# Sommaire

# Introduction

**Mes missions**

* Participer aux comités éditoriaux, pour obtenir la matière, les informations nécessaires à la création de contenus : textes, vidéos, visuels, podcasts, données contextuelles pour le référencement naturel de Google et données contextuelles pour les réseaux sociaux.
* Intégrer ces contenus dans l’outil de gestion de contenus
* Rédiger des documents (Word ou PowerPoint) d’expression de besoins métier à destination de l’équipe UX / UI et de la Digital Factory.

# Présentation de l’entreprise

## L’entreprise en général

« L’assureur d’un monde qui change » : tel est le slogan du leadeur mondial en assurance des emprunteurs. Cardif du groupe BNP Parisbas est donc leader mondial des partenariats en bancassurance et en assurance emprunteur. En offrant la meilleure assurance à ses clients, BNP Parisbas Cardif va donc permettre à ceux-ci de réaliser leurs projets tout en se prémunissant des aléas de la vie tels que : le décès, l'invalidité, l'incapacité de travail et/ou de perte d’emploi. Cardif ne va pas se limiter aux assurances de prêt, elle va cocréer avec ses partenaires des offres qui correspondent à l’évolution et aux changements dans le monde. Ces offres sont diverses. On pourra noter les secteurs suivants : assurance, épargne, prévoyance, retraite, assurance-crédit, assurance emprunteurs, assurance moyens de paiement et bien d’autres. Les principales seront donc l’assurance emprunteur (ADE), l’assurance habitation, assurance prévoyance, épargne et retraite.

Les chiffres clés de Cardif sont les suivants :

Tableau 1 : chiffres clés de Cardif

|  |  |
| --- | --- |
| N° 1 | MONDIAL EN ASSURANCE DES EMPRUNTEURS |
| 500 | PARTENAIRES DISTRIBUTEURS |
| 8000 | COLLABORATEURS DANS LE MONDE |
| 32.6 | MD€ DE CHIFFRE D'AFFAIRES EN 2021 |
| 33 | PAYS |

## Présentation de la Digital Factory

## Description du stage

# Déroulement du stage

Pendant mon stage, j’ai été amené à travailler sur différents univers, plus précisément sur différents sites web. Il s’agit du site cardif.fr qui se divise en deux grands sites : Cardif Particulier et Cardif Partenaire. Puis le site Finagora auquel on accède grâce à un lien sur l’espace Partenaire. Pour toute la suite, on va désigner les différents sites comme suit :

* Cardif Partenaire 🡪 B2B (Business to Business)
* Cardif Particulier 🡪 B2C (Business to Consumer)
* Finagora 🡪sites des courtiers

Tous ces sites web ont été conçus à partir d’un **Système** de **Gestion** de **Contenu** (SGC) en anglais Content Management System (CMS). Un CMS va permettre la création et l’administration d’un site web. Dans notre cas, le CMS que nous utilisons à Cardif est Liferay.

## Présentation de Liferay

Liferay est un éditeur de logiciels open source permettant à ses clients de créer des portails, intranets et des sites web. Il est tout aussi un éditeur indépendant de solutions de portail open source, de gestion de contenu, de collaboration sociale et de commerce en ligne B2B, en langage Java. La première version de Liferay est apparue en 2004, puis d’autres versions ont vu le jour et aujourd’hui, nous sommes à la version 7.

Pour la suite, nous allons limiter Liferay a sa fonction de CMS. Les fonctionnalités de celui-ci qui vont nous intéresser sont les suivantes :

* Il permet de hiérarchiser les utilisateurs en leur attribuant des rôles, des droits d’accès et/ou des permissions.
* Il permet de structurer les contenus qui peuvent être du contenu web ou des documents et médias.
* Il inclut la gestion des versions.
* Il inclut la planification des publications.
* Il va inclure le référencement SEO

Liferay se compose de 3 principaux produits :

* Liferay Analytics Cloud est un outil d'analyse web
* Liferay Commerce est une solution pour créer des sites web dédiés au B2B.
* Liferay Digital Expérience Platform (DXP) qui est ici une plateforme numérique pour la création des portails clients, des intranets et des sites web ouverts au public.

Ce dernier produit comporte deux versions différentes :

* + Community Edition (CE) qu’on va désigner comme une version moins stable aux mises à jour, mais elle reste tout de même fonctionnelle.
  + Enterprise Edition (EE ou DXP) qui est comme son nom l’indique une version faite pour les portails d’entreprises. Elle va être payante, mais néanmoins, elle est plus stable, sécurisé et va garantir la disponibilité des données en cas de bug avéré.

Au sein de Cardif, nous allons utiliser la version Enterprise Edition de Liferay car elle offre un grand nombre de Portlet, une meilleure sécurité des données pour le passage à la version la plus récente et elle permet également de créer et synchroniser plusieurs sites.

## Gestion du site Finagora

Finagora est le site web des partenaires et des courtiers en assurance de Cardif. Sur ce site web, les partenaires de Cardif vont trouver une partie de l’actualité de Cardif, mais aussi et surtout les différents éléments nécessaires au bon fonctionnement de leur activité. Ils vont retrouver :

* Les **produits de la gamme Cardif** dont les plus importants sont : Cardif Libertés Emprunteur, Cardif Élite, Cardif Élite Retraite, Cardif élite Capitalisation…
* L’**actualité de Cardif** et des autres partenaires.
* La **Gamme Financière** qui a son tour est divisée en plusieurs sous parties :
  + **La vie des supports** qui comprend : les Mandats de gestion Cardif Élite, la Gestion pilotée/Gestion Horizon, les Fonds euro et les supports structurés. Nous citons ici les plus importants sur lesquels nous sommes très souvent amenés à travailler.
  + Les **Opérations sur titre**
  + Les **Partenaires Gold** et les **Partenaires Privilégiés**
  + L’**Actualité des partenaires Gold**
  + Les **nouveaux OPC de la gamme Cardif**

Nous allons revenir plus en détail sur les différents points.

* **Juridique et fiscal** où l'on va notamment retrouver l’actualité contenant des points juridiques et tous les documents juridiques.

Mon rôle en tant que membre de la Digital Factory va être de gérer le back office du site web Finagora. Gérer ce site va revenir à : mettre à jour les différents fichiers devenus obsolètes, mettre à jour les actualités et le contenu web de tout le site, publier de nouveaux articles dans les différentes rubriques et section qui les concerne. Pour faire simple, sur Finagora, les éléments qu’on va être appelé à modifier seront soit du contenu web, soit des documents et médias.

Pour la suite, nous allons désigner par **contribution** toute opération visant à apporter une modification au site web Finagora.

Les différentes missions sur Finagora qui parviennent à la digital DF (Digital Factory) sont confiés par la Digital Courtage.

### Contribution Finagora

Comme nous l’avons dit précédemment, la contribution va être centrée sur la création, la modification, la suppression, la dépréciation et la planification des contenus web et des documents & médias. Cette contribution va donc s’appliquer sur les différentes rubriques du site citées ci-dessus, allant ainsi de la rubrique « produits Cardif » à la rubrique « Juridique et fiscal ».

#### Contribution d’une actualité

Une actualité est une page web qui aura un contenu web et/ou un onglet avec des documents à télécharger. Le contenu quant à lui aura un titre, un sommaire ou non, un résumé ou non et enfin un corps de l’actualité.

Pour la contribution du contenu web, il faut dans un premier temps formater le contenu de l’actualité en Html. Nous allons prendre en compte ici les entêtes, les paragraphes, la police, les images, les icônes, le format du texte et tout autre élément nécessaire au passage du contenu textuel à une page web. Nous allons particulièrement faire attention aux accents, aux apostrophes et aux points de suspensions, car ceux-ci sont susceptibles de ne pas présenter comme on l’aurait souhaité dans la page web.

Ensuite, il faut prendre en compte les catégorisations. En d’autres termes, il faut savoir quelles sont les pages qui ont le droit d’afficher l’actualité (‘a la une’, ‘first page’…). Donner les droits d’accès en lectures aux utilisateurs.

**Exemple de contribution d’une actualité**

Je vais ici vous présenter en détail les étapes nécessaires pour contribuer une actualité. Nous allons prendre l’exemple d’une actualité que j’ai mise en ligne : **Altixia**.

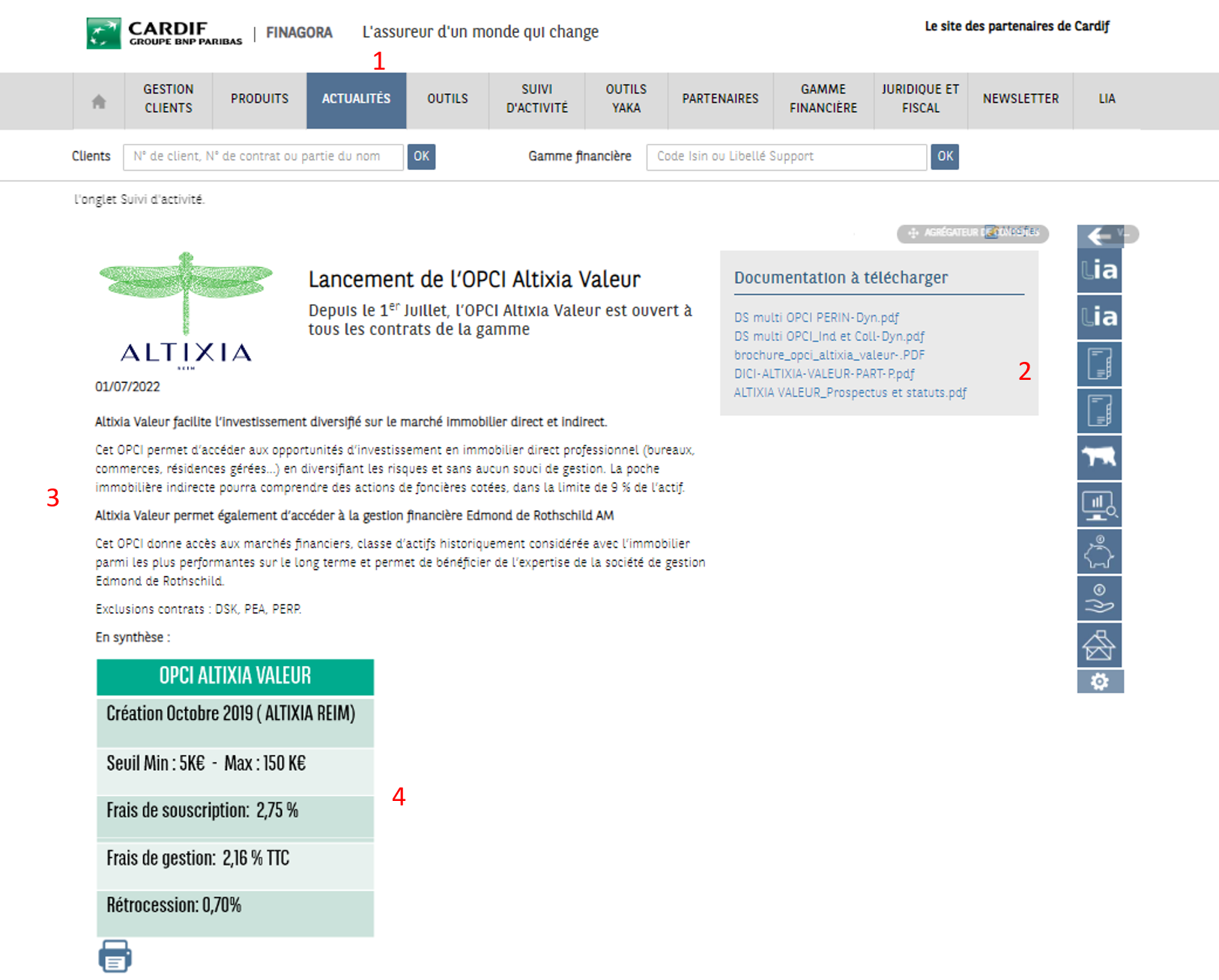


Figure 1 : actualité Altixia

Sur cet article, on peut remarquer facilement qu’il est une actualité en ‘1’ ; il contient des pièces jointes à télécharger en ‘2’ ; il a un contenu web avec une image en ‘3’ et ‘4’.

* **Formatage du contenu textuel en code Html**. Dans un premier temps on va formater le contenu pour obtenir un code Html. Ce qui constitue à identifier les paragraphes, les textes en gras et d’appliquer les balises correspondantes. Pour le l’image dans le contenu, je l’ai stocké dans les documents et médias puis j’ai utilisé son chemin d’accès pour l’afficher dans le contenu web à l’aide de la balise <img>.
* **Préparation du visuel**

Étape 1 : Le panneau 1 montre tous les onglets nécessaires à la configuration.

Étape 2 : Ça correspond à l’onglet contenu de l’étape 1. C’est là qu’on va entrer le résumé de l’article, le contenu formaté précédemment en Html, et si nécessaire les fichiers à télécharger.

Étape 3 : A ce niveau, on ajoute l’image qui décrit le contenu web ou le logo de l’entreprise.

Étape 4 : Comme nous en avons parlé plus haut, c’est à ce niveau que l’on va préciser rubriques dans lesquelles on veut afficher l’article. En actualité, ‘first page’ demande l’affiche sur la page d’accueil et sur le page des actualités.

Étape 5 : A cette étape, on a la possibilité de planifier la date et l’heure à laquelle on souhaite mettre l’actualité en ligne. On peut aussi modifier l’heure et la date de mise a jour dans le cas d’une mise à jour de l’article.

Étape 6 : Elle définit la mise en ligne dans l’espace restreint de Finagora et reste accessible uniquement aux personnes connectées

Pour résumer, le gros du travail est fait lors du formatage du contenu de l’article proprement dit. Le reste est de la configuration et des paramétrages. En contribuant pour la publication sur Finagora, je me suis familiarisé avec le CMS Liferay (Content Management System).

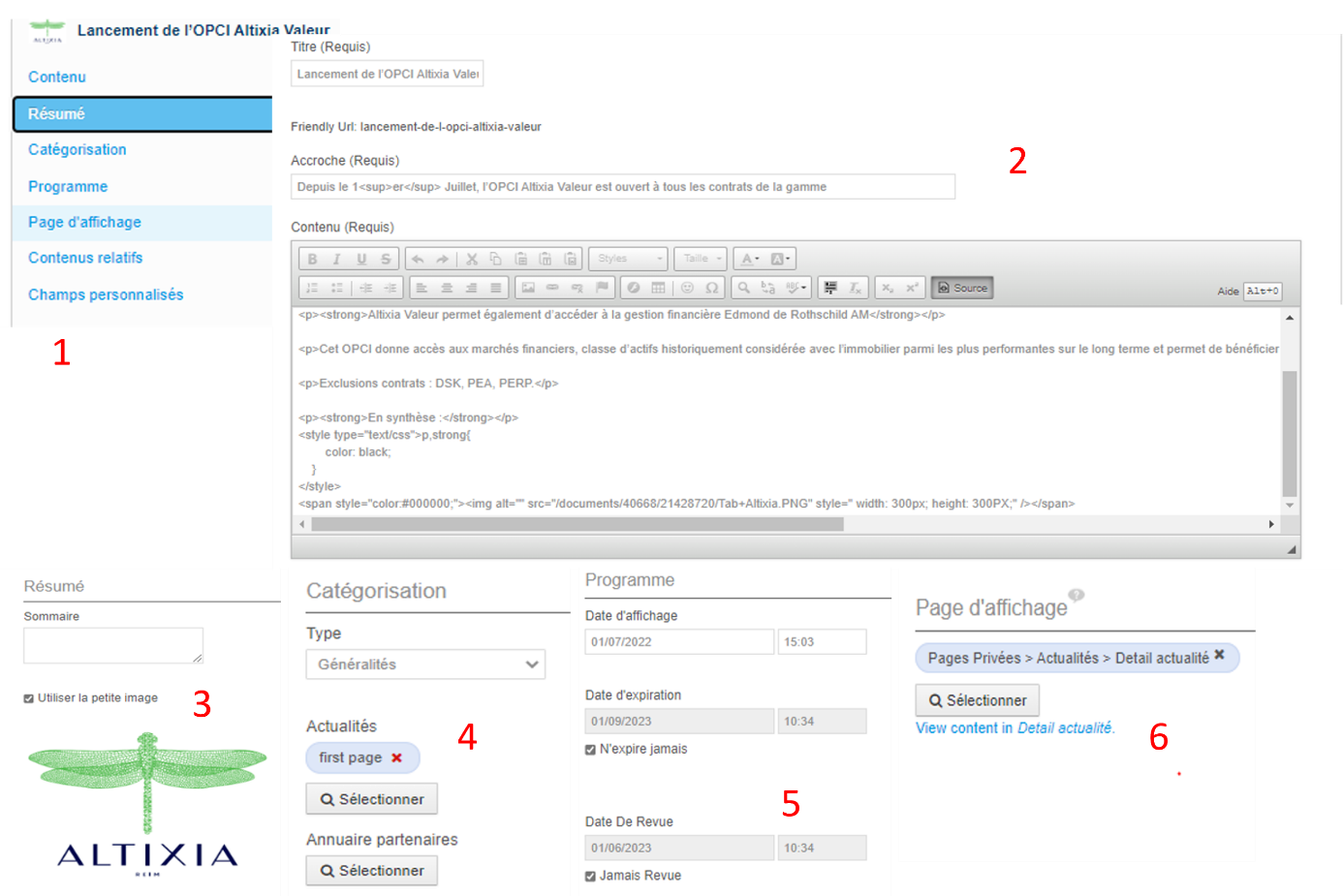


Figure 2: étapes de préparation du visuel de publication

#### Contribution dans la gamme financière

Cette rubrique est celle sur laquelle je suis très souvent amené à travailler. Comme nous l’avons expliqué, elle comporte plusieurs sous-sections.

* **La vie des supports** : cette sous rubrique est la plus importante de la gamme financière. Elle comprend les gestions déléguées et sous mandants, l’offre immobilière, les supports structurés et bien d’autres. La contribution visant à modifier le visuel de cette sous-section va se faire à un seul endroit précis. Dans le contenu web de la vie des supports. Ce contenu est localisé à : Web Content/Accueil/Contribution/Gamme financière/la vie des supports.

Pour la modification des éléments qui la constitue, la procédure sera souvent analogue.

Certains éléments de cette rubrique sont des liens qui mènent au téléchargement d’un document ; c’est le cas de l’offre immobilière. D’autres sont des liens qui pointent vers des pages spécifiques ; c’est le cas des gestions déléguées.

* **Contribution offre immobilière.** La contribution des docs et Média se fait à l’emplacement suivant : Accueil/Contribution/Gamme financière/Immobilier. Dans ce dossier, il faudra remplacer les documents obsolètes par les documents fournis par la Digital Courtage.

Dans le cas où l’offre immobilière n’existe pas, il faut la créer. Il va falloir créer son dossier dans les Docs & Médias, y ajouter les documents, modifier le contenu web de la vie des supports afin d’ajouter le lien de cette nouvelle offre et enfin ajouter ses documents en lien (voir image ci-dessous).



Figure 3: extraits d'offre immobilière de la gamme financière

Les offres SC Pythagore et SCI Cap Santé sont deux des offres que j’ai eu à créer durant mon stage.

* **Contribution des gestions déléguées.** Il s’agit là de la rubrique qui donne des informations sur les partenaires auprès de qui Cardif s'appuie pour gérer au mieux une partie ou tous les contrats de ses clients. Nous allons retrouver pour tous les partenaires une actualité et/ou des documents d’information, de présentation et d’évolution du partenaire.

La contribution à ce niveau peut être soit la modification du contenu web d’un partenaire ou l’application des opérations de d’ajout, suppression ou de modification des documents du partenaire. La modification du contenu web se fait suivant le nom du partenaire et à l’emplacement ci-contre : Web Content/ Accueil/ Contribution/ Actualités/ Gestion déléguée et sous-mandat Elite. Toujours suivant les noms des partenaires, la contribution des Documents et médias se fait à l’emplacement ci-contre : Accueil/ Contribution/ Produits épargne/ CARDIF ELITE/ Gestion déléguée et sous mandat (voir image ci-dessous). En 5 mois de stage, j'ai contribué sur les produits de tous les partenaires de Cardif.

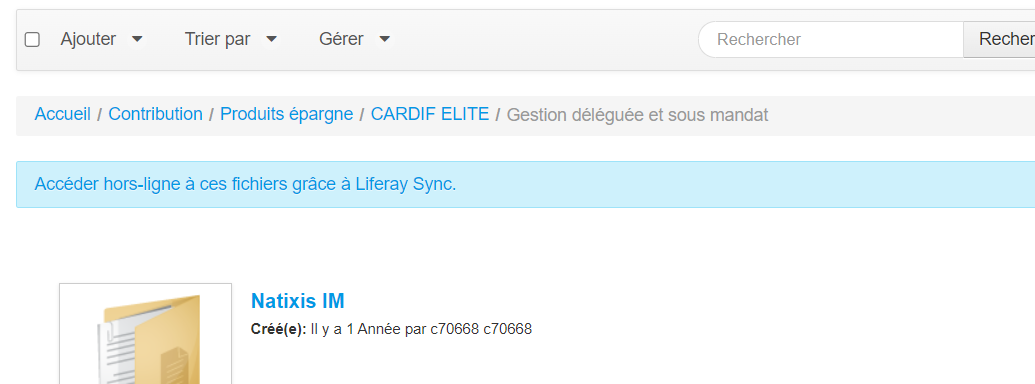


Figure 4: emplacement des documents de gestion délégués

* **Les nouveaux et les anciens fonds** : ils vont de pair. La modification de l’un entraine très souvent la modification de l’autre. Nous allons décrire ça le plus facilement possible. L’image ci-dessous montre les ‘cards’ d’entrée dans les rubriques citées à partir de l’onglet ‘gamme financière’.



Figure 5: les fonds de la gamme Cardif.

Il s’agit là de la rubrique où l'on met à jour les fonds des sociétés de gestion partenaire de Cardif. Ces fonds sont ajoutés dans une page web en tant qu’actualité. L’interface des nouveaux fonds ressemble à l’image ci-dessous.



Figure 6: les nouveaux fonds de la gamme Cardif

On remarque que les fonds sont entrés sous forme de tableau Html qui est mis à jour régulièrement. Chaque ligne du tableau correspond à un fond et pointe donc vers un fichier PDF à télécharger. Les fichiers doivent être stockés à l’emplacement suivant : Accueil/ Contribution/ Gamme financière/ Les nouveaux fonds. Dans ce répertoire, nous ferons très rarement des modifications mas plus des ajouts. On va stocker les fichiers, récupérer leurs chemins d’accès, créer une ligne dans le tableau avec les éléments correspondant au fond qu’on souhaite ajouter et enfin ajouter en lien le chemin d’accès ce fichier. Le nouveau fond a ainsi été créé.

Très souvent comme nous l’avons dit, l’ajout d’un fonds entraine le retrait d’un autre du tableau des nouveaux fonds. Les fonds qui sont retirés des nouveaux fonds sont mis dans le tableau des anciens fonds (voir image ci-dessous).



Figure 7 : les fonds des mois précédents.

Pour récapituler, la gamme financière possède à elle seule plusieurs sous-rubriques sur lesquelles j’ai contribué, je pense notamment aux partenaires gold et partenaire privilégiés ; les opérations sur les titres ; les gestions pilotées et les gestions horizon ; les fonds euros et euro croissance ; les supports structurés et buy & hold, ces 3 derniers contenus dans la vie des supports.

## Gestion du site Cardif B2B & B2C

## Gestion des différents projets

1. **Projet d’intégration d’une page web**

Ce projet est une demande de mon manager. Cette demande a été formulée à partir d’une expression de besoin.

### Projet d’accessibilité numérique

D’après l’article 47 de la loi n°2005-102 du 11 février 2005 pour l’égalité des droits et des chances, la participation et la citoyenneté des personnes handicapées, article que je paraphrase en disant : tous les services numériques publics doivent être accessibles à tous et surtout aux personnes handicapées. Ainsi donc, Les recommandations internationales pour l'accessibilité de l'internet doivent être appliquées pour les services de communication publique en ligne, quel que soit le contenu, le mode de d’accès ou de consultation. Les règles relatives à l’accessibilité numérique sont fixées, les délais d’adaptation et les sanctions encourues par les organismes qui ne la respecteront pas sont aussi fixés.

Un site web ouvert au public va de ce fait être un site web permettre aux personnes handicapées d’accéder à son contenu sans avoir de difficulté. Pour dire d’un site qu'il est accessible, on peut vérifier les éléments suivants :

* Toutes les informations sont reportées lorsqu’on navigue à l’aide d’un outil de synthèse vocale. Si oui, on est certain qu’un mal voyant peut avoir accès à notre site facilement,
* La navigation au clavier permet d’atteindre toutes les informations pertinentes,
* Le site web reste correct, peu importe la taille de l’écran sur lequel on le consulte, peu importe l’agrandissement et l’espacement qu’on lui applique,
* Le contraste de couleur (couleur du texte / couleur de fond) doit être supérieur à 4.5,
* Chaque image porteuse d’information doit avoir un texte alternatif,
* Il n’existe pas des balises utilisées juste à des fins de décoration,
* Les pseudo-classes (hover, focus…) sont bien définies sur les éléments interactifs,

La liste reste non exhaustive. Lors de la création et/ou des modifications d’un site web, on doit tenir compte des éléments d’accessibilité de la norme en vigueur. Pour cela, il existe des référentiels comme le RGAA (Référentiel Général d’Amélioration de l’Accessibilité) pour mettre en conformité les applications numériques créées.

Afin de se conformer au décret d’application de la loi (2019) de l’article 45 de la loi handicap de 2005, Cardif a décidé de faire un audit de ses applications web et de les rendre accessible au besoin. D’après le document d’information de Cardif que l'on peut consulter sur le lien 2 en annexe 1, il est marqué :

« BNP Paribas Cardif s’engage à rendre accessibles ses sites web (internet, intranet et extranet), ses applications mobiles, ses progiciels et son mobilier urbain numérique conformément à l’article 47 de la loi n°2005-102 du 11 février 2005.

[…] Nous présentons nos excuses à toutes celles et ceux qui ne pourraient pas, à ce jour, accéder de manière simple et complète à l'ensemble du contenu proposé. »

Ils informent également que l’audit du site a révélé un niveau faible d’accessibilité. C’est de là que vient le projet « d’accessibilité numérique » chez Cardif, projet sur lequel j’ai eu le plaisir de travailler avec l’équipe en charge.

NB : Cardif dispose d’un site de recette pour faire les tests avant publication et d’un site en production pour le code final à destination des internautes. Nous avons effectué toutes nos modifications d’accessibilité sur le site en recette. Ces modifications ne seront portées en production qu’une fois que l’accessibilité aura atteint un taux convenable.

Afin de rendre accessible le site de Cardif, nous sommes intervenus à plusieurs niveaux :

#### Accessibilité sur les images

Elle reste la plus facile.

* Lorsque la balise image est porteuse d’information, une alternative textuelle doit être passée à cette balise au travers de la balise ‘***alt*’.** L’alternative doit être la plus précise et la plus descriptive possible.
* Lorsque la balise image est utilisée à une fin de décoration, l’attribut ‘***alt***’ doit être vide (alt= ‘’). Cependant, les balises possédant un attribut WAI-ARIA ***role="img"*** ne peuvent faire office d'image de décoration qu'à la condition qu'elles possèdent un attribut WAI-ARIA aria-hidden="true".
* L’utilisation des images contenant des textes est déconseillée, nous avons préféré reprendre le texte présent sur ces images en CSS ou de les remettre intégralement dans l’attribut ‘***alt***’.
* Lorsque nous avons un groupe d’images, nous les regroupons en une seule image et donnons une alternative textuelle qui décrit l’image.

#### Accessibilité sur les Listes

La notion d’accessibilité pousse à privilégier l’utilisation des listes (ordonnées ou non ordonnées) pour décrire des contenus proches et argumentant la même thématique. Nous nous devons donc de regrouper visuellement sous forme de liste nos contenus web. La notion de ‘regroupement visuel’ fait référence à :

* La présence de marqueur de liste pour permettre d’identifier visuellement qu’il s’agit d’une liste non ordonnée,
* La présence de marqueur tel qu’un numéro ou une lettre pour faire comprendre qu’il s’agit d’une liste ordonnée.

#### Accessibilité sur les liens

Pour rendre notre site web accessible sur le critère de lien, nous avons dû intervenir à différents niveaux :

La navigation au clavier est un élément important de l’accessibilité. Les liens constituent le fondement du web et un élément incontournable pour un site web, il est nécessaire et primordial qu’en naviguant au clavier qu’on puisse atteindre les liens présents sur le site. Les liens doivent avoir le focus lorsqu’on navigue au clavier et être reconnaissable de façon distincte au survol avec la souris.

* **Le focus :** pour parfaire le focus sur les liens du site, nous avons désactivé le soulignement par défaut. À la navigation au clavier, on active la bordure autour de l’élément.
* **Le hover :** au survol d’un lien, on va faire apparaitre le soulignement. On identifie donc plus facilement le lien qui est interactif contrairement aux autres éléments du contenu web.

#### Accessibilité sur les balises de décoration

Sur le site en production aujourd’hui, il existe plusieurs balises de décoration. C’est le cas de :

* La balise ‘***div****’*, **‘*span’*** et/ou **‘i’**. Cette balise est souvent utilisée pour ajouter des icônes à l’aide des classes Css. Comme on peut le voir sur la figure ci-dessous, en 2 on peut voir la balise qui sert à obtenir l’icône 1. Ce pendant, cette balise ne vérifie pas les critères d’accessibilité, car elle est vide. Un lecteur d’écran qui parcourt cette balise ne restituera rien. Pour résoudre ce souci, nous devons dire au lecteur d’écran de ne pas prendre en compte cette balise. Ceci se fait à l’aide de l’attribut ‘***aria-hidden’*** qui prend la valeur ‘***true’***.

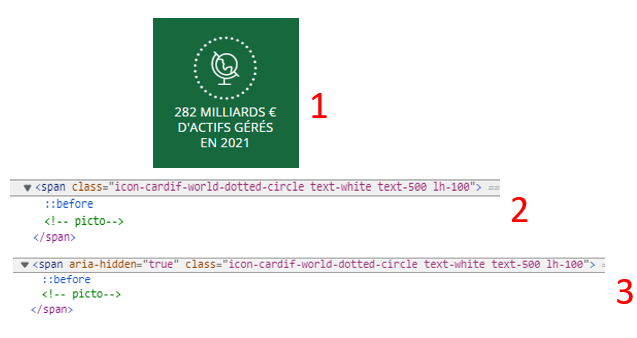


Figure 8 : accessibilité sur la balise <span> de décoration.

1. La balise **<hr>.** Sur le site de la production, la balise <hr> est souvent utilisé pour faire office de soulignement comme on peut voir sur le 1 et 2 de la figure 2. Son avantage est que l’on peut définir la taille du trait. L’équipe UI/UX avoir le trait à un niveau bien précis du texte, donc impossible d’utiliser un soulignement classique. Cependant, pour des raisons d’accessibilité, nous ne pouvons pas utiliser une balise <hr> car d’après la documentation Html, elle doit être utilisée pour changer de thématique (voir lien 1 en annexe 1).

Pour résoudre ces problèmes, nous avons opté pour l’ajout d’une bordure à partir du pseudo élément ‘ ***before****’* et l’ajout de certaines classes comme on peut le voir en 3 sur la figure 2.

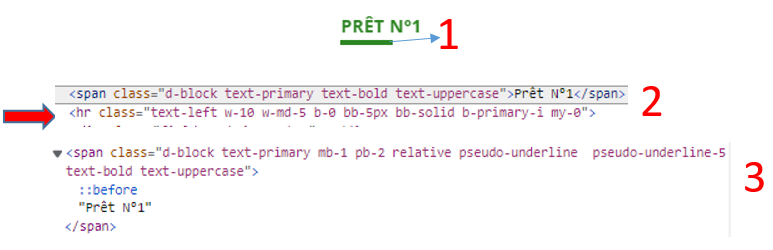


Figure 9 : accessibilité sur la balise <hr> de décoration.

En 3, nous ajoutons :

* La classe ‘**relative**’ pour donner la position relative à l’élément qui est le **span** dans ce cas.
* La classe ‘**pseudo-underline**’ qui s’applique sur le pseudo-élément ***before*** va créer un nouveau contenu vide, le placer en position absolu par rapport au premier parent qui est en position relative (notre span), placer ce contenu en bas à droite, lui appliquer une bordure de ‘Solid’ en bas uniquement avec une épaisseur de 5 px, et lui donner une taille par défaut de 13% de la longueur de l’élément **span.**
* La classe ‘**pseudo underline-5**’va permettre de modifier à 5% la taille qui a été fixé à 13% précédemment.

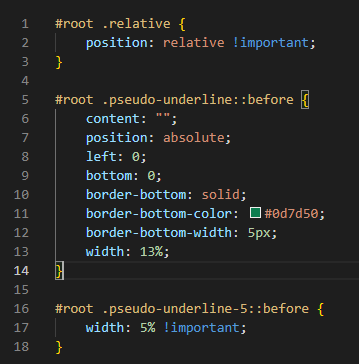


Figure 10 : classes nécessaires pour corriger l'accessibilité des balises ‘hr'

Ayant remarqué que c’est un problème qui revient souvent et dont la taille de la bordure n’est pas toujours pareille, nous avons décidé d’inclure dans le Css du site web un ensemble de classe qui feront varier cette taille. Nous avons aussi tenu compte du fait que la bordure puisse être à gauche et pas à droite, en haut ou en bas.

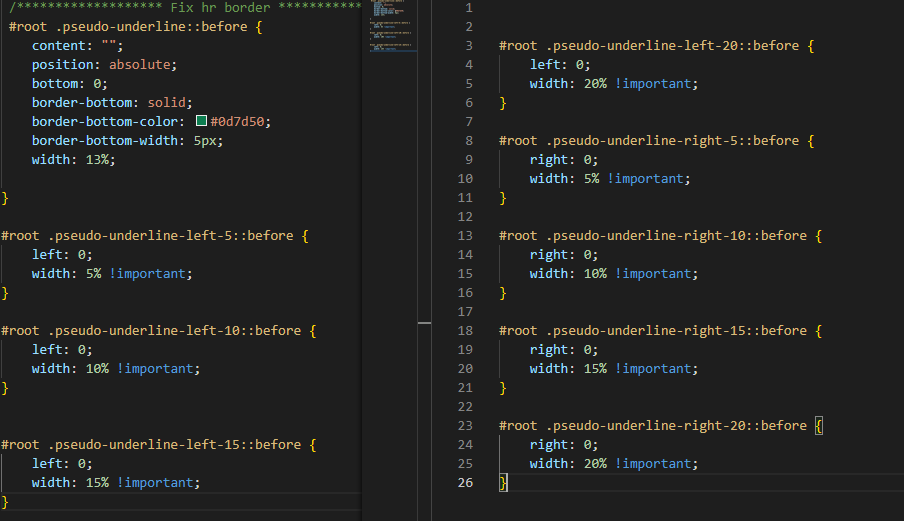


Figure 11: ensemble de classes pour rendre accessible le soulignement

#### Accessibilité sur les formulaires

Afin de rendre un contenu de type formulaire accessible, plusieurs éléments sont à prendre en compte.

* Utilisation de la balise < label > et des attributs ***for*** & ***id :*** 
  + Lorsque l’étiquette d’un champ doit être visible, il faut utiliser la balise <label> pour l’étiqueter.
  + Utiliser l’attribut ***for*** dans les labels et ***id*** dans les champs.
  + Chaque couple champ/label doit contenir la même valeur pour l’attribut for et id et ceux-ci doivent être unique (voir image ci-dessous).



Figure 12 : accessibilité label/champ d’un formulaire

* Utilisation de l’attribut ***title***: elle va être utile pour décrire des champs qui n’ont pas une étiquette. Il doit être renseigné en décrivant la fonction du champ. Cet attribut va trouver toute son importance lorsque le site est parcouru par un lecteur d’écran, la valeur de cet attribut sera restituée. L’attribut ***placeholder*** ne peut remplacer ***title.***
* Utilisation de l’attribut ***autocomplete.*** Comme nous pouvons le voir sur la figure 5, j’ai ajouté cet attribut sur tous les champs de formulaire. Sa valeur est fonction du champ. L’utilisation de cet attribut facilite le remplissage des champs qui constitue des informations personnelles.
* Utilisation des balises de regroupement tel que <fieldset>. Comme nous pouvons le voir ci-dessous, elle est utilisée pour regrouper des groupes de champs dont l’intitulé est similaire. Nous nous sommes assurés au fur et à mesure que tout le formulaire est accessible à la navigation au clavier.

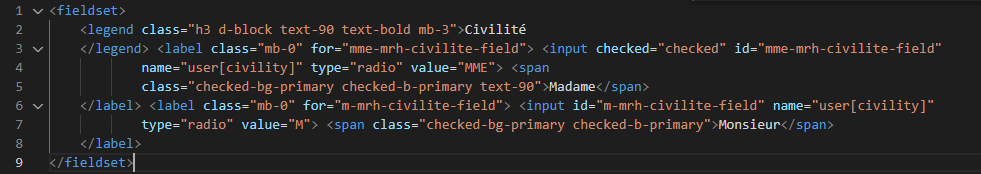


Figure 13 : accessibilité sur un formulaire avec la balise fieldset

#### Accessibilité sur les accordéons

La version des accordéons en production aujourd’hui ne permet pas de naviguer au clavier. L’objectif a donc été de rendre cette version navigable au clavier et que l’élément qui possède le cuiseur soit identifiable de façon distincte.

Sachant que Liferay 7 utilise la version 4 de Bootstrap, nous avons jugé plus simple de passer par ce Framework d’interface utilisateur qui offre une bibliothèque de classe importante pour résoudre ce problème.

Pour faire simple, afin de rendre un accordéon accessible, certaines classes et attributs doivent être présente.

* Sur le bouton qui fait l’action :
  + ***Data-toggle = "collapse"*** c’est cet attribut qui fait l’action **hide** / **show** à l’aide de JavaScript.
  + ***Data-target = "#collapseOne" & aria-control = ‘collapseOne’ :*** ces attributs désignent vers l’élément qui va subir l’action.
  + Le dernier et le plus important est ***aria-expanded="true".***  C’est lui qui indique aux lecteurs d’écran si l’élément est déroulé ou non.
* Sur la ***div*** qui subit l’action Collapse :
  + L’attribut ***id*** est nécessaire.
  + Un attribut data-parent et un autre aria-labelledby ne sont pas obligatoires, mais ils viendraient parfaire l’accessibilité.
  + En fin la classe collapse qui va servir à faire un ***display = ‘none’*** sur le contenu (masque le contenu).

Consulter l’annexe 3 pour voir le résultat des accordéons que j’ai rendus accessible. De manière analogue, j’ai rendu accessible tous les accordéons du site web.

#### Accessibilité sur les modales

### Projet d’optimisation du temps de formatage

Afin de faciliter la tâche aux développeurs de la Digital Factory qui font de la publication d’article, j’ai décidé de créer une page web pour formater les articles en Html.

#### Contexte et problématique

Deux mois après avoir rejoint l’équipe, j'étais déjà opérationnel et autonome. J’ai pendant ces premiers mois appris et je me suis familiarisé avec les outils qu’utilise mon équipe au quotidien. J’ai cependant remarqué que la publication d’un article demandait beaucoup de temps. Ce temps était long à cause d’une étape clé qui est la plus fastidieuse. Je me suis donc posé la question de savoir s’il est possible de palier à ce problème. Ne souhaitant plus passer autant de temps pour formater juste un article, j’ai décidé de me pencher sur la question et d’en faire un problème dont je dois trouver la solution.

#### Justification des choix faits

À cette période, je m’autoformais sur la bibliothèque JavaScript qu’est React JS. Il n’a pas fallu chercher longtemps pour que je me dise : « c’est cette bibliothèque que je veux pour résoudre ce problème ». Un élément principal m’a orienté à l’aveugle sur cette solution :

* Le JSX. Cette syntaxe particulière qui est en fait une extension de JavaScript qui utilise le Html pour décrire à quoi va ressembler l’interface utilisateur.
* La place de React dans le développement web des IU aujourd’hui. C’était pour moi le timing parfait pour mettre en pratique les base que je venais d’apprendre et d’en apprendre davantage
* En fin tout simplement le fait que React est la bibliothèque par excellence pour ce projet particulier.

Après ma décision de résoudre ce problème et après avoir trouvé le langage à utiliser, j’en ai discuté avec mon manager en lui présentant l’idée et les plus que la solution peut apporter à l’équipe. Il a tout de suite été séduit par l’idée et m’a donné son feu vert.

Après son accord qui a suivi les réponses aux questions « pour qui, et pourquoi et avec quoi développer cette page web ? », j’ai commencé à réfléchir sur les fonctionnalités que j’attends de mon application ; et comment je peux les mettre en œuvre.

#### Description des fonctionnalités

1. En tant qu’utilisateur de la page web, je dois pouvoir :
2. Entrer l’intitulé d’un sous-titre, son numéro et récupérer en sorti du code Html qui correspond à la syntaxe de Cardif.
3. Entrer un texte et obtenir en sortie un ou plusieurs paragraphes en fonction du nombre de retour à la ligne du texte.
4. Entrer des éléments d’une liste avec des retours à la ligne et obtenir en sortie une liste formatée en html et toujours suivant la syntaxe de Cardif.
5. Entrer l’url d’une vidéo, faire un retour à la ligne, entrer le titre de la vidéo et récupérer en sortie un « iframe » correspondant au format iframe de Cardif.
6. Disposer d’un box de d’éléments contenant tous les éléments (sous-titre, liste, paragraphe, vidéo et box ‘bon à savoir’).
7. Ajouter des éléments () que je souhaite utiliser pour formater mon article.
8. Supprimer un composant quand je n’en ai plus besoin,
9. Formater un seul élément et obtenir le code correspondant.
10. Formater tout un article et obtenir le code correspondant.
11. Modifier un élément incorrect de façon instantané.
12. Disposer d’un élément me permettant de valider mon formatage (bouton, lien).
13. Être automatique redirigé vers le contenu important lorsque je valide le formatage

#### Description du projet

Comme nous l’avons mentionné plus haut, le Project a été réalisé avec la bibliothèque JavaScript qu’est React JS.

Il faut retenir deux choses de React :

* Il est déclaratif : on définit des vues simples pour chaque état du UI. Une fois qu’un changement doit être fait, React s’en occupe de façon optimale en ne modifiant que le contenu qui a changé d’état sans toucher ceux qui n’ont pas changé.
* Se structure à base de composant : on crée des composant autonome qui vont constituer l’application et ainsi favoriser le débogage de celle-ci.
* React fonctionne avec un DOM virtuel : une représentation du dom est stockée en mémoire et React se charge de la réconciliation.

Je vais vous expliquer ici les différentes étapes qui ont permis d’obtenir notre page web.

* **Création des composants :** j’ai commencé par la création des composant qui vont constituer mon application. Il s’agit ici des composants liste, paragraphe, sous-titre, vidéo et box « bon à savoir » comme nous pouvons vois sur l’image ci-dessous qui montre le composant liste.



Figure 14 : création du composant liste

Dans notre application, il existe des composants qui ont été créé pour factoriser le code. Il n’est pas nécessaire de le présenter. C’est le cas d’un composant crée pour ajouter un titre a un autre composant, ce composant de titre est réutilisé dans tous nos élément liste, paragraphe…

Dans le code, on peut voir que la classe Liste hérite de la classe React Component. Son constructeur définit un état de l’application qui peut varier selon la valeur de la clé ‘***key***’. La classe liste va retourner du JSX. Il s’agit d’une div qui va contenir :

* **Un titre**. Ce titre est passé au composant Liste grâce au composant PartTitle qui prend en attribut titre=’Liste’.
* **Un textarea.** Il a un attribut ***ref*** qui lui sera donné par son parent. Nous reviendrons sur l’utilité des références plus tard
* **Un bouton supprimer.**  C’est grâce à ce bouton qu’on va pouvoir supprimer un élément si on en a plus besoin (voir fonctionnalité 8 dans la section II, D, 3, c). La gestion de l’évènement est faite à partir d’une méthode de la classe parente et on lui passe en paramètre la clé du composant courant.
* **Création du box de boutons :** il s’agit ici du box qui va contenir tous nos boutons représentant les différents éléments. Il est lui aussi un composant au sens React. Il prend en paramètre les propriétés de son ancêtre le plus proche. Il utilise la méthode ***choix\_btn ()*** de la classe ancêtre. Cette étape s’appelle la remonter d’état, à la seule différence qu’au lieu de faire remonter un état localement défini, on fait remonter des constantes en fonction du choix de l’utilisateur.



Figure 15: code du box d'éléments

Pour résumer, on a créé nos éléments (composants), on a créé le box de bouton pour afficher un élément en fonction du clic sur un bouton du box (voir image ci-dessous). Nous allons à présent voir ce qui se passe lorsqu’on clique sur un bouton de ce box donc ce que fait la méthode ***choix\_btn ()***.

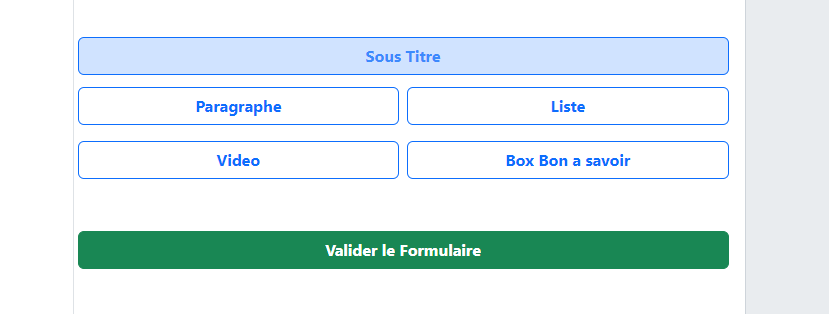


Figure 16: résultat du box de bouton

* **L’application dans le fichier App.js.**  C’est la partie de notre application la plus importante.

Le code sur l’image ci-dessous montre le rendu que l’App qui sera fait au DOM React. Il est scindé de la façon suivante :

* Header à la ligne 165,
* La div qui va contenir les entrées de l’application entre la ligne 168 et 173,
* La div qui va contenir la sortie de l’application entre la ligne 175 et 185,

Seule la div des entrées va retenir notre attention.



Figure 17: rendu de app.js

Cette div a une instruction JS et une balise composant :

1. **Balise composant : Box de composant**

On va noter ici la valeur de l’attribut choix\_btn qui est une méthode de la classe App. Pour mieux comprendre le fonctionnement de cette fonction choix\_btn, regardons cet extrait de la classe App. On voit que la classe App définit :

* **Map\_ALL\_Elt** (ligne 12)**.** Lorsqu’on clique sur un bouton pour afficher un élément graphiquement, cet élément va d’abord être stocké dans cette Map.
* **Ref : qui est aussi une Map** (ligne 21)**.** Lorsqu’on clique sur un bouton du box de bouton dans le but d’ajouter un élément, on doit créer une référence cet élément, cette référence est stockée dans cette Map. Nous l’utiliserons pour gare une traçabilité sur tous les éléments qui sont créée dans l’application. Notamment lors de la suppression d’un élément.
* La variable d’Etat **increment\_nbre\_elts.** Ellejoue un rôle crucial dans notre application, elle s’incrémente quand on clique sur un bouton pour ajouter un élément

**NB : c’est le changement d’état de cette variable qui entraine une mise à jour de l’état du DOM virtuel et donc ce qui aura changé dans note App sera automatiquement mis à jour dans l’interface utilisateur.**

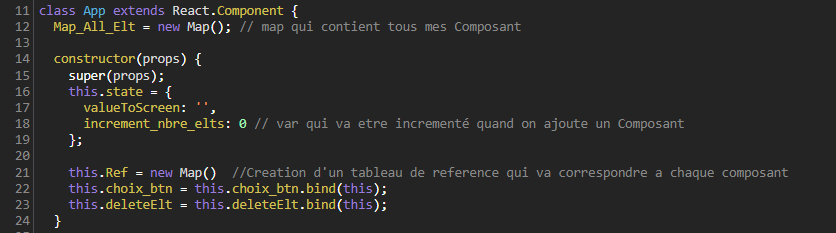


Figure 18: constructeur de la classe App

Pour revenir à notre fonction choix\_btn ():

* Elle est appelée lorsqu’on clique sur un bouton du box (sous-titre, paragraphe, liste, vidéo…).
* Chaque bouton de ce box appelle la fonction avec un paramètre qui lui est propre ; valeur 0 pour un sous-titre, valeur 1 pour un paragraphe, valeur 2 pour une liste, valeur 3 pour une vidéo, valeur 4 pour un pour le box « bon à savoir » et la valeur 5 pour le bouton de validation du formulaire.
* Si la valeur reçue est différente de 5, 0 par exemple, on a le code suivant pour ajouter un sous-titre.

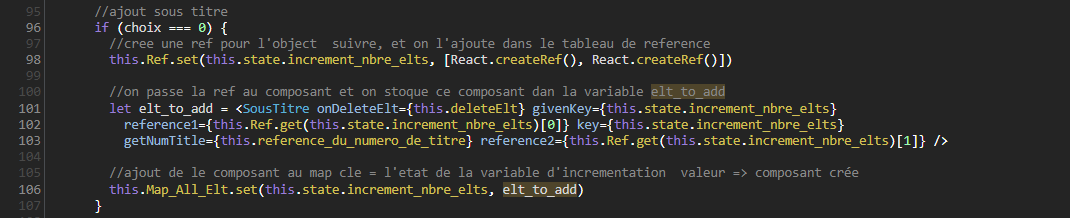


Figure 19: code exécuté dans la fonction choix\_btn pour le clique sur sous-titre

À la ligne 98, on crée une référence et on la stocke dans la Map des références. Dans ce cas précis, on crée deux références pour ce composant, car l’une est pour le titre et l’autre pour le numéro du titre. Dans ce cas, a cette clé dans la Map, on va attribuer un tableau de 2 valeurs (2 réf.).

À la ligne 101, on crée un objet sous-titre (c’est l’initialisation de l’objet sous-titre qu’on a créé précédemment) et on lui attribue la référence crée à la ligne 98. Pour l’instant cet objet va être stocké dans la Map prévu pour contenir les différents composants. À la sortie de la fonction choix\_btn, on incrémente la variable **increment\_nbre\_elts.** Le changement d’état de cette variable va mettre à jour le DOM virtuel et donc notre composant va apparaitre dans la liste des composants sur l’interface utilisateur.

* Si la valeur reçue est 5, l’utilisateur souhaite valider le formulaire et obtenir son résultat.

Pour chaque pair key/value de la map Ref, on vérifie que la valeur de la Ref existe et n'est pas nulle. En fonction de l’id de l’élément (paragraphe, vidéo…), on applique une fonction de formatage particulière (voir image ci-dessous) parmi les fonctions de formatage qu’on verra plus tard. Le texte formaté est incrémenté à la variable valeur (voir ligne 77) qui sera retourné. Une fois la boucle terminée, on modifie l’état de la variable valueToScreen (voir ligne 82) présente dans le constructeur. Une fois de plus, c’est la modification d’une variable d’état qui va mettre à jour l’UI.

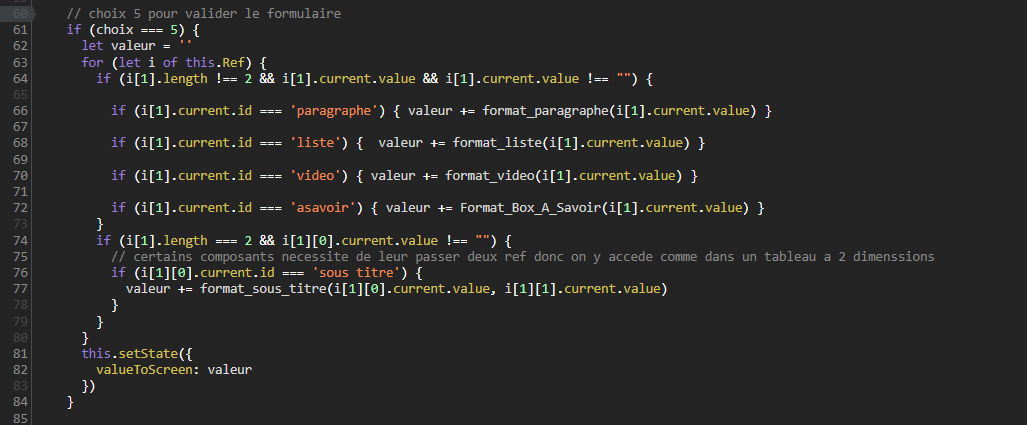


Figure 20: fonction choix\_btn pour un choix de valider le formulaire

Maintenant que les attributs et méthodes de la classe App ont été expliqués, il sera plus facile d’expliquer le fonctionnement de la méthode. Supprimer un élément de l’affichage va revenir à retirer sa référence dans la table des références et de le retirer lui aussi de la table des éléments. Ainsi, lors du prochain changement d’état, cet élément ne sera plus présent dans les boucles.

NB : Sans consulter le code, un expert React peut me poser au premier abord la question de savoir pourquoi le choix des composants non contrôlé alors que React le recommande ? pendant le développement avec les composants contrôles, j’ai eu du mal à garder la traçabilité sur les éléments qui sont ajoutés de façon aléatoire. Je dois savoir exactement quel composant est ajouté, quel est modifié, et quel est supprimé ; aussi et surtout pouvoir supprimer un composant particulier. Il est parfois fastidieux d’utiliser les composants contrôlés, car il vous faut écrire un gestionnaire d’événement pour chaque possibilité de changement des données, et gérer toute modification des saisies via un composant React. Donc dans mon cas, j’ai dû passer par les composants non contrôlés pour avoir la main sur chaque composant qui est créé afin d’y apporter mes modifications et mes suppressions. C’est là la nécessiter de donner à chaque composant une référence. Pour en savoir plus, voir l’annexe 1, lien 3 et 4

1. **Instruction JS**

Il s’agit de l’instruction {this.FromMapToArray(this.Map\_All\_Elt).map((e) => e)}. Elle permet simplement d’afficher sur l’interface utilisateur les composants présents dans la collection de composants **Map\_All\_Elt.**

Pour récapituler :

* J’ai créé des composants et un box de bouton correspondant chacun a un composant.
* Chaque composant est contrôlé à l’aide d’une fonction choix\_btn qui prend un paramètre de 0 à 5.
* Si ce paramètre est différent de 5, on crée une référence React, on crée un composant non contrôlé auquel on attribue cette référence. On stocke cet élément dans une collection de composant. On modifie l’état de l’application pour mettre à jour et afficher le composant qui a été créé.
* Si ce paramètre est 5, les valeurs des inputs dont on dispose de la référence sont formatées à l’aide des fonctions adéquates. Ces valeurs sont jointes et passé à la variable d’état qui va permettre de mettre à jour le DOM React et donc d’afficher le résultat attendu par l’utilisateur.
* À la suppression d’un composant, on valide à nouveau le formulaire, ce qui permet de mettre à jour l’affichage.

#### Présentation des fonctions de formatage utilisées

Pour résoudre notre problème, nous avons dû utiliser certaines expressions rationnelles. Elles sont listées dans le tableau ci-dessous.

Tableau 2 : tableau des expressions rationnelles utiles pour les fonctions de formatage

|  |  |
| --- | --- |
| **Caractères** | **Significations** |
| {n,} | Il y a au moins n occurrences.  Exemple : /a{3,}/ = ‘aaa’, ‘aaaa’ mais pas ‘aa’ |
| \s | Correspond à un espace, une tabulation ou un saut de ligne. |
| \t | Tabulation |
| /[^A-Za-z0-9\_]/ | Correspond à tous les caractères non alphanumériques tels que : %, @, $, £, ¨… |
| \n | Retour à la ligne |
| \g | Sélectionne toutes les occurrences |
| ^ | Exprime la négation d’une expression |

* Formatage des paragraphes : je veux pour voir récupérer en entrer un texte et renvoyer un ou plusieurs paragraphes en fonction de la structure du texte. Le résultat doit être un code formaté en Html avec la balise ***‘<p>***’.

Le texte est découpé en paragraphe avec la méthode split () qui prend en paramètre une expression rationnelle qui est « \n » et stocké dans un tableau. Ensuite, pour chaque élément du tableau, on ajoute une balise de paragraphe ouvrante et une fermante.

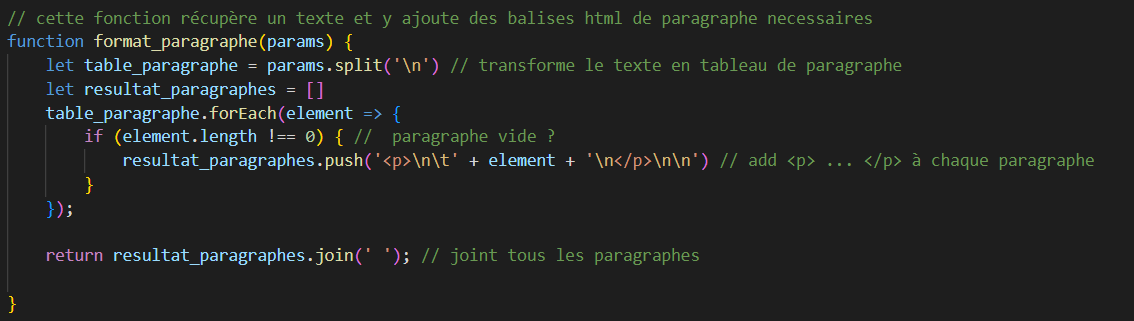


Figure 21 : fonction de formatage des paragraphes

* Formatage des sous-titres : l’idée est de récupérer un sous-titre et son numéro afin de renvoyer le code Html qui correspond à la syntaxe de code des sous-titres au sein de la Digital Factory (DF). Au sein de la DF, dans un article, tout sous-titre doit avoir un identifiant. Pendant le formatage des sous-titres, il faut ajouter cet identifiant unique en attribut.

L’identifiant correspond au titre auquel on retire les accents, on remplace les espaces par les tirets et on retire tout caractère qui n’est pas alphanumérique. Pour une question d’esthétique et de prévoyance, on retire les espaces en double puis les espaces en début et en fin de phrase.

On peut voir en annexe à quoi ressemble une ensemble sous-titre + paragraphe sur le site de Cardif (Voir annexe 3) et comment se présente le code formaté avec la page web conçu (Voir annexe 3).

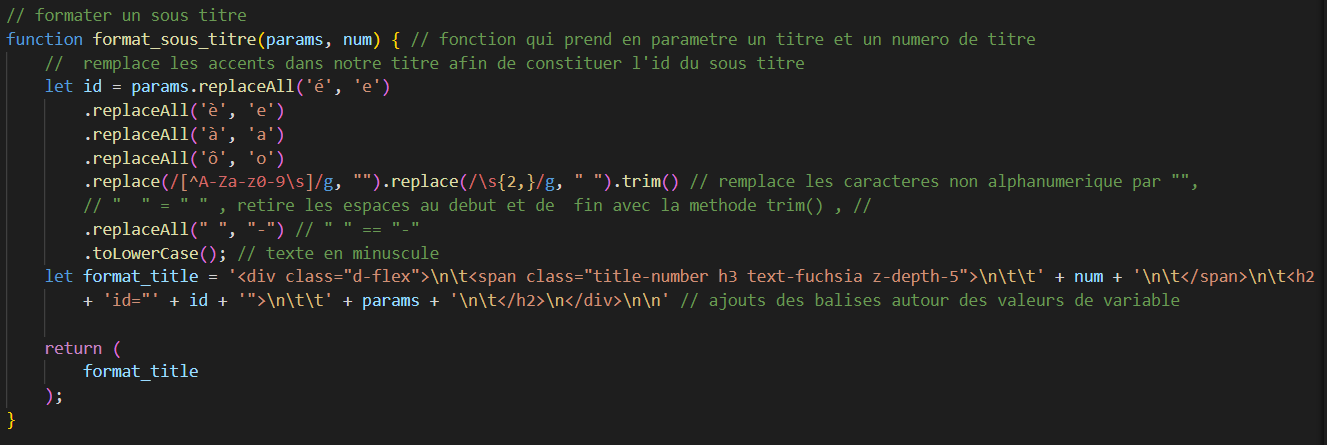


Figure 22 : fonction de formatage des sous-titres

* Formatage des listes : je veux ici constituer un élément de liste avec chaque paragraphe d’un texte. Je prends un texte en entrée, je le découpe en fonction des retours à la ligne, j’entoure ces paragraphes avec les balises <li>…</li>, je joins ces paragraphes et je concatène le tout avec les balises <ul>…</ul> toute fois avec les classes qu’il faut.

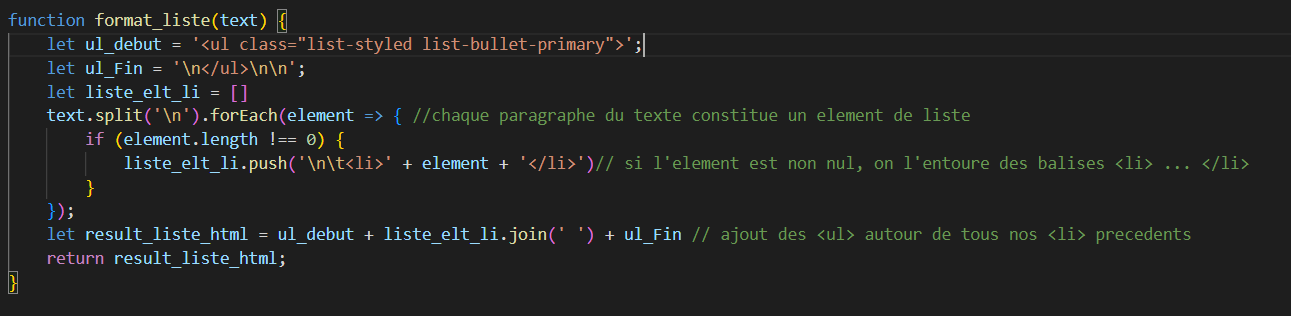


Figure 23 : fonction de formatage des sous-titres

* Formatage du box « bon à savoir » : pour ce cas principalement, je veux appliquer un style particulier au premier paragraphe uniquement ; je le découpe donc en paragraphe comme précédemment. J’insère les éléments dans ce tableau de façon ordonné. Et puis je joins ses éléments que j’envoie à la sortie de la fonction.

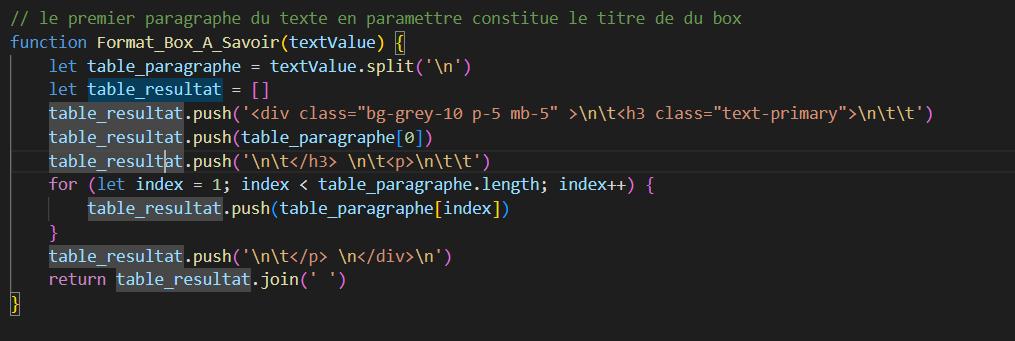


Figure 24 : fonction de formatage du box « bon à savoir »

* Formatage d’un iframe vidéo : l’input contient un texte avec 2 paragraphes. Le premier est le lien de la vidéo et le second est le titre. De façon analogue aux façons précédentes, le découpe le texte. Ces deux éléments vont être concaténés à des endroits précis de l’iframe. En sortie, je revois un iframe pouvant être utilisé.

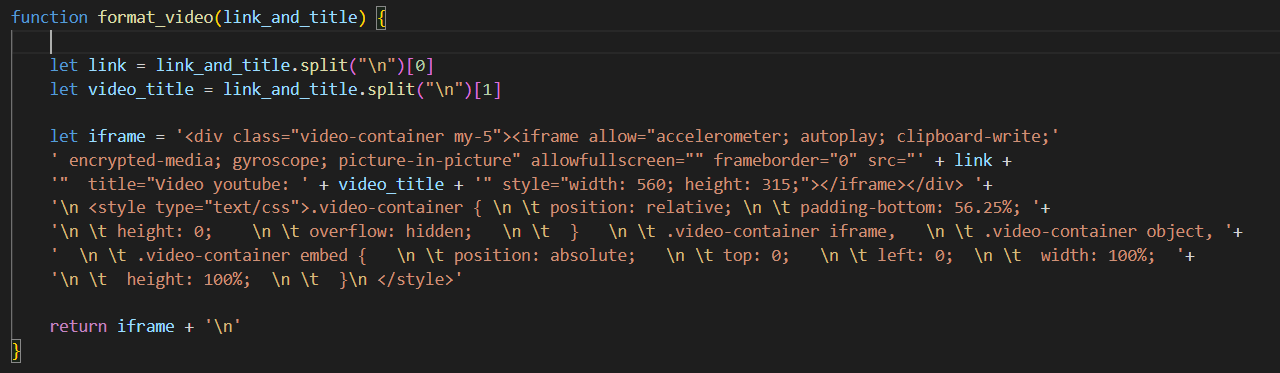


Figure 25: fonction de formatage iframe vidéo

#### Présentation des résultats

J’ai travaillé avec le logiciel de gestion des versions GitHub pour conserver l’évolution du projet. Le projet est actuellement sur le dépôt privé GitHub suivant <https://github.com/axeljunior1/bnp_projet_formatage_v3>. J'ai déployé l’application sur le sur l’hébergeur Netlify à l’adresse suivante : <https://formatage.netlify.app/>.

L’image ci-dessous nous donne une présentation de la page.



Figure 26: résultat de la page web développé

#### Pistes d’améliorations

Le travail réalisé est parfaitement fonctionnel. Mais pour rester dans un esprit d’amélioration perpétuelle, je peux proposer de passer par un WYSIWYG pour parfaire le formatage, prendre en compte tous types d’éléments, et ainsi minimiser au maximum le temps de formatage.

# Difficultés rencontrées et solutions proposées

## Difficultés rencontrées

## Solution proposée

## Les acquis au terme du stage

Au terme de ce stage, j'ai pu :

* Découvrir ce qu’est le métier de Content Manager dans la vie d’un site web.
* Travailler avec tous les profils métiers du digital autour de la vie d’un site : webmasters, chef de projet digital, UX designers, rédacteurs de contenus.
* Comprendre comment fonctionne le métier de téléconseiller et celui d’experts en acquisition de trafic au sein de Cardif.
* Apprendre à intégrer des contenus digitaux dans différents outils de gestion de contenus.
* Acquérir de nombreuses notions comme le SEO (référencement naturel Google).
* Acquérir de nombreuses notions de base du Framework React JS, de la bibliothèque jQuery.
* Améliorer grandement mes connaissances de Html, Css et Bootstrap autant globalement que sur des notions particulières.
* Monter en compétente de façon fulgurance sur le concept qu’est l’accessibilité web

# Conclusion

Bibliographie :

[Accueil - RGAA | numerique.gouv.fr](https://www.numerique.gouv.fr/publications/rgaa-accessibilite/#contenu) ou on peut télécharger les deux documents d’accessibilité.

# Annexes

## Annexe 1 : Liens

Lien 1 : https://document-information-cle.cardif.fr/accessibilite

Lien 2 : <https://developer.mozilla.org/fr/docs/Web/HTML/Element/hr>

Lien 3 : <https://fr.reactjs.org/docs/forms.html#alternatives-to-controlled-components>

Lien 4 : <https://fr.reactjs.org/docs/uncontrolled-components.html>

## Annexe 2 : Table des matières

Table des matières

[Sommaire 1](#_Toc112309500)

[Introduction 1](#_Toc112309501)

[I. Présentation de l’entreprise 2](#_Toc112309502)

[A. L’entreprise en général 2](#_Toc112309503)

[B. Présentation de la Digital Factory 2](#_Toc112309504)

[C. Description du stage 2](#_Toc112309505)

[II. Déroulement du stage 4](#_Toc112309506)

[A. Présentation de Liferay 4](#_Toc112309507)

[B. Gestion du site Finagora 5](#_Toc112309508)

[1- Contribution Finagora 6](#_Toc112309509)

[C. Gestion du site Cardif B2B & B2C 6](#_Toc112309510)

[D. Gestion des différents projets 7](#_Toc112309511)

[1. Projet Priips 7](#_Toc112309512)

[2. Project d’accessibilité numérique 8](#_Toc112309513)

[3. Projet d’optimisation du temps de formatage 14](#_Toc112309514)

[III. Difficultés rencontrées et solutions proposées 16](#_Toc112309515)

[A. Difficultés rencontrées 16](#_Toc112309516)

[B. Solution proposée 16](#_Toc112309517)

[C. Les acquis au terme du stage 16](#_Toc112309518)

[Conclusion 17](#_Toc112309519)

[Annexes 19](#_Toc112309520)

[Annexe 1 : Liens 19](#_Toc112309521)

[Annexe 2 : Table des matières 20](#_Toc112309522)

[Annexe 3 : Images 21](#_Toc112309523)

## Annexe 3 : Images

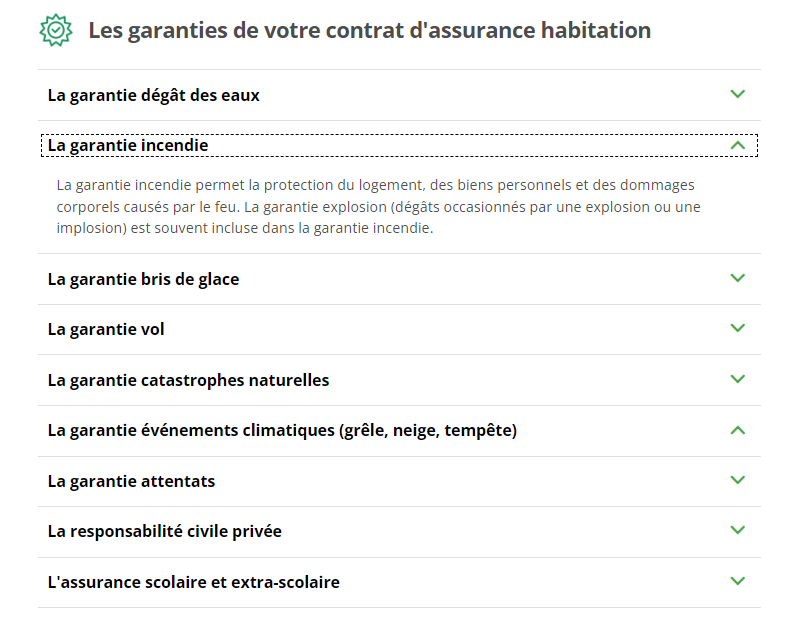


Figure 27: accessibilité sur les accordéons

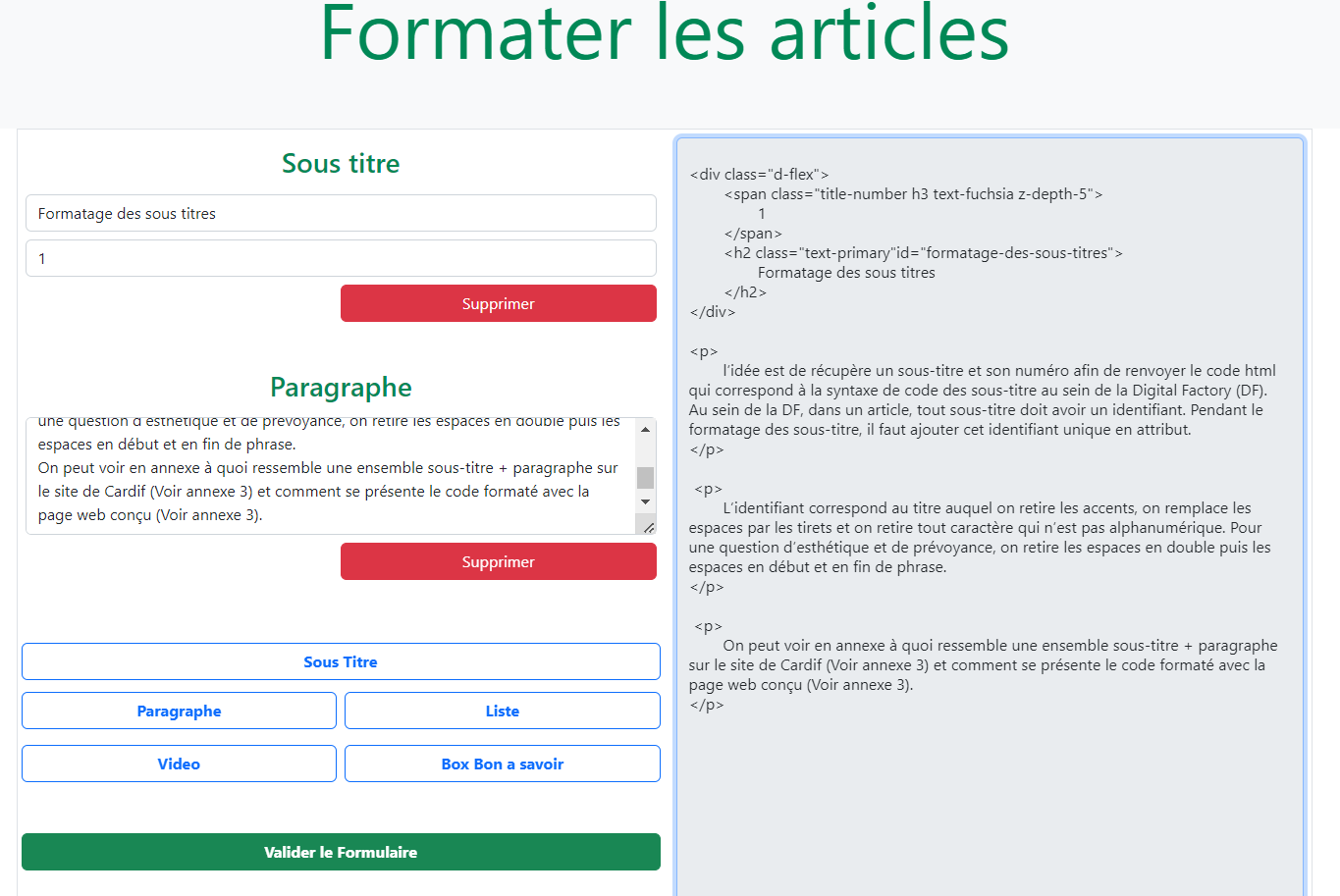


Figure 28: code obtenu pour le formatage d’un sous-titre et ses paragraphes

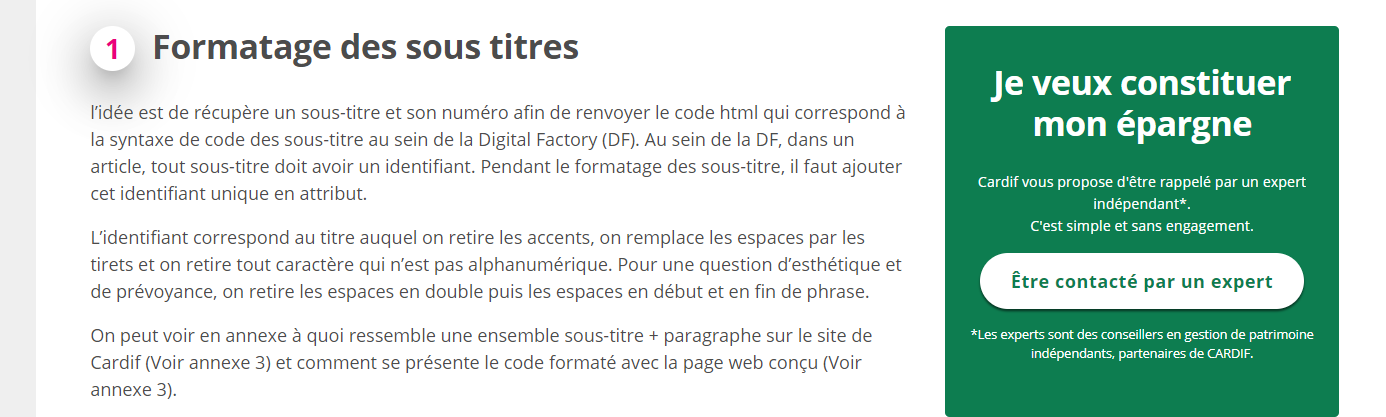


Figure 29: résultat du formatage d'un sous-titre + paragraphe

## Annexe 3 : Table des abréviations

|  |  |
| --- | --- |
| Abréviations | Sigle |
| DF | Digital Factory |
| JS | JavaScript |
| DOM | Document Object Model |
| CMS opc | Content Management System |
|  |  |