



STAGE - e-Liberty Services

Développeur Front End

Effectué par **BOUDIN Rodolphe**

Du **27/05/2024** au **05/07/2024**

Etablissement de formation : Lycée Dominique Villars - BTS SIO 1ère année

Entreprise d'accueil : e-Liberty Services, Savoie Technolac - Bat. Andromède, 108 Av. du Lac Léman, 73290 La Motte-Servolex

Maître de stage : GENEVE Jordan (jordan.geneve@eliberty.fr, 04 58 16 00 10)

Sommaire

Glossaire des Outils, Services et Ressources Informatique	3
Logiciels / services	3
Ressources numériques	4
Langages de programmation, bibliothèques et librairies	4
Méthodes de Développement et Conception	5
Outils et concept	5
Extensions utiles VS Code	5
Ressources informationnelles	6
Semaine 1	7
JOUR 1 (Lundi 27/05)	7
JOUR 2 (Mardi 28/05)	9
JOUR 3 (Mercredi 29/05)	10
Rendu final du mini jeu "Tic-Tac-Toe"	10
JOUR 4 (Jeudi 30/05)	11
JOUR 5 (Vendredi 31/05)	12
Rendu final de la To-do list	12
Semaine 2	13
JOUR 6 (Lundi 03/06)	13
JOUR 7 (Mardi 04/06)	14
JOUR 8 (Mercredi 05/06)	15
Rendu final du composant Divider :	16
JOUR 9 (Jeudi 05/06)	17
JOUR 10 (Vendredi 06/06)	18
Semaine 3	19
JOUR 11 (Lundi 10/06)	19
JOUR 12 (Mardi 11/06)	20
JOUR 13 (Mercredi 12/06)	21
JOUR 14 (Jeudi 13/06)	22
JOUR 15 (Vendredi 14/06)	23

Semaine 4	24
JOUR 16 (Lundi 17/06)	24
Rendu final du composant Calendar :	25
JOUR 17 (Mardi 18/06)	26
JOUR 18 (Mercredi 18/06)	27
JOUR 19 (Jeudi 20/06)	28
JOUR 20 (Vendredi 21/06)	29
Semaine 5	30
JOUR 21 (Lundi 24/06)	30
JOUR 22 (Mardi 25/06)	31
JOUR 23 (Mercredi 26/06)	32
JOUR 24 (Jeudi 27/06)	33
JOUR 25 (Vendredi 28/06)	35
Semaine 6	36
JOUR 26 (Lundi 01/07)	36
Rendu final du composant Textinput :	37
JOUR 27 (Mardi 02/07)	38
JOUR 28 (Mercredi 03/07)	39
Rendu final du composant Checkbox :	40
JOUR 29 (Jeudi 04/07)	41
JOUR 30 (Vendredi 05/07)	42

Glossaire des Outils, Services et Ressources Informatique

Logiciels / services

- **Figma** : Éditeur de graphiques vectoriels et outil de prototypage, essentiel pour les maquettes des designers et utilisé par les développeurs.
- **Gmail** : Service de messagerie professionnelle.
- **Google Meet** : Service de visioconférence utilisé pour les réunions.
- **GitHub** : Plateforme de partage de code et de gestion de versions avec Git.
- **Jira** : Outil de suivi des bugs et de gestion de projets.
- **Linux** : Système d'exploitation open source.
- **Storybook** : Outil open source pour le développement de composants UI en isolation, compatible avec la plupart des frameworks front-end.
- **Slack** : Plateforme de communication et de gestion de projets, utilisée pour la communication interne.
- **Visual Studio Code** : Éditeur de code extensible.
- **Zeroheight** : Plateforme de stockage cloud pour le système de conception, accessible à toutes les équipes collaboratives.

Ressources numériques

- **Amazon Web Services (AWS)** : Services de cloud computing.
- **BrowserStack** : Plate-forme de test web et mobile à la demande
- **IcoMoon** : Outil pour créer des polices d'icônes à partir de SVG.
- **MUI** : Bibliothèque de composants React personnalisables.

Langages de programmation, bibliothèques et librairies

- **HTML / CSS / JavaScript**
- **il8next** : Bibliothèque pour la gestion des traductions.
- **Jest** : Framework de test JavaScript.
- **Npm** : Gestionnaire de paquets JavaScript.
- **Playwright** : Bibliothèque d'automatisation open source pour tests de navigateur et web scraping, génère le code pour reproduire des scénarios d'utilisation. Utilisé avec **Cucumber** pour le développement axé sur le comportement (BDD) via le langage **Gherkin**.
- **React** : Bibliothèque pour créer des interfaces utilisateur composables.
- **Sass** (+ design tokens) : Langage de préprocesseur CSS.
- **Sonata** : Extension pour la gestion des bases de données Symfony dans VS Code.
- **TypeScript** : Langage open source pour le développement JavaScript sécurisé.

Méthodes de Développement et Conception

- **Atomic Design** : Approche de conception pour créer des interfaces modulaires.
- **Block Elements Modifiers** (BEM) : Méthodologie de nommage des classes CSS.
- **KANBAN** : Méthode visuelle de gestion de projet.

Outils et concept

- **Cheat Sheets** : Feuilles de triche fournissant des rappels rapides et pratiques.
- **Regex** : Expressions régulières pour le traitement de texte avancé.

Extensions utiles VS Code

- **Figma for VS Code** : Extension pour intégrer Figma à VS Code.
- **Jest + Jest Runner + Test Adapter Converter** : Extensions pour l'intégration de Jest dans VS Code.
- **Live Server** : Extension pour un serveur de développement local.
- **Local History** : Extension pour la gestion de l'historique des fichiers.
- **Path Intellisense** : Extension pour l'autocomplétion des chemins de fichiers.
- **Prettier** : Extension pour le formatage automatique du code.
- **SonarLint** : Extension pour la détection de code malveillant.
- **Turbo Console Log** : Extension pour le débogage et les logs console.
- **VScode-icons** : Extension pour des icônes personnalisées dans VS Code.

Ressources informationnelles

MDN : <https://developer.mozilla.org/fr/>

Stack Overflow : <https://stackoverflow.com/>

OpenClassrooms : <https://openclassrooms.com/fr/>

Reddit : <https://www.reddit.com/>

React : <https://fr.react.dev/>

TotalTypeScript : <https://www.totaltypescript.com/>

Semaine 1

JOUR 1 (Lundi 27/05)

Matin :

- **9h00** : Petit-déjeuner d'accueil.
- **9h15 - 9h30** : Mêlée quotidienne
- Présentation de l'entreprise, de ses principaux produits, des employés et de leurs postes.
- Création de mes comptes entreprise (Slack, mail professionnel, Github, nouveau poste, etc.).

Après-midi :

- Début de l'apprentissage du TypeScript avec une formation en ligne sur [Total TypeScript](#)
- Lecture du manuel TypeScript.

Explication de la mêlée quotidienne :

Tous les matins, de 9h15 à 9h30, l'équipe de développement se réunit pour une "mêlée quotidienne", une pratique clé des méthodologies agiles.

Objectifs de la Mêlée Quotidienne :

- **Définir les priorités** : Chaque développeur fait le point sur les tâches en cours et à venir, alignant ainsi l'équipe sur les priorités du jour.
- **Lever les blocages** : Les obstacles rencontrés sont identifiés et résolus, permettant de maintenir la productivité.
- **Suivi des sujets** : L'avancement des tâches est discuté pour s'assurer du respect des deadlines et de la bonne communication.

Cette réunion courte mais essentielle synchronise les efforts de l'équipe, favorise la communication et aide à surmonter les obstacles rapidement, illustrant ainsi les principes agiles appliqués dans l'entreprise.

JOUR 2 (Mardi 28/05)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne
- Fin de la formation sur TypeScript et lecture complète du manuel TypeScript.

Après-midi :

- Session de questions / réponses sur TypeScript avec mon formateur.
- Début de l'apprentissage de React avec l'objectif de réaliser le mini-jeu "Tic-Tac-Toe" sur le site de [React](https://react.dev/).

JOUR 3 (Mercredi 29/05)

Matin :

- **9h00** : Point de synchronisation sur les travaux effectués la veille sur le mini-jeu "Tic-Tac-Toe".
- **9h15 - 9h30** : Mêlée quotidienne.
- Finalisation du mini-jeu en React, objectif : design du mini-jeu afin de le rendre visuellement plus attractif.

Après-midi :

- Design du mini-jeu.
- Rappel des bases de JavaScript.

Rendu final du mini jeu "Tic-Tac-Toe"

X a gagné

O	X	X
O	X	O
X	O	X

1. Revenir au début
2. Aller au coup #1
3. Aller au coup #2
4. Aller au coup #3
5. Aller au coup #4
6. Aller au coup #5
7. Aller au coup #6
8. Aller au coup #7
9. Aller au coup #8
10. Aller au coup #9

-> Développé avec **React/TypeScript** et stylisé en **CSS**.

JOUR 4 (Jeudi 30/05)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Début de la création d'une to-do list en React pour consolider mes bases, après une phase de définition du besoin et de conception des fonctionnalités (ajouter, supprimer, cocher, etc.).

Après-midi :

- Poursuite de la création de la to-do list.
- **14h00 - 15h30** : Réunion de présentation et de partage de connaissances sur Playwright avec une démonstration en direct.
- Finalisation de la to-do list en React avec l'aide de Jordan. Quelques finitions à apporter demain.

JOUR 5 (Vendredi 31/05)

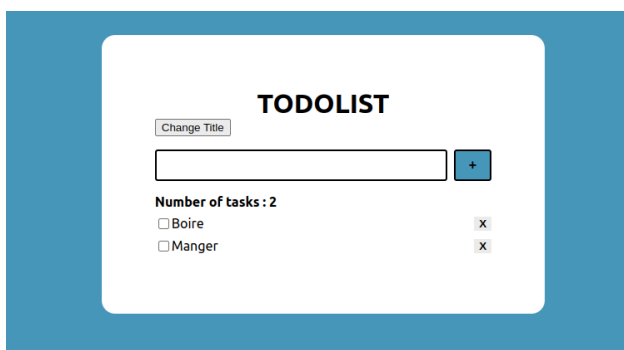
Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Amélioration de la to-do list en React avec un meilleur design.
- **11h00 - 12h00** : Réunion avec l'équipe s'occupant du design system (UX / UI designers, intégrateurs et développeurs front-end).

Après-midi :

- Poursuite de l'amélioration de la to-do list en React.

Rendu final de la To-do list



-> Développé avec **React/TypeScript** et stylisé en **CSS**.

Semaine 2

JOUR 6 (Lundi 03/06)

Matin :

- **9h00** : Point de synchronisation des exercices réalisés.
- **9h15 - 9h30** : Mêlée quotidienne.
- Explication du design system avec Jordan pour clarifier mon projet de stage, afin de créer des composants cohérents et réutilisables qui amélioreront l'efficacité et la qualité du développement.
- **11h00** : Réunion avec les designers et développeurs pour discuter du nouvel élément dans le design system : input.

Après-midi :

- Création d'un nouvel élément dans le design system : le Divider (un séparateur représenté par une ligne). Pour ce projet, j'ai utilisé Storybook (outil), la méthode de conception Atomic Design, et la méthode BEM de E-Liberty pour garantir une structure CSS modulaire et maintenable. Le projet est lié à Github et utilise des composants Figma et ZeroHeight. Premier ticket créé sur Jira.

JOUR 7 (Mardi 04/06)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- **9h30-10h** : Explication approfondie de l'utilisation de Playwright avec Cucumber et Gherkin dans l'entreprise, permettant de simuler des comportements utilisateur pour naviguer sur internet.
- **10h-12h** : Atelier pratique Playwright, exemple avec l'ajout d'un produit dans un panier

Après-midi :

- Création du nouvel élément "Divider" terminée
- **16h30-17h30** : Présentation d'un nouveau site publié, [Pic du Midi](#), développé par l'agence digitale UAM rattachée à e-Liberty, depuis le design jusqu'au développement.
- **17h30-18h** : Goûter commun pour célébrer la nouvelle publication (site Pic du Midi)

JOUR 8 (Mercredi 05/06)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Stylisation du composant "Divider" avec Sass avec les design tokens de E-Liberty.
- Création des stories du composant "Divider", pour le documenter sur Storybook

Après-midi :

- Test du composant "Divider" avec Jest en utilisant react-testing-library.
- Optimisation de mon environnement de travail : configuration des raccourcis puis ajout d'extension telles que Prettier, Local History, Path intellisense, SonarLint, Turbo Console Log, VsCode-icons, ...
- Apprentissage des commandes Github et explication de son fonctionnement en entreprise.

Rendu final du composant "Divider" :

Divider

The Divider component provides a thin, unobtrusive line for grouping elements to reinforce visual hierarchy.

For more information, see ZeroHeight documentation: [Divider Documentation](#)



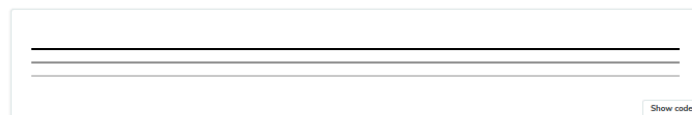
Name	Description	Default	Control	
color	The color of the Divider. <code>DividerColor</code>	<code>DividerColor.Black</code>	Black	

STORIES

Default



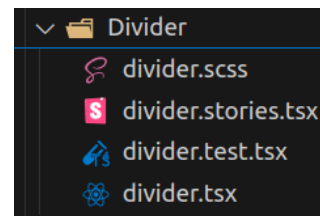
Colors



Vertical



Horizontal



-> Développé avec **React/TypeScript** et stylisé avec **Sass** (en utilisant des **design tokens**). Documentation via **Storybook**. Tests effectués avec **Jest**.

Lien Storybook :

<https://eliberty-design-system-react-storybook.s3.eu-west-1.amazonaws.com/prod/index.html?path=/docs/components-divider--documentation>

JOUR 9 (Jeudi 05/06)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Création d'un nouveau composant dans le design system, le "Calendar" : un calendrier. Avec les mêmes outils et méthodes que le composant Divider. Réutilisation d'un composant (venant d'une librairie npm) déjà créé ([site](#)) puis modification du style et du comportement.
- **11h-11h30** : Discussion avec les designers sur le design system, présentation du nouveau composant créé (Divider), puis mise au clair sur le composant "Calendar"

Après-midi :

- Conception du nouveau composant "Calendar" à partir d'un composant existant, intégrant Sass et une librairie déjà munie du code HTML. Mise en place de l'environnement avec les fichiers nécessaires : stories, tests, Sass, TypeScript React, etc.

JOUR 10 (Vendredi 06/06)

- **9h15 - 9h30** : Mêlée quotidienne.
- Design du composant "Calendar" avec Sass

Semaine 3

JOUR 11 (Lundi 10/06)

- **9h15 - 9h30** : Mêlée quotidienne.
- Suite développement du composant "Calendar" avec Sass et React

JOUR 12 (Mardi 11/06)

- **9h15 – 9h30** : Mêlée quotidienne.
- **9h45–10h** : Réunion axée sur les objectifs des développeurs
- Développement du composant Calendar : partie React et les Stories pour l'affichage sur Storybook.

JOUR 13 (Mercredi 12/06)

- **9h15 - 9h30** : Mêlée quotidienne.
- Développement du composant "Calendar" avec React, finition des stories et début d'implémentation des tests.

JOUR 14 (Jeudi 13/06)

- **9h15 - 9h30** : Mêlée quotidienne.
- Phase d'implémentation des tests sur le composant "Calendar".
Tests réalisés :
 - Devrait s'afficher correctement avec les propriétés par défaut
 - Devrait afficher un point sous la date du jour
 - Si la date présélectionnée correspond à la date souhaitée
 - Si la date indisponible n'est pas sélectionnable
 - S'assurer que la range affichée soit correcte
 - S'assurer que la date sélectionnée correspond à la date renvoyée par l'événement onClick
- **14h** : Réunion avec l'équipe du design system pour discuter des nouveaux composants à ajouter. Présentation du composant créer : "Divider", puis de l'avancement du composant "Calendar".

JOUR 15 (Vendredi 14/06)

- **9h15 - 9h30** : Mêlée quotidienne.
- L'implémentation des tests avec Jest sur le composant "Calendar" est terminée et opérationnelle. Tests réalisés sur ce composant :
 - Devrait s'afficher correctement avec les accessoires par défaut => Fait
 - Devrait afficher un point sous la date du jour => Fait
 - Si la date présélectionnée correspond à la date souhaitée => Fait
 - Si la date indisponible n'est pas sélectionnable => Fait
 - S'assurer que la range affichée soit correcte => Fait
 - S'assurer que la date sélectionnée correspond à la date renvoyée par l'événement onClick => Fait
- Début d'exportation du projet sur Github, mais pas réussi à aller au bout.

Semaine 4

JOUR 16 (Lundi 17/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Exportation du projet sur **Github** en utilisant de nouvelles **commandes** apprises :
 - **git stash** -> met en attente les modif
 - **git checkout main** -> aller sur la branche main
 - **git pull upstream main** -> se mettre à jour avec le code actuel
 - **git checkout -b <nomBranche>** -> aller sur la branche nomBranche (rajouter -b pour créer la branche en même temps)
 - **git stash pop** -> récupérer les changements mis en attente
 - **git add nomFichier** ou **.** -> pour ajouter un ou tous les fichiers
 - **git commit -m "message"** -> ensemble des modifs sous un même titre
 - **git push** -> passer de serveur local à distant

Après-midi :

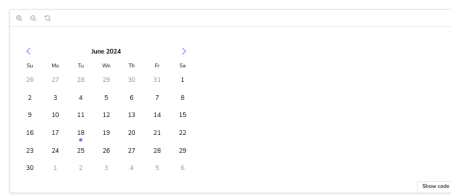
- Correction des bugs et du code du composant "Calendar"
- Nouveau composant "Textinput" à faire. Installation du projet.

Rendu final du composant "Calendar" :

Calendar

A calendar allows a user to select a date or a range of dates.

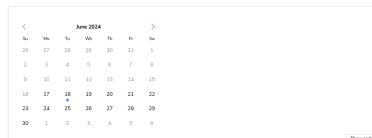
For more information, see Zeroheight documentation: [Calendar Documentation](#)



Calendar component preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. Below the calendar is a table of props.

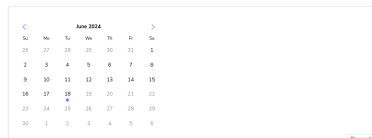
Name	Description	Default	Control	Type
minDate	The minimum selectable date	-	<code>[['m' / 'a' / 'a' / 'a']]</code>	Date
maxDate	The maximum selectable date	-	<code>[['m' / 'a' / 'a' / 'a']]</code>	Date
selectedDate	The start of the selected range	-	<code>[['m' / 'a' / 'a' / 'a']]</code>	Date
rangeDuration	The duration of the range in days	1	Set number	number
excludedDates	Dates that the user cannot select	[]	Set object	Date[]
locale	The locale	'en-US'	Choose option...	string
onClick	Triggered when the user selects a day	(date: Date) => void	-	-

Min Date



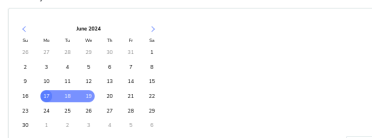
Min Date calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The minimum date is set to June 1st.

Max Date



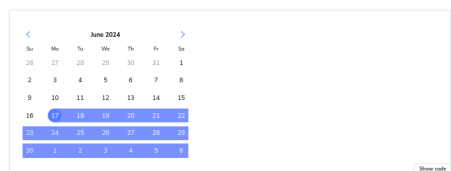
Max Date calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The maximum date is set to June 1st.

Three Days Duration



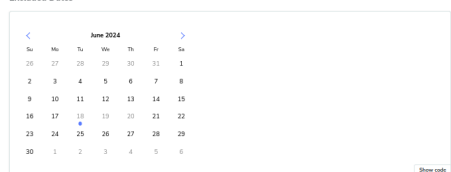
Three Days Duration calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The duration is set to 3 days.

Three Weeks Duration



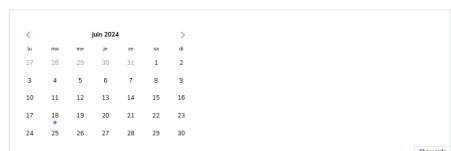
Three Weeks Duration calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The duration is set to 3 weeks.

Excluded Dates

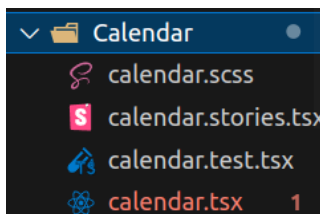


Excluded Dates calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The excluded dates are June 1st and 2nd.

French Locale



French Locale calendar preview showing a date selection interface for June 2024. The interface includes a calendar grid with days of the week and dates. The locale is set to French.



→ Développé avec **React/TypeScript** et stylisé avec **Sass** (en utilisant des design tokens). Documentation via **Storybook**. Tests effectués avec **Jest**.

Lien Storybook :

https://elibrary-design-system-react-storybook.s3.eu-west-1.amazonaws.com/branch/feat_DS-34-calendar/index.html?path=/docs/molecules-calendar--documentation

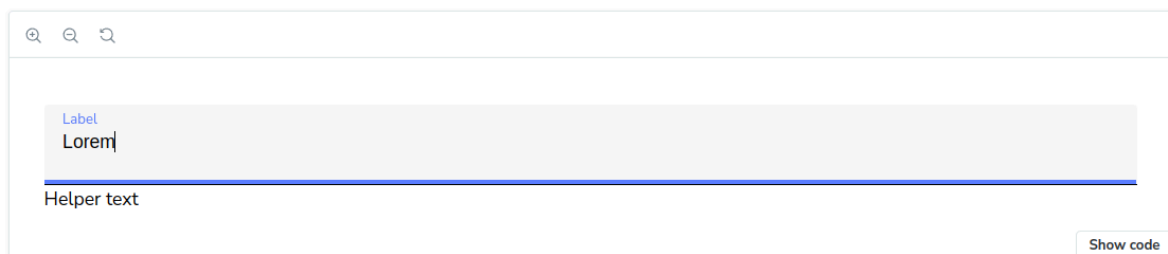
JOUR 17 (Mardi 18/06)

Matin :

- **9h15 – 9h30 : Mêlée quotidienne.**
- Mise au point du compte rendu selon les critères donnés.

Après-midi :

- Suite du développement du composant "TextInput" avec React. Ajout du Helper text ainsi que le Placeholder avec Sass et React.



JOUR 18 (Mercredi 18/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Suite du développement du composant "TextInput". Partie React presque terminée.

Après-midi :

- Début de développement des stories pour Storybook avec Jest (react testing librairie). Poursuite de la stylisation du composant avec Sass.
- Différents états du composant "Textinput" créé : disabled, required, helperText, errortext. Ajout des icônes. Ajout de fonctions.
- Retour de bugs sur le composant "Calendar" par une designeuse avec Safari sur Iphone :
 - On a un petit bug d'affichage (en mobile uniquement) au niveau des pictos de navigation : "previous month" et "next month" qui apparaissent sur l'icône.
 - On a également ce bug où l'état "hover" (normalement non visible en mobile) apparaît après avoir cliqué sur un jour et reste visible même après mise à jour de l'affichage du calendrier.
- Présentation de l'utilisation de AWS dans l'entreprise

-> progression du composant TextInput :
https://eliberty-design-system-react-storybook.s3.eu-west-1.amazonaws.com/branch/feat_DS-34-calendar/index.html?h=/story/molecules-calendar--default

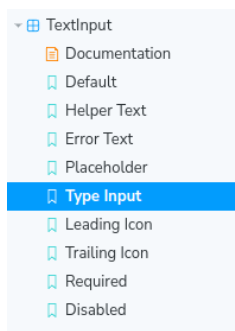
JOUR 19 (Jeudi 20/06)

Matin :

- **9h15 – 9h30 : Mêlée quotidienne.**
- Développement côté React du composant TextInput terminé
- **11h-12H :** Réunion avec le groupe Design System pour les avancés par rapport à la semaine dernière. Discussion des bugs du “Calendar”, ajout de nouveaux éléments dans le design system (icons), etc.

Après-midi :

- Travail en autonomie cet après-midi
- Ajout de toutes les stories sur Storybook avec les différentes fonctionnalités du composant “TextTinput”:



- Poursuite de la stylisation du composant “TextInput”
- Reprise des bugs du composant “Calendar” avec une Intégratrice :
 - Bug d’affichage (Iphone sur Safari uniquement) au niveau des pictos de navigation : “previous month” et “next month” qui apparaissent sur les icônes de navigation.
 - Raison : display flex n’est pas compatible avec le text-indent, à remplacer par display block.
 - A retenir : tester ses composants sur mobile également et pas non seulement sur ordinateur

JOUR 20 (Vendredi 21/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Poursuite du développement du composant "textinput".

Après-midi :

- Poursuite du développement du composant "Textinput".
- Correction des bugs du composant "Calendar", suivi d'une mise en production sur le main dans Github pour le rendre accessible à l'ensemble des utilisateurs.

Semaine 5

JOUR 21 (Lundi 24/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Suite du développement du composant "Textinput".

Après-midi :

- Développement du composant "Textinput" terminé -> plus qu'à faire confirmer par Jordan puis réaliser les tests sur ce composant.

JOUR 22 (Mardi 25/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Restructuration du composant "Textinput" avec Jordan, pour avoir un code plus lisible.
- **11h-12h :** Réunion avec les designers pour discuter du design system. Discussion sur mes avancés : correction des bugs Calendar, avancement du composant textinput ainsi que quelques modifs pour le "Textinput"

Après-midi :

- Grosses modifications du composant "Textinput" avec une meilleure structure et les même fonctionnalités (qui sont reprises)
- **16h -17h30 :** Participation à un apéro avec tous les membres ainsi que le CEO pour discuter des avancées.

JOUR 23 (Mercredi 26/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Correction des problèmes du "Textinput" avec Jordan puis avec une intégratrice :
 - Structure : comment enlever la classe Textinput_padding => hauteur de 56px mais sans padding, comment faire pour avoir 56px avec le padding ?
 - Pourquoi height 100% est plus grand que le parent ? (48px) => input / label notamment
 - Comment faire en sorte que le clique sur l'input passe par dessus la div de label ?
 - Mettre les line height dans le design vu que c'est sur 1 ligne?
 - Margin-bottom pour pas faire bouger les éléments en dessous

Après-midi :

- Suite développement "Textinput" et optimisation du code.

JOUR 24 (Jeudi 27/06)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Travail sur le composant "Textinput" avec Jordan, dans le but d'optimisation de celui-ci => comportements du composant regroupés (focused, filled, disabled, error), simplification de la partie React, Sass, etc.

Après-midi :

- Ajout des design tokens dans le design system car certains étaient en manque + synchronisation dans AWS.
- Enregistrement des avancées du composant "Textinput" sur Github avec les commandes apprises.
- Composant "Textinput" réellement terminé, plus que des tests unitaires à réaliser sur celui-ci.
- Liste des futures tests à réaliser sur le composant "Textinput" :
 - Vérifier que le label s'affiche correctement
 - Vérifier que la value passée soit correctement affichée dans l'input
 - Simuler un changement dans l'input et vérifier que la fonction onChange est appelée avec la valeur correcte
 - Vérifier que le state local inputValue est mis à jour correctement après un changement dans l'input
 - Vérifier que le leadingIcon soit affiché correctement lorsque leadingIconName est spécifié
 - Vérifier que showTrailingIcon s'affiche et fonctionne correctement en effaçant le contenu de l'input
 - Vérifier que l'input est required lorsque additionalLabelText est : Required ou Asterisk
 - Vérifier que le placeholder soit affiché lorsqu'il est défini
 - Vérifier que helperText est affiché correctement lorsque helperText est spécifié

- Vérifier que le type de l'input est correctement appliqué en fonction du type
- focused
- Vérifier que la classe CSS 'TextInput-filled' est appliquée lorsque l'input contient une valeur.
- Vérifier que le composant se désactive lorsque disabled est true
- Vérifier que le style d'erreur fonctionnent correctement lorsque errorText est spécifié

JOUR 25 (Vendredi 28/06)

Matin :

- **9h15 – 9h30 : Mêlée quotidienne.**
- Correction des derniers bugs du composant “Calendar” :
 - Lors du hover sur l'un des boutons, un texte apparaît en pseudo-élément :before avec les mentions 'Previous month' et 'Next month' => Texte n'apparaît plus lors du hover.
 - Lors du survol (hover), la couleur de survol du bouton ne s'applique pas.
 - Enlever le survol (hover) sur les dispositifs n'ayant pas la propriété.
- MAJ de la branche du composant “Calendar” sur Github.
- Création des premiers tests sur le composant “Textinput”.

Après-midi :

- Liste de tous les tests réalisés sur le composant “Textinput” :
 - Vérifier que la valeur passée soit correctement affichée dans l'input + Vérifier que la classe CSS 'TextInput-filled' est appliquée lorsque l'input contient une valeur.
 - Simuler un changement dans l'input et vérifier que la fonction onChange est appelée avec la valeur correcte
 - Vérifier que le leadingIcon soit affiché correctement lorsque leadingIconName est spécifié
 - Vérifier que showTrailingIcon s'affiche et fonctionne correctement en effaçant le contenu de l'input
 - Vérifier que l'input est required lorsque additionalLabelText est : Required ou Asterisk + affichage de la div
 - Vérifier que le placeholder soit affiché lorsqu'il est défini + class 'textInput-interactive'
 - Vérifier que helperText est affiché correctement lorsque helperText est spécifié
 - Vérifier que le type de l'input est correctement appliqué en fonction du type
- Tests restants :
 - Vérifier focused
 - Vérifier que le composant se désactive lorsque disabled est true
 - Vérifier que le style d'erreur fonctionnent correctement lorsque errorText est spécifié

Semaine 6

JOUR 26 (Lundi 01/07)

Matin :

- **9h15 - 9h30 : Mêlée quotidienne.**
- Réalisation des tests unitaires sur le composant "Textinput" terminé.

Après-midi :

- Amélioration et correction des tests unitaires ainsi que quelques changements vu avec Jordan.
- Composant push sur Github.

Rendu final du composant "TextInput" :

TextInput

Text inputs are used to gather text-based input using a keyboard. They are a type of input that allows users to enter and edit text.

For more information, see Zeroheight documentation: [Text input Documentation](#)

Name	Description	Default	Control
label	Field name	'Label'	Label
type	Type of input	'text'	Text
value	Value entered by the user	-	Set string
leadingIconName	Name of the icon to be displayed (left)	-	Choose option...
showTrailingIcon	Show the trailing icon (right)	-	Set boolean
additionalLabelText	An additional text for the Label	-	Choose option...
placeholder	Additional information about the field	-	Set string
helperText	Helper text	-	Set string
disabled	Input disabled	-	Set boolean
errorText	Error text	-	Set string
onChange*	Change the input text	-	-

Placeholder

Q Label Placeholder X

Helper Text

Label
Helper Text

Types

Label

Label

Label
g/m/m/aaaa

Label

Label

Label

Disabled

Q Label (Required) X

Error Text

Q Label (Required) X

Error Text

STORIES

Default

Label

Value

Q Label (Required) X

Lorem ipsum dolor sit amet

Leading Icon

Q Label

Show Trailing Icon

Label X

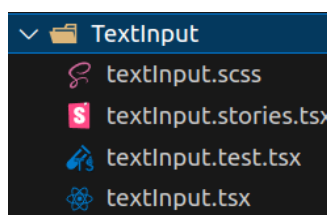
Additional Label Text

Label

Label (Required)

Label (Optional)

Label *



-> Développé avec **React/TypeScript** et stylisé avec **Sass** (en utilisant des design tokens). Documentation via **Storybook**. Tests effectués avec **Jest**.

Lien Storybook :

https://elibrary-design-system-react-storybook.s3.eu-west-1.amazonaws.com/branch/feat_DS-20-textinput/index.html?path=/docs/components-textinput--documentation

JOUR 27 (Mardi 02/07)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Nouveau composant à développer : Checkbox
- Création du nouveau composant "Checkbox" (composant simple et rapide).
 - Déclarations des props en React
 - label : string (obligatoire)
 - type : enum string
 - (none)
 - check
 - indeterminate
 - state : enum string
 - disabled
 - error
 - size : enum string (M)
 - onClick : fonction renvoi booléen
 - Stylisation du composant ainsi que ses différents états.
 - Stories sur Storybook

Après-midi :

- Poursuite du développement => presque terminé !

JOUR 28 (Mercredi 03/07)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Poursuite du développement du composant "Checkbox".
- Composant React ainsi que stylisation terminée.
- Stories terminées :
 - Default
 - Sizes
 - Types
 - Disabled State
 - Error State
- Réalisation des tests unitaires sur le composant :
 - Vérifier que le composant est correctement affiché avec les props par défaut (label) + non check, type check par défaut, sans états
 - Vérifier que la classe CSS correcte est appliquée pour une taille
 - Vérifier que la props checked est affichée lorsqu'elle est true, et, n'est pas affichée lorsqu'elle est false
 - Vérifier que la classe CSS correcte est appliquée pour chaque type ('check', 'indeterminate') + icon correctement affiché pour chaque type
 - Vérifier que la checkbox est désactivée lorsque l'état est disabled + qu'il ne soit pas cliquable
 - Vérifier que la classe CSS correcte est appliquée pour l'état error
 - Vérifier que le composant appelle la fonction onClick avec la valeur correcte lorsque la case est cochée/décochée.
- Composant push sur Github => attends les retour sur celui-ci par la designeuse via le ticket Jira.
- Retour de bugs sur le composant "Textinput" par une designeuse :
 - - La taille des icônes (leading et trailing) n'est pas cohérente avec la maquette. Le container de l'icon est bien à 24 x 24 pixels mais l'icon en lui-même semble être à 14 px au lieu de 18, à voir comment corriger ça car là il est vraiment super petit.
 - - Pour les déclinaisons S3V et MBNR, les tokens pour le label gardent ceux de Redpill (c'est le cas pour les couleurs et les typos, je ne peux pas recetter les spacing, à vérifier de votre côté du coup)

Rendu final du composant "Checkbox" :

Checkbox

A checkbox allows users to select one or more values from a small set of options.

For more information, see ZeroHeight documentation : [Checkbox Documentation](#)

☐ Label

Show code

Name	Description	Default	Control
label	Label for the checkbox (string)	'Label'	Label
size	Size of the checkbox (CheckboxSize)	CheckboxSize.M	M
checked	Checkbox is checked (boolean)	false	Set boolean
type	Selection state of the checkbox (CheckboxType)	CheckboxType.Check	Choose option...
state	State of the checkbox (CheckboxState)	-	Default
onClick*	Click handler for the checkbox (checked: boolean) => void undefined	-	-

STORIES

Default

☐ Label

Show code

Sizes

☐ Label ☐ Label ☐ Label ☐ Label

Show code

Types

☐ Label ☒ Label ☐ Label

Show code

Disabled State

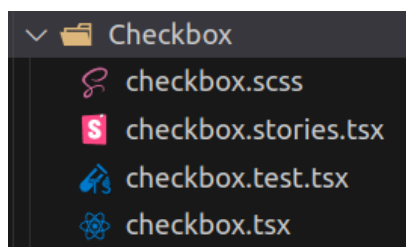
☐ Label ☐ Label ☐ Label

Show code

Error State

☐ Label ☒ Label ☐ Label

Show code



-> Développé avec **React/TypeScript** et stylisé avec **Sass** (en utilisant des design tokens). Documentation via **Storybook**. Tests effectués avec **Jest**.

Lien Storybook :

https://eliberty-design-system-react-storybook.s3.eu-west-1.amazonaws.com/branch/feat_DS-27-checkbox/index.html?path=/docs/components-checkbox--documentation

JOUR 29 (Jeudi 04/07)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.
- Correction bugs du composant "Textinput" avec Jordan. Manque quelques bugs à corriger.
- Remplissage du tableau de synthèse avec Jordan.

Après-midi :

- **13h-14h** : Conférence e-Liberty x Johan Martinsson : "3P pour sortir du legacy", lien site <https://martinsson-johan.blogspot.com/>
lien diapo :
<https://speakerdeck.com/martinsson/getting-out-of-legacy-with-3p-protect-prepare-produce-techexcellence-meetup>
- Correction bugs du composant "Textinput" avec intégrateur
- Retour sur le composant "Checkbox" par la designeuse. Tout semble ok.
- Préparations pour le composant "RadioButton".

JOUR 30 (Vendredi 05/07)

Matin :

- **9h15 - 9h30** : Mêlée quotidienne.

Après-midi :

- Questions / réponses avec développeur back-end - directeur technique :
 - Peux-tu décrire brièvement ton rôle et tes principales responsabilités ?
 - Quels langages de programmation et frameworks utilises-tu le plus souvent ?
 - Quels outils utilises-tu pour le contrôle de version et la gestion du code ?
 - Comment gère-tu les bases de données (types de bases de données et outils) ?
 - Peux-tu décrire le processus de développement, de la conception à la mise en production ?
 - Quelles sont les meilleures pratiques pour assurer la qualité du code ?
 - Quels conseils donnerais-tu à quelqu'un qui débute en développement back-end ?
 - Comment reste-tu à jour avec les nouvelles technologies et tendances dans le développement back-end ?
- Réponses :
 - Rôle et responsabilités :
 - Prise de décisions techniques, veille technologique, développement.
 - Aide et partage d'expérience avec l'équipe.
 - 15 ans d'expérience dans le domaine.
 - Langages et frameworks :
 - Utilise principalement PHP Symfony.
 - Choix des technologies dépendant des besoins (Python, Node.js, etc.).
 - Outils de contrôle de version :
 - Utilisation de Git.
 - Gestion des bases de données :
 - Utilisation de PostgreSQL pour son rapport qualité-prix (gratuit), meilleur du marché selon lui.
 - Confirme que PostgreSQL était un bon choix.

- Processus de développement :
 - Identification des besoins clients.
 - Création de maquettes et spécifications.
 - Définition des tâches et architecture du système.
 - Codage, tests, CI avec GitHub, déploiement en production.
 - Bonnes pratiques :
 - Réalisation de tests unitaires.
 - Importance de la clarté (nommage) et de la structure du code.
 - Écrire le minimum de code nécessaire et le maintenir à jour.
 - Conseils :
 - Importance de la compréhension plutôt que de tout savoir.
 - Ne pas se reposer uniquement sur l'IA.
 - Lire, comprendre les problématiques, et s'appropriier le code.
 - Mise à jour des compétences :
 - Documentation et curiosité.
 - Abonnement à des plateformes de développement (comme daily.dev).
 - Se tenir informé des dernières tendances
-
- Questions / réponses avec support client :
 - Peux-tu décrire une journée typique dans le support client ?
 - Une journée typique implique de recevoir des demandes clients, principalement par mails et tickets, car il y a peu de contacts par téléphone dans cette entreprise.
 - Quelles sont tes principales responsabilités ?
 - Les principales responsabilités incluent maintenir un service de qualité, être efficace et rapide dans la prise en charge des demandes des clients.
 - Quelles compétences sont essentielles pour ce rôle ?
 - Les compétences essentielles comprennent la capacité à résoudre des casse-têtes et des énigmes, à comprendre la situation des clients, et à posséder des compétences spécifiques au domaine de l'entreprise.
 - Comment gères-tu le stress avec les clients ?
 - Il est crucial de prioriser les clients tout en restant efficace et rapide, de gérer le stress des clients, de comprendre la notion d'urgence, et de gérer les afflux de travail selon les fortes saisons de l'entreprise.
 - Quels outils utilises-tu régulièrement ?
 - Les outils utilisés régulièrement incluent des outils de ticketing, comme Jira, et le back-office de l'entreprise.

- Peux-tu décrire le processus de traitement d'une demande de support ?
 - Le processus commence par la réception d'une demande client. Ensuite, il faut s'affecter la demande. Trois scénarios peuvent se présenter :
 - Le support peut résoudre le problème seul.
 - Le ticket est transféré à une personne plus compétente.
 - Le ticket est envoyé aux développeurs lorsque ce sont eux qui doivent résoudre le problème.
- Quels conseils donnerais-tu à quelqu'un qui débute dans le support client ?
 - Les conseils incluent de s'accrocher, d'être curieux, d'avoir une bonne compréhension des solutions de l'entreprise, et de bien comprendre les demandes et les problèmes des clients.
- Comment restes-tu à jour avec les nouvelles technologies ?
 - Bien que non spécifié directement, rester à jour avec les nouvelles technologies pourrait inclure suivre des formations continues, lire des articles et des blogs technologiques, participer à des webinaires et des conférences, et échanger avec des collègues et des experts du domaine.
- Questions / réponses avec Administrateur ops / devops cloud :

- Quel est le rôle principal d'un Administrateur Ops/DevOps Cloud ?
 - Le rôle principal d'un Administrateur Ops/DevOps Cloud est de gérer les serveurs et les infrastructures sur des plateformes comme AWS. Cela inclut la planification des ressources nécessaires, la configuration des connexions réseau, ainsi que la gestion administrative des systèmes. Du côté DevOps, l'accent est mis sur l'automatisation des déploiements de serveurs et de code, ainsi que sur l'intégration continue des changements dans les environnements de production.
- Quels outils et technologies sont couramment utilisés dans ce domaine ?
 - Les outils couramment utilisés comprennent AWS pour l'infrastructure cloud, les systèmes d'exploitation Linux pour la gestion des serveurs, et des outils d'automatisation tels que Packer, Docker, ainsi que des plateformes de CI/CD comme Jenkins. Les machines virtuelles (VM) sont également utilisées pour créer des environnements isolés et flexibles.
- Peux-tu décrire une journée typique pour un administrateur Ops/DevOps Cloud ?
 - Une journée typique commence par la surveillance des systèmes en place, suivie par la mise à jour des services et des logiciels. On configure et on met en réseau de nouveaux services, tout en s'assurant que les processus de déploiement sont automatisés pour optimiser l'efficacité opérationnelle.
- Comment les processus de déploiement et de gestion des applications sont-ils automatisés ?
 - Les processus de déploiement et de gestion des applications sont automatisés à l'aide de pipelines d'intégration continue (CI) et de déploiement continu (CD). Ces pipelines permettent de livrer rapidement et de manière fiable les changements logiciels tout en assurant la qualité des déploiements.
- Quelles compétences techniques et qualités personnelles sont essentielles dans ce domaine ?
 - Les compétences techniques essentielles incluent une expertise dans l'utilisation des services cloud comme AWS, une solide maîtrise des systèmes d'exploitation Linux, ainsi que des compétences avancées en automatisation et en gestion des infrastructures. Sur le plan personnel, la curiosité intellectuelle, la capacité à rester à jour avec les nouvelles technologies, et une volonté d'apprentissage continu sont cruciales pour réussir.
- Comment surveiller les performances et la disponibilité des services cloud ?
 - Pour surveiller les performances et la disponibilité des services cloud, on utilise des outils tels que AWS Management Console, Grafana pour les tableaux de bord détaillés, et Datadog pour recevoir des notifications sur les actions réalisées et les incidents éventuels.
- Quels conseils donnerais-tu à ceux qui débutent dans ce domaine ?
 - Pour ceux qui débutent dans ce domaine, il est recommandé de développer une forte curiosité pour les nouvelles technologies et de s'engager dans un apprentissage continu. Il est crucial de pratiquer régulièrement avec les outils et les plateformes cloud, de suivre les certifications pertinentes, et de rester informé des évolutions dans le domaine de l'Ops/DevOps Cloud.

Conseils :

Organisation et Méthodologie

- Être organisé dans son travail : diviser les tâches en étapes (par exemple, créer une to-do list) et réfléchir au procédé.

Jordan

- Finir ce qu'on a commencé : ne pas entreprendre plusieurs tâches en parallèle.

Jordan

Code et Conventions

- Ne laisse pas le code te déranger : éviter d'utiliser des balises comme `<p>` si elles risquent de casser avec les changements de conventions; préférer les `<div>`.

Xavier

- Déclaration des éléments et fonctions : trier les déclarations en mettant les éléments en haut et les fonctions en bas.

Jordan

- Éviter les répétitions dans le code.

Jordan

- Mettre régulièrement son code à jour.

Maxime

Apprentissage et Compétences

- Comprendre profondément ce que l'on apprend : il est préférable de bien comprendre moins de choses que de connaître superficiellement beaucoup de sujets.

Xavier

- Documenter ses actions et créer du contenu.

Camilo

- Cultiver la curiosité et partager ses découvertes et compréhensions.

Camilo

- Partager ses compétences et apprentissages sur des plateformes comme YouTube ou Medium pour assurer une traçabilité et développer une audience ou une communauté.

Camilo

Persévérance et Compréhension

- S'accrocher dans les situations difficiles et rester persévérant.

Noémie

- Cultiver la curiosité pour mieux comprendre les besoins des clients.

Noémie

- Acquérir une bonne compréhension des solutions proposées par l'entreprise.

Noémie

- Bien comprendre les demandes et les problèmes des clients pour offrir un service efficace.

Noémie

Curiosité et Innovation

- Cultiver une curiosité pour les nouvelles technologies et les méthodes innovantes.

Swan

- Investir dans un apprentissage continu et autodirigé.

Swan

- Pratiquer régulièrement pour renforcer ses compétences pratiques.

Swan

- Acquérir des certifications et des qualifications pertinentes pour progresser.

Swan

- Maintenir une veille constante sur les tendances et les développements du secteur.

Swan