2020-2021 : DUT2 Informatique

Anicet Nougaret

Université de Paris

Rapport de stage

Développeur web

Entreprise d'accueil

Coddity

37bis rue de Montreuil, 75011 Paris

Maître de stage

Yoann Legrand

Tech Lead et développeur sénior chez Coddity

Tutrice de stage

Karima Mersad

Enseignante-chercheur à l'Université de Paris

Résumé/Abstract

Lors de mon stage de dix semaines au sein de Coddity, j'ai été principalement chargé de développer *Freeday Admin*, un logiciel d'administration pour *Freeday*, le nouveau produit de Coddity.

J'ai eu le privilège de développer les fonctionnalités principales du logiciel en autonomie. À l'aide des conseils et de la supervision de Yoann Legrand, ainsi que de mes propres recherches, j'ai pu répondre au cahier des charges tout en apprenant des techniques et des outils qui m'étaient inconnus.

Grâce à la confiance que Coddity m'a accordée, j'ai pu enrichir mes compétences au-delà de mes espérances et en adéquation totale avec mon projet professionnel.

During my ten weeks internship at Coddity I was assigned to develop *Freeday Admin* from scratch. *Freeday Admin* is a custom administration panel for Coddity's newest commercial product called *Freeday*.

I had the privilege to develop all of *Freeday Admin*'s features by myself. Thanks to Yoann Legrand's supervision and pieces of advice I managed to fulfill my mission as planned, doing personal research and learning many skills in the process.

Thanks to Coddity's trust I got to improve my software engineering skills and knowledge beyond my expectations.

Remerciements

Je remercie Mme. Mersad d’avoir supervisé mon stage avec enthousiasme et de s’être mise à disposition pour répondre à mes questions.

Je remercie M. Belliard et M. Moreau, gérants de Coddity, de m’avoir fait confiance et d’avoir veillé avec grande attention à ce que je m’épanouisse le plus que possible au sein de leur entreprise.

Je remercie Yoann Legrand pour avoir supervisé mon stage avec attention et bonne humeur, malgré la distance. Son aide et sa confiance m'ont permis de m'améliorer significativement en tant que développeur.

Je remercie Sosthène Leroy, stagiaire chez Coddity, pour ses réponses précieuses à mes difficultés et interrogations techniques.

Je remercie finalement tous les employés de Coddity que j’ai eu la chance immense de rencontrer, pour leur bienveillance, leur enthousiasme et tout ce qu’ils m’ont appris du métier de développeur.

Présentation de Coddity

L'entreprise, son activité

Coddity est une Entreprise de Service du Numérique (ESN) de plus de 20 employés. Elle fut créée en 2016 par Matthieu Moreau et Christopher Belliard après avoir tous deux étés employés d'Alten, une grande ESN française.

Coddity se spécialise dans le développement web et la science des données. Les ingénieurs de Coddity épaulent des équipes au sein d'entreprises de toute taille exprimant un besoin en expertise web et data.

Mais en parallèle de cette activité principale, Coddity développe de plus en plus son "Atelier", un incubateur de projets informatiques en interne pour ses consultants et experts. L'Atelier a notamment donné naissance à *Freeday*, le projet pour lequel j'ai principalement travaillé, et qui est aujourd'hui développé et commercialisé à plein temps par une équipe dédiée. Deux autres logiciels, *Hubot* et *Confusioner*, sont actuellement développés au sein de Coddity par des experts *data* et des développeurs web.

L'équipe *Freeday*

L'équipe travaillant sur Freeday est constituée comme suit :

- Christopher Belliard, chargé des tâches administratives, commerciales et de management.
- Louis Marslen, chargé des tâches commerciales et de la communication
- Chloé Recalt, chargé des tâches commerciales et de la relation client
- Yoann Legrand, mon tuteur de stage, en charge de superviser le développement de *Freeday* et *Freeday Admin*
- Sosthène Leroy, en charge de développer *Freeday.*

Au sein de cette équipe j'ai travaillé sur Freeday Admin, un projet issu de Freeday comme je l'expliquerai plus tard.

Marché et concurrence

Comme dit précédemment, Coddity est avant tout une ESN. Le marché des ESN étant aujourd'hui saturé, Coddity est mis en concurrence forte avec nombre d'autres entreprises.

Pour se démarquer auprès de ses clients, Coddity mise sur l'excellence technique. Une attention au détail toute particulière qu'ils aiment qualifier d'Artisanat du développement informatique.

Mais trouver des clients n'est pas tout. Une ESN doit aussi recruter les rares experts dans le domaine du développement web et de la science des données. Pour ce faire, Coddity mise sur sa communication décalée et sur l'aspect accueillant de son équipe, de sa culture d'entreprise, et de ses locaux situés dans le 11e arrondissement de Paris.

Ses concurrents notables sont Carbon IT, Sfeir, ou encore Theodo, trois entreprises qui offrent elles aussi une expertise web et *data*.

Travail effectué

Lors de mon stage de dix semaines, j'ai eu l'occasion d'effectuer des tâches variées. Comme la programmation de serveurs, l'intégration d'interfaces et la conception d'expériences utilisateur pendant la totalité du stage, principalement pour le projet Freeday Admin, mais aussi l'audit de code du site de Coddity, ainsi que la refonte du blog de l'entreprise. Cette variété de missions m'a permis de diversifier mon expérience.

Le métier de développeur web

Le développement web est une sous-branche du développement logiciel, elle-même sous-branche de l'informatique.

Par définition, les développeurs web développent des logiciels pour le web. C'est-à-dire des programmes fonctionnant exclusivement au sein des navigateurs web (ex: Firefox, Chrome …), principalement dans un souci d'accessibilité. Les navigateurs web sont eux-mêmes des logiciels fonctionnant grâce au système d'exploitation, lui-même pouvant être considéré comme un logiciel manipulant directement la machine. Ce qui montre que le métier de développeur web implique de se reposer sur des fondations élevées qui n'ont cessé de grimper avec le temps. L'on parle généralement de code haut niveau, se reposant sur les fondations, et le code bas niveau, avec lequel ces fondations sont bâties.

Le code bas niveau est potentiellement utilisé à plusieurs endroits différents ce qui le rend tout particulièrement critique. C'est pourquoi il est généralement rendu plus performant et sécurisé que la moyenne. Il est aussi plus proche du fonctionnement de la machine donc nécessairement plus difficile à manipuler. Alors que le code haut niveau est plus proche de l'utilisateur ce qui veut dire qu'il doit constamment s'adapter aux nouveaux besoins. C'est pourquoi il est généralement plus proche du langage humain, mais en conséquence il est aussi plus ambigüe, moins sécurisé et moins performant. L'on utilise souvent la "métaphore de l'iceberg". Le code dit "haut niveau", proche de l'utilisateur, est visible à la surface de l'iceberg, mais la majorité du code est en réalité immergée, il s'agit du code "bas niveau", les utilisateurs ne le voient pas.

Pour le développeur web, rester à la surface de l'iceberg en travaillant principalement avec du code "haut niveau" est essentiel. Cela permet de répondre aux besoins des utilisateurs avec plus de rapidité. En effet, il y a aujourd'hui un enjeu grandissant de la rapidité de développement des logiciels afin de satisfaire des utilisateurs en nombre toujours croissant. Les contraintes principales sont la rapidité de développement et l'adaptation aux changements. Même s'il faut faire attention aux performances, à la sécurité et à la pérennité du code, il faut rester le plus haut possible à la surface de l'iceberg afin de ne pas perdre de temps. Quitte à déléguer le plus de choses possible à du code généraliste situé plus profond et qui n'a pas vocation à changer aussi vite. Le développeur web doit donc trouver un entre-deux entre rapidité de développement et qualité du code produit, aspect auquel Coddity accorde une importance toute particulière.

Dans le cadre du stage, j'ai travaillé à la fois sur le code des interfaces utilisateur, ou code *frontend*, et sur le code dédié à la logique métier, ou code *backend*. Le tout suivant une logique centralisée sur laquelle je reviendrai par la suite.

Sur des projets de taille conséquente, les développeurs frontend et backend sont généralement spécialisés, avec quelques développeurs seulement qui se placent dans un entre-deux. Mais de mon point de vue, je trouve qu'avoir pratiqué un mélange des deux s'est avéré d'autant plus enrichissant.

Présentation du projet *Freeday Admin*

Lors de mes dix semaines de stage j'ai principalement travaillé sur *Freeday Admin*, une application web servant de panneau de contrôle centralisé en ligne pour *Freeday,* un logiciel commercial de Coddity. *Freeday Admin* dépendant entièrement de *Freeday*, il est important de comprendre le fonctionnement de ce dernier.

Présentation de *Freeday*

Freeday est un outil de gestion de congés conçu pour les petites et moyennes entreprises. Il propose aux employés un moyen simple de soumettre leurs demandes de congés et à leurs responsables de les valider ou non.

Développé par l'Atelier de Coddity depuis septembre 2018, ce logiciel est entièrement centralisé. Les droits d'accès au logiciel sont ensuite loués aux entreprises clientes. On peut parler de *Software as a Service* ou *SaaS*.

L'accès au logiciel ou "service" est permis de deux manières :

Premièrement par l'intermédiaire d'une page web servant de panneau de contrôle à l'intention du département des ressources humaines du client. On peut y valider les demandes de congés, y consulter des plannings des congés et des récapitulatifs.

Deuxièmement, par l'intermédiaire de *Slack*, un service tiers de messagerie particulièrement populaire au sein des entreprises. En effet, un utilisateur robot pour *Slack* a été développé par Coddity afin de capter les demandes de congés que les employés lui envoient par message textuel.

Lorsqu'une action est effectuée par un employé de l'entreprise cliente sur le panneau de contrôle web ou à l'aide de l'utilisateur robot sur *Slack*, des requêtes sont envoyées via le réseau internet à l'infrastructure de Coddity qui les répartit équitablement entre plusieurs ordinateurs qu'on appelle "instances". Chaque instance traite ses requêtes à l'aide d'un logiciel prévu à cet effet nommé "serveur". Ce serveur traite les requêtes indépendamment des autres instances et met à jour les données du client (absences, utilisateurs, plannings...) tout en garantissant leur cohérence si besoin. Les données sont ensuite envoyées à un système de stockage central ou "base de données" situé sur un autre ordinateur.

Le besoin à l'origine de *Freeday Admin*

Le caractère centralisé de *Freeday* permet une maîtrise et une surveillance en temps réel du bon fonctionnement du produit et de son utilisation par les clients. Ce qui représente un réel atout. Pas besoin pour le client de mettre à jour le logiciel, à la moindre connexion la dernière version leur est directement envoyée depuis les serveurs. Mais la centralisation apporte aussi certains risques, comme la vulnérabilité aux pannes (si le serveur stoppe de façon imprévisible alors tous les clients sont potentiellement impactés), la vulnérabilité aux attaques informatiques (si la base de donnée est détruite ou volée alors tous les clients sont potentiellement impactés), et les performances (l'intégralité du logiciel est (re)téléchargé de A à Z par le client à chaque connexion ce

qui implique de la latence). C'est autour de ces possibilités et contraintes que les besoins en ingénierie de Coddity se sont agencés au sein du projet *Freeday Admin*.

En effet, en raison du lancement commercial imminent de *Freeday* lors de mon arrivée en avril 2021, il devenait primordial pour Coddity de pouvoir surveiller le bon fonctionnement de *Freeday*, de le mettre à jour et d'en extraire des données utiles sans avoir à faire systématiquement appel à des techniques chronophages et répétitives. En somme, il y avait besoin d'un outil d'automatisation de la maintenance et de visualisation des données d'utilisation.

Tout cela étant intrinsèquement lié au fonctionnement de *Freeday*, il a donc été décidé de développer un logiciel entièrement sur mesure. C'est dans ce contexte qu'est né le projet *Freeday Admin* sur lequel j'ai travaillé.

Cahier des charges de *Feeday Admin*

Après quelques jours d'adaptation et de recherches personnelles, Yoann Legrand et moi commençons à concevoir le cahier des charges. Ce dernier prévoyait de faire une application web, c'est à dire un logiciel complet au sein d'un site web, constitué de quatre lots de fonctionnalités répartis en quatre pages :

- Page *Tenants* :
    - Mettre en place une mesure automatique quotidienne de l'utilisation de Freeday par les clients. Par exemple le nombre de connexions au site web, le nombre de messages envoyés à l'utilisateur robot sur Slack, le nombre de congés enregistrés...
- Page *Instances* :
    - Mettre en place un système de surveillance en temps réel des serveurs pour savoir à tout moment s'ils sont allumés et à jour.
- Page *Backups* :
    - Mettre en place un système de surveillance des sauvegardes de la base de données pour savoir si elles viennent à manquer à un moment donné.
- Page *Operations* :
    - Permettre le lancement d'opérations de maintenance automatisées à distance. Comme une nouvelle installation des serveurs, une mise à jour des serveurs ou un changement de configuration.
- Fonctionnalités additionnelles facultatives :
    - Ajout de statistiques de charges des serveurs pour s’assurer qu’ils n’ont pas une consommation électrique, une température, ou une utilisation de la mémoire disponible trop importante.

Planification du temps et des tâches

Gantt initial

Taches \Semaines	1	2	3	4	5	6	7	8	9	10
Parcours de formation										
Freeday Admin : Cahier des charges										
Freeday Admin : Mise en place des outils										
Freeday Admin : Authentification										
Freeday Admin : Raccordage avec Freeday										
Freeday Admin : Page Tenants										
Freeday Admin : Page Instances										
Freeday Admin : Page Operations										
Freeday Admin : Page Backups							(optionnel)	(optionnel)		
Freeday Admin : Statistiques de charge								(optionnel)		
Freeday Admin : Phase de fiabilisation										

En pratique

Taches \Semaines	1	2	3	4	5	6	7	8	9	10
Parcours de formation										
Freeday Admin : Cahier des charges										
Freeday Admin : Mise en place des outils										
Freeday Admin : Authentification										
Freeday Admin : Raccordage avec Freeday										
Freeday Admin : Page Tenants										
Freeday Admin : Page Instances										
Freeday Admin : Page Operations										
Freeday Admin : Page Backups										
~~Freeday Admin : Statistiques de charge~~										
Freeday Admin : Phase de fiabilisation										
Freeday : correction de bugs										
Site Coddity : Audit du code										
Blog Coddity : Réécriture et modernisation										
Préparation d'un établi Coddity										

J'ai pu réaliser plus de tâches que prévu grâce à une avance d'une semaine. En effet, le début de projet fut plus fluide que prévu.

L'ajout de "statistiques de charges" à *Freeday Admin*, une tâche facultative qui aurait normalement pu prendre environ une semaine, a été annulé vers la fin du stage pour me permettre d'effectuer des petites tâches en dehors de *Freeday Admin*. Ce qui fut positif puisque ces tâches nouvelles ont apporté encore plus de richesse à mon expérience chez Coddity.

Compte-rendu des travaux réalisés

Freeday Admin

Organisation et méthodes de travail

La réalisation du projet s'organisait en cycles courts de quelques semaines maximum axés autour des lots de fonctionnalités cités précédemment.

Au début de chaque cycle, Yoann Legrand créait une *issue* sur l'outil de gestion de projet utilisé chez Coddity nommé *Gitlab*. Cette *issue* décrivait le lot de fonctionnalités à implémenter et les défis éventuels.

Je réalisais ensuite une maquette de l'interface utilisateur ainsi que des recherches sur la faisabilité technique et les technologies *open-source* à utiliser au sein de la fonctionnalité afin d'économiser la quantité de code à réaliser.

Yoann Legrand me donnait alors un retour détaillé sur la maquette et mon analyse technique, puis à l'issue de cet échange je commençais à coder.

Je procédais étape par étape selon ma propre analyse. Lorsqu'une fonctionnalité était finie, je la testais à la main, mais je réalisais aussi des outils automatisés afin de pouvoir tester un maximum de cas. Par exemple, il m'arrivait d'écrire des tests unitaires et des tests d'intégration pour garantir automatiquement et rapidement le bon fonctionnement du code. Il m'arrivait aussi d'écrire des générateurs de données aléatoires pour voir la réaction du logiciel avec des données toujours plus variées.

Lorsque le lot de fonctionnalités était fini, je le soumettais sur *Gitlab* par le biais d'une *merge request*. Cette *merge request* montrait la différence entre le code source du projet avant et après l'implémentation du lot. J'ajoutais aussi un descriptif détaillé et des captures d'écran. Mon tuteur venait ensuite auditer le code source et nous effectuions des améliorations et des corrections de bogues si besoin.

Chaque soir nous avions une réunion quotidienne par appel vocal où nous discutions de l'avancement du projet. Mais il arrivait aussi que je lui pose des questions par message textuel sur *Slack* pendant la journée et il me répondait aussitôt.

L'organisation était flexible et fluide, c'est un point que j'ai beaucoup apprécié. Mais sachant que j'étais le seul programmeur sur le projet et que mon tuteur était le seul évaluateur, il n'y avait pas possibilité de rencontrer les défis qu'on pourrait avoir à surmonter au sein d'équipes plus grandes et complexes.

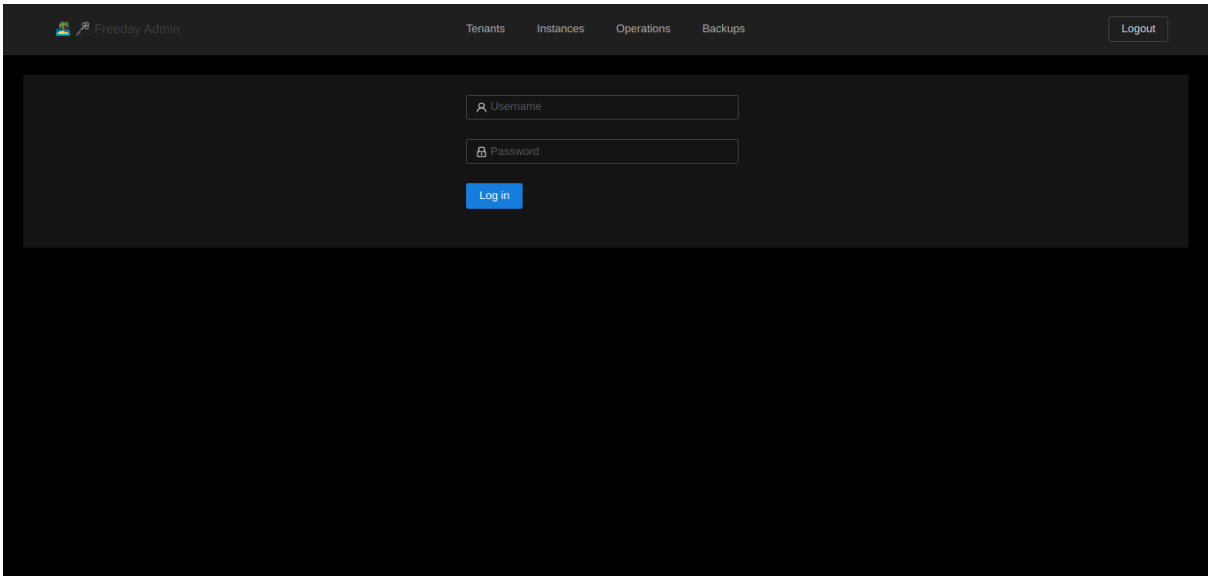
Système d’authentification

Freeday Admin présente un fonctionnement similaire à celui de *Freeday*. L’interface est une page web renvoyée à l’utilisateur à chaque utilisation et fonctionnant sur sa machine. C'est pourquoi, pour des raisons de sécurité, la logique et les procédures réelles que *Freeday Admin* peut effectuer, sont exécutées par un serveur centralisé géré par Coddity. Ainsi, pour permettre au serveur de différencier un opérateur légitime du

site *Freeday Admin* d'un internaute tombé là par hasard, il fallait créer un système d'authentification.

Ce système permet de se connecter à *Freeday Admin* avec un nom d'utilisateur et un mot de passe à l'aide d'un formulaire. Ce mot de passe est vérifié par le serveur à partir de sa propre configuration.

Une fois connecté de la sorte, toutes les requêtes envoyées au serveur depuis l'interface (servant à manipuler à distance les fonctionnalités de *Freeday Admin*) sont annotées d'un jeton secret de connexion pour que le serveur les accepte. Ce jeton est régénéré à chaque nouvelle connexion et expire au bout d'une heure. Il s'agit d'un mécanisme de sécurité tout à fait classique.



Capture d'écran de Freeday Admin, authentification

Page *Tenants*

Cette page permet de visualiser les statistiques d'utilisation de *Freeday* pour chaque entreprise cliente (en anglais, *tenant*). Les statistiques sont calculées quotidiennement par le serveur de *Freeday Admin* et sauvegardées dans des rapports successifs jours après jours.

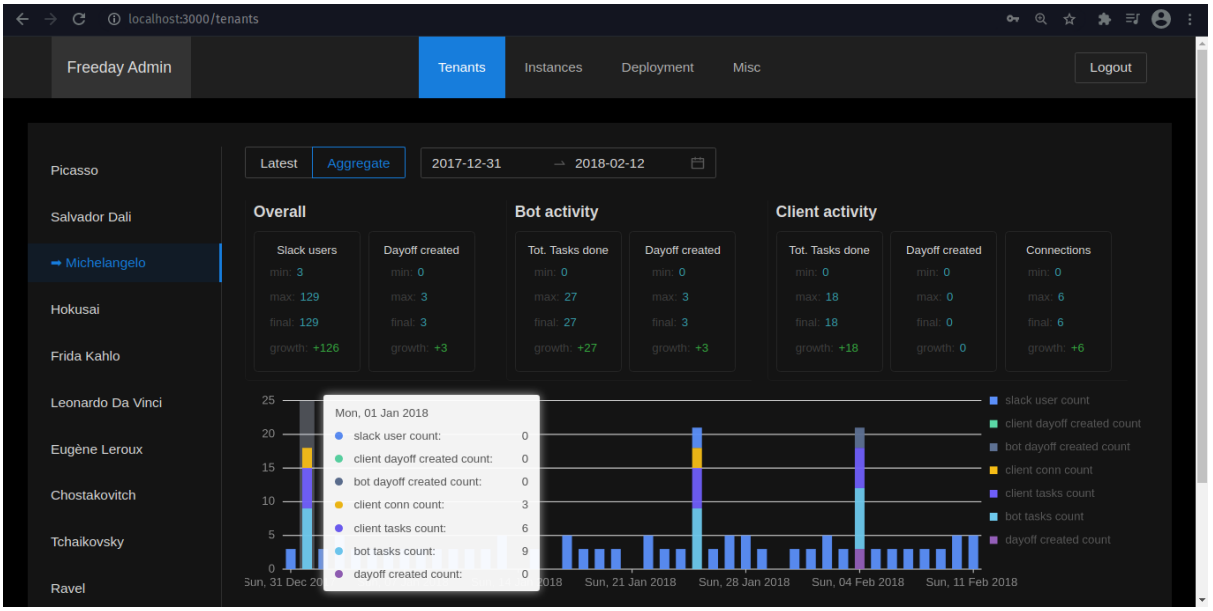
L'on peut visualiser le rapport du jour, ou un rapport calculé à partir de tous les rapports compris entre deux dates.

Les données furent choisies pour convenir aux besoins de l'équipe commerciale de *Freeday*, mais un soin particulier a été apporté à la facilité de modification du code afin de permettre des affichages supplémentaires ou corrigés dans le futur.

Côté serveur, le défi était de communiquer avec Freeday afin de récupérer les informations importantes. Une mise à jour de *Freeday* a été développée par Sosthène Leroy, un autre stagiaire, afin de permettre cela. De mon côté, j'ai dû créer des générateurs de données de test afin de pallier le manque de données pour tester le bon

fonctionnement des différentes visualisations. Ce qui représente un certain défi en raison de la multiplicité des bases de données impliquées dans le processus.

Côté client, le défi fut de mettre en place l'interface complexe de cette page. J'ai réalisé une maquette pour pouvoir faire corriger par Yoann Legrand d'éventuelles mauvaises idées de conception avant de me lancer dans le code. Puis j'ai suivi la documentation d'*Ant Design*, une collection *open-source* de composants d'interfaces, que j'ai utilisés pour gagner du temps.

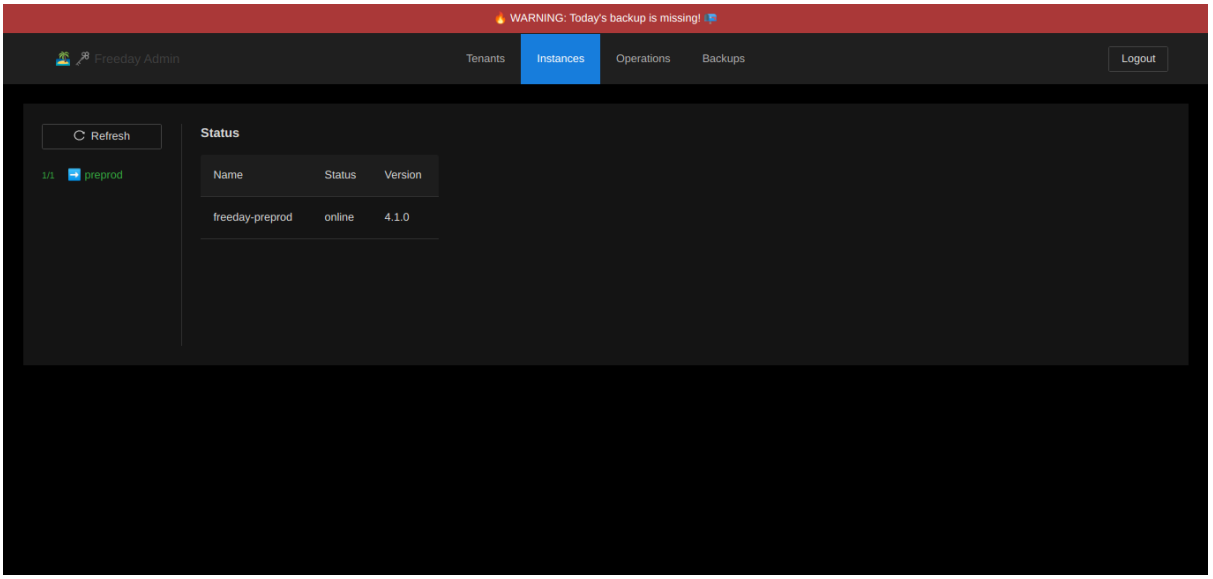


Capture d'écran de Freeday Admin, page Tenants

Page *Instances*

La page *Instances* permet de vérifier en un coup d'œil si les serveurs de *Freeday* sont en ligne et à jour. Chaque *instance* correspond à une machine physique sur laquelle tourne un ou plusieurs serveurs. L'on peut voir pour chaque *instance* les serveurs allumés ou éteints et la version de *Freeday* avec laquelle ils opèrent. C'est un outil essentiel pour détecter les pannes et garantir le bon fonctionnement du produit.

Le nombre d'*instances* et de serveurs pouvant diminuer ou augmenter à chaque mise à jour, il fallait proposer un système dynamique. Pour obtenir cela, il fallait raccorder automatiquement *Freeday Admin* et le système de déploiement de *Freeday* pour qu'il y trouve la dernière liste de serveurs.



Capture d'écran de Freeday Admin, page Instances

La difficulté principale fut de trouver un moyen de vérifier le statut de chaque serveur *Freeday* depuis *Freeday Admin*. En effet, le mécanisme de répartition de charge de l'infrastructure *Freeday* ne permettait pas d'accéder à un serveur en particulier sur demande pour vérifier son statut. Il a fallu essayer plusieurs méthodes et mettre à jour le code de *Freeday* pour permettre cette fonctionnalité. C'était une difficulté intéressante puisqu'elle m'a permis de voir de près le fonctionnement de l'architecture en *round robin* de *Freeday*, et comment elle peut présenter des avantages en termes de performance au prix d'une complexité un peu encombrante.

Page *Operations*

Cette page sert d'outil de maintenance à distance. En effet, elle permet de lancer des *playbooks Ansible* sur les serveurs de *Freeday* à distance.

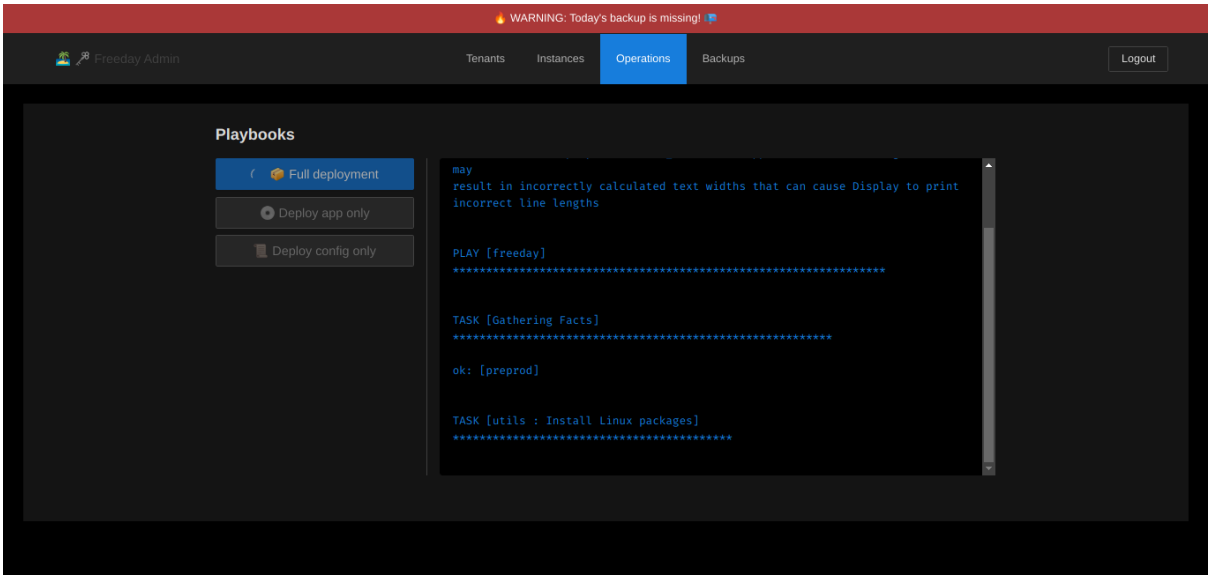
Les *playbooks Ansible* sont des collections de fichiers qui décrivent chacune une procédure de maintenance automatisée, comme l'installation des serveurs *Freeday*, leur mise à jour, leur protocole de démarrage … Lorsqu'on en lance un, le protocole se déroule automatiquement, tout en envoyant régulièrement des *logs*, des messages textuels brefs informant l'utilisateur du bon fonctionnement ou non de l'opération.

En temps normal ces *playbooks* sont installés sur une machine privée et sont lancés depuis un terminal par un employé formé à *Ansible*, comme Yoann Legrand. Mais ici, le défi était de pouvoir les lancer depuis n'importe où et plus ergonomiquement. Pour résoudre ce problème, nous avons décidé de permettre au serveur de *Freeday Admin* de lancer lui-même des *playbooks Ansible*, ce qui a nécessité quelques jours d'expérimentation, dans le but de les rendre contrôlables depuis l'interface utilisateur ergonomique de *Freeday Admin*.

En raison du caractère potentiellement dangereux que représentent ces *playbooks* en cas d'utilisation par un individu malintentionné, des mots de passe supplémentaires sont demandés au lancement par l'apparition d'une fenêtre modale.

Une fois la procédure enclenchée, le serveur écoute la sortie standard d'*Ansible* et renvoie les *logs* à l'interface utilisateur en temps réel à l'aide du protocole de communication *WebSocket*. Ils sont ensuite affichés dans le rectangle noir comme si l'on avait réellement accès au terminal du serveur de *Freeday Admin*.

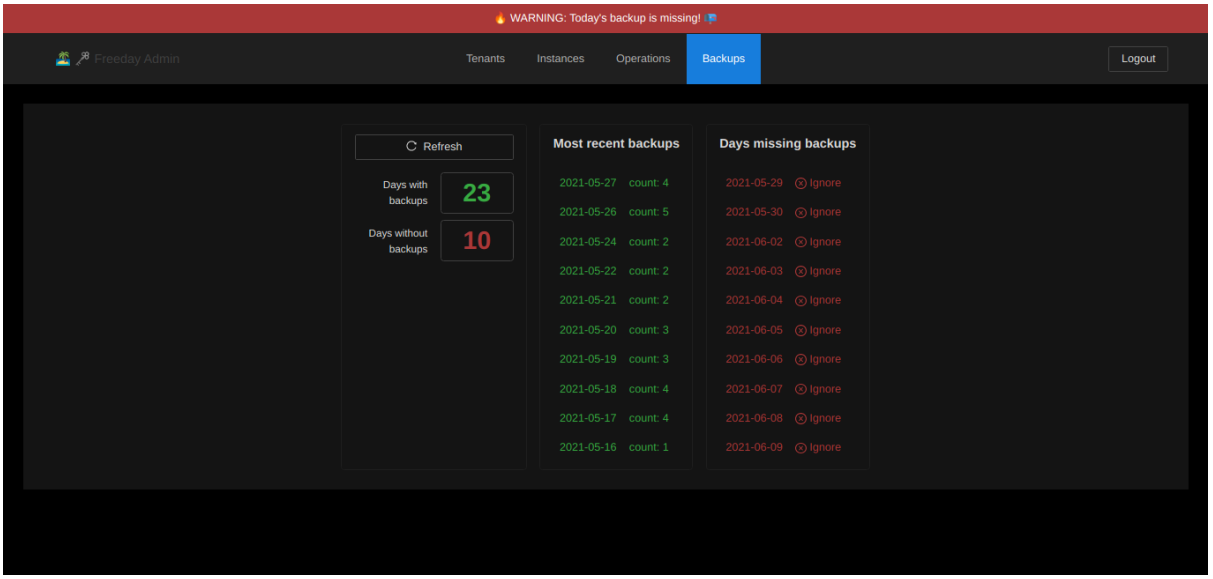
La page *Operations* est probablement la fonctionnalité la plus unique et techniquement intéressante de *Freeday Admin* en raison du protocole de sécurité renforcé, de la capacité à interagir avec les *playbooks Ansible*, et de l'utilisation du protocole *WebSocket*.



Capture d'écran de Freeday Admin, page Operations

Page *Backups*

Dernière page et lot de fonctionnalités du projet Freeday Admin, elle permet de visualiser l'état des sauvegardes (en anglais, *backups*) de la base de données. Le but étant de permettre aux mainteneurs de Freeday de s'assurer quotidiennement que la ou les sauvegardes du jour sont bien présentes.



Capture d'écran de Freeday Admin, page Backups

La surveillance et la sauvegarde des bases de données au quotidien est une sécurité essentielle pour une entreprise comme Coddity qui développe des logiciels SaaS. En effet, comme je l'ai expliqué précédemment, le problème des logiciels centralisés est qu'ils dépendent d'un système de stockage central, qui, s'il est défaillant ou pire encore, perdu, peut signifier un retour en arrière de plusieurs mois et des dommages conséquents pour les clients.

Tous les jours le serveur de *Freeday Admin* va vérifier dans le système du fichier la présence des fichiers de sauvegarde. L'interface utilisateur reçoit ensuite le rapport quotidien de leur présence et affiche des messages d'alerte aux bons endroits si certaines sauvegardes viennent à manquer.

Site de Coddity

J'ai eu l'occasion de travailler sur le site de Coddity en réalisant un audit du code source de ce dernier. Cet audit visait à identifier les parties du code ne répondant plus aux nouveaux standards de qualité de Coddity. S'il y a une compétence que j'ai particulièrement améliorée lors de ce stage, il s'agit très certainement de ma capacité à lire et comprendre le code des autres. Cet exercice fort enrichissant m'a permis de décortiquer un projet complexe pour en identifier les failles et y apporter mon analyse personnelle basée sur ma propre expérience avec la création de sites web.

Blog de Coddity

Refonte

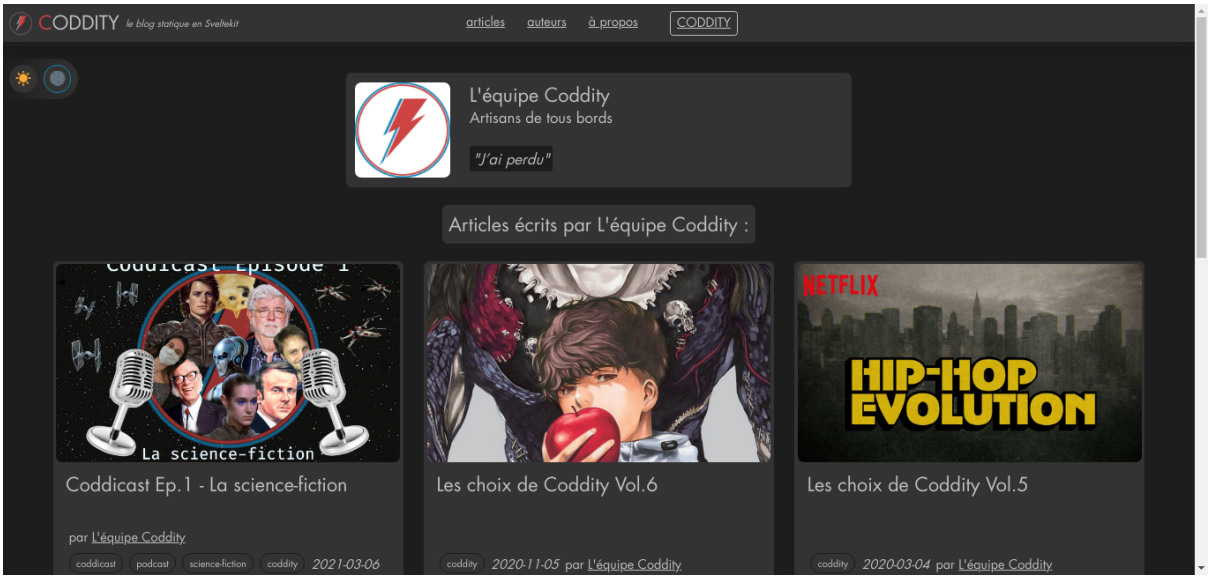
Après mon audit du site de Coddity, M. Belliard, gérant de Coddity, me proposa d'apporter aussi un regard critique sur le blog de l'entreprise, surtout en matière d'expérience utilisateur et d'interface. En effet, certains aspects esthétiques du site devaient être retravaillés.

Mais lorsque j'ai décidé de mettre en place le serveur du blog sur ma propre machine pour apporter les modifications esthétiques souhaitées, j'ai remarqué que les problèmes techniques étaient tout aussi présents que ceux esthétiques. À l'aide de Sosthène Leroy qui avait aussi travaillé sur le blog par le passé, j'ai aussi remarqué que tous ces problèmes techniques empêchant de mettre à jour l'esthétique du blog étaient bien plus enracinés que prévu. En effet, ils étaient un symptôme de l'utilisation dès le départ de technologies qui se sont avérées trop encombrantes au fil du temps pour la fonctionnalité relativement simple que le blog implémentait. C'est un scénario relativement classique dans le monde du développement informatique où les technologies évoluent vite et où leurs avantages court-terme sont parfois mieux connus que leurs inconvénients long-terme.

Face à l'impossibilité de corriger tous ces problèmes en un laps de temps raisonnable, et de manière à éviter des problèmes similaires dans le futur, Sosthène et moi avons donc décidé de refaire le blog à partir de zéro en utilisant des techniques plus simples et droit au but. Nous sommes passés d'une solution nécessairement dynamique et encombrante à une solution légère, statique, facile à maintenir, mais pouvant être augmentée dans le futur si le besoin s'en faisait.

En l'espace de 5 heures de travail nous avions un blog fonctionnel, sans aucun problème technique, avec presque autant de fonctionnalités que l'ancien. Ainsi qu'un design satisfaisant et toujours en accord avec la charte graphique de Coddity.

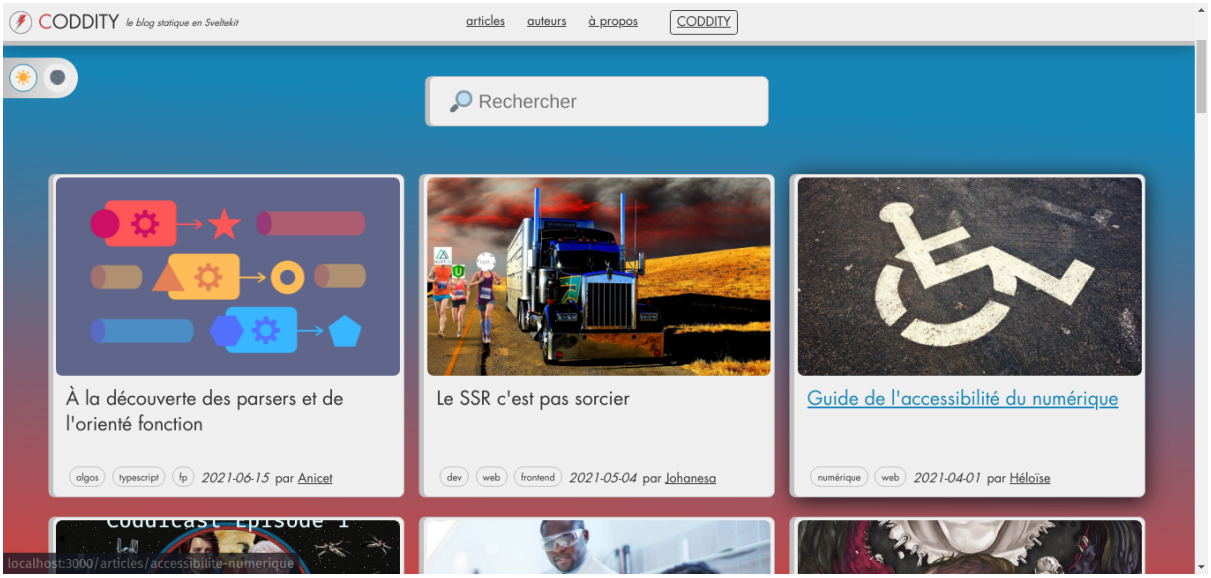
Nous n'avons à ce stade pas de démonstration irréfutable que cette refonte survivra nécessairement mieux aux bogues et des maintenances au fil du temps. Mais nous pensons que son fonctionnement, significativement plus léger et droit au but, ainsi que le peu de temps qui fut nécessaire à son élaboration, montrent déjà un net avantage en termes de maintenabilité.



Capture d'écran du nouveau blog en mode d'affichage sombre.

Cette refonte du blog a été approuvée par les gérants de Coddity qui nous ont félicité pour notre rapidité et pour la qualité du produit fini. Le nouveau blog devrait donc être

mis en ligne publiquement d'ici le 14 juin, 2 jours seulement avant la fin de mon stage. Je pense que je ne peux pas trouver de meilleures manières de finir mon expérience chez Coddity.



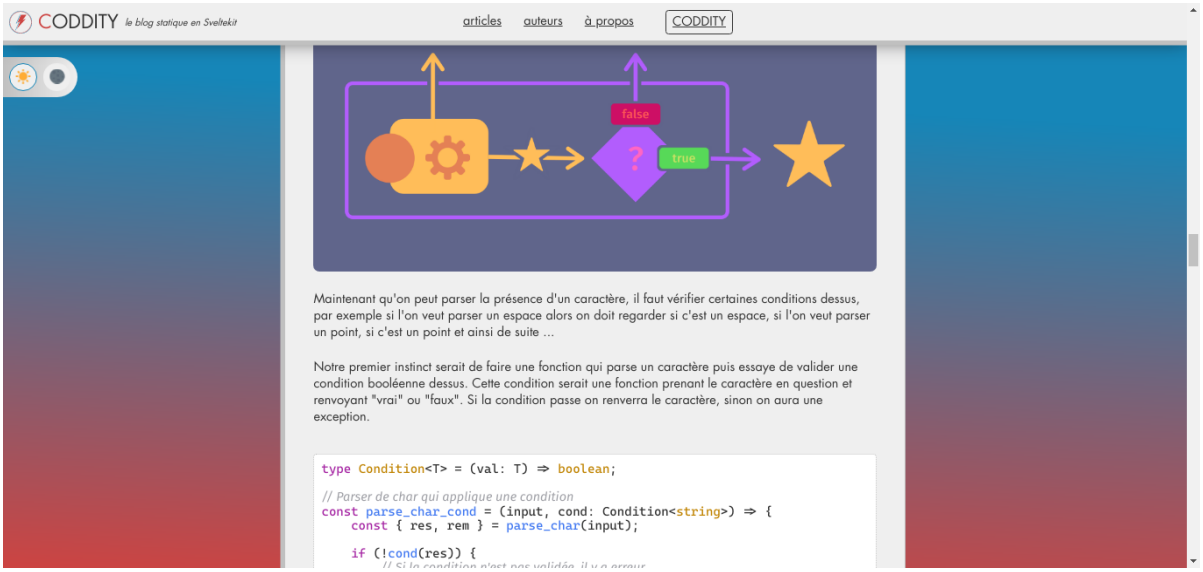
Capture d'écran du nouveau blog en mode d'affichage clair

Article

Lors de mon stage j'ai aussi eu l'occasion de rédiger un article technique pour le blog de Coddity. L'article en question parle d'un sujet qui me passionne et que peu de programmeurs connaissent : la programmation orientée fonction. Il s'agit d'un "paradigme" de programmation, c'est-à-dire une famille de méthodes pour exprimer les algorithmes et organiser le code.

Dans cet article je montre un cas concret de l'utilisation de ce paradigme en réalisant un programme capable de décomposer une phrase à l'aide d'une technique orientée fonction inspirée d'une branche des Mathématiques appelée théorie des Catégories.

Il sera publié le 14 juin prochain.



Capture d'écran de mon article sur le nouveau blog.

Conclusion

Mon stage a permis de significativement enrichir et renforcer mes compétences de développeur, mais surtout de découvrir des personnes passionnées par cette discipline ainsi qu'un cadre de travail épanouissant.

Je suis fier de mes productions, qui je le pense auront un réel impact sur Coddity. Mais aussi de ma progression personnelle. Les méthodes de travail, compétences techniques, processus de réflexion et de conception que j'ai pu observer et vivre au sein de l'entreprise m'ont fait grandir dans ce milieu.

L'informatique, et plus particulièrement le développement web, est un domaine qui évolue à toute vitesse. Les technologies utilisées, les méthodes de travail, les objectifs et les besoins se métamorphosent d'année en année. Avoir été capable d'observer pendant dix semaines comment une entreprise comme Coddity procède pour s'adapter et prendre des décisions fortes en permanence m'a fortement inspiré.

Au-delà de ma poursuite d'étude et de ma rentrée future dans le marché du travail, la programmation est une passion que je pratique aussi sur mon temps libre depuis plusieurs années. Ce stage m'a donc été d'autant plus utile.

Mon enthousiasme grandissant envers le milieu du développement web et envers les différentes missions pour Coddity au sein desquelles j'ai pris part, me permettent aujourd'hui d'enrichir mes propres objectifs académiques et professionnels.

Table des matières

Glossaire

ESN : Anciennement appelées SSII, ce sont des sociétés de service dans le domaine de l'informatique et des nouvelles technologies (conseil, maintenance, expertise, programmation, gestion de projets …).

Backend : Anglicisme, littéralement "l'arrière-boutique", en programmation désigne le code non visible d'un logiciel ou d'un composant de logiciel lors de son utilisation par d'autres programmeurs.

Frontend : Anglicisme, littéralement "la devanture [d'un magasin]", en programmation désigne le code visible d'un logiciel ou d'un composant de logiciel lors de son utilisation par d'autres programmeurs ou par l'utilisateur. Idéalement, le code frontend est particulièrement documenté et bien conçu de façon à simplifier le travail des utilisateurs.

Logique métier : En ingénierie logiciel désigne le code décrivant les demandes du client et/ou le modèle simulé par le logiciel. Par exemple, le code qui calcule le taux imposable dans un simulateur des impôts, le code qui trie les candidats dans Parcoursup…

SaaS : Software as a Service, logiciel centralisé fonctionnant sur un ou plusieurs serveurs et dont l'accès est loué aux clients, à travers un compte et un système d'autorisations.

Open-source : Mode de distribution d'un logiciel où leurs ayant-droits rendent le code source public. Pratique très courante, le code open-source peut souvent être utilisé dans des projets commerciaux (en fonction des licences), ce qui facilite l'innovation et la collaboration.

Théorie des Catégories : Théorie mathématique utilisée dans plusieurs branches des Mathématiques, de la Physique, mais aussi en programmation orientée fonction. Elle permet notamment de décrire des classes d'objets et des opérations permettant de passer d'une classe à une autre, opérations appelées "morphismes" et pouvant être composées les unes avec les autres, en analogie à la composition de fonctions entre elles en programmation.

Bibliographie et Webographie

Web development sur *Wikipedia* : https://en.wikipedia.org/wiki/Web_development

Entreprise de services du numérique sur *Wikipedia* : https://fr.wikipedia.org/wiki/Entreprise_de_services_du_num%C3%A9rique

Category Theory for Programmers, Bartosz Milewski, auto-publié le 12 août 2019 sous licence CC 4.0 : https://github.com/hmemcpy/milewski-ctfp-pdf/releases/download/v1.3.0/category-theory-for-programmers.pdf