

# RAPPORT DE STAGE

Stage du 2 mai au 30 juin 2023

# REMERCIEMENTS

Chers membres de l'équipe ou devrai-je je dire Alexandre, Romain et François Guillaume.

Je tiens à vous exprimer mes plus sincères remerciements pour cette incroyable expérience de stage au sein de votre équipe. Travailler avec vous a été un privilège et une source de joie tout au long de cette période.

Je suis reconnaissant de l'accueil chaleureux que vous m'avez réservé et de la bienveillance avec laquelle vous m'avez encadré. Vos conseils avisés, votre expertise et votre disponibilité m'ont permis de progresser et d'apprendre énormément. Chaque interaction avec vous a été une occasion d'enrichissement professionnel et personnel.

Ce stage a été une expérience inestimable qui m'a permis de mettre en pratique mes connaissances, d'explorer de nouvelles technologies et de renforcer mes compétences. Je suis conscient de la chance que j'ai eue de travailler sur des projets concrets et de contribuer à leur succès.

Je souhaite également exprimer ma profonde gratitude pour les moments de partage et d'échange que nous avons eus. Vos conseils, vos retours constructifs et vos discussions ont enrichi ma perspective et m'ont permis de grandir professionnellement. Je tiens à vous remercier pour cette expérience unique et pour l'impact positif qu'elle a eu sur moi. Votre confiance, votre soutien et votre encouragement ont été les moteurs de ma motivation et de ma réussite.

Je quitte cette équipe avec une immense gratitude et une grande tristesse de voir cette expérience prendre fin. Je garderai précieusement les souvenirs et les leçons apprises durant ce stage, et je suis convaincu qu'elles me serviront tout au long de ma carrière professionnelle.

Encore une fois, merci infiniment pour cette opportunité et pour tout ce que vous m'avez apporté. Je vous souhaite à tous un succès continu dans vos projets futurs.

Cordialement,

Mathéo DELAUNAY

# TABLE DES MATIERES

1 – Introduction	
1.1 – Contexte du stage	04
1.2 – Objectifs du stage	05
2 – Présentation de l'entreprise	
2.1 – Historique et domaine d'activité de l'entreprise	06
2.2 – Structure organisationnelle	07
2.3 – Présentation du service	08
3 – Expérience du travail à distance	
3.1 – Introduction au télétravail	09
3.2 – Présentation de mon environnement de travail	10
3.3 – Avantages et opportunités du télétravail	11
3.4 – Défis et solutions du télétravail	12
3.5 – Réflexions sur l'expérience du télétravail	13
4 – Présentation du projet et des missions liées	
4.1 – Description du projet	14
4.2 – Description de la première mission	15
4.3 – Description de la deuxième mission	16
4.4 – Description de la troisième mission	17
5 – Démarche suivie pour réaliser les missions	
5.1 – Analyse préliminaire des besoins	18
5.2 – Planification des activités	20
5.3 – Sélection des outils et des ressources nécessaires	22
6 – Réalisation des missions	
6.1 – Tâches effectuées et activités réalisées	24
6.2 – Résultats obtenus	58
6.3 – Difficultés rencontrées et solutions apportées	61
7 – Évaluation des réalisations et des compétences mobilisées	
7.1 – Analyse des résultats par rapport aux objectifs fixés	62
7.2 – Évaluation des compétences techniques acquises ou mobilisées	63
7.3 – Réflexion sur les compétences transversales développées	64
8 – Conclusion	
8.1 – Bilan du stage	65
8.2 – Retour d'expérience personnel	66
8.3 – Perspectives d'évolution et d'amélioration	68

# 1 - INTRODUCTION

## 1.1 - CONTEXTE DU STAGE

Dans le cadre de ma première année de BTS SIO à l'école EPSI, j'ai eu l'opportunité de réaliser un stage au sein de l'entreprise "NoBullShit Tech Engineers". Fondée en septembre 2022, cette entreprise regroupe des passionnés de la technologie et se spécialise dans l'assistance aux entreprises qui rencontrent des difficultés liées à l'urbanisation, à l'abstraction et à la modernisation de leurs systèmes d'information et de leurs logiciels en tant que service (SaaS).

Le stage s'est déroulé sur une période de neuf semaines, du 2 mai 2023 au 30 juin 2023. L'objectif principal de ce stage était de mettre en pratique les connaissances acquises lors de ma formation et d'acquérir une expérience professionnelle concrète dans le domaine de la technologie.

Pendant mon stage, ma mission principale était de créer un formulaire d'incident en utilisant l'API Better Uptime. L'objectif était de développer un formulaire convivial permettant aux utilisateurs de signaler facilement les incidents techniques rencontrés. Pour réaliser cette mission, j'ai travaillé en étroite collaboration avec mon maître de stage, Alexandre Cailler, ainsi qu'avec Romain Boudot et François Guillaume Ribreau. Ils m'ont encadré, guidé et fourni des orientations tout au long de la réalisation de ma mission.

Étant donné que l'entreprise est constituée de freelances et qu'elle défend le droit à chacun de travailler depuis chez lui, dans son environnement pour des raisons écologiques évidentes et de tranquillité, mon stage s'est déroulé en remote, c'est-à-dire que j'ai travaillé à distance à la fois depuis mon établissement scolaire et depuis mon domicile. Cette expérience m'a permis de développer des compétences en matière d'organisation et de gestion du travail à distance, compétences de plus en plus essentielles dans notre société en constante évolution.

Ce rapport de stage vise à rendre compte de mon expérience chez "NoBullShit Tech Engineers" en mettant en évidence les compétences techniques acquises, les connaissances développées et les résultats obtenus. Je tiens à exprimer ma gratitude envers toute l'équipe pour leur collaboration, leur soutien et leur mentorat qui ont contribué à la réussite de ce stage enrichissant.

# 1 - INTRODUCTION

## 1.2 - OBJECTIFS DU STAGE

L'objectif principal de mon stage était d'acquérir une expérience pratique et théorique dans le domaine professionnel du développement. Je souhaitais découvrir les technologies utilisées dans l'entreprise et déterminer celles sur lesquelles je souhaiterais me concentrer pour mon parcours professionnel. Mon stage visait également à combler les lacunes de connaissances que j'ai identifiées au cours de ma scolarité.

Grâce à ce stage, j'ai pu développer d'incroyables compétences professionnelles, notamment dans l'utilisation d'API, de JavaScript, de Node.js, du SCSS, de variables d'environnement, de Docker, de Kubernetes, de Cypress, et bien d'autres encore.

Un autre objectif essentiel était d'améliorer mes compétences en résolution de problèmes. Les problèmes rencontrés dans un contexte professionnel sont souvent plus complexes que ceux abordés en cours, et mon stage m'a permis de développer ma capacité à analyser et à trouver des solutions efficaces.

La collaboration efficace en équipe était également un objectif important. J'ai appris à travailler en collaboration avec mes collègues à distance, en développant mes compétences en communication et en réalisant des comptes rendus journaliers pour assurer une bonne coordination et une transparence au sein de l'équipe.

J'avais pour objectif d'obtenir des retours et des évaluations de mon maître de stage et des membres de l'équipe. Ces retours me permettraient de mesurer ma progression et d'identifier les points à améliorer. J'apprécierais l'opportunité de poursuivre ma deuxième année de stage avec cette entreprise, dans le but de continuer à progresser avec leur soutien.

Enfin, je souhaitais développer mon autonomie et ma prise d'initiative au fil du stage. Travailler à distance peut comporter davantage de distractions, mais j'ai relevé le défi de rester concentré et de proposer des améliorations ou des idées pour le développement du formulaire d'incident. J'ai cherché à devenir de plus en plus autonome et à prendre des initiatives pour mener à bien ma mission.

# **2 - PRESENTATION DE L'ENTREPRISE**

## **2.1 - HISTORIQUE ET DOMAINE D'ACTIVITE**

NoBullShit Tech Engineers est une entreprise dynamique et innovante spécialisée dans le domaine de la technologie et des services informatiques. Fondée en septembre 2022, notre entreprise a rapidement acquis une solide réputation en tant que partenaire de confiance pour résoudre les problématiques les plus complexes liées à l'urbanisation, à l'abstraction et à la modernisation des systèmes d'information.

Notre domaine d'activité couvre une large gamme de services, allant de l'architecture logicielle à la conception et au développement de solutions informatiques sur mesure. Nous intervenons auprès de diverses entreprises, des start-ups aux grandes entreprises françaises, en mettant notre expertise et notre expérience à leur service.

Notre approche se distingue par notre engagement envers la transparence, la communication et la collaboration étroite avec nos clients. Nous croyons fermement que comprendre les défis auxquels nos clients sont confrontés est essentiel pour proposer des solutions efficaces et adaptées à leurs besoins spécifiques.

Grâce à notre équipe composée de talents exceptionnels, nous sommes en mesure d'offrir des services de haut niveau dans des domaines tels que l'ingénierie logicielle, l'architecture système, le développement web, la sécurité informatique et bien plus encore. Nous nous tenons constamment informés des dernières tendances et technologies afin de rester à la pointe de l'innovation et de répondre aux exigences changeantes du marché.

Notre objectif ultime est de redresser la France de son exécution technique sub-optimale et de former les leaders de demain.

En résumé, NoBullShit Tech Engineers est une entreprise passionnée et engagée, prête à relever les défis les plus complexes dans le domaine de la technologie et à aider ses clients à réussir grâce à des solutions informatiques innovantes et adaptées.

# **2 - PRESENTATION DE L'ENTREPRISE**

## **2.2 - STRUCTURE ORGANISATIONNELLE**

Le projet sur lequel j'ai travaillé avait une structure organisationnelle flexible et collaborative. Plutôt que de suivre une hiérarchie stricte, notre équipe était axée sur la coopération et l'entraide.

Le travail était supervisé par Alexandre Cailler, Romain Boudot et François-Guillaume Ribreau. Alexandre et Romain jouaient un rôle essentiel en fournissant des orientations, des conseils et des directives tout au long du projet. Leur expérience et leur expertise étaient précieuses pour nous guider dans nos tâches et nos prises de décision.

François-Guillaume intervenait à la fin du processus pour apporter son dernier avis sur notre travail. Son rôle consistait à évaluer en détail nos réalisations et à s'assurer qu'elles répondaient aux normes de qualité élevées de l'entreprise. Son expertise et son regard critique nous ont permis d'améliorer nos compétences et de garantir la réussite du projet.

En termes de structure, nous fonctionnions en tant qu'équipe collégiale, où chacun avait la possibilité de contribuer et de partager ses idées. Les décisions étaient prises collectivement, en favorisant les échanges d'informations et la communication ouverte entre les membres de l'équipe.

Cette approche nous a permis de tirer parti des compétences et des connaissances de chacun, créant ainsi un environnement propice à l'apprentissage et à la croissance professionnelle. La structure organisationnelle flexible nous a également permis de nous adapter rapidement aux défis et aux exigences changeantes du projet.

## 2 - PRESENTATION DE L'ENTREPRISE

### 2.3 - PRESENTATION DU SERVICE

Le service dans lequel j'ai effectué mon stage était un petit groupe de développement composé de trois développeurs Full Stack expérimentés : Alexandre Cailler, Romain Boudot et François-Guillaume Ribreau. Ensemble, nous formions une équipe agile et dynamique, chargée de la réalisation d'un projet technologique passionnant.



Alexandre Cailler



Romain Boudot



François Guillaume  
Ribreau



Moi (Mathéo DELAUNAY) Stagiaire

En tant que développeurs Full Stack, nous étions responsables de toutes les étapes du processus de développement, depuis la conception initiale jusqu'à la livraison finale du produit. Notre expertise couvrait à la fois le développement côté serveur (backend) et le développement côté client (frontend), ce qui nous permettait d'avoir une vision globale du projet et de prendre des décisions éclairées à chaque étape.

During my internship, I realized daily reports. These reports were essential to ensure good communication and transparency within the team. They allowed me to share my progress, my difficulties encountered and my proposed solutions. Thanks to these reports, my colleagues and my internship supervisor were constantly informed of the state of advancement of my tasks and could provide me with advice and orientations.

# **3 - EXPERIENCE DU TRAVAIL A DISTANCE**

## **3.1 - INTRODUCTION AU TELETRAVAIL**

Dans le cadre de mon stage au sein de cette entreprise axée sur le travail en freelance et la promotion du travail à distance, j'ai eu l'opportunité de vivre une expérience significative de télétravail. Cette approche s'inscrit dans une démarche écologique visant à réduire les déplacements et à favoriser un environnement de travail propice à la concentration et à la tranquillité.

Durant cette période, j'ai eu la flexibilité de travailler à distance depuis mon établissement scolaire ainsi que depuis mon domicile. Ce mode de travail à distance a nécessité une adaptation et une organisation spécifiques afin de maintenir un haut niveau de productivité et d'efficacité. J'ai dû développer des compétences en matière de gestion du temps, de planification autonome, de communication à distance et d'utilisation des outils technologiques adaptés (Slack, Google meet, etc).

L'expérience du télétravail m'a permis de comprendre les avantages et les opportunités qu'il offre. J'ai pu apprécier, la réduction des temps de trajet, la possibilité de travailler dans un environnement familial et confortable, ainsi que la diminution des interruptions fréquentes présentes dans un environnement de bureau traditionnel.

Cependant, j'ai également été confronté à certains défis propres au télétravail. La gestion de la solitude, la nécessité de maintenir une discipline personnelle, la communication efficace à distance et la séparation entre vie professionnelle et vie personnelle sont autant d'aspects qui ont nécessité une réflexion et des solutions adaptées.

Cette expérience de travail à distance a été enrichissante et m'a permis de développer des compétences clés telles que l'autonomie, la gestion du temps, l'adaptabilité et la résolution de problèmes à distance. Elle a également renforcé ma capacité à m'organiser de manière efficace et à travailler de manière indépendante, tout en maintenant un haut niveau de productivité.

Dans les sections suivantes de ce rapport de stage, je détaillerai plus en profondeur les avantages et les inconvénients du télétravail que j'ai pu observer, ainsi que les solutions que j'ai mises en place pour surmonter les défis rencontrés.

# 3 - EXPERIENCE DU TRAVAIL A DISTANCE

## DISTANCE

### 3.2 - PRESENTATION DE MON ENVIRONNEMENT DE TRAVAIL

Pour présenter mon environnement de travail, je suis fier de vous décrire un espace soigneusement aménagé pour favoriser ma productivité et ma concentration. J'ai pris le soin de sélectionner des équipements adaptés, notamment un MacBook Air M2, un ordinateur portable puissant et compact, qui constitue l'outil central de mes activités professionnelles. Afin d'améliorer mon expérience de travail, j'ai également intégré un écran de 42 pouces, offrant une généreuse surface d'affichage pour une meilleure visualisation de mes applications et documents.

Mon bureau se distingue par son approche minimaliste, conçue dans le but de réduire les distractions et de maintenir un environnement propice à la concentration. J'ai volontairement éliminé les éléments superflus, ne conservant que l'essentiel à portée de main : mon ordinateur, mon clavier et ma souris. Cette approche a créé un espace épuré et fonctionnel, favorisant ainsi ma productivité et ma clarté d'esprit.

Afin de vous donner un aperçu visuel de cet agencement minimaliste, j'ai joint une photo de mon environnement de travail qui illustrent les choix que j'ai faits pour limiter les distractions et maximiser mon efficacité.



Image de mon lieu de travail

# **3 - EXPERIENCE DU TRAVAIL A DISTANCE**

## **3.3 - AVANTAGE ET OPPORTUNITES DU TELETRAVAIL**

Le télétravail offre de nombreux avantages qui favorisent la concentration, la productivité et le bien-être au travail. Travailler dans un environnement familial, que ce soit à mon établissement scolaire ou chez moi, me permet de me plonger pleinement dans mes missions sans les distractions habituelles d'un lieu de travail traditionnel. Je peux ainsi me concentrer efficacement et produire un travail de qualité.

De plus, le télétravail me permet de m'adapter aux outils numériques, une compétence de plus en plus importante dans le monde professionnel d'aujourd'hui. En travaillant à distance, je suis amené à utiliser des plateformes de communication en ligne, des logiciels collaboratifs et d'autres technologies, ce qui me permet d'acquérir de nouvelles compétences techniques et de me familiariser avec les pratiques de travail à distance.

Un autre avantage du télétravail est la possibilité de créer un environnement de travail personnalisé. J'ai aménagé mon espace de travail de manière à optimiser ma productivité et mon confort. En éliminant les distractions inutiles, j'ai créé un environnement propice à la concentration et à la clarté d'esprit.

Par ailleurs, le télétravail contribue à la réduction de l'empreinte environnementale. En évitant les déplacements quotidiens, je limite les émissions de gaz à effet de serre et la pollution atmosphérique. Cela participe à la préservation de l'environnement et à la lutte contre le changement climatique.

Enfin, travailler à distance améliore ma productivité globale. Dans un environnement familial et personnalisé, je peux minimiser les distractions et les interruptions, ce qui me permet de me concentrer davantage sur mes tâches. Je peux également gérer mon emploi du temps de manière plus flexible, ce qui me permet d'optimiser mon efficacité.

En somme, le télétravail présente de nombreux avantages, tels que la concentration accrue, l'adaptation aux outils numériques, un environnement de travail personnalisé, une réduction de l'empreinte environnementale et une amélioration globale de la productivité. Ces avantages contribuent à rendre mon expérience de stage plus enrichissante et efficace.

# **3 - EXPERIENCE DU TRAVAIL A DISTANCE**

## **3.4 - DEFIS ET SOLUTIONS DU TELETRAVAIL**

Le télétravail, bien qu'il présente de nombreux avantages, peut également présenter des défis auxquels j'ai dû faire face durant mon stage. Cependant, j'ai pu trouver des solutions efficaces pour les surmonter et tirer le meilleur parti de cette modalité de travail à distance.

L'un des défis du télétravail est la gestion de la communication et de la collaboration avec l'équipe. Étant éloigné physiquement de mes collègues, il était important de maintenir une communication fluide et efficace. Pour cela, nous avons utilisé des outils de communication en ligne tels que Slack, nous permettant de rester connectés en temps réel et d'échanger des informations essentielles.

Un autre défi auquel j'ai été confronté était de trouver un équilibre entre le travail et le développement de mes connaissances. Étant passionné par mon domaine d'études, j'étais motivé à approfondir mes compétences et à explorer de nouveaux domaines. Cela signifiait que je ne comptais pas mes heures et que je travaillais souvent au-delà des horaires normaux.

Un autre défi fut la gestion de ma motivation et l'isolement ressenti en travaillant seul à distance. Travailler sans la présence physique de collègues peut parfois être isolant, et cela peut affecter notre motivation et notre engagement.

Pour surmonter ce défi, j'ai décidé de diversifier mon environnement de travail en me rendant régulièrement dans mon école. Cela m'a permis de bénéficier de la présence d'autres étudiants et d'avoir des interactions sociales. En travaillant dans un environnement partagé, j'ai pu retrouver une certaine dynamique de groupe, échanger des idées avec mes pairs et recevoir des avis extérieurs sur mes projets. Cette expérience m'a offert la possibilité de sortir de l'isolement et de maintenir ma motivation grâce à une certaine dose de sociabilité.

# **3 - EXPERIENCE DU TRAVAIL A DISTANCE**

## **3.5 - REFLEXIONS SUR L'EXPERIENCE DU TELETRAVAIL**

En conclusion, le télétravail a été une expérience extrêmement bénéfique durant mon stage. Il m'a offert la flexibilité et la liberté de travailler depuis des lieux variés tels que mon établissement scolaire et mon domicile. J'ai pu profiter d'un environnement de travail soigneusement aménagé, avec mon MacBook Air M2 et mon écran de 42 pouces, favorisant ainsi ma concentration et ma productivité. Travailler à distance m'a permis de développer mes compétences dans l'utilisation des outils numériques et de m'adapter aux pratiques de travail à distance, des compétences de plus en plus importantes dans le monde professionnel moderne.

J'ai également été confronté à des défis, tels que maintenir une routine de travail disciplinée et gérer les frontières entre le travail et la vie personnelle. Cependant, j'ai pu trouver des solutions en établissant une routine structurée et en créant un espace de travail dédié. De plus, le fait de travailler dans mon école m'a permis de bénéficier de la sociabilité et des avis extérieurs, ajoutant ainsi une dimension enrichissante à mon expérience de travail à distance.

Enfin, en tant que passionné de café, j'ai apprécié pouvoir déguster un bon café tout en travaillant, sans avoir à passer du temps sur les routes pour me rendre au travail. Cette petite touche de plaisir a contribué à renforcer ma motivation et mon enthousiasme au quotidien.

Dans l'ensemble, le télétravail a été une expérience positive, me permettant de concilier productivité, confort et équilibre entre vie professionnelle et vie personnelle. Il m'a offert de nouvelles perspectives sur les pratiques de travail modernes et a renforcé mes compétences dans l'utilisation des outils numériques. Je suis convaincu que le télétravail continuera à jouer un rôle essentiel dans le monde professionnel, offrant de multiples avantages aux travailleurs et aux entreprises.

# 4 - PRESENTATION DES MISSIONS

## 4.1 - DESCRIPTION DU PROJET

Le projet sur lequel j'ai travaillé consistait à recréer un formulaire d'incident en utilisant l'API de Better Uptime. Notre objectif principal était de développer un formulaire flexible, permettant de moduler les champs en fonction des besoins spécifiques de chaque utilisateur.

Pour ce faire, nous avons utilisé les fonctionnalités de l'API de Better Uptime pour récupérer les informations nécessaires et les afficher dans notre formulaire personnalisé. Nous avons pris en compte les différents types de champs, les validations et les formats de données requis pour assurer une expérience utilisateur optimale.

Une fois le formulaire d'incident développé, nous l'avons déployé sur une fonction cloud de Google Cloud Platform (GCP). Cela nous a permis de bénéficier de la scalabilité et de la flexibilité offertes par GCP, garantissant ainsi une disponibilité maximale du service.

En parallèle, nous avons utilisé Cypress, un outil de test automatisé, pour créer des scénarios de test exhaustifs. Ces tests automatisés nous ont permis de vérifier le bon fonctionnement du formulaire, de détecter d'éventuels bugs et de nous assurer que les validations étaient correctement appliquées.

Pour faciliter la mise en production du formulaire d'incident, nous avons utilisé Docker pour créer un conteneur. Cette approche nous a permis d'empaqueter toutes les dépendances et configurations nécessaires dans un environnement isolé, garantissant ainsi une portabilité et une cohérence du système.

Enfin, nous avons utilisé Trivy, un outil d'analyse de vulnérabilités, pour inspecter le contenu du conteneur Docker. L'objectif était d'identifier et de remédier à d'éventuelles failles de sécurité, assurant ainsi la protection des données sensibles traitées par le formulaire.

Ce projet a été une occasion unique d'explorer différentes technologies et d'appliquer des bonnes pratiques de développement logiciel. Grâce à cette expérience, j'ai pu renforcer mes compétences en intégration d'API, en déploiement sur le cloud, en tests automatisés, en conteneurisation et en sécurité des applications.

# 4 - PRESENTATION DES MISSIONS

## 4.2 - DESCRIPTION DE LA 1er MISSION

Recréation du formulaire d'incident avec l'API de Better Uptime

La mission de recréation du formulaire d'incident avec l'API de Better Uptime consiste à développer un nouveau formulaire permettant aux utilisateurs de signaler des incidents ou des problèmes techniques. L'API de Better Uptime sera utilisée pour gérer et enregistrer les incidents signalés.

L'objectif principal de cette mission est de concevoir et de mettre en place un formulaire convivial et intuitif, offrant aux utilisateurs la possibilité de fournir toutes les informations nécessaires concernant l'incident, telles que sa description, sa priorité, sa catégorie, etc. Le formulaire devra également permettre de joindre des fichiers ou des captures d'écran pour mieux documenter l'incident.

Une attention particulière sera accordée à la personnalisation du formulaire, en offrant la possibilité de moduler les champs selon les besoins spécifiques de l'entreprise. Cela garantira une flexibilité et une adaptabilité maximales aux différentes situations et contextes.

L'utilisation de l'API de Better Uptime permettra de créer un lien direct entre le formulaire d'incident et la gestion des incidents au sein de l'entreprise. Les incidents signalés par les utilisateurs seront automatiquement enregistrés dans le système de gestion des incidents de Better Uptime, facilitant ainsi leur suivi et leur résolution par l'équipe responsable.

Cette mission requiert des compétences en développement web, en particulier dans les langages de programmation tels que HTML, CSS et JavaScript. Une connaissance de l'API de Better Uptime sera également nécessaire pour assurer l'intégration réussie du formulaire avec la plateforme de gestion des incidents.

# 4 - PRESENTATION DES MISSIONS

## 4.3 - DESCRIPTION DE LA 2eme MISSION

Déploiement du formulaire sur Google Cloud Platform (GCP) en utilisant Cloud Functions et sur Docker en utilisant des conteneurs

La deuxième mission du projet consiste à déployer le formulaire d'incident sur Google Cloud Platform (GCP) en utilisant Cloud Functions, ainsi que sur Docker en utilisant des conteneurs.

Pour le déploiement sur GCP, nous utiliserons Cloud Functions, qui est un service sans serveur permettant d'exécuter du code en réponse à des événements. Nous configurerons une fonction Cloud qui sera déclenchée lorsque le formulaire d'incident est soumis. Cette fonction sera responsable de traiter les données du formulaire, de les valider et de les transmettre à l'API de Better Uptime. Nous veillerons à ce que la fonction Cloud soit correctement configurée, avec les autorisations appropriées et les dépendances nécessaires.

En parallèle, nous déployerons également le formulaire d'incident sur Docker en utilisant des conteneurs. Docker est une plateforme de virtualisation légère qui permet de créer, de déployer et de gérer des applications dans des conteneurs. Nous créerons un conteneur Docker contenant l'application du formulaire d'incident, ainsi que toutes les dépendances nécessaires. Ce conteneur pourra ensuite être facilement déployé sur n'importe quel environnement compatible avec Docker, offrant ainsi une portabilité et une flexibilité accrues.

Le déploiement sur GCP et Docker offre plusieurs avantages. Sur GCP, nous bénéficierons de la scalabilité automatique et de la gestion transparente de l'infrastructure, ce qui garantit une disponibilité élevée et des performances optimales. En utilisant Docker, nous assurons une isolation et une gestion efficace des dépendances, facilitant ainsi le déploiement de l'application dans différents environnements de manière cohérente.

# **4 - PRESENTATION DES MISSION**

## **4.4 - DESCRIPTION DE LA 3eme MISSION**

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

La troisième mission du projet consiste à réaliser des tests automatisés avec Cypress et à effectuer une analyse de sécurité avec Trivy.

Pour les tests automatisés, nous utiliserons Cypress, un framework de test End-to-End qui permet de simuler les interactions d'un utilisateur avec l'application et de vérifier le bon fonctionnement de celle-ci. Nous développerons des scénarios de test qui couvrent les fonctionnalités clés du formulaire d'incident, en simulant les actions de l'utilisateur, en vérifiant les résultats attendus et en détectant les éventuelles erreurs. Les tests automatisés nous permettront de s'assurer de la qualité du formulaire et d'identifier rapidement les éventuels problèmes ou dysfonctionnements.

En ce qui concerne l'analyse de sécurité, nous utiliserons Trivy, un outil open-source spécialisé dans la détection des vulnérabilités des conteneurs Docker. Nous analyserons le conteneur dans lequel le formulaire d'incident est déployé à l'aide de Trivy. Cet outil effectuera une analyse approfondie des dépendances, des bibliothèques et des images utilisées dans le conteneur, afin d'identifier les éventuelles failles de sécurité connues. Cette analyse nous permettra de prendre des mesures préventives pour corriger les vulnérabilités et garantir la sécurité de l'application.

Les tests automatisés avec Cypress et l'analyse de sécurité avec Trivy offrent une double approche pour assurer la qualité et la sécurité de l'application. Les tests automatisés permettent de détecter les erreurs fonctionnelles et d'assurer un bon fonctionnement du formulaire d'incident, tandis que l'analyse de sécurité avec Trivy permet de détecter et de résoudre les éventuelles failles de sécurité.

# 5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS

## 5.1 - ANALYSE PRELIMINAIRE DES BESOINS

Pour l'analyse préliminaire des besoins liés aux trois missions du projet, nous avons identifié les éléments suivants :

Mission 1 - Recréation du formulaire d'incident avec l'API de Better Uptime :

- Compréhension approfondie des fonctionnalités et des exigences du formulaire d'incident existant.
- Étude approfondie de l'API de Better Uptime et de ses fonctionnalités pour intégrer les données de l'incident.
- Identification des champs requis pour le formulaire d'incident et possibilité de les personnaliser selon les besoins spécifiques du projet.
- Conception d'une interface utilisateur conviviale et intuitive pour faciliter la saisie des informations de l'incident.
- Intégration de la validation des données pour garantir la saisie correcte et cohérente des informations.
- Mise en place d'une gestion des erreurs appropriée pour informer les utilisateurs en cas de problèmes lors de la soumission du formulaire.

Mission 2 - Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker :

- Évaluation des exigences en termes d'environnement de déploiement et de ressources nécessaires pour exécuter le formulaire d'incident.
- Sélection et configuration de Google Cloud Platform (GCP) en fonction des besoins du projet.
- Utilisation de Cloud Functions de GCP pour déployer le formulaire en tant que service évolutif et hautement disponible.
- Configuration et déploiement d'un conteneur Docker pour exécuter le formulaire d'incident de manière isolée et portable.
- Assurer la compatibilité et l'intégration correcte du formulaire avec les services et les fonctionnalités de GCP et Docker.
- Configuration des mécanismes de surveillance et de journalisation appropriés pour assurer une visibilité et une gestion efficaces du formulaire déployé.

# **5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS**

## **5.1 - ANALYSE PRELIMINAIRE DES BESOINS**

Mission 3 - Tests automatisés avec Cypress et analyse de sécurité avec Trivy :

- Identification des scénarios de test clés pour le formulaire d'incident à l'aide de Cypress.
- Mise en place de l'environnement de test avec Cypress, y compris la configuration des dépendances et des données de test.
- Développement des scripts de test automatisés pour simuler les interactions de l'utilisateur et vérifier les fonctionnalités du formulaire.
- Exécution régulière des tests automatisés pour détecter les erreurs fonctionnelles et garantir la qualité du formulaire d'incident.
- Utilisation de Trivy pour analyser le conteneur Docker du formulaire et détecter les éventuelles vulnérabilités et failles de sécurité connues.
- Mise en place de mesures correctives pour résoudre les vulnérabilités identifiées et renforcer la sécurité du formulaire.

# 5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS

## 5.2 - PLANIFICATION DES ACTIVITES

Semaine 1 :

- Définition des objectifs du projet.
- Analyse des exigences et des fonctionnalités attendues.
- Étude des technologies et des outils à utiliser.
- Mise en place de l'environnement de développement.

Semaine 2 :

- Conception de l'architecture de l'application.
- Création de la structure du projet.
- Mise en place des bases de données et des modèles de données.
- Implémentation des fonctionnalités de base, telles que l'inscription, la connexion et la gestion des utilisateurs.

Semaine 3 :

- Finalisation des fonctionnalités de gestion des utilisateurs.
- Implémentation des fonctionnalités liées aux articles, comme la création, la modification et la suppression.
- Mise en place de l'interface utilisateur pour les différentes fonctionnalités.
- Tests unitaires et d'intégration pour assurer la qualité du code.

Semaine 4 :

- Création de scripts de tests automatisés avec Cypress.
- Mise en place d'un pipeline pour exécuter les tests de manière régulière.
- Correction des problèmes rencontrés lors de l'exécution des tests.
- Investigation sur les problèmes liés au formulaire public et à l'envoi des données à l'API.

# **5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS**

## **5.2 - PLANIFICATION DES ACTIVITES**

Semaine 5 :

- Révision et optimisation du code existant pour améliorer les performances et la maintenabilité.
- Ajout de fonctionnalités avancées, telles que la recherche d'articles, le tri et le filtrage.
- Implémentation d'une fonctionnalité de commentaires pour permettre aux utilisateurs d'interagir avec les articles.
- Mise en place d'un système de notifications pour informer les utilisateurs des activités pertinentes.
- Travail sur l'interface utilisateur pour améliorer l'expérience utilisateur.

Semaine 6 :

- Ajout de tests automatisés pour vérifier les erreurs affichées en rouge
- Utilisation de Mustache pour permettre la modification dynamique des textes avec des variables d'environnement
- Modification du code pour utiliser Mustache et adaptation des tests
- Participation à des réunions pour discuter des modifications à apporter

# **5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS**

## **5.3 - SELECTION DES OUTILS ET DES RESSOURCES NECESSAIRES**

Dans cette phase du projet, j'ai procédé à la sélection des outils et des ressources nécessaires pour mener à bien mes missions. J'ai pris en compte différents aspects tels que la gestion des tests, le développement du code, la gestion des dépendances et l'hébergement de mon application.

Pour la gestion des tests, j'ai opté pour l'utilisation de Postman et Cypress. Postman est un outil populaire qui m'a permis de tester les API et de valider leur bon fonctionnement. Il m'a offert la possibilité d'envoyer des requêtes HTTP, de visualiser les réponses et de réaliser des tests automatisés. Quant à Cypress, il s'agit d'un framework de tests End-to-End qui m'a permis de vérifier le bon comportement de mon application dans un environnement simulé. J'ai ainsi pu réaliser des tests de bout en bout pour garantir la qualité et la fiabilité de mon application.

Pour le développement du code, j'ai utilisé plusieurs outils. Tout d'abord, j'ai utilisé WebStorm, un environnement de développement intégré (IDE) développé par JetBrains. Cet outil puissant m'a offert des fonctionnalités avancées pour le développement JavaScript. J'ai pu programmer de manière efficace en bénéficiant de fonctionnalités de complémentation automatique, de débogage et de gestion de projets. Ensuite, j'ai utilisé Docker pour la création et la gestion des conteneurs. Docker m'a permis de packager mon application avec toutes ses dépendances dans un conteneur, garantissant ainsi la portabilité et l'isolation de mon application.

# **5 - DEMARCHE SUIVIE POUR REALISER LES MISSIONS**

## **5.3 - SELECTION DES OUTILS ET DES RESSOURCES NECESSAIRES**

En ce qui concerne l'hébergement de mon projet, j'ai choisi d'utiliser Google Cloud Platform (GCP). J'ai utilisé les fonctionnalités des Cloud Functions pour héberger mon projet de manière évolutive et sans effort de gestion des serveurs. GCP m'a offert une infrastructure solide et fiable pour exécuter mon application dans le cloud.

Par ailleurs, j'ai utilisé GitLab pour la gestion de mes fichiers de code source. GitLab m'a permis de versionner mon code, de collaborer avec d'autres développeurs et de gérer efficacement les différentes branches de mon projet. J'ai également réalisé un benchmark entre GitLab et GitHub pour évaluer les performances et les fonctionnalités offertes par ces deux plateformes de gestion de code.

Enfin, pour gérer les variables d'environnement, j'ai utilisé le fichier .envrc plutôt que simplement le fichier .env. Cette approche m'a permis de sécuriser mes variables d'environnement et de les gérer de manière plus fine. J'ai également réalisé un benchmark pour comparer les avantages et les performances entre l'utilisation de .envrc et celle de .env.

L'ensemble de ces outils et ressources a joué un rôle crucial dans le bon déroulement de mon projet, en me permettant d'optimiser mon développement, d'assurer la qualité de mon code, de faciliter la collaboration et de garantir la stabilité de mon application.

Dans l'ensemble, la sélection minutieuse de ces outils et ressources a contribué à la réussite de nos missions en garantissant une gestion efficace des tests, du développement du code, des dépendances et de l'hébergement de notre application.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

Avant de recréer le formulaire d'incident avec l'API de Better Uptime, j'ai effectué une analyse approfondie du site existant pour comprendre sa structure et ses fonctionnalités. J'ai également pris le temps de me documenter sur l'API de Better Uptime afin de comprendre son fonctionnement et ses endpoints. Parallèlement, j'ai étudié le BEM pour adopter une approche de développement cohérente et maintenable, et j'ai appris le SCSS pour écrire un code CSS plus structuré. Grâce à ces préparatifs, j'étais prêt à recréer le formulaire d'incident en utilisant l'API de Better Uptime avec une meilleure compréhension du contexte et des outils nécessaires.

The screenshot shows the 'Report new incident to Additi - Funéraire' page. At the top right, there's a user profile for 'Mathéo DELAUNAY'. The main form fields include:

- What's going on?**: A text area for describing the issue, with a note: "Please describe what's going on including steps to reproduce the issue. The more specific you are, the easier it's going to be for your colleagues to locate the issue and resolve the incident."
- Brief description (maximum of 100 characters)**: An input field containing the placeholder "Example: New users can't sign up".
- In-depth description**: A large text area for detailed information.
- Attach files or screenshots**: A section with a "Drag & drop files or screenshots" button and a "Choose files" button.
- Your contact details**: A section with a note: "We'll keep you posted about the incident updates." It includes an "Your e-mail" input field with the value "matheodelaunay04@gmail.com".
- Who should we notify?**: A section with a note: "Immediately notify via" followed by a dropdown menu.

Aperçu du site de Better Uptime à refaire

Avant de commencer à coder, j'ai également appris mes connaissances sur les différents workflows de gestion de versions, notamment GitFlow et Trunk Based Development. J'ai compris les principes et les avantages de chaque approche. Pour ce projet, j'ai opté pour Trunk Based Development, une méthode qui favorise une branche principale (trunk) où se trouvent les développements en cours et les déploiements fréquents. Cette approche m'a permis de simplifier la gestion des versions, de collaborer de manière fluide et de garantir des cycles de développement rapides et itératifs.

# 6 - REALISATION DES MISSIONS

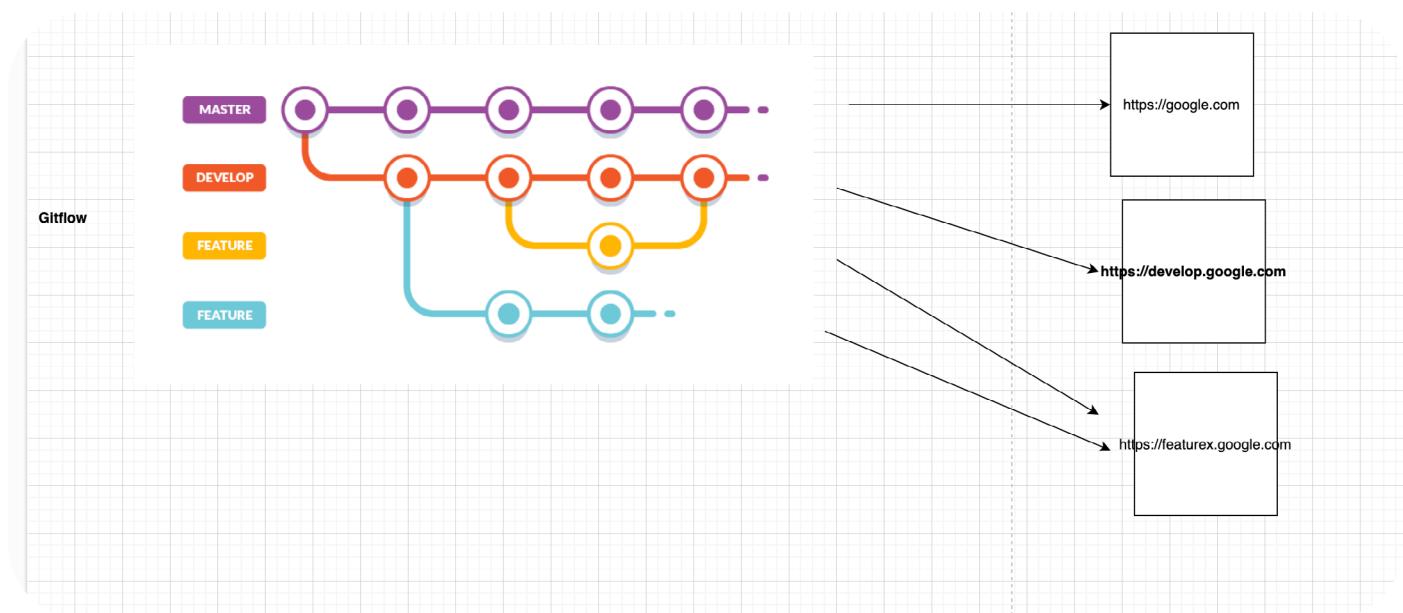
## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

La différence entre GitFlow et Trunk Based Development

La différence entre GitFlow et Trunk Based Development réside dans la façon dont les branches sont utilisées et gérées dans un workflow de gestion de versions.

GitFlow est un modèle de gestion de versions qui utilise plusieurs branches pour séparer les différentes étapes du développement. Il comprend des branches de fonctionnalités, de développement, de version, de correction de bugs, etc. Les développements sont réalisés dans des branches de fonctionnalités distinctes, puis fusionnés dans des branches de développement et finalement dans une branche principale (habituellement "master" ou "main") lors des releases. GitFlow permet une gestion plus complexe des versions, des releases et des hotfixes, mais peut entraîner une surcharge de branches et de fusions.



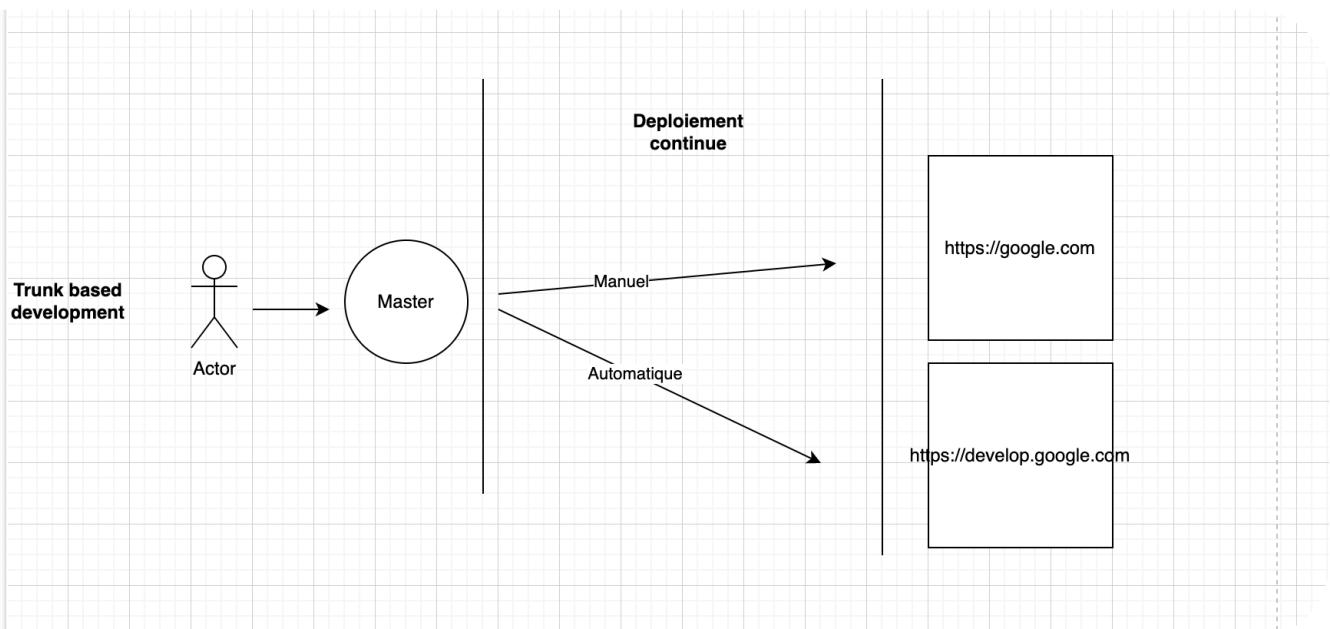
Le schéma du Gitflow

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

En revanche, Trunk Based Development est un modèle plus simple où le développement se fait principalement sur une seule branche principale (trunk). Les nouvelles fonctionnalités sont développées directement sur cette branche principale, ce qui facilite la collaboration en temps réel et la livraison fréquente de nouvelles fonctionnalités. Il met l'accent sur une intégration continue et des déploiements réguliers. Trunk Based Development favorise un développement plus agile, avec moins de complexité de gestion des branches, mais nécessite une approche rigoureuse des tests et des validations pour garantir la stabilité du tronc principal.



Le schéma du trunk based development

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

### Recréation du formulaire d'incident avec l'API de Better Uptime

J'ai commencé par coder la structure HTML et les styles CSS pour recréer le formulaire d'incident. J'ai utilisé les balises HTML appropriées pour structurer le contenu, tels que `<header>`, `<nav>`, `<main>`, et `<form>`. J'ai attribué des classes CSS aux éléments pour les styliser en utilisant des sélecteurs et des règles définies dans le fichier `styles.css`.

J'ai également utilisé des balises `<h1>`, `<h2>`, `<p>`, `<label>`, `<input>`, `<textarea>`, et `<select>` pour ajouter du contenu texte et des champs interactifs au formulaire. J'ai utilisé des attributs comme `class`, `placeholder`, et `value` pour personnaliser le comportement des éléments.

En termes de styles, j'ai créé des classes CSS correspondantes pour chaque élément du formulaire, en utilisant une approche basée sur BEM (Block-Element-Modifier). Cela m'a permis de structurer mes styles de manière modulaire et réutilisable.

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="styles.css">
    <title>Signaler un nouvel incident à Additi - Funéraire
        | Meilleur temps de disponibilité</title>

    <link href="https://fonts.cdnfonts.com/css/sf-pro-display" rel="stylesheet">
</head>
<body>
    <header></header>
    <div class="flex">
        <main>
            <h1 class="main__title">Signaler un nouvel incident à Additi - Funéraire</h1>
            <form action="#" class="form">
                <div class="form__sections form__section1">
                    <div class="form__heading">
                        <div class="form__heading-title">
                            <h2>Que se passe-t-il ?</h2>
                        </div>
                        <div class="form__heading-text">
                            <p>Veuillez décrire ce qui se passe, y compris les étapes pour reproduire le problème. Plus vous serez précis, plus il
                            </p>
                        </div>
                    </div>
                    <div class="form__body">
                        <label class="text-label">
                            <a>Brève description (100 caractères maximum)</a>
                            <input type="text" placeholder="Example : Les nouveaux utilisateurs ne peuvent pas s'inscrire">
                        </label>
                    <label class="text-label">
                        <a>Description détaillée</a>
                        <textarea></textarea>
                    </label>
                </div>
            </form>
        </main>
    </div>
</body>
</html>
```

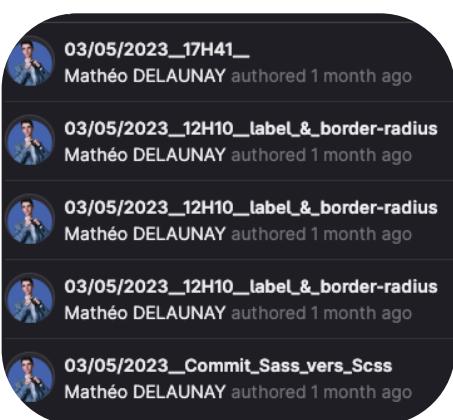
Le fichier `index.html` au début du projet

# 6 - REALISATION DES MISSIONS

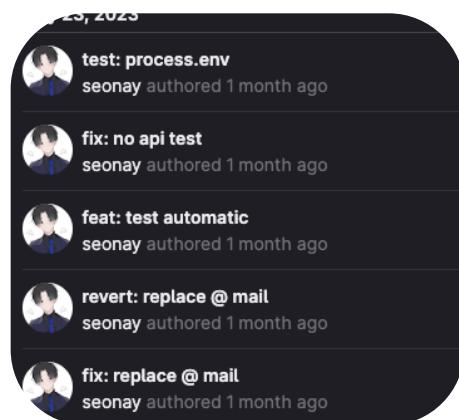
## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

### Recréation du formulaire d'incident avec l'API de Better Uptime

Plus tard, lors d'un des review revue de mon code, il a été remarqué que mes messages de commit ne suivaient pas les bonnes pratiques de nommage. L'équipe de développement m'a alors fourni des conseils sur les conventions de nommage des commits Git. J'ai appris qu'il est important d'écrire des messages de commit clairs et concis, en commençant par une phrase impérative et en utilisant des verbes d'action. Cela permet de faciliter la compréhension des changements effectués dans le code et d'assurer une meilleure traçabilité des modifications. J'ai donc pris en compte ces recommandations et j'ai ajusté mes messages de commit en conséquence pour améliorer la lisibilité de l'historique du projet.



AVANT



APRÈS

Le type du commit est un élément important pour informer les autres développeurs sur la nature des changements effectués. Il existe neuf types de commit couramment utilisés :

- build : pour les changements qui affectent le système de build ou les dépendances externes utilisées, comme npm, make, etc.
- ci : pour les changements liés aux fichiers et scripts d'intégration ou de configuration, tels que Travis, Ansible, BrowserStack, etc.
- feat : pour l'ajout d'une nouvelle fonctionnalité.
- fix : pour la correction d'un bug.
- perf : pour l'amélioration des performances du code.
- refactor : modification qui n'apporte ni nouvelle fonctionnalité ni d'amélioration de performances
- style : pour les changements qui n'apportent aucune altération fonctionnelle ou sémantique, tels que l'indentation, la mise en forme, l'ajout d'espaces, le renomage d'une variable, etc.
- docs : pour la rédaction ou la mise à jour de la documentation.
- test : pour l'ajout ou la modification des tests.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

### Recréation du formulaire d'incident avec l'API de Better Uptime

Après avoir terminé le développement initial de mon application, j'ai entrepris le déploiement sur la plateforme GCP Cloud Function. Pour ce faire, j'ai créé un routeur en utilisant Node.js, qui joue un rôle central dans la gestion des requêtes entrantes et le routage vers les fichiers appropriés de mon application.

Le routeur a été conçu pour traiter les différentes URL et rediriger les requêtes vers les fichiers correspondants. Cela permet à mon application de répondre de manière adéquate aux différentes fonctionnalités demandées par les utilisateurs. J'ai pu définir les routes en utilisant des fonctionnalités telles que les routes express ou les routes intégrées à mon framework de développement.

En utilisant les bibliothèques Axios et Functions, j'ai pu étendre les fonctionnalités de mon routeur. Axios, une bibliothèque JavaScript populaire, m'a permis de réaliser facilement des requêtes HTTP depuis mon code, que ce soit pour interagir avec mon API interne ou pour accéder à des services externes. J'ai pu effectuer des appels GET, POST, PUT ou DELETE et traiter les réponses pour fournir les données demandées ou effectuer des actions spécifiques.

Quant à Functions, il s'agit d'une fonctionnalité de la plateforme GCP qui permet d'exécuter du code JavaScript de manière serverless. En déployant mon application en tant que Cloud Function, j'ai pu héberger mon code et le rendre accessible via des URL spécifiques. Functions a également pris en charge la gestion des appels entrants et l'exécution du code correspondant à chaque route définie dans le routeur.

L'utilisation du "switch" pour gérer les différentes conditions de routage a été une amélioration significative par rapport à l'approche initiale basée sur de multiples déclarations "if". Les déclarations "if" successives rendaient le code complexe et difficile à maintenir, surtout lorsque le nombre de routes augmentait. En utilisant le "switch", j'ai pu regrouper toutes les routes dans une seule structure de contrôle, ce qui a rendu mon code plus clair, plus lisible et plus facile à maintenir. Chaque route était définie comme un "case" dans le "switch", et le code correspondant à chaque route était exécuté de manière cohérente et structurée.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

Plus tard dans le développement de mon application, j'ai entrepris de factoriser le code pour le rendre encore plus lisible et explicite. La factorisation consiste à réorganiser et à simplifier le code existant tout en conservant la même fonctionnalité.

Pour rendre le code plus lisible, j'ai utilisé des techniques telles que l'extraction de fonctions ou de modules réutilisables. J'ai identifié les parties du code qui se répetaient et les ai isolées dans des fonctions distinctes. Cela m'a permis de réduire la redondance et d'améliorer la compréhension globale du code.

J'ai également cherché à rendre le code plus explicite en choisissant des noms de variables, de fonctions et de modules significatifs. Des noms bien choisis facilitent la compréhension du code, en indiquant clairement le rôle et la fonction de chaque élément.

La factorisation du code a également contribué à améliorer sa maintenabilité. En isolant les fonctionnalités dans des modules distincts, j'ai facilité la modification ou la mise à jour de chaque partie du code sans affecter le reste de l'application. De plus, en réduisant la redondance et en utilisant des noms explicites, j'ai rendu le code plus facile à comprendre et à déboguer.

Dans cet exemple j'ai créé une fonction answer qui est utilisée pour les Codes de réponse HTTP (500, 404, 401, etc) :

```
function answer(res, status_code, message, header : {} = {}) : void {  
    res.writeHead(status_code, header);  
    res.write(message);  
    res.end();  
}
```

La fonction answer

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

J'ai utilisé les variables d'environnement pour sécuriser les informations sensibles, telles que les URL d'API, les jetons d'authentification et d'autres données confidentielles nécessaires à mon application.

Pour gérer les variables d'environnement, j'ai utilisé l'outil `direnv`, qui me permet de charger automatiquement les variables d'environnement à partir d'un fichier ` `.envrc` situé dans le répertoire de mon projet.

Dans ce fichier ` `.envrc` , j'ai défini les différentes variables d'environnement nécessaires à mon application, en utilisant la syntaxe `export VARIABLE=valeur`. Par exemple, j'ai défini `API\_URL\_BETTER\_UPTIME` avec l'URL de l'API Better Uptime et `BETTER\_UPTIME\_TOKEN` avec le jeton d'authentification requis.

Il est important de noter que le fichier ` `.envrc` contenant les informations sensibles ne doit pas être versionné et partagé publiquement. Pour éviter cela, j'ai utilisé un modèle ` `.envrc.fr.dist` . Ce fichier modèle contient les noms des variables d'environnement nécessaires, mais les valeurs sont vides ou fictives. Ainsi, lorsqu'une autre personne récupère mon projet, elle peut créer son propre fichier ` `.envrc` en renseignant les valeurs correctes pour les variables d'environnement.

```
1  export BETTER_UPTIME_TOKEN=*****;  
2  
3  export CYPRESS_BETTER_UPTIME_TOKEN=*****;  
4  
5  export API_URL_BETTER_UPTIME=https://betteruptime.com/api/v2/incidents;  
6  
7  export CYPRESS_API_URL_BETTER_UPTIME=https://betteruptime.com/api/v2/incidents;  
8  
9  export SITE_URL=https://formulaire-better-uptime-otjhoifyna-ew.a.run.app/;  
10  
11 export CYPRESS_SITE_URL=https://formulaire-better-uptime-otjhoifyna-ew.a.run.app/;
```

Le fichier `.envrc.fr.dist`

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

J'ai utilisé la bibliothèque Mustache en conjonction avec les variables d'environnement pour rendre le texte affiché sur la page de manière modulaire et personnalisée.

Avec Mustache, j'ai créé des modèles de texte HTML contenant des balises spéciales, appelées "balises Mustache". Ces balises indiquent où les variables d'environnement doivent être insérées dans le texte final.

Par exemple, j'ai pu définir des messages d'erreur ou des instructions spécifiques à afficher sur la page en utilisant des balises Mustache. Les valeurs de ces balises sont alors remplies dynamiquement lors de l'exécution de l'application, en utilisant les variables d'environnement correspondantes.

Cette approche a permis de personnaliser facilement le contenu de la page en fonction des variables d'environnement spécifiées. Par exemple, si j'ai une variable d'environnement "APP\_NAME" contenant le nom de mon application, je peux l'insérer dans le modèle HTML en utilisant la balise Mustache {{ APP\_NAME }}. Lorsque le modèle est rendu, la balise sera remplacée par la valeur de la variable d'environnement, affichant ainsi le nom de l'application sur la page.

De plus, cette modularité s'étend également aux champs du formulaire. J'ai pu utiliser les balises Mustache pour définir les attributs des champs du formulaire, tels que les valeurs par défaut, les libellés et les attributs supplémentaires. Cela m'a permis d'adapter dynamiquement les champs du formulaire en fonction des variables d'environnement, offrant ainsi une plus grande flexibilité dans la configuration de l'application.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

Avec l'utilisation de Mustache, les champs HTML ne sont plus définis avec des valeurs fixes, comme on l'a vu précédemment avec l'exemple <title>Le titre de la page</title>. Au lieu de cela, on utilise des balises Mustache pour rendre les champs dynamiques.

Dans le front-end, on peut utiliser une balise Mustache pour définir le titre de la page, par exemple : <title>{{page.title}}</title>. Cette balise indique que la valeur du titre sera fournie dynamiquement lors du rendu de la page.

<title>{{page.title}}</title>

Dans le back-end, lors de l'appel de la page HTML, un "view" est créé. Ce "view" correspond aux valeurs définies dans le fichier "view\_mustache", qui contient les valeurs utilisées par Mustache. Dans ce fichier, la valeur de page.title est associée à process.env.PAGE\_TITLE, qui est le nom de la variable d'environnement contenant le titre de la page. Si cette variable d'environnement n'est pas définie, une valeur par défaut en anglais peut être utilisée pour éviter de laisser un champ vide dans la page.

Avec Mustache, il est également possible de créer des booléens liés aux variables d'environnement pour la modularité des champs optionnels. Cela signifie que ces champs ne sont pas envoyés au front-end, de sorte que même en examinant la page, on ne peut pas y accéder. Cela peut être utile pour masquer certaines informations sensibles ou rendre des fonctionnalités spécifiques accessibles uniquement à certains utilisateurs.

```
page: {
  /**
   * Language of the page.
   * @type {string}
   */
  language: process.env.PAGE_LANGUAGE || 'en',
  /**
   * Title of the page.
   * @type {string}
   */
  title:
    process.env.PAGE_TITLE ||
    'Report a New Incident to Additi - Funeraire | Best Uptime',
},
```

Le fichier view\_mustache.js

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Recréation du formulaire d'incident avec l'API de Better Uptime

Dans le fichier ` `.envrc` , toutes les valeurs textuelles de la page sont définies à l'aide de variables d'environnement.

```
export PAGE_TITLE="Signaler un nouvel incident à Additi - Funéraire | Meilleur temps de disponibilité"
export PAGE_LANGUAGE="fr"

export MAIN_TITLE="Signaler un nouvel incident à Additi - Funéraire"

export DESCRIPTION_HEADING="Que se passe-t-il ?"
export DESCRIPTION_TEXT="Veuillez décrire ce qu'il se passe, y compris les étapes pour reproduire le problème. Plus vous décrivez, meilleures seront nos chances de résoudre rapidement votre problème."
```

Le fichier .envrc.fr.dist

Il est important de comprendre qu'il existe trois niveaux de sécurité pour gérer les variables d'environnement :

- Niveau 1 : Utilisation simple du fichier ` `.env` .
- Niveau 2 : Utilisation de ` `direnv` , qui permet de valider le contenu du fichier ` `.env` avant son utilisation. De plus, lorsque vous effectuez un ` `cd ..` , cela décharge la liste des variables d'environnement qu'il a ajoutées. Cela peut expliquer pourquoi vous rencontrez des problèmes lorsque vous essayez de les accéder.
- Niveau 3 : Utilisation de Vault d'HashiCorp, qui est un outil complexe à mettre en place. Il permet de stocker les variables d'environnement de manière chiffrée, afin qu'aucun développeur n'ait accès à ces variables en clair. Vous ne pouvez accéder à ces variables chiffrées que si vous disposez des bons droits d'accès.

Dans votre cas, vous utilisez le niveau 2 de sécurité en utilisant ` `direnv` . Cela vous permet de valider le contenu du fichier ` `.env` avant de l'utiliser, ce qui est une bonne pratique pour éviter les erreurs de configuration. De plus, ` `direnv` décharge également les variables d'environnement lors de la navigation vers un autre répertoire, ce qui aide à maintenir un environnement propre et évite les conflits potentiels.

La raison pour laquelle vous avez choisi d'utiliser le niveau 2 de sécurité avec ` `direnv` peut être liée à la simplicité de configuration par rapport au niveau 3 avec Vault d'HashiCorp. Le niveau 3, bien que plus sécurisé, peut être plus complexe à mettre en place et nécessiter des compétences supplémentaires pour la gestion de Vault.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

### Recréation du formulaire d'incident avec l'API de Better Uptime

Pour sécuriser les données envoyées depuis le formulaire vers Better Uptime, vous avez utilisé JSON Schema. JSON Schema est un outil puissant qui permet de définir des schémas de validation pour les données JSON.

Pour créer votre système de validation, vous avez défini un schéma de validation JSON qui décrit la structure et les contraintes des données attendues. Par exemple, vous avez spécifié que le champ "summary" doit être de type "string" et avoir une longueur comprise entre 1 et 100 caractères, car l'API Better Uptime exige une brève description qui ne dépasse pas 100 caractères.

Vous avez également appliqué des contraintes similaires à d'autres champs, tels que le champ "email" qui doit avoir le format d'une adresse email valide. Ces contraintes garantissent que seules les données conformes aux spécifications sont acceptées.

Lorsque le formulaire est soumis, vous utilisez le schéma de validation JSON avec une bibliothèque compatible JSON Schema pour valider les données saisies par l'utilisateur. Cette validation permet de s'assurer que les données respectent les contraintes spécifiées avant de les envoyer à Better Uptime.

En utilisant JSON Schema pour valider les données, vous évitez les problèmes potentiels causés par l'envoi de fausses informations ou de données incorrectes à l'API. Cela contribue à garantir l'intégrité des données et à maintenir un fonctionnement fluide de votre application.

Le code effectue une validation des données de l'incident en les comparant avec un schéma JSON. Si les données ne respectent pas le schéma, une réponse avec le code d'état 400 (Bad Request) est renvoyée, indiquant que les données JSON fournies sont invalides.

```
if (!incidentValidatorResult.valid) {  
    res.writeHead( statusCode: 400);  
    res.write( chunk: 'Invalid JSON data'  
    res.end();  
    return;
```

La if de validation

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES

### REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Lorsque j'ai souhaité déployer mon formulaire sur Google Cloud Platform (GCP), j'ai suivi une série d'étapes simples pour rendre mon application accessible en ligne. En utilisant la ligne de commande (CLI) de GCP, j'ai pu déployer mon formulaire en tant que fonction cloud sur GCP Cloud Functions. Ce processus m'a permis de rendre mon formulaire disponible pour les utilisateurs via une URL sécurisée.

Dans cet environnement cloud, j'ai pu bénéficier de la puissance et de la fiabilité de GCP pour héberger mon application et gérer le trafic entrant. Grâce à l'utilisation de la CLI, j'ai pu automatiser le déploiement avec Gitlab, ce qui m'a fait gagner du temps et simplifié le processus.

Dans les sections suivantes, je vais décrire les étapes que j'ai suivies pour déployer mon formulaire sur GCP Cloud Functions en utilisant la CLI. Ces étapes incluent la configuration de la CLI, le déploiement de la fonction, la vérification du déploiement et la création d'une pipeline Gitlab.

J'ai commencé par la configuration du CLI

Pour commencer, j'ai installé la CLI de Google Cloud Platform (GCP) sur ma machine. J'ai suivi les instructions fournies par GCP pour installer la CLI correspondant à mon système d'exploitation dans mon cas pour MacOS. Une fois la CLI installée, j'ai exécuté la commande `gcloud init` pour effectuer la configuration initiale. Cela m'a permis de me connecter à mon compte GCP et de spécifier les paramètres de configuration tels que la région par défaut et le projet actif, si je n'avais pas eu de projet GCP, j'aurais dû créer un projet GCP en utilisant la commande `gcloud projects create`, fournir un nom et un identifiant uniques du projet.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Après avoir configuré mon projet GCP, j'ai procédé au déploiement de ma fonction sur Google Cloud Functions. J'ai utilisé la commande gcloud functions deploy en spécifiant les paramètres nécessaires pour déployer ma fonction de manière adéquate.

Tout d'abord, j'ai donné un nom significatif à ma fonction en remplaçant <NOM\_DE\_LA\_FONCTION> par le nom que je souhaitais lui attribuer dans mon projet formulaire-better-upptime.

Ensuite, j'ai utilisé l'option --runtime=<RUNTIME> pour spécifier le runtime dans lequel ma fonction s'exécutera. Par exemple, si j'ai utilisé Node.js, j'ai défini --runtime=nodejs18.

J'ai également utilisé l'option --trigger-http pour indiquer que ma fonction sera déclenchée par une requête HTTP. Cela signifie que ma fonction sera accessible via une URL.

Pour spécifier le chemin vers mon code source, j'ai utilisé l'option --source=<RÉPERTOIRE\_SOURCE>. J'ai veillé à ce que le répertoire source contienne tous les fichiers nécessaires à l'exécution de ma fonction, tels que mes fichiers HTML, CSS, JavaScript, etc donc la --source fut . dans le répertoire dans lequel j'ai init le gcloud.

Le point d'entrée de ma fonction a été spécifié à l'aide de l'option --entry-point=<POINT\_D'ENTRÉE>. J'ai indiqué le nom de la fonction ou de la méthode qui sera utilisée comme point d'entrée de mon code.

Enfin, j'ai défini la région dans laquelle ma fonction sera déployée à l'aide de l'option --region=<RÉGION>. J'ai choisi la région la plus appropriée en fonction de mes besoins.

Une fois toutes ces options configurées, j'ai exécuté la commande gcloud functions deploy pour déployer ma fonction sur Google Cloud Functions. Le processus de déploiement a été lancé et j'ai pu suivre son avancement à l'aide des informations affichées dans la console.

Une fois le déploiement terminé, j'ai obtenu l'URL de ma fonction qui me permettait d'accéder à mon formulaire via une requête HTTP.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

```
gcloud functions deploy formulaire-better-uptime \
--gen2 \
--memory=256MB \
--max-instances=10 \
--set-env-vars BETTER_UPTIME_TOKEN=$BETTER_UPTIME_TOKEN \
--runtime=nodejs18 \
--region=europe-west1 \
--source=. \
--entry-point=better_uptime_form_incident \
--trigger-http \
--allow-unauthenticated
```

Commande pour le déploiement de la fonction GCP

Voici tous les options qui ont été configurées pour le déploiement, toujours avec des variables d'environnement pour le token privé.

Après le déploiement de ma fonction, j'ai utilisé la commande gcloud functions describe pour vérifier l'état et les détails de ma fonction. Cela m'a permis de confirmer que ma fonction était correctement déployée et d'obtenir des informations telles que l'URL de déclenchement.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Pour déployer ma fonction à chaque nouveau commit, j'ai mis en place une pipeline GitLab. Cette pipeline me permet d'automatiser les étapes de construction, de test et de déploiement de ma fonction sur Google Cloud Functions.

Pour commencer, j'ai créé un fichier appelé `.gitlab-ci.yml` à la racine de mon projet. Ce fichier contient la configuration de ma pipeline et définit les différentes étapes à exécuter.

Dans ce fichier, j'ai d'abord défini les stages que j'ai utilisés pour ma pipeline. Dans mon cas, j'ai défini les stages suivants : build, test et deploy. Cependant, il existe d'autres stages que vous pouvez utiliser en fonction des besoins de votre projet.

Le stage build est responsable de la construction de mon application. J'y ai inclus toutes les étapes nécessaires pour compiler et préparer mon code source à être déployé. Par exemple, j'ai pu y inclure la compilation des fichiers JavaScript, la gestion des dépendances, etc.

Le stage test est utilisé pour exécuter les tests automatiques sur mon application. Cela me permet de m'assurer que mon code fonctionne correctement avant d'effectuer le déploiement. Dans cette étape, j'ai pu inclure des commandes pour exécuter mes tests unitaires, d'intégration ou de performance.

Enfin, le stage deploy est responsable du déploiement de ma fonction sur Google Cloud Functions. J'y ai inclus les commandes nécessaires pour utiliser la CLI de GCP et déployer ma fonction dans l'environnement cible. J'ai également pu configurer les paramètres de déploiement, tels que le nom de la fonction, le point d'entrée, le chemin vers le code source, etc.

De plus de ces 3 stages il en existe d'autres comme : security, lint release, cleanup, etc.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Dans le fichier `.gitlab-ci.yml`, j'ai défini les variables nécessaires pour mon déploiement. Les variables peuvent contenir des informations sensibles ou des valeurs spécifiques à mon projet. Par exemple, j'ai utilisé la variable `BETTER_UPTIME_TOKEN` pour stocker le jeton d'authentification de l'API Better Uptime et la variable `GCP_CREDENTIALS` pour stocker les informations d'identification de Google Cloud Platform.

```
variables:  
  GCLOUD_PROJECT: molink-373610  
  GCP_CREDENTIALS: $GCLOUD_CREDENTIALS_DEPLOY  
  BETTER_UPTIME_TOKEN: $BETTER_UPTIME_TOKEN  
  SITE_URL: $SITE_URL  
  API_URL_BETTER_UPTIME: $API_URL_BETTER_UPTIME
```

Les variables configurées dans la pipeline

Le stage “build” est responsable de la construction de l’application. Il utilise l’image Docker “node:18” pour exécuter les commandes Node.js nécessaires. Les étapes de script incluent l’installation des dépendances avec la commande “`npm install`” et la génération des fichiers de build avec “`npm run build`”. Ce stage est déclenché uniquement lorsqu’un push est effectué sur la branche “master”.

```
build:  
  stage: build  
  image: node:18  
  script:  
    - npm install  
    - npm run build  
  rules:  
    - if: '$CI_PIPELINE_SOURCE == "push" && $CI_COMMIT_BRANCH == "master"'
```

Le code du stage : build

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES

### REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Le stage “test” est dédié à l’exécution des tests de l’application. Il utilise l’image Docker “cypress/base:latest” et les étapes de script comprennent l’installation de Cypress avec “npm install cypress” et l’initialisation de Cypress avec “npx cypress install”. Avant d’exécuter les tests, l’application est démarrée en arrière-plan avec “npm start”. Ce stage est également exécuté uniquement lorsqu’un push est effectué sur la branche “master”. Les captures d’écran et les vidéos des tests réalisés par Cypress sont enregistrés en tant qu’artefacts.

```
test:
  stage: test
  image: cypress/base:latest
  before_script:
    - npm install cypress
    - npx cypress install
  script:
    - npm start &
    - npm run test
  rules:
    - if: '$CI_COMMIT_BRANCH == "master"'
  artifacts:
    paths:
      - /builds/getnobullshit/open-source/bu-uptime-form/cypress/screenshots/*
      - /builds/getnobullshit/open-source/bu-uptime-form/cypress/videos/*
```

Le code du stage : test

J’ai créé une pipeline schedule qui déclenche le stage “test” du code toutes les 3 heures. Cela permet d’exécuter automatiquement les tests à intervalles réguliers, indépendamment des pushes sur la branche “master”. Cette planification périodique assure que le code est continuellement testé et que les éventuels problèmes sont rapidement identifiés.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Le stage “deploy” est chargé du déploiement de l’application sur Google Cloud Functions. Il utilise l’image Docker “google/cloud-sdk” pour accéder aux outils de déploiement GCP. Les étapes de script comprennent le décodage des informations d’identification GCP à partir de la variable \$GCP\_CREDENTIALS, l’activation du compte de service GCP avec “gcloud auth activate-service-account”, et la configuration du projet GCP avec \$GCLOUD\_PROJECT. Ensuite, la commande “gcloud functions deploy” est utilisée pour déployer la fonction cloud sur GCP, en spécifiant les options appropriées telles que le nom de la fonction, le runtime, la région, le point d’entrée, etc. Ce stage est également déclenché uniquement lorsqu’un push est effectué sur la branche “master”.

```
deploy:
  stage: deploy
  image: google/cloud-sdk
  before_script:
    - echo "$GCP_CREDENTIALS" | base64 -d > /tmp/gcp-creds.json
    - gcloud auth activate-service-account --key-file=/tmp/gcp-creds.json
    - gcloud config set project $GCLOUD_PROJECT
  script:
    - |
      gcloud functions deploy formulaire-better-uptime \
        --gen2 \
        --memory=256MB \
        --max-instances=10 \
        --set-env-vars BETTER_UPTIME_TOKEN=$BETTER_UPTIME_TOKEN \
        --runtime=nodejs18 \
        --region=europe-west1 \
        --source=. \
        --entry-point=better_uptime_form_incident \
        --trigger-http \
        --allow-unauthenticated
  rules:
    - if: '$CI_PIPELINE_SOURCE == "push" && $CI_COMMIT_BRANCH == "master"
```

Le code du stage : deploy

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Grâce à notre processus de déploiement sur Google Cloud Platform (GCP) en utilisant GitLab CI/CD et Google Cloud Functions, nous avons réussi à déployer notre formulaire sur internet de manière efficace. Une fois le déploiement terminé, le formulaire est accessible en ligne, ce qui permet aux utilisateurs d'y accéder et de l'utiliser.

En utilisant Google Cloud Functions, notre formulaire est maintenant hébergé sur le cloud de GCP, ce qui offre une disponibilité élevée et une évolutivité automatique en fonction de la demande. Cela signifie que notre formulaire peut être accessible à tout moment, de n'importe où dans le monde, ce qui facilite l'interaction des utilisateurs avec notre application.

Grâce à l'intégration de GitLab CI/CD dans notre processus de déploiement, nous avons automatisé les étapes clés, ce qui permet des déploiements rapides et cohérents de notre formulaire. Chaque fois qu'un push est effectué sur la branche principale (master), notre pipeline est déclenchée, exécutant les étapes de construction, de test et de déploiement. Cela garantit que les modifications apportées à notre formulaire sont rapidement mises en ligne et accessibles aux utilisateurs.

The screenshot shows the Google Cloud Functions interface. At the top, there's a navigation bar with 'Cloud Functions' and a back arrow labeled 'Informations sur la fonction'. Below the navigation are four buttons: 'MODIFIER', 'SUPPRIMER', 'COPIER', and a question mark icon. The main content area displays the function name 'formulaire-better-uptime', its generation ('2e génération'), deployment time ('Déploiement : 22 juin 2023, 12:14:29'), and URL ('URL: https://europe-west1-molink-373610.cloudfunctions.net/formulaire-better-uptime'). There are also icons for a clipboard and a refresh symbol.

La fonction affichée dans GCP

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

### Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Dans le cadre du déploiement de mon formulaire, j'ai décidé d'utiliser Docker pour faciliter le processus d'exécution de mon application. Docker est une plateforme open-source qui permet de créer, déployer et exécuter des applications dans des conteneurs légers et isolés. Cela garantit une portabilité et une reproductibilité accrues, indépendamment de l'environnement d'exécution.

En utilisant Docker, j'ai pu créer une image contenant toutes les dépendances et configurations nécessaires pour exécuter mon formulaire. Cette image peut être facilement déployée sur différents systèmes, ce qui facilite le déploiement sur divers environnements, tels que des machines locales ou des serveurs cloud.

L'utilisation de Docker présente de nombreux avantages. Tout d'abord, cela permet d'isoler l'application et ses dépendances, évitant ainsi les conflits potentiels avec d'autres applications ou environnements. De plus, Docker simplifie le processus de gestion des dépendances, car toutes les dépendances sont encapsulées dans l'image Docker.

Dans la suite, je vais décrire les étapes pour déployer le formulaire en utilisant Docker, en partant de la construction de l'image Docker jusqu'à l'exécution du conteneur.

Pour commencer, je me suis documenté sur la documentation en ligne de Docker afin de comprendre comment utiliser cette plateforme de déploiement. J'ai ensuite procédé à l'installation de Docker sur mon Mac et j'ai ouvert un terminal pour exécuter les commandes.

Ensuite, je me suis rendu dans le répertoire où se trouve mon fichier Dockerfile. Ce fichier contient les instructions nécessaires pour construire l'image Docker de mon application.

Pour construire l'image Docker, j'ai exécuté la commande suivante dans mon terminal : `docker build -t form:v1.0.0`.

Cette commande construit l'image Docker en se basant sur les instructions spécifiées dans le Dockerfile. L'option `-t` permet de donner un nom et une étiquette à l'image, dans mon cas `form:v1.0.0`.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES

### REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Une fois que l'image est construite, j'ai lancé un conteneur Docker à partir de cette image en utilisant la commande suivante :

```
docker container run -d -p 8080:8080 form:v1.0.0
```

Cette commande démarre un conteneur Docker en arrière-plan (-d pour détaché) et relie le port 8080 du conteneur au port 8080 de mon système hôte (-p pour le port).

Pour vérifier si le conteneur est en cours d'exécution, j'ai exécuté la commande suivante : `docker container ls`

Cette commande affiche la liste des conteneurs actuellement en cours d'exécution sur mon système. Je peux ainsi vérifier si mon conteneur est présent dans la liste.

Si j'ai besoin d'accéder à l'intérieur du conteneur pour exécuter des commandes ou obtenir un shell, j'ai utilisé la commande `docker exec`. Par exemple, pour accéder au shell du conteneur, j'ai exécuté la commande suivante : `docker exec -it <container-id> /bin/bash`

J'ai remplacé `<container-id>` par l'identifiant ou le nom réel du conteneur, ce qui m'a permis d'obtenir un accès au shell du conteneur.

Si je souhaite arrêter le conteneur, j'utilise la commande suivante :

```
docker container stop <container-id>
```

Une fois de plus, j'ai remplacé `<container-id>` par l'identifiant ou le nom réel du conteneur.

Grâce à ces étapes, j'ai pu déployer mon application en utilisant Docker, ce qui présente plusieurs avantages. Mon application est maintenant encapsulée dans un conteneur isolé, ce qui facilite sa portabilité et son déploiement sur différents environnements.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES

### REALISEES

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Il était également prévu de migrer mon application vers Kubernetes (K8s) en utilisant une approche “Infrastructure as Code” (IaC), mais malheureusement, je n'ai pas eu le temps de le réaliser.

Kubernetes est une plateforme de conteneurisation et d'orchestration qui permet de gérer et de déployer des applications de manière scalable et résiliente. En adoptant Kubernetes, j'aurais pu bénéficier d'avantages supplémentaires tels que la gestion automatique des conteneurs, l'équilibrage de charge, la redondance et la mise à l'échelle automatique.

L'utilisation de l'approche “Infrastructure as Code” aurait permis de décrire mon infrastructure Kubernetes sous forme de code, ce qui aurait facilité la gestion, la reproductibilité et le versionnage de mon environnement de déploiement. J'aurais pu utiliser des outils tels que Terraform ou Kubernetes YAML pour définir les ressources, les services et les politiques nécessaires pour exécuter mon application sur Kubernetes.

En adoptant Kubernetes avec une approche “Infrastructure as Code”, j'aurais pu bénéficier d'une gestion plus automatisée de mon infrastructure, d'une meilleure évolutivité et d'une plus grande résilience de mon application.

Malheureusement, en raison de contraintes de temps, je n'ai pas pu réaliser cette migration vers Kubernetes. Cependant, cela reste une perspective intéressante pour améliorer encore davantage le déploiement et la gestion de mon application à l'avenir.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES

### REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Cypress est un outil de test automatisé largement utilisé dans les pipelines de développement et d'intégration continue GitLab. En l'intégrant à votre pipeline GitLab, vous pouvez automatiser l'exécution des tests de votre application web à chaque modification de code.

L'ajout de Cypress à votre pipeline GitLab vous permet de bénéficier de ses fonctionnalités avancées de test automatisé, telles que la simulation des interactions utilisateur, la vérification des éléments de l'interface utilisateur, la gestion des dépendances et bien plus encore. Vous pouvez écrire des scénarios de test complets et les exécuter de manière fiable à chaque étape de votre pipeline GitLab.

L'intégration de Cypress à votre pipeline GitLab garantit que vos tests sont exécutés automatiquement à chaque push sur votre branche de développement, votre branche principale (par exemple, la branche "master") ou toute autre branche spécifique que vous avez définie. Cela permet d'identifier rapidement les problèmes potentiels et de s'assurer que votre application fonctionne correctement avant de la déployer.

De plus, en intégrant Cypress à votre pipeline GitLab, vous pouvez bénéficier des fonctionnalités de génération de rapports et d'artefacts de Cypress. Cela signifie que vous pouvez consulter les résultats des tests, y compris les captures d'écran et les vidéos des tests exécutés, directement dans l'interface GitLab, ce qui facilite la visualisation et l'analyse des résultats de test.

En utilisant Cypress dans ma pipeline GitLab, j'ai mis en place des scénarios de test complets qui simulent les interactions utilisateur et vérifient les éléments de l'interface utilisateur de mon site. Ces tests automatisés me permettent de détecter rapidement les problèmes potentiels tels que des fonctionnalités défectueuses, des erreurs d'affichage ou le langage de la page.

Je vais maintenant vous expliquer en détail comment j'ai mis en place ces tests automatisés avec Cypress et pourquoi ils sont essentiels pour assurer la qualité de mon application.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Pour commencer je vais vous expliquer la différence en les 2 tests disponibles si Cypress :

- Le E2E testing (End-to-End testing), comme son nom l'indique, vise à tester l'application dans son ensemble, en simulant les interactions réelles des utilisateurs avec l'application. Il s'agit d'un type de test qui vérifie le bon fonctionnement de l'application de bout en bout, en testant l'intégration entre ses différentes parties. Les tests E2E sont généralement effectués sur une application déployée, simulant des scénarios d'utilisation réels, tels que la navigation entre les pages, la saisie de données dans des formulaires, l'interaction avec des éléments de l'interface utilisateur, etc. Ces tests vérifient la cohérence de l'application dans son ensemble et sont plus proches de l'expérience utilisateur réelle.
- Le Component testing, quant à lui, se concentre sur le test individuel des composants de l'application. Il s'agit de tester chaque composant de manière isolée, en vérifiant son comportement, sa logique et ses fonctionnalités spécifiques. Les tests de composants sont généralement effectués au niveau du code source, avant le déploiement de l'application. Ils permettent de s'assurer du bon fonctionnement de chaque composant individuellement, en simulant les différentes conditions et en testant les différents cas de figure. Ces tests sont plus ciblés et se concentrent sur la qualité interne des composants.

Pour mes tests j'ai donc utilisé le End-to-End testing pour mes tests qui furent au nombre de 7 tests.

1. Lors du premier test, je simule une saisie de données valide dans les champs du formulaire, puis je soumets le formulaire. Mon objectif est de m'assurer que le formulaire est capable de traiter les données correctement et de renvoyer une réponse appropriée. Je veille à remplir tous les champs obligatoires avec des données valides, conformément aux exigences spécifiées. Ensuite, en soumettant le formulaire. Si le formulaire fonctionne correctement, je peux être confiant quant à sa capacité à collecter et à traiter les informations soumises par les utilisateurs de manière précise et fiable.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

2. Dans le deuxième test, je vais reproduire le même scénario que dans le premier test, mais cette fois-ci, je vais utiliser des valeurs différentes dans les champs du formulaire. L'objectif est de vérifier si le formulaire est capable de gérer correctement différentes combinaisons de données et de générer les résultats attendus en fonction de ces valeurs.
3. Dans le troisième test, je vais me concentrer sur la vérification du comportement du formulaire lorsque des numéros de téléphone invalides sont saisis. L'objectif est de déterminer si le formulaire parvient à détecter les numéros de téléphone incorrects et à afficher un message d'erreur approprié. Dans le fichier app.js, j'ai mis en place une validation du numéro de téléphone à l'aide d'une expression régulière (regex). Cette expression régulière permet de définir un modèle auquel le numéro de téléphone doit correspondre pour être considéré comme valide et si cela n'est pas le cas d'afficher un message d'erreur.
4. Dans le test numéro quatre, nous vérifions si le formulaire réagit correctement lorsqu'un e-mail invalide est soumis. L'objectif est de s'assurer que le formulaire effectue une validation appropriée sur le champ de l'e-mail et réagit en conséquence.
5. Dans le test numéro cinq, nous vérifions comment le formulaire réagit lorsqu'une brève description avec plus de 100 caractères est soumise. L'objectif est de s'assurer que le formulaire effectue une validation appropriée sur le champ de la brève description et réagit en conséquence.  
Dans le fichier app.js, il y a une vérification de la longueur de la brève description. Cette vérification vise à s'assurer que la longueur de la brève description ne dépasse pas une limite spécifiée, qui est de 100 caractères.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

6. Le test numéro six est un scénario automatisé qui vise à vérifier la validation des données du formulaire en effectuant une requête POST vers une API spécifiée.

Dans ce test, nous utilisons l'URL de l'API Better Uptime et le jeton d'authentification BETTER\_UPTIME\_TOKEN pour effectuer des requêtes POST et GET toujours avec des variables d'environnement.

Nous créons un objet "incident" qui représente les données soumises via le formulaire. Cet objet contient des informations telles que le résumé, la description, l'e-mail du demandeur et d'autres paramètres.

Ensuite, nous envoyons une requête POST à l'API Better Uptime en incluant les données de l'incident dans le corps de la requête. Nous ajoutons également les en-têtes requis, y compris le type de contenu et l'autorisation. Nous vérifions ensuite la réponse de la requête pour s'assurer que le statut de la réponse est 201, indiquant que la requête a été traitée avec succès.

Après cela, nous effectuons une requête GET pour récupérer les données de l'API Better Uptime. Nous extrayons les informations pertinentes de la réponse, telles que l'ID de l'incident, la cause, l'e-mail, les SMS, etc. Ensuite, nous effectuons plusieurs assertions pour vérifier que les données récupérées correspondent aux données que nous avons soumises via le formulaire. Par exemple, nous vérifions que la cause de l'incident correspond au résumé de l'incident, que l'e-mail, les SMS, les notifications push et les appels sont correctement enregistrés.

Ces assertions nous permettent de confirmer que les données soumises via le formulaire sont correctement enregistrées dans l'API Better Uptime, ce qui démontre le bon fonctionnement de la validation des données du formulaire.

Ce test automatisé est essentiel car il nous permet de vérifier rapidement et de manière fiable si les données soumises via le formulaire sont correctement traitées et enregistrées. Il nous aide à détecter d'éventuels problèmes de validation ou de traitement des données, garantissant ainsi la qualité et l'intégrité des données dans notre application.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Le test “Test Language” est un scénario automatisé qui vise à vérifier la langue et le texte affichés sur le site. Il s’agit d’un test d’interface utilisateur qui utilise Cypress pour effectuer des assertions sur le contenu de la page.

Dans ce test, nous commençons par récupérer l’URL du site à tester à partir de la variable d’environnement SITE\_URL. Ensuite, nous utilisons la commande cy.visit(SITE\_URL) pour charger la page dans le navigateur Cypress.

Une fois la page chargée, nous utilisons la commande cy.get(‘html’) pour sélectionner l’élément HTML de la page. Ensuite, nous utilisons la méthode .then() pour accéder à la valeur de l’attribut lang de cet élément HTML.

En fonction de la valeur de l’attribut lang, nous effectuons des assertions spécifiques. Si la langue est définie sur “en” (anglais), nous vérifions que le texte “Create a New Incident” existe sur la page. Si la langue est définie sur “fr” (français), nous vérifions que le texte “Créer un nouvel incident” existe sur la page.

Si la langue n’est ni “en” ni “fr”, nous affichons un message d’erreur dans la console indiquant que la langue n’est pas prise en charge.

Ce test automatisé est utile pour s’assurer que le bon contenu linguistique est affiché sur le site.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Pour faire les 5 premiers tests j'ai factoriser avec un beforeEach pour la création du form qui sera utilisé dans les tests.

```
5   form = {
6     brief_description: {
7       value: '',
8       selector: '#brief_description',
9       action: 'type',
10    },
11    detailed_description: {
12      value: '',
13      selector: '#detailed_description',
14      action: 'type',
15    },
16    email: {
17      value: '',
18      selector: '#requester_email',
19      action: 'type',
20    },
21    phone_number: {
22      value: '',
23      selector: '#phone_number',
24      action: 'type',
25    },
26    checkbox: {
27      selector: ':nth-child(3) > .form__body-checkbox__content',
28      action: 'click',
29    },
30    submitButton: {
31      selector: '.form__submit-button',
32      action: 'click',
33    },
34  };
35};

36
```

Le form et comment il est construit

La fonction beforeEach() est utilisée pour initialiser la variable "form" avant chaque test. Cette variable est un objet contenant les différentes informations sur les champs du formulaire à remplir. Voici une explication détaillée de chaque propriété de l'objet "form" :

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

**brief\_description** : C'est un objet qui représente le champ de description brève. La propriété "value" est initialement vide et sera remplie avec une valeur spécifique lors de l'exécution du test. La propriété "selector" contient le sélecteur CSS utilisé pour cibler ce champ dans la page HTML, et la propriété "action" spécifie l'action à effectuer sur le champ, qui est "type" dans ce cas.

**detailed\_description** : C'est un objet qui représente le champ de description détaillée. Les propriétés "value", "selector" et "action" sont similaires à celles du champ de description brève.

**email** : C'est un objet qui représente le champ d'adresse e-mail. Les propriétés "value", "selector" et "action" sont similaires à celles des champs de description.

**phone\_number** : C'est un objet qui représente le champ de numéro de téléphone. Les propriétés "value", "selector" et "action" sont similaires à celles des champs précédents.

**checkbox** : C'est un objet qui représente une case à cocher sur le formulaire. La propriété "selector" contient le sélecteur CSS utilisé pour cibler cette case à cocher dans la page HTML, et la propriété "action" spécifie l'action à effectuer sur la case à cocher, qui est "click" dans ce cas.

**submitButton** : C'est un objet qui représente le bouton de soumission du formulaire. La propriété "selector" contient le sélecteur CSS utilisé pour cibler ce bouton dans la page HTML, et la propriété "action" spécifie l'action à effectuer sur le bouton, qui est "click" dans ce cas.

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Dans chaque test, comme dit juste avant j'ai configuré la structure de données form pour définir les valeurs des champs du formulaire. Par exemple, dans le premier test, je définis les valeurs suivantes :

```
// Set form field values
form.brief_description.value = 'Brief description';
form.detailed_description.value = 'Detailed description';
form.email.value = 'matheodelaunay04@gmail.com';
form.phone_number.value = '0782199754';

for (const field of Object.values(form)) {
  cy.get(field.selector)[field.action](field.value);
}
```

Valeur du test 1 (avec des valeurs qui fonctionne)

Dans chaque test suivant, je peux modifier ces valeurs spécifiques pour effectuer des tests avec différentes configurations. Par exemple, dans le deuxième test, je peux changer les valeurs pour tester des scénarios différents. Cela me permet d'avoir une flexibilité dans mes tests en ajustant facilement les données du formulaire pour couvrir différents cas de test, sans avoir à répéter le code de sélection des éléments du formulaire. Comme par exemple le test avec un numéro de téléphone invalide où j'ai remplacé le champ téléphone :

```
// Set form field values
form.brief_description.value = 'With an invalid phone value';
form.detailed_description.value = 'Different detailed description';
form.email.value = 'contact@seonay.eu';
form.phone_number.value = '1234567890';

for (const field of Object.values(form)) {
  cy.get(field.selector)[field.action](field.value);
}
```

Valeur du test 3 (avec la valeur téléphonée invalide)

# 6 - REALISATION DES MISSIONS

## 6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

Dans les tests qui vérifient les erreurs, j'utilise des assertions pour vérifier si les messages d'erreur appropriés s'affichent correctement sur la page. Par exemple, dans le test "Does not pass with tel error", je configure les valeurs du formulaire de manière à provoquer une erreur de numéro de téléphone invalide. Ensuite, j'utilise la commande cy.get pour sélectionner l'élément HTML qui affiche le message d'erreur correspondant au numéro de téléphone.

Ensuite, j'utilise l'assertion .should('be.visible') pour vérifier si l'élément est visible, ce qui indique que le message d'erreur est correctement affiché. Si l'élément n'est pas visible, l'assertion échoue et le test indique qu'il y a un problème avec la validation du numéro de téléphone.

```
cy.get('#phone_number_alert').should(chainer: 'be.visible');
```

Les tests Cypress jouent un rôle clé dans mon processus de développement, me permettant de gagner en confiance dans mon application et d'assurer une meilleure expérience utilisateur en détectant et en corrigeant rapidement les erreurs. Grâce à ces tests automatisés, je peux valider la fonctionnalité de mon formulaire, vérifier la saisie des données et m'assurer que les messages d'erreur s'affichent correctement lorsque des valeurs invalides sont soumises.

En intégrant ces tests dans mon pipeline GitLab, je peux exécuter mes tests à chaque modification de code et obtenir un retour rapide sur la qualité de mon application. Cela me permet d'identifier et de résoudre les problèmes rapidement, ce qui contribue à améliorer la stabilité de mon application et à offrir une meilleure expérience utilisateur.

# **6 - REALISATION DES MISSIONS**

## **6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES**

### **Tests automatisés avec Cypress et analyse de sécurité avec Trivy**

La sécurité est un aspect crucial dans le développement et le déploiement des applications. Les vulnérabilités dans les logiciels peuvent être exploitées par des attaquants pour compromettre la sécurité d'un système. C'est pourquoi il est essentiel d'identifier et de remédier rapidement à ces vulnérabilités.

Trivy est un outil open source de scan de vulnérabilités spécialement conçu pour les conteneurs et les images de conteneurs. Il permet de détecter les vulnérabilités connues dans les bibliothèques et les dépendances utilisées par les images de conteneurs, ce qui aide les développeurs et les équipes DevOps à maintenir un niveau de sécurité élevé dans leurs environnements de conteneurs.

En utilisant Trivy, vous pouvez automatiser le processus de détection des vulnérabilités dans vos images de conteneurs, en les analysant à la recherche de failles connues dans les bases de données de vulnérabilités telles que CVE (Common Vulnerabilities and Exposures) et d'autres sources. Trivy prend en charge différents formats d'image tels que Docker, OCI, et bien d'autres.

Trivy fournit des rapports détaillés sur les vulnérabilités détectées, y compris des informations sur les CVE correspondantes, la gravité des vulnérabilités, les correctifs disponibles et les références pour en savoir plus. Cela permet aux développeurs de prendre des mesures correctives appropriées pour remédier aux vulnérabilités identifiées.

En intégrant Trivy dans votre pipeline de développement et de déploiement, vous pouvez effectuer des scans de vulnérabilités de manière régulière et automatisée, vous assurant ainsi que vos images de conteneurs sont exemptes de vulnérabilités connues avant leur déploiement. Cela renforce la sécurité de votre infrastructure de conteneurs et réduit les risques d'exploitation des vulnérabilités.

# **6 - REALISATION DES MISSIONS**

## **6.1 - TACHES EFFECTUEES ET ACTIVITES REALISEES**

### **Tests automatisés avec Cypress et analyse de sécurité avec Trivy**

Pour scanner les vulnérabilités avec Trivy sur macOS, vous pouvez suivre les étapes suivantes :

1. Installation de Trivy : Utilisez Homebrew, un gestionnaire de paquets pour macOS, pour installer Trivy. Ouvrez un terminal et exécutez la commande suivante : brew install trivy
2. Préparation de l'image de conteneur : Assurez-vous d'avoir une image de conteneur disponible localement ou dans un registre accessible. Vous pouvez récupérer une image à partir d'un registre public ou la construire localement à l'aide d'un fichier Dockerfile.
3. Exécution du scan de vulnérabilités : Une fois l'image de conteneur prête, vous pouvez lancer le scan de vulnérabilités en utilisant la commande Trivy.

Exécutez la commande suivante dans le terminal : trivy image <nom\_image>

Remplacez `<nom\_image>` par le nom de votre image de conteneur.

4. Analyse des résultats du scan : Trivy analysera l'image de conteneur et affichera les résultats du scan dans le terminal. Vous pourrez voir les vulnérabilités détectées, leurs CVE correspondantes, leur gravité et les correctifs disponibles.

Pour une meilleure lisibilité et pour faciliter le partage des résultats du scan de vulnérabilités, vous pouvez rediriger la sortie de la commande Trivy vers un fichier texte. Cela vous permettra de conserver une trace des résultats ou de les partager avec d'autres membres de votre équipe.

Pour ce faire, vous pouvez utiliser l'opérateur de redirection > dans le terminal pour écrire la sortie dans un fichier texte. Voici comment vous pouvez le faire : trivy image <nom\_image> > resultat\_scan.txt

# 6 - REALISATION DES MISSIONS

## 6.2 - RESULTATS OBTENUS

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

La recréation du formulaire d'incident avec l'API de Better Uptime a été réalisée avec succès. Ce formulaire permet aux utilisateurs de saisir les détails d'un incident, tels que la brève description, la description détaillée, l'e-mail du demandeur, le numéro de téléphone et les options de notification.

L'intégration de l'API de Better Uptime permet de soumettre les données du formulaire pour un suivi ultérieur. Lorsque le formulaire est soumis, les données sont envoyées à l'API via une requête POST, en respectant les paramètres requis tels que l'URL de l'API et les en-têtes d'authentification.

Cette recréation du formulaire d'incident offre une interface utilisateur conviviale et réactive. Les utilisateurs peuvent facilement remplir les champs requis et choisir les options de notification qui leur conviennent.

En utilisant l'API de Better Uptime, vous pouvez collecter les incidents signalés par les utilisateurs, les stocker dans une base de données et les traiter ultérieurement. Cela facilite la gestion des incidents et permet une communication plus efficace entre les utilisateurs et les équipes chargées de la résolution des problèmes.

L'intégration de l'API de Better Uptime dans le formulaire d'incident améliore la transparence, la traçabilité et l'efficacité du processus de gestion des incidents.

Le formulaire hébergé sur la Cloud fonction GCP

# 6 - REALISATION DES MISSIONS

## 6.2 - RESULTATS OBTENUS

Déploiement du formulaire sur Google Cloud Platform (GCP) et Docker

Le déploiement du formulaire sur Google Cloud Platform (GCP) en utilisant Docker et Cloud Functions a été réalisé avec succès. Cette approche offre plusieurs avantages en termes de simplicité, d'évolutivité et de gestion des ressources.

J'ai utilisé Docker pour encapsuler le formulaire dans un conteneur, ce qui permet une meilleure isolation et une gestion simplifiée des dépendances. Cette encapsulation facilite également le déploiement du formulaire sur différents environnements, sans se soucier des différences de configuration.

Pour le déploiement sur GCP, j'ai choisi d'utiliser Cloud Functions, qui est un service de calcul sans serveur. Les Cloud Functions permettent d'exécuter le code du formulaire en réponse à des événements spécifiques, tels que les requêtes HTTP. Cela permet d'avoir une approche légère et sans gestion d'infrastructure.

Le déploiement du formulaire en tant que Cloud Function offre une scalabilité automatique, où les ressources sont allouées dynamiquement en fonction de la charge de travail. Cela garantit une réponse rapide et fiable, même en cas de fluctuations de trafic.

L'utilisation de Cloud Functions simplifie également la gestion des ressources, car la plateforme prend en charge automatiquement la mise à l'échelle, la surveillance et la gestion des ressources nécessaires à l'exécution du formulaire.

The screenshot shows the Google Cloud Functions interface in the Google Cloud console. The page title is "Fonctions - Cloud Functions". The URL is "console.cloud.google.com/functions/list?referrer=search&hl=fr&project=molink-373610". The search bar contains "fun". The main area displays a table of functions:

Environnement	Nom	Dernier déploiement	Région	Recommandation	Déclencheur	Exécution	Mémoire allouée	Fonction exécutée	Actions
2nd gen	formulaire-better-uptime	16 mai 2023, 18:09:08	us-central1		HTTP	Node.js 18	256 Mo	helloHttp	⋮
2nd gen	formulaire-better-uptime	22 juin 2023, 12:14:29	europe-west1		HTTP	Node.js 18	256 Mo	better_uptime_form_incident	⋮

Les Cloud Functions sur GCP

# 6 - REALISATION DES MISSIONS

## 6.2 - RESULTATS OBTENUS

Tests automatisés avec Cypress et analyse de sécurité avec Trivy

L'intégration des tests automatisés avec Cypress et l'analyse de sécurité avec Trivy a produit des résultats prometteurs.

Les tests automatisés avec Cypress ont permis de vérifier le bon fonctionnement du formulaire d'incident. Différents scénarios ont été couverts, tels que la soumission du formulaire avec des données valides, des données incorrectes et des champs vides. Les assertions ont été utilisées pour vérifier que les réponses renvoyées étaient conformes aux attentes. Ces tests ont aidé à identifier d'éventuels problèmes et à améliorer la qualité globale du formulaire.

L'analyse de sécurité avec Trivy a permis de scanner l'image Docker utilisée dans le déploiement de l'application. Trivy a identifié les vulnérabilités potentielles présentes dans l'image, en fournissant des informations détaillées sur les CVE (Common Vulnerabilities and Exposures) associées. Cela a permis de prendre des mesures correctives pour remédier à ces vulnérabilités et garantir un niveau de sécurité plus élevé.

En combinant les tests automatisés avec Cypress et l'analyse de sécurité avec Trivy, vous avez renforcé la robustesse de votre application. Les tests automatisés garantissent le bon fonctionnement du formulaire et permettent de détecter rapidement d'éventuels problèmes. L'analyse de sécurité avec Trivy garantit la détection proactive des vulnérabilités dans l'image Docker, réduisant ainsi les risques potentiels.

Ces pratiques d'automatisation des tests et d'analyse de sécurité contribuent à améliorer la qualité, la fiabilité et la sécurité de votre application. Elles permettent également de gagner du temps et des efforts en détectant les problèmes plus tôt dans le processus de développement, ce qui facilite la correction et la prévention des éventuelles vulnérabilités.



The screenshot shows a terminal window with the following output:

```
Running: spec.cy.js (1 of 1)

Cypress Tests: Form Submission
✓ Passes http://localhost:8080 (1976ms)
✓ Passes with different values (1799ms)
✓ Does not pass with tel error (1776ms)
✓ Does not pass with mail error (1794ms)
✓ Does not pass with short description error (2967ms)

Cypress Tests: Form Data Validation
✓ Verifies the submitted data via POST request (714ms)

Test Language
✓ Verifies the language and text (92ms)

7 passing (11s)

(Results)

Tests:    7
Passing:  7
Failing:  0
Pending:  0
Skipped: 0
Screenshots: 0
Video:   true
Duration: 11 seconds
Spec Ran: spec.cy.js
```

# **6 - REALISATION DES MISSIONS**

## **6.3 - DIFFICULTES RENCONTREES ET SOLUTION APORTEES**

Pendant ce stage, j'ai rencontré plusieurs difficultés liées aux technologies que je ne connaissais pas encore. J'ai dû explorer de nouveaux concepts, me documenter, tester, apprendre et faire des erreurs pour progresser et mieux comprendre ces technologies.

J'ai également fait face à des défis liés au télétravail. Au début, m'adapter à travailler à distance a été un problème, mais j'ai réussi à trouver des solutions en changeant de lieu de travail et en établissant une routine pour rester productif.

La gestion du temps a également été une difficulté. J'ai constaté que je perdais beaucoup de temps à résoudre de petits problèmes au fil des semaines, ce qui m'a empêché de progresser aussi rapidement que je l'aurais souhaité. Cela m'a poussé à être plus conscient de la gestion de mon temps et à trouver des moyens d'optimiser mon travail.

Un autre défi auquel j'ai été confronté était la mise en place des pipelines, des variables d'environnement et de Cypress. Comprendre comment ces éléments interagissent et les configurer correctement a nécessité des efforts supplémentaires de ma part, mais j'ai pu résoudre ces problèmes en recherchant des ressources et en sollicitant l'aide de mes collègues.

En plus de ces difficultés, j'ai également rencontré de nombreux petits problèmes tout au long de mon stage. Bien que je ne puisse pas les énumérer tous ici, ils ont tous contribué à mon apprentissage et à ma croissance professionnelle.

Dans l'ensemble, malgré ces difficultés, j'ai pu surmonter les obstacles grâce à ma détermination, ma curiosité et ma capacité à apprendre de mes erreurs. Chaque défi rencontré m'a permis de développer de nouvelles compétences et de devenir plus autonome dans mon travail.

# **7 - EVALUATION DES REALISATIONS ET DES COMPETENCE MOBILISEES**

## **7.1 - ANALYSE DES RESULTATS PAR RAPPORT AUX OBJECTIFS FIXES**

Lors de mon stage, je me suis engagé à donner le meilleur de moi-même, à apprendre autant que possible et à ne pas décevoir mes responsables et mes collègues. Je n'avais pas fixé d'objectifs spécifiques en termes de résultats tangibles, mais plutôt une approche axée sur l'apprentissage et l'expérience.

Dans cette optique, j'ai été très satisfait des résultats que j'ai obtenus. J'ai eu l'occasion de travailler sur des projets concrets, d'appliquer mes connaissances techniques et de relever des défis stimulants. J'ai pu acquérir de nouvelles compétences, explorer des technologies que je ne connaissais pas auparavant et développer ma capacité à résoudre des problèmes.

La collaboration avec mon équipe a été un aspect clé de mon stage. J'ai pu travailler aux côtés de développeurs expérimentés, qui m'ont guidé et soutenu dans mon apprentissage. J'ai également appris à travailler en équipe, à communiquer efficacement et à contribuer de manière significative aux projets.

Cependant, malgré ma satisfaction quant aux résultats obtenus, je ressens également une certaine tristesse à l'idée que mon stage se termine. Je suis conscient qu'il y a encore tant de choses à apprendre et à explorer dans le domaine du développement logiciel. Si j'avais eu la possibilité de prolonger mon stage, j'aurais saisi cette opportunité avec enthousiasme pour continuer à développer mes compétences et ma compréhension.

Néanmoins, je suis reconnaissant de l'opportunité qui m'a été offerte et de la confiance accordée en mes capacités. Ce stage m'a permis de grandir professionnellement et personnellement, et je suis confiant dans le fait que les compétences et les expériences acquises pendant cette période me serviront dans mes futurs projets et dans ma carrière.

En fin de compte, mon objectif principal était de faire de mon mieux et d'apprendre le plus possible, et je pense avoir atteint cet objectif avec succès. Je suis ravi des résultats obtenus et je suis impatient de continuer à développer mes compétences et à progresser dans le domaine du développement logiciel.

# **7 - EVALUATION DES REALISATIONS ET DES COMPETENCE MOBILISEES**

## **7.2 - EVALUATION DES COMPETENCES**

### **TECHNIQUES ACQUISES OU MOBILISEES**

Au cours de mon stage, j'ai eu l'opportunité de mobiliser et d'acquérir diverses compétences techniques qui ont été essentielles pour mener à bien mes missions et contribuer aux projets de l'entreprise.

Tout d'abord, j'ai renforcé mes compétences en développement web en travaillant avec des technologies telles que HTML, CSS, JavaScript, et SCSS. J'ai pu mettre en pratique mes connaissances en utilisant Node.js pour développer des applications côté serveur. J'ai également utilisé des outils et des plugins tels que ESLint, Husky, Docker, et les variables d'environnement pour améliorer la qualité du code et faciliter le déploiement des applications.

En ce qui concerne les tests automatisés, j'ai utilisé Cypress pour écrire des tests fonctionnels et m'assurer du bon fonctionnement de mes applications.

Dans la gestion de projet, j'ai utilisé une approche de rapport quotidien pour suivre mes progrès, partager mes réalisations et discuter des défis rencontrés. J'ai utilisé des outils de gestion de versionnement comme Git pour collaborer avec d'autres développeurs et gérer efficacement les branches et les conflits de code.

Enfin, j'ai également acquis des compétences en gestion des environnements de développement, en utilisant des outils tels que Docker pour créer des environnements reproductibles et faciliter le déploiement de mes applications.

Dans l'ensemble, mon stage m'a permis de mobiliser et d'acquérir un large éventail de compétences techniques, notamment en développement web avec HTML, CSS, JavaScript et SCSS, en tests automatisés avec Cypress, en gestion de projet avec des rapports quotidiens et des outils de versionnement, et en gestion des environnements de développement avec Docker. Ces compétences seront précieuses pour ma carrière professionnelle future, me permettant d'aborder de nouveaux défis et de continuer à évoluer en tant que développeur compétent.

# **7 - EVALUATION DES REALISATIONS ET DES COMPETENCE MOBILISEES**

## **7.3 - EVALUATION DES COMPETENCES**

### **TECHNIQUES ACQUISES OU MOBILISEES**

Au-delà des compétences techniques, mon stage m'a également permis de développer et de renforcer des compétences transversales qui sont essentielles dans le monde professionnel. Voici quelques réflexions sur ces compétences :

**Autonomie** : Pendant mon stage, j'ai dû prendre des initiatives, travailler de manière indépendante et prendre des décisions éclairées. J'ai été capable de gérer mes propres tâches, de fixer des objectifs et de les atteindre avec peu de supervision. Cela m'a permis de développer ma capacité à travailler de manière autonome et à faire preuve d'initiative.

**Résolution de problèmes** : Tout au long de mon stage, j'ai été confronté à des défis techniques et j'ai dû trouver des solutions efficaces pour les surmonter. J'ai appris à analyser les problèmes, à rechercher des solutions, à expérimenter différentes approches et à prendre des décisions basées sur des informations limitées. Cette expérience m'a permis d'améliorer mes compétences en résolution de problèmes et ma capacité à trouver des solutions créatives.

**Collaboration** : Bien que j'aie travaillé de manière indépendante sur la plupart de mes tâches, j'ai également eu l'occasion de collaborer avec d'autres membres de l'équipe. J'ai participé à des réunions, partagé mes idées, échangé des connaissances et travaillé en étroite collaboration sur certains projets. Cela m'a permis de développer mes compétences en communication, en écoute active et en travail d'équipe.

**Gestion du temps** : Pendant mon stage, j'ai été confronté à des délais serrés et à plusieurs tâches à accomplir simultanément. J'ai dû gérer efficacement mon temps, établir des priorités, planifier mes activités et respecter les échéances. Cette expérience m'a permis d'améliorer mes compétences en gestion du temps et ma capacité à travailler de manière productive sous pression.

**Adaptabilité** : Le domaine de la technologie est en constante évolution, et pendant mon stage, j'ai dû m'adapter à de nouvelles technologies, à de nouvelles méthodes de travail et à de nouveaux défis. J'ai développé ma capacité à m'adapter rapidement aux changements, à apprendre de nouvelles compétences et à m'adapter à de nouveaux environnements.

# 8 - CONCLUSION

## 8.1 - BILAN DU STAGE

Mon stage a été une expérience extrêmement enrichissante sur de nombreux aspects. J'ai pu acquérir de nouvelles compétences techniques, développer mes connaissances dans des domaines spécifiques et renforcer mes compétences transversales. Dans l'ensemble, le bilan de mon stage est très positif.

Tout d'abord, j'ai eu l'opportunité de travailler avec des technologies et des outils modernes tels que Node.js, HTML, CSS, JavaScript, SCSS et des plugins tels que ESLint, Husky et Docker. Ces expériences m'ont permis de consolider mes compétences en développement web et de me familiariser avec les bonnes pratiques et les outils utilisés dans l'industrie.

Ensuite, j'ai été exposé à un environnement de travail professionnel, ce qui m'a donné une perspective précieuse sur la réalité du monde du travail. J'ai appris à travailler en équipe, à collaborer avec mes collègues et à participer à des réunions. J'ai également développé ma capacité à gérer mon temps de manière autonome et à respecter les délais.

Une autre facette positive de mon stage a été l'opportunité de travailler sur des projets concrets et de voir mes contributions prendre vie. J'ai pu participer à la création d'un formulaire d'incident, mettre en place des tests automatisés avec Cypress et effectuer des analyses de sécurité avec Trivy. Ces réalisations m'ont permis de constater l'impact de mon travail et de voir comment mes compétences techniques peuvent être appliquées dans un contexte réel.

Enfin, j'ai eu la chance de travailler avec une équipe dynamique et bienveillante qui m'a soutenu tout au long de mon stage. J'ai pu poser des questions, recevoir des conseils et bénéficier de l'expérience de mes collègues. Cette atmosphère de travail positive a contribué à mon développement personnel et professionnel.

Dans l'ensemble, mon stage a été une expérience très positive. J'ai acquis de nouvelles compétences, consolidé mes connaissances techniques et développé mes compétences transversales. Je suis reconnaissant d'avoir eu cette opportunité et je suis confiant que les apprentissages et l'expérience que j'ai acquises pendant mon stage seront extrêmement bénéfiques pour ma future carrière.

# 8 - CONCLUSION

## 8.2 - RETOUR D'EXPERIENCE PERSONNEL

Mon retour d'expérience personnel de ce stage a été extrêmement positif. J'ai eu la chance de travailler dans un environnement stimulant et d'apprendre auprès d'une équipe talentueuse et bienveillante.

Tout d'abord, j'ai pu développer mes compétences techniques en me familiarisant avec de nouvelles technologies et en les appliquant dans des projets concrets. J'ai appris à utiliser des langages de programmation tels que Node.js, HTML, CSS, JavaScript, et j'ai également exploré des outils et des frameworks pertinents tels que SCSS, ESLint, Husky et Docker. Cela m'a permis d'élargir mes compétences et de gagner en confiance dans mes capacités de développement.

En outre, j'ai eu l'occasion de travailler sur des tâches variées et intéressantes, ce qui m'a permis d'acquérir une expérience pratique précieuse. J'ai participé à la création d'un formulaire d'incident, ce qui m'a permis de mettre en pratique mes compétences en conception et en développement. J'ai également été impliqué dans la mise en place de tests automatisés avec Cypress, ce qui m'a donné une compréhension approfondie des pratiques de test et de la qualité logicielle.

Une autre facette importante de mon expérience a été la collaboration avec l'équipe. J'ai pu travailler en étroite collaboration avec des collègues expérimentés, ce qui m'a permis d'apprendre de leurs connaissances et de bénéficier de leurs conseils. J'ai également apprécié l'ambiance de travail positive et le soutien mutuel au sein de l'équipe.

En termes de développement personnel, ce stage m'a permis de renforcer des compétences transversales telles que la gestion du temps, la résolution de problèmes et la communication. J'ai dû gérer plusieurs tâches simultanément, respecter les délais et communiquer efficacement avec les membres de l'équipe. Ces compétences seront précieuses dans ma future carrière professionnelle.

De plus, je suis conscient qu'en tant que stagiaire, je suis encore en phase d'apprentissage et de développement professionnel, mais je souhaitais être le plus proche de la perfection afin de remercier et de ne pas décevoir mes supérieurs qui ont eu la gentillesse de me prendre en stage. Bien que j'aie fait de grands progrès pendant ce stage, je suis également conscient qu'il me reste encore beaucoup à apprendre et à améliorer.

# **8 - CONCLUSION**

## **8.2 - RETOUR D'EXPERIENCE PERSONNEL**

J'ai réalisé que le monde professionnel est en constante évolution, et qu'il est important d'être adaptable et ouvert à l'apprentissage continu. J'ai compris qu'il est essentiel de rester curieux et de chercher activement de nouvelles connaissances et compétences pour rester pertinent dans l'industrie.

De plus, j'ai également réalisé l'importance de développer des compétences non techniques telles que la gestion du stress, la résolution de problèmes et la communication efficace. Ces compétences sont essentielles pour réussir dans un environnement professionnel et je suis déterminé à les développer davantage.

De plus, pour cette année à venir, je suis déterminé à continuer à m'investir pleinement dans mon développement professionnel. Je vais mettre à profit toutes les technologies, compétences et connaissances que j'ai acquises pendant ce stage pour continuer à progresser.

Je suis motivé à explorer de nouvelles opportunités, à relever de nouveaux défis et à continuer à élargir mes compétences techniques. Je suis prêt à investir du temps et des efforts supplémentaires pour approfondir mes connaissances dans les domaines qui m'intéressent et pour rester à jour avec les dernières tendances de l'industrie.

Je suis reconnaissant envers l'équipe qui m'a encadré pendant ce stage et je tiens à exprimer ma gratitude en continuant à me donner à fond dans mes futurs projets. Je suis convaincu que cette expérience m'a préparé de manière significative pour mes prochaines étapes professionnelles et je suis impatient de mettre en pratique tout ce que j'ai appris.

En somme, ce stage a été une expérience extrêmement enrichissante et je suis déterminé à poursuivre ma croissance professionnelle avec passion et détermination. Je suis prêt à relever de nouveaux défis, à continuer à apprendre et à évoluer dans le monde professionnel passionnant qui m'attend.

# **8 - CONCLUSION**

## **8.2 - PERSPECTIVES D'EVOLUTION ET D'AMELIORATION**

Les perspectives d'évolution et d'amélioration pour moi à la suite de ce stage sont nombreuses. Tout d'abord, je souhaite continuer à approfondir mes connaissances techniques dans les domaines que j'ai explorés pendant ce stage, tels que le développement web, les tests automatisés et la sécurité. Je vais continuer à me former et à explorer de nouvelles technologies et outils pour rester à jour avec les dernières avancées de l'industrie.

En ce qui concerne mes compétences transversales, je souhaite renforcer ma capacité à gérer mon temps de manière plus efficace, à résoudre les problèmes de manière autonome et à améliorer ma communication interpersonnelle. Je vais chercher des opportunités de collaboration et de leadership au sein d'équipes de projet pour développer ces compétences.

En outre, je suis conscient de l'importance de développer ma pensée critique et ma capacité à prendre des décisions éclairées. Je vais chercher des projets plus complexes et plus stratégiques pour affiner ces compétences et prendre des responsabilités plus importantes.

Enfin, je suis déterminé à poursuivre mon apprentissage et à rester ouvert aux nouvelles opportunités d'apprentissage et de croissance. Je vais chercher des projets innovants et stimulants qui me permettront de continuer à développer mes compétences et à repousser mes limites.

De plus je suis conscient que je dois m'améliorer grandement dans deux domaines spécifiques : la communication en télétravail et la qualité de mes écrits. En travaillant à distance, j'ai réalisé que je dois être plus attentif à la clarté de mes messages et à la façon dont je m'exprime. De plus, je vais veiller à améliorer la qualité de mes écrits en les relisant attentivement et en les révisant pour éviter les erreurs et les formulations ambiguës. Je considère ces deux aspects comme des opportunités d'apprentissage et je suis déterminé à faire les efforts nécessaires pour m'améliorer.

Dans l'ensemble, ce stage a été une étape importante dans mon parcours professionnel, et j'ai acquis de précieuses compétences et expériences. Je suis prêt à relever de nouveaux défis et à continuer à progresser dans ma carrière en tirant parti des leçons apprises pendant ce stage.