur du conteneur à 100% de la hauteur de son conteneur parent.
padding: Définit les espacements internes du conteneur.
.user-tabs: Cette règle de style s'applique à la classe "user-tabs" qui représente la section des onglets d'utilisateurs. Elle définit plusieurs propriétés :

width: Définit la largeur de la section des onglets d'utilisateurs à 600 pixels.
height: Définit la hauteur de la section des onglets d'utilisateurs à 500 pixels.
background-color: Définit la couleur de fond de la section des onglets d'utilisateurs.
border: Définit une bordure de 1 pixel avec la couleur spécifiée.
border-radius: Définit le rayon de courbure des coins de la section des onglets d'utilisateurs.
padding: Définit les espacements internes de la section des onglets d'utilisateurs.
margin-right: Définit la marge à droite de la section des onglets d'utilisateurs.
.user-tabs ul: Cette règle de style s'applique aux listes non ordonnées (ul) à l'intérieur de la classe "user-tabs". Elle définit plusieurs propriétés :

list-style: Définit le style de la liste comme étant sans puce.
padding et margin: Définissent les marges et les espacements internes de la liste.
.user-tabs li: Cette règle de style s'applique aux éléments de liste (li) à l'intérieur de la classe "user-tabs". Elle définit plusieurs propriétés :

align-items: center: Centre les éléments verticalement à l'intérieur de chaque élément de liste.
padding: Définit les espacements internes des éléments de liste.
cursor: pointer: Change le curseur de la souris lorsqu'il survole les éléments de liste pour indiquer qu'ils sont cliquables.
margin-bottom: Définit la marge en bas de chaque élément de liste.
.user-tabs li.active: Cette règle de style s'applique aux éléments de liste (li) qui ont la classe "active" à l'intérieur de la classe "user-tabs". Elle définit plusieurs propriétés :

background-color: Définit la couleur de fond des éléments de liste actifs.
border: Définit une bordure de 1 pixel avec la couleur spécifiée pour les éléments de liste actifs.
border-radius: Définit le rayon de courbure des coins des éléments de liste actifs.

```css
.user-tabs ul li {
    display: flex;
    align-items: center;
    gap: 30px;
  }

  .user-tabs ul li img {
    width: 50px;
    height: 50px;
    border-radius: 50%;
  }

.chat-container {
    width: 500px;
    border: 1px solid #ebebeb;
    border-radius: 50px;
    padding: 10px;
}

.chat-header {
    color: #ffffff;
    text-align: center;
    margin-bottom: 10px;
}

.chat-messages {
    word-wrap: break-word;
    color: #ffffff;
    height: 300px;
    border: 0px solid #ccc;
    overflow-y: scroll;
    padding: 5px;
}

.chat-input {
    border: 15px solid #616161;
    border-radius: 30px;
    display: flex;
    margin-top: 10px;
}

.chat-input input[type="text"] {
    flex: 1;
    padding: 1px;
}
```

.user-tabs ul li: Cette règle de style s'applique aux éléments de liste (li) à l'intérieur des listes non ordonnées (ul) à ll'intérieur de la classe "user-tabs". Elle définit plusieurs propriétés :

display: flex: Définit le type d'affichage des éléments de liste en tant que conteneur flexible.
align-items: center: Centre les éléments horizontalement à l'intérieur de chaque élément de liste.
gap: 30px: Définit un espacement horizontal de 30 pixels entre les éléments de liste.
.user-tabs ul li img: Cette règle de style s'applique aux images (img) à l'intérieur des éléments de liste dans la classe "user-tabs". Elle définit plusieurs propriétés :

width: Définit la largeur des images à 50 pixels.
height: Définit la hauteur des images à 50 pixels.
border-radius: Définit le rayon de courbure des coins des images pour obtenir une forme de cercle.
.chat-container: Cette règle de style s'applique à la classe "chat-container" qui représente le conteneur principal des messages de chat. Elle définit plusieurs propriétés :

width: Définit la largeur du conteneur des messages de chat à 500 pixels.
border: Définit une bordure de 1 pixel avec la couleur spécifiée pour le conteneur des messages de chat.
border-radius: Définit le rayon de courbure des coins du conteneur des messages de chat.
padding: Définit les espacements internes du conteneur des messages de chat.
.chat-header: Cette règle de style s'applique à la classe "chat-header" qui représente l'en-tête du chat. Elle définit plusieurs propriétés :

color: Définit la couleur du texte de l'en-tête du chat.
text-align: Définit l'alignement du contenu de l'en-tête du chat au centre.
margin-bottom: Définit la marge en bas de l'en-tête du chat.
.chat-messages: Cette règle de style s'applique à la classe "chat-messages" qui représente la zone où s'affichent les messages du chat. Elle définit plusieurs propriétés :

word-wrap: Définit le comportement de retour à la ligne des mots longs à l'intérieur de la zone des messages de chat.
color: Définit la couleur du texte à l'intérieur de la zone des messages de chat.
height: Définit la hauteur de la zone des messages de chat à 300 pixels.
border: Définit une bordure de 0 pixel avec la couleur spécifiée pour la zone des messages de chat.
overflow-y: Définit le comportement de défilement vertical lorsque le contenu dépasse la hauteur définie.
padding: Définit les espacements internes de la zone des messages de chat.
.chat-input: Cette règle de style s'applique à la classe "chat-input" qui représente la section de saisie de texte et de bouton d'envoi du chat. Elle définit plusieurs propriétés :

border: Définit une bordure de 15 pixels avec la couleur spécifiée pour la section de saisie de texte et de bouton d'envoi.
border-radius: Définit le rayon de courbure des coins de la section de saisie de texte et de bouton d'envoi.
display: flex: Définit le type d'affichage de la section de saisie de texte et de bouton d'envoi en tant que conteneur flexible.
margin-top: Définit la marge en haut de la section de saisie de texte et de bouton d'envoi.

.chat-input input[type="text"]: Cette règle de style s'applique à l'élément de saisie de texte (input) de type "text" à l'intérieur de la classe "chat-input". Elle définit plusieurs propriétés :

flex: 1: Définit la flexibilité de l'élément de saisie de texte pour occuper tout l'espace disponible horizontalement.
padding: Définit les espacements internes de l'élément de saisie de texte.

```
.chat-input button {
    padding: 10px 10px;
    cursor: pointer;

}

.active-user-button {
    background-color: #ccc;

}
```

Le premier bloc de règles de style s'applique à un bouton à l'intérieur de l'élément avec la classe "chat-input". Voici une explication des différentes propriétés :

padding: 10px 10px;: Cette propriété définit les espacements internes du bouton. Les valeurs "10px 10px" indiquent que le bouton aura un espacement de 10 pixels en haut et en bas, ainsi qu'un espacement de 10 pixels à gauche et à droite.

cursor: pointer;: Cette propriété définit le curseur de la souris lorsqu'il survole le bouton. En utilisant la valeur "pointer", le curseur prendra la forme d'une main, indiquant qu'il est cliquable.

Le deuxième bloc de règles de style s'applique aux éléments qui ont la classe "active-user-button". Voici une explication de la propriété :

background-color: #ccc;: Cette propriété définit la couleur de fond des éléments ayant la classe "active-user-button". La valeur "#ccc" représente une nuance de gris définie en utilisant le code hexadécimal. Dans ce cas, la couleur de fond sera un gris clair.

3.3 Explication des fichiers JavaScript :

```
var activeUser = "Tous";
var userCount = 1;

function switchUser(user) {
    activeUser = user;
    var userTabs = document.querySelectorAll(".user-tabs li");
    userTabs.forEach(function(tab) {
        tab.classList.remove("active");
    });
    document.querySelector(".user-tabs li[data-user='" + user + "']").classList.add("active");

    userTabs.forEach(function(tab) {
        tab.classList.remove("active-user-button");
    });

    document.querySelector(".user-tabs li[data-user='" + user + "']").classList.add("active-user-button");

}
```

Fonction:
switchUser(user) est une fonction qui prend un paramètre user. Cette fonction est appelée pour changer l'utilisateur actif. Elle effectue les actions suivantes :
Met à jour la variable activeUser avec la valeur du paramètre user, ce qui permet de mettre à jour l'utilisateur actif.
Sélectionne tous les éléments <li> qui sont enfants de l'élément avec la classe "user-tabs" à l'aide de la méthode document.querySelectorAll(). Ces éléments représentent les onglets des utilisateurs.
Parcourt chaque élément <li> à l'aide de la méthode forEach().
Supprime la classe "active" de chaque élément <li> en utilisant la méthode classList.remove("active"). Cela permet de désactiver tous les onglets des utilisateurs.
Sélectionne l'élément <li> qui a un attribut data-user égal à la valeur du paramètre user en utilisant la méthode document.querySelector(). Cet élément représente l'onglet de l'utilisateur spécifié par le paramètre user.
Ajoute la classe "active" à l'élément <li> sélectionné en utilisant la méthode classList.add("active"). Cela permet d'activer l'onglet de l'utilisateur spécifié.
Répète les étapes précédentes en remplaçant la classe "active-user-button" par la classe "active". Ainsi, l'onglet de l'utilisateur spécifié aura également la classe "active-user-button".

```
function createNewTab(user) {
    var userTabsList = document.getElementById("user-tabs-list");
    var newTab = document.createElement("li");
    newTab.innerText = "Utilisateur " + userCount;
    newTab.setAttribute("data-user", "user" + userCount);
    newTab.onclick = function() {
        switchUser("user" + userCount);
    };
    userTabsList.appendChild(newTab);
    userCount++;
}
```

La variable userTabsList est déclarée et initialisée avec l'élément qui a l'ID "user-tabs-list". Cet élément représente la liste des onglets d'utilisateurs où le nouvel onglet sera ajouté.
Création de l'élément <li> :

La variable newTab est déclarée et initialisée avec un nouvel élément <li> créé à l'aide de la méthode document.createElement("li"). Cet élément représente le nouvel onglet d'utilisateur.
Définition du texte de l'onglet :

La propriété innerText de l'élément newTab est définie avec la valeur "Utilisateur " concaténée avec la valeur de la variable userCount. Cela permet de donner un nom unique à chaque nouvel onglet d'utilisateur.
Attribution de l'attribut data-user :

L'attribut data-user de l'élément newTab est défini avec la valeur "user" concaténée avec la valeur de la variable userCount. Cela permet d'identifier l'utilisateur associé à cet onglet.
Définition de l'événement onclick :

Lorsque l'onglet est cliqué, la fonction anonyme function() {...} est exécutée. Cette fonction appelle la fonction switchUser() avec le paramètre "user" concaténé avec la valeur de la variable userCount. Cela permet de passer le nom d'utilisateur associé à l'onglet en tant que paramètre à la fonction switchUser().
Ajout de l'onglet à la liste parente :

L'élément newTab est ajouté à la liste des onglets d'utilisateurs représentée par l'élément userTabsList en utilisant la méthode appendChild().
Incrémentation du compteur d'utilisateurs :

La variable userCount est incrémentée pour assurer que chaque nouvel onglet d'utilisateur a un identifiant unique.

```
function sendMessage() {
    var messageInput = document.getElementById("message-input");
    var message = messageInput.value;

    if (message.trim() !== "") {
        var chatMessages = document.getElementById("chat-messages");
        var messageElement = document.createElement("div");
        messageElement.innerText = "[" + activeUser + "] " + message;
        chatMessages.appendChild(messageElement);

        messageInput.value = "";
    }
}
```

Récupération de l'élément d'entrée du message :

La variable messageInput est déclarée et initialisée avec l'élément qui a l'ID "message-input". Cet élément représente la zone de texte où l'utilisateur saisit son message.
Récupération du contenu du message :

La variable message est déclarée et initialisée avec la valeur de la propriété value de l'élément messageInput. Cela récupère le texte saisi par l'utilisateur dans la zone de texte.
Vérification du contenu du message :

Une condition if est utilisée pour vérifier si le contenu du message, après avoir supprimé les espaces vides au début et à la fin avec trim(), n'est pas une chaîne vide. Cela permet de s'assurer que le message n'est pas vide ou constitué uniquement d'espaces vides.
Ajout du message à la fenêtre de chat :

Si le message n'est pas vide, la variable chatMessages est déclarée et initialisée avec l'élément qui a l'ID "chat-messages". Cet élément représente la zone où les messages de chat sont affichés.
La variable messageElement est déclarée et initialisée avec un nouvel élément créé à l'aide de la méthode document.createElement("div"). Cet élément représente le message à afficher.
La propriété innerText de l'élément messageElement est définie avec une chaîne de texte qui inclut le nom de l'utilisateur actif (activeUser) et le contenu du message saisi par l'utilisateur.
L'élément messageElement est ajouté à l'élément chatMessages en utilisant la méthode appendChild(). Cela affiche le message dans la fenêtre de chat.
Réinitialisation de la zone de saisie du message :

La propriété value de l'élément messageInput est définie avec une chaîne vide pour effacer le contenu de la zone de saisie du message.

Annexes

4.1 Capture d'écran du site web