Rapport de projet - Plan de l'appartement connecté DOMUS

Rappel du sujet/besoin et cahier des charges

La Plateforme Domus

Appartenant au LIG (Laboratoire d'Informatique de Grenoble) et située à la MACI (Maison de la Création et de l'Innovation), la plateforme Domus est un espace servant à experimenter le comportement humain dans l'environnement d'un appartement connecté.

Domus est équipé d'un appartement de 80m², aménagé pour pouvoir y vivre, d'une régie pour surveiller les expérimentations et de bureaux pour le personnel qui travaille sur le projet. L'appartement connecté possède des capteurs et des effecteurs en tout genre pour contrôler lampes, volets, etc. ou encore consulter la température ou le taux de Co2.

Plan intéractif

Notre projet consiste en l'élaboration d'un plan interactif de cet appartement. Celui-ci doit répondre au fonctionnement suivant : un utilisateur peut cliquer sur les pièces pour en afficher les photos et peut cliquer sur les capteurs et effecteurs pour connaitre leur état et les contrôler. Le plan sera mis sur le site de la plateforme Domus, actuellement en construction, via une balise HTML iframe. Il permet de mettre en avant la plateforme et les expériences qui y ont lieu, via la consultation de photos et des différents éléments de domotique disponibles. Cependant, le contrôle et la consultation de l'état des capteurs/effecteurs n'est possible que si le plan est consulté depuis le réseau de Domus.

Exigences client

Le but principal du plan interactif est d'être utilisé par des utilisateurs extérieurs à la plateforme. Ainsi le plan se doit d'être simple d'utilisation et intuitif. Cela se traduit par des interactions basiques proches de ce qu'une télécomande et/ou des actionneurs physiques pourraient proposer (par exemple, reproduire les 3 boutons qui commandent les volets roulants). Il est également nécessaire que l'utilisateur fasse le rapprochement entre le plan et les éléments qu'il souhaite consulter/contrôler. Ainsi toute information donnée par le plan se doit de faire correspondre la localisation de l'objet physique et son pendant virtuel. Un dernier point demandé est de faire un code facile à réutiliser dans le futur, en particulier sur la possibilité d'ajouter facilement de nouveaux capteurs.

Technologies employées

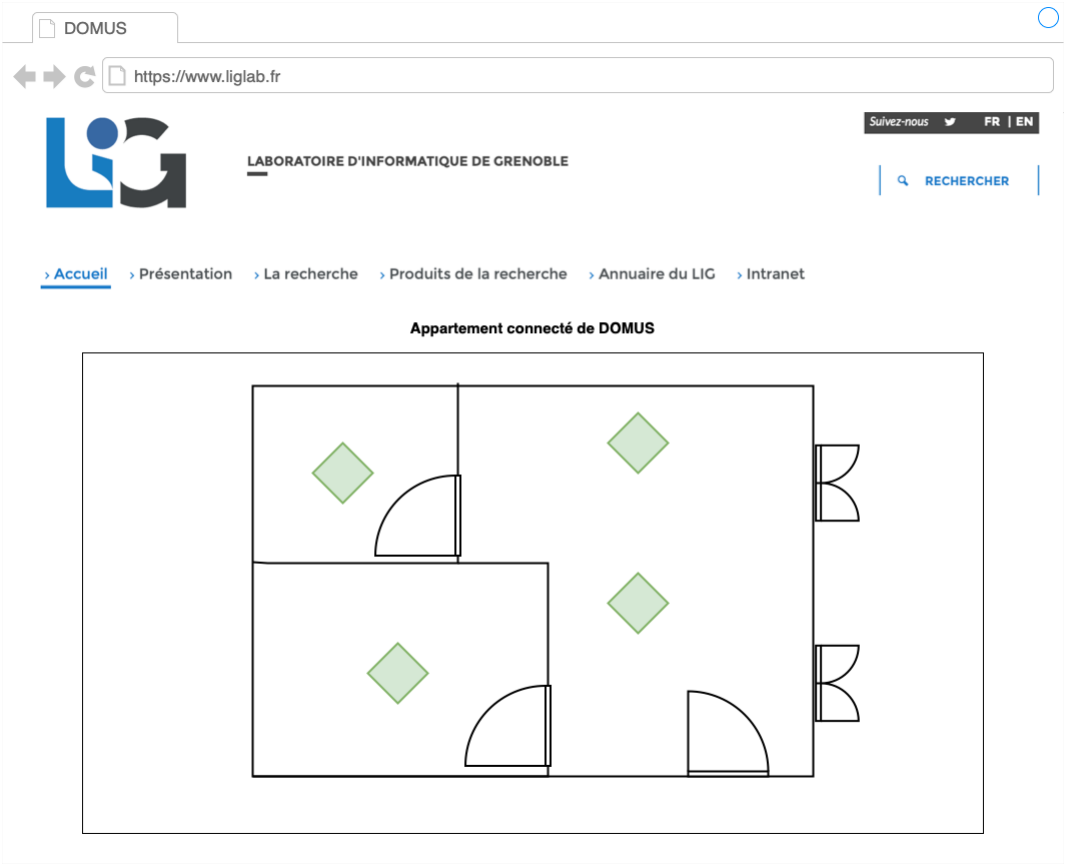
- Vue.js (aussi appelé plus simplement Vue) est un framework JavaScript open-source utilisé pour construire des interfaces utilisateur et des applications web monopages.

- Open Home Automation Bus (openHAB) est un logiciel de domotique open source écrit en Java. Il est déployé dans les locaux et se connecte aux dispositifs et services de différents fournisseurs.

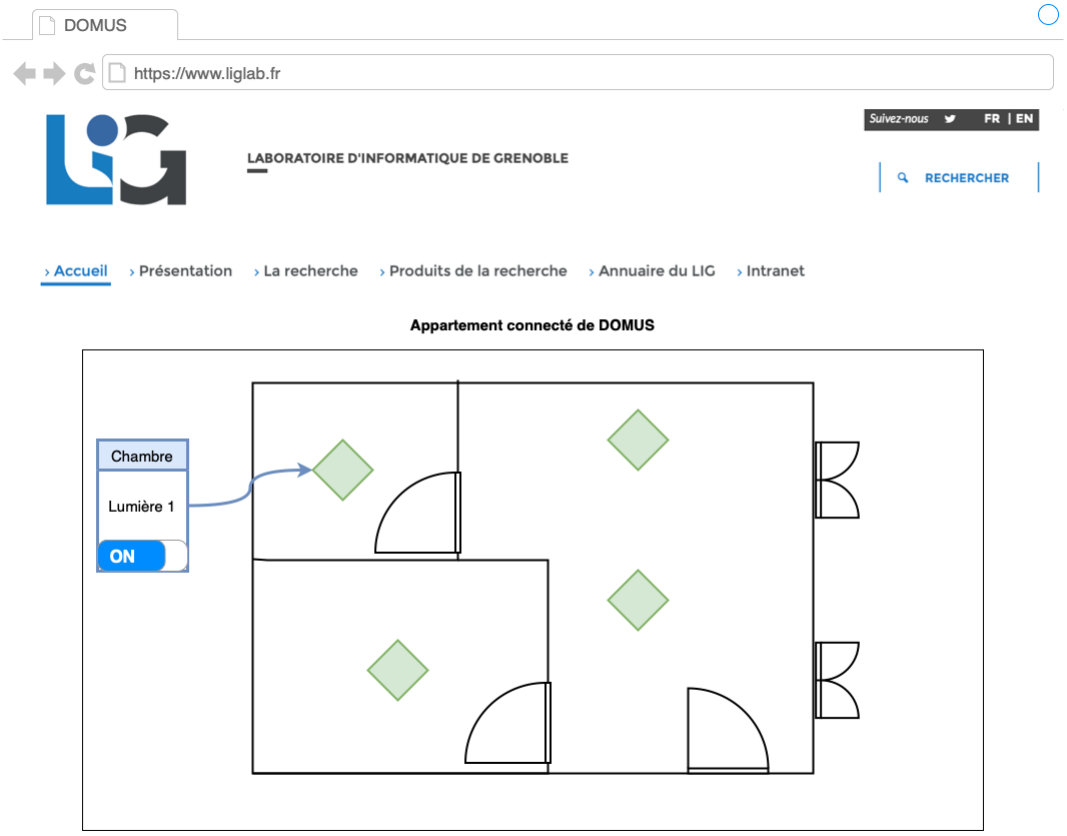
Architecture technique

Front-end

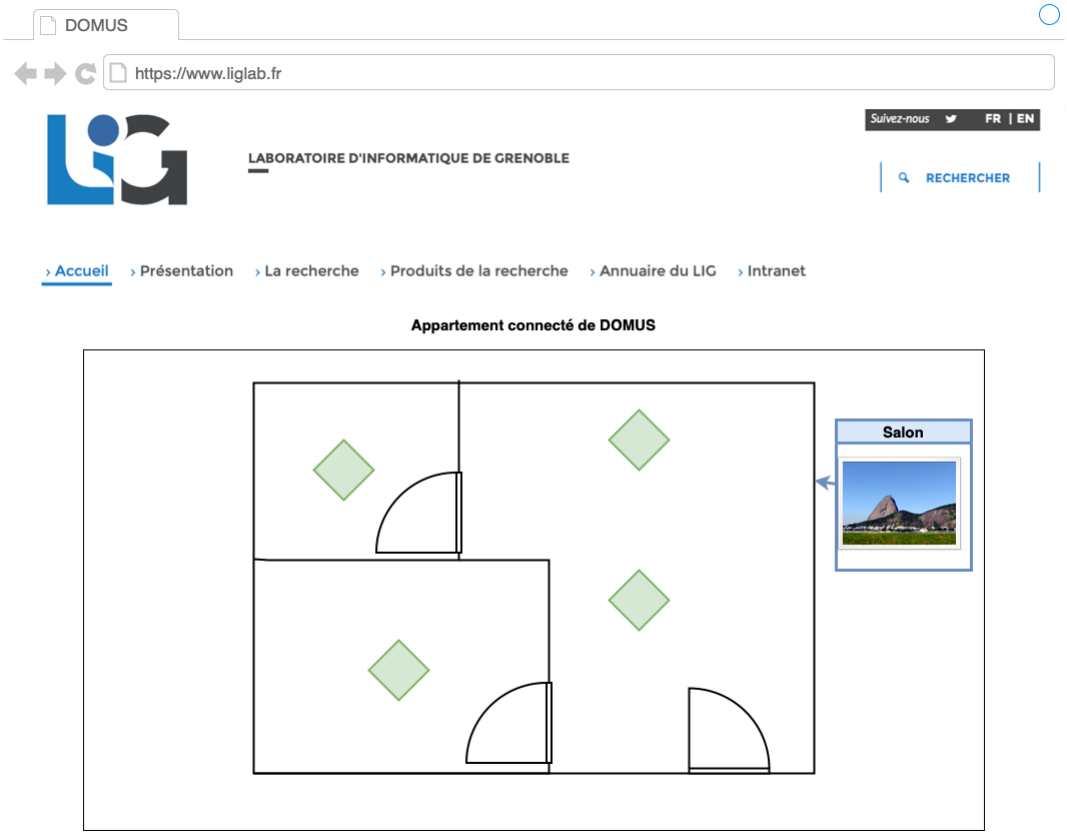
Maquettes du plan dynamique



Plan global



Popup de contrôle d'un appareil



Popup avec photo d'une pièce

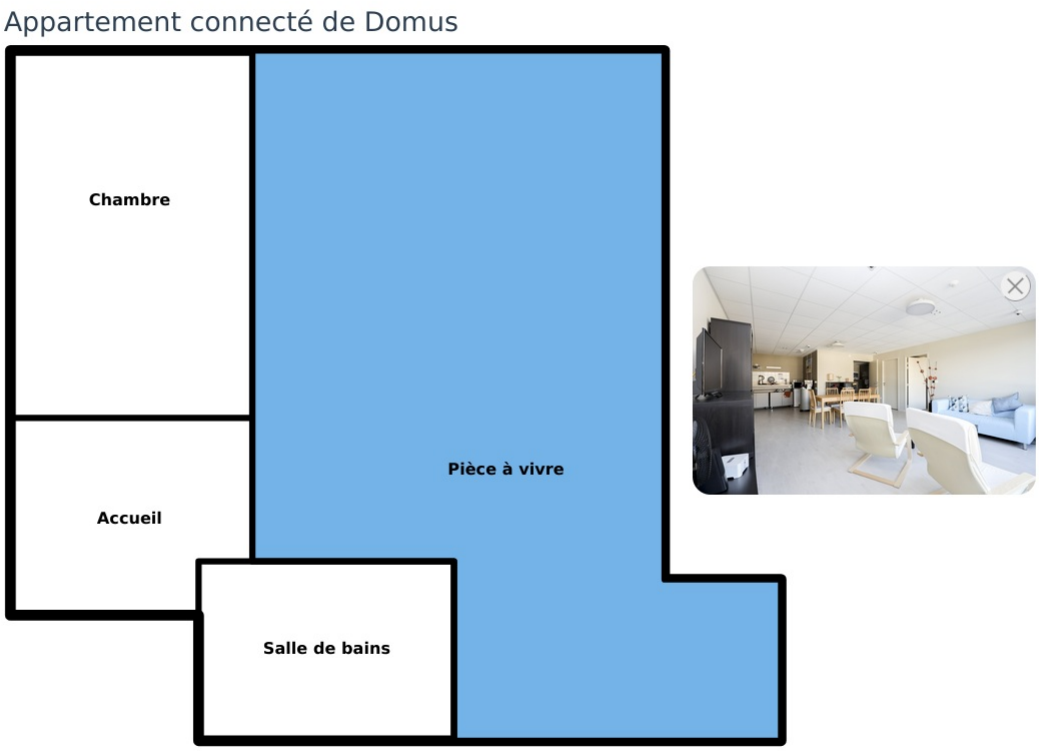
Back-end

En ce qui concerne l'architecture globale back-end, nous avons utilisé l'API d'openHAB qui sert de médiateur entre les différents appareils de domotique et le site web. La communication entre le plan et le réseau OpenHAB se fait par le biais de requêtes HTML sur l'API REST. Ces requêtes étant générées dans les Pop-ups des capteurs/effecteurs, elles se basent sur les informations données par les widgets de contrôle.

Réalisations techniques

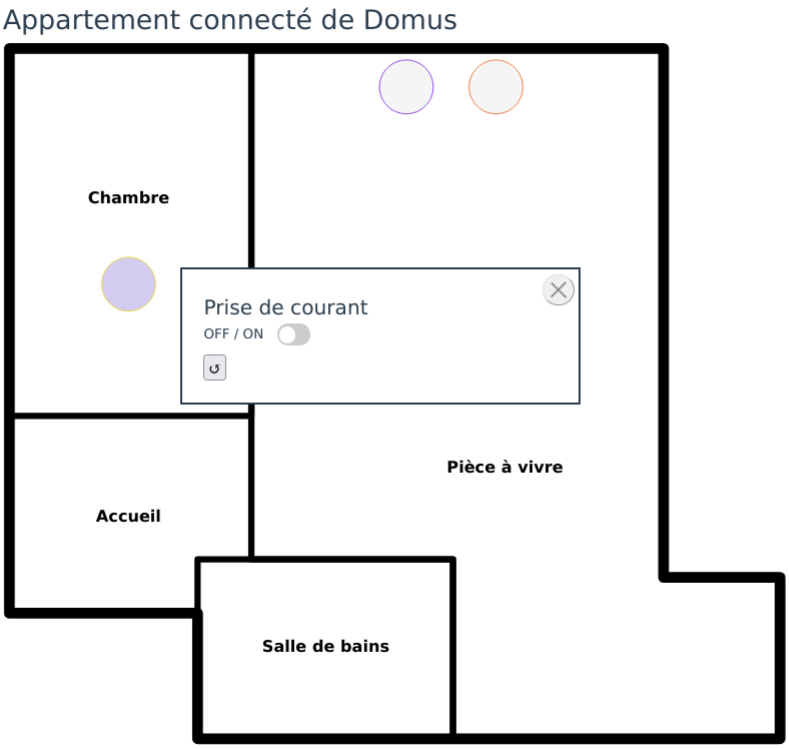
Visualisation des photos

La partie visualisation des photos de l'appartement a pu être entièrement réalisée. Le clic sur une pièce du plan de l'appartement affiche une pop-up contenant une photo de la pièce concernée. Le client souhaitait avoir un retour visuel de la pièce sélectionnée c'est pourquoi cette dernière est colorée sur le plan tant que sa photo est visible. Il était également souhaité que la photo soit située à proximité de la pièce associée sans la cacher sur le plan. À noter que le SAS d'accueil est visible sur le plan mais non cliquable, suite à la demande du client.



Télécommande des effecteurs

Le deuxième aspect du projet consistait à créer une interface permettant de visualiser les informations des capteurs et contrôler les effecteurs de l'appartement connecté, à la manière d'une télécommande. Cet aspect là a été implémenté sous le nom "mode contrôle". Ce mode permet de visualiser sur le plan des éléments de domotiques et d'obtenir des informations les concernant en cliquant dessus. Une fenêtre pop-up s'affiche lors du clic, affichant des informations relative au capteur/effecteur considéré, tel que son nom, une description et son état actuel. Pour les effecteurs (= éléments de domotique controllables), des contrôles associés sont proposés à l'utilisateur. Nous avons créé les widgets de contrôle suivants : bouton d'action (ex : pour baisser ou remonter un store), slider (ex : pour régler l'intensité lumineuse d'une lampe) et bouton switch (ex : pour activer ou désactiver une prise de courant). Du point de vue technique, l'interrogation et le contrôle des capteurs se font en utilisant l'API REST d'OpenHAB, via de simples requêtes GET et POST.



Difficultés et limites

Dans l'idéal de ce que voulait le client, la partie télécommande du site ne devait être visible et accessible que depuis l'intérieur de l'appartement connecté. Cependant, nous avons rencontré des difficultés sur le moyen de savoir si l'utilisateur consultait le site depuis l'appartement ou non. Afin de quand même différencier les deux modes (télécommande et visualisation des photos) nous avons opté pour une solution temporaire, permettant de passer d'un mode à l'autre à l'aide d'un simple bouton. Il faudra faire des recherches supplémentaires afin d'avoir le comportement originellement souhaité.

Nous avons également rencontré des limites en voulant tester le site dans les locaux de l'appartement connecté et non simplement en environnement local sur nos machines, afin de réellement intéragir avec les capteurs. À cause de limitations réseau principalement, le test a été impossible en condition réelle et nous avons finalement dû simuler les objets connectés avec OpenHAB sur nos propres ordinateurs.

Gestion de projet

Méthodes

Le client a tenu à faire une réunion toutes les semaines, les lundis ou jeudis matins afin de connaître notre avancement et de s'assurer que nos visions du projet concordaient. Ces réunions nous ont permis de communiquer de manière fluide et rapide, de poser facilement les questions qui nous venaient au fur et à mesure du projet.

Planning prévisionnel

Nous avons passé les deux premières semaines à développer un prototype basique (en HTML et JavaScript) afin d'avoir une idée du produit final que nous souhaitions développer et afin d'avoir des retours de la part du porteur de projet. Lorsque nous sommes arrivés à une base de projet solide, nous nous sommes ensuite lancés dans une migration vers le framework Vue.js.

Rôles des membres

Oscar avait le rôle de chef de projet. Il définissait les fonctionnalités nécessaires dans le projet en les mettant sous forme de tickets sur Trello et les ordonnait par ordre de priorité. Son rôle lui demandait aussi de valider les fonctionnalités implémentées avant de pouvoir les considérer comme terminées. Margaux a assumé le rôle de responsable communication avec le client, en gérant les questions, les échanges par mail et en notant les remarques faites. Sophie a repris en cours de route le rôle de scrum master (attribué à Théo en début de projet), aidant Oscar sur la création des tickets et l'organisation interne. Amad fut à plein temps sur le développement et a eu comme mission principale d'imaginer et d'implémenter le plan de l'appartement sous le format SVG afin de le rendre interactif.

Outils

Collaboration

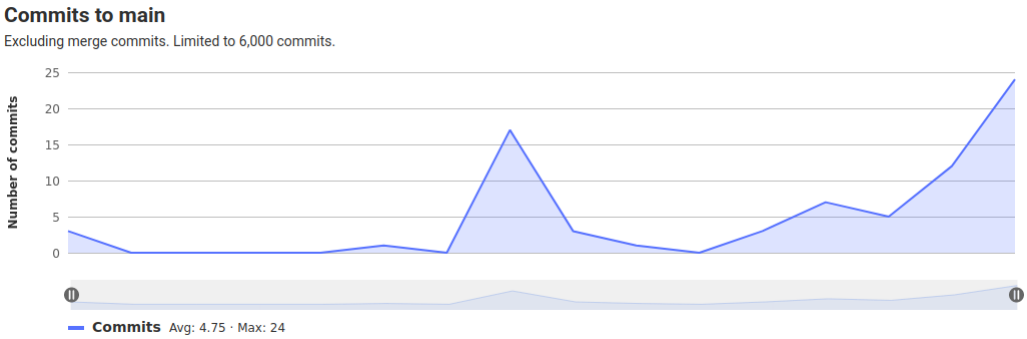
L'organisation de notre équipe a dû être adaptée à certaines conditions particulières. En effet, un membre du groupe devait travailler en distanciel car étant en séjour Erasmus à l'étranger. Nous avons donc dû utiliser plusieurs outils collaboratifs : - Trello pour la distribution du travail - GoogleDoc / HackMD lorsqu'il y avait besoin de rédiger des documents - Discord sur un serveur dédié pour nos réunions, discussions et partage de ressources

CI/CD et gestion de code

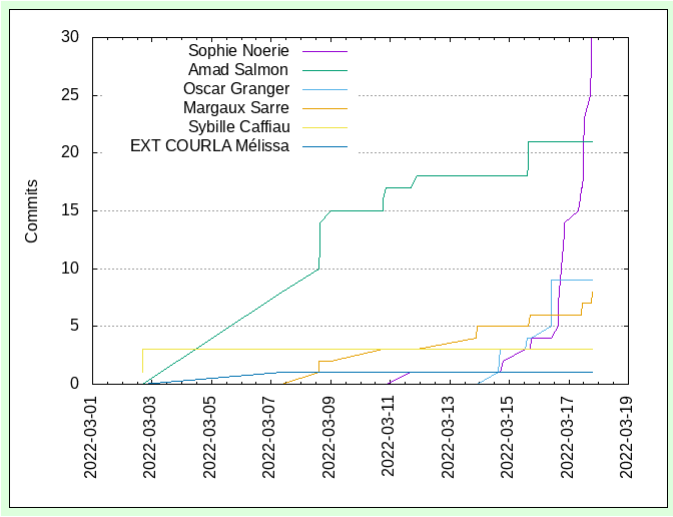
Nous avons travaillé avec Git comme outil de versionnement du code. Nos répertoires de travail ont été hébergés sur GitLab. Nous avons fait des réunions pour regrouper les différentes branches et maintenir la cohésion du code, malgré le fait que la plupart des modifications entre les branches touchaient aux mêmes fichier. Pour ce qui est de la mise en production, un problème technique venant de la plateforme Domus ne nous a pas permis de pouvoir tester notre code dans les conditions réelles. Nous avons donc mis en place un moyen de simuler des appareils connectés avec openHAB pour pouvoir tester le code.

Métriques logicielles

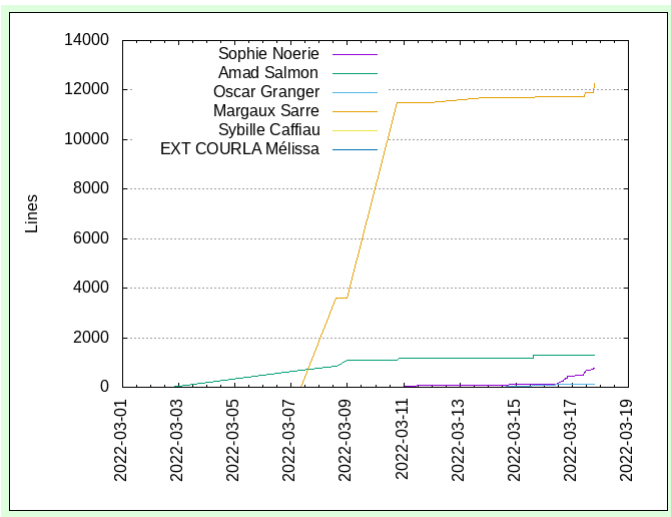
Lignes de code



Nombre de commits au cours du projet



Nombre de commits par personne



Nombre de lignes de code par personne au cours du projet (valeur non représentative de la quantité et la qualité de travail effectué)

Langages utilisés

Extension	Files (%)	Lines (%)	Lines/file
	3 (6.52%)	38 (0.38%)	12
PDF	1 (2.17%)	4734 (47.54%)	4734
cjs	1 (2.17%)	14 (0.14%)	14
css	2 (4.35%)	353 (3.54%)	176
html	2 (4.35%)	42 (0.42%)	21
ico	1 (2.17%)	0 (0.00%)	0
jpg	12 (26.09%)	202967 (2038.23%)	16913
js	4 (8.70%)	216 (2.17%)	54
json	4 (8.70%)	7958 (79.92%)	1989
md	2 (4.35%)	46 (0.46%)	23
svg	1 (2.17%)	46 (0.46%)	46
vue	13 (28.26%)	1239 (12.44%)	95

Conclusion (Retour d'expérience)

Les nombreux aléas de ce projet ne l'ont pas rendu simple. En effet, entre l'organisation incertaine du début et les problèmes de la plateforme pour tester notre code, il a fallu faire des compromis. Cependant, la bonne communication avec le client et la bonne synergie de travail au sein du groupe nous ont permis d'avoir un résultat final répondant aux attentes du client.