

Projet Professionnel Dévops

Sayfoutdinov Said-Timour

2023/2024

*Drivemanager*

Outil de gestion des taxis médicalisés

Rapport  
mensuel  
de lʼentreprise

**P**

our commencer, touchez ou cliquez sur ce paramètre fictif, et débutez la saisie. Vous pouvez voir et modifier ce document sur votre Mac, iPad ou iPhone, ou sur iCloud.com.

Vous pouvez modifier du texte et des polices et ajouter de beaux graphiques. Utilisez des styles de paragraphes pour un document à lʼaspect uni. Par exemple, le style Corps est utilisé ici. Vous pouvez le modifier dans lʼonglet Texte des commandes Format.

Remerciement

*Ce dossier représente l'aboutissement de mon année d'études chez Simplon (à distance). Je tiens à remercier l'ensemble des personnes m'ayant accompagné dans cette tâche.*

* *Patrice Guillaume (DSI), Responsable d'activité Service Informatique (AURA), Tiphaine Hanry (Simplon) et Leslie Bitene-Verrier (Simplon), Responsables de la promo, pour leur accompagnement et leurs conseils.*
* *DSI pour m'avoir accepté et accompagné malgré les difficulté possiblement contraignant et un profil peu orthodoxe.*
* *Simplon Grenoble pour m'avoir accepté malgré les aménagements peu compatible avec la formation.*
* *Bien sûr je ne peux omettre ma reconnaissance envers les enseignants Thomas Laforge et Anthony Youssef pour leur implication dans notre réussite, leur organisation et leur enseignement de façon général*
* *Et pour finir à mes proches pour leur soutien moral et leurs conseils tout au long de cette formation.*

*A toutes ces personnes, je présente mes remerciements, mon respect et ma gratitude.*

Sommaire

Introduction. 4

Présentation du projet 5

Mise en place de la méthodologie devops. 8

Partie 2. 12

Partie 3. 12

Conclusion 12

Bibliographie 12

Introduction.

En 2007, Patrick Debois a inventé le terme DevOps, qui est la fusion des trois premières lettres du mot anglais "development" et de l'abréviation "ops" du mot anglais "operations" (exploitation). Il existe plusieurs illustrations du concept, notre choix a été porté sur la plus commune, d’après notre opinion (voir fig.1).

Souvent désignée comme une approche méthodologique, le DevOps vise à intégrer harmonieusement les équipes responsables du développement et de l'exploitation. L'objectif est de permettre le déploiement fréquent de mises à jour logicielles en production, favorisant ainsi la création d'un environnement de travail plus cohérent. Sur le plan organisationnel, la mise en œuvre du concept de DevOps implique un rapprochement des équipes, encourageant ainsi une coordination et une collaboration plus étroites par rapport aux structures traditionnelles en silos.

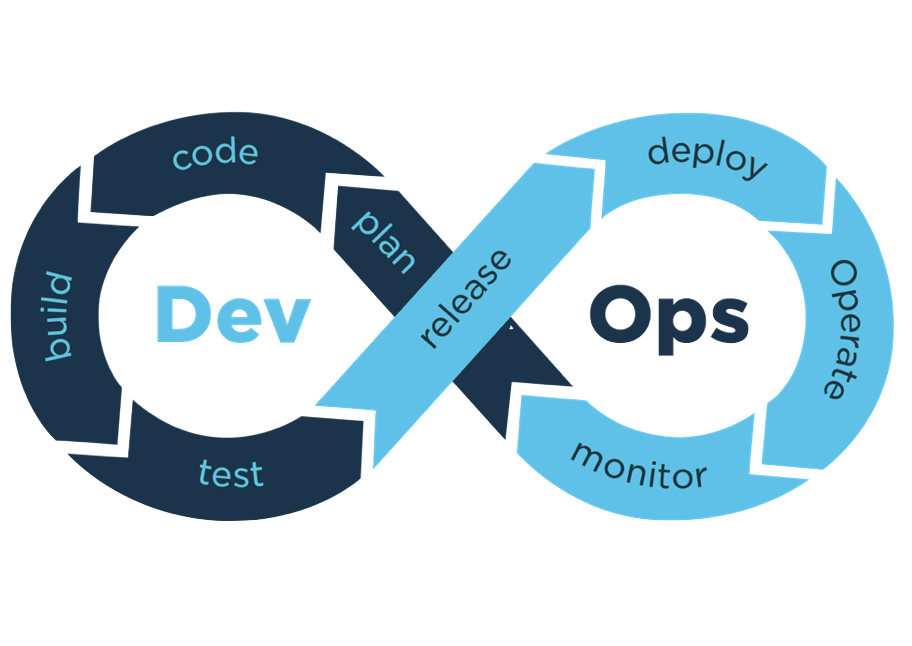


Fig. 1

De cette façon, en adoptant les principes du DevOps, nous avons pour objectif de surmonter les obstacles traditionnels qui entravent souvent le déploiement efficace des logiciels. Car, en instaurant une culture de collaboration et d'automatisation, nous souhaitons améliorer le cycle de développement, réduire les risques associés aux déploiements et améliorer la qualité du produit final.

Également cette approche favorisera l'utilisation d'outils et de pratiques qui permettent une surveillance continue des performances, une rétroaction rapide et une résolution proactive des problèmes, ce qui contribue à une plus grande agilité et à une meilleure satisfaction des clients. Ainsi, en suivant dès le début, les principes[[1]](#footnote-1) du DevOps, nous nous assurons la possibilité d’amélioration de la solution dans le future.

Présentation du projet

Contexte et la problématique.

L’idée de cette solution, pour les chauffeurs des taxis médicalisé, est venu suite aux plusieurs échanges, que nous avons pu avoir avec un professionnel de telle entreprise. Il a évoqué le souci que l’aspect bureaucratique de son travail est très prenant, mais n’est pas moins important; car pour faire la facturation des courses, auprès de la sécurité sociale, un document établi par le médecin, attestant la nécessité du transport accompagné mais aussi le chauffeur doit produire un document, à chaque course, dans lequel il fournie un certain nombre d’informations sur la course et le patient.

Le planning des courses et leur suivi est assez complexe car il faut alterné entre l’application de calendrier avec le planning et les feuilles volantes où sont noté les courses effectué, ce qui fait que le suivi général est assez complexe car il faut éviter de prendre les feuilles avec les courses inscrite. En plus de ça il faut aussi savoir retrouver toutes les informations, sur le patient, nécessaire au remplissage de l’Annexe 4[[2]](#footnote-2) à la fin de la course. Par conséquent, le chauffeur doit avoir en permanence les documents relative à tous les patients connu de l’entreprise, car ils ne peuvent pas prévoir tous les ajouts de courses possible dans la journée, et il serait trop compliqué à obliger chaque chauffeur venir au siège chaque matin ou chaque semaine pour récupérer ces documents, car ça serait une consommation d’essence inutile. Sachant que les chauffeurs ont souvent le droit de rentrer avec les voitures de service à la maison.

Mais ces problème ne sont pas uniques source de préoccupation du gérant; car employant plusieurs chauffeurs au sein de l’entreprise, il ne peut jamais être certain où se trouve les chauffeurs. Car ils a rencontré à plusieurs reprise les soucis avec les informations que les chauffeurs lui fournissent sur leur localisation, ce qui conduit à des plaintes des patients concernant les délais d’attente trop important. Puis que, souvent les patients qui ont recours à ce type de service, sont des patients qui suivent des traitement lourd et n’ont pas assez de ressource pour rester dans la salle d’attente après une séance chimiothérapie ou de l’hémodialyse.

Il serait possible de noter d’autres soucis qui ont donné envie de créer une solution centralisé mais pour cela il faudra faire un dossier à part pour cela. Toutefois nous avons exposé les principaux, ainsi dans la suite nous allons résumer, l’objectif du projet, les contraintes et les fonctionnalités principales pour donner une vision globale sur la solution, avant d’aborder la méthodologie du travail.

objectif du projet.

"DriveManager" est solution de gestion des taxis médicalisés. L'objectif principal de l'outil est de proposer une solution complète qui vise à fournir un outils centralisée et efficace de la planification des courses, la gestion des clients, et le suivi des chauffeurs et gestions des voitures. En mettant en place une application maître pour les gérant et une application dédiée aux chauffeurs.

Contraintes du projet.

La sécurité des données et des données l'authentification des utilisateurs sont des aspects critique du projet. La protection des données sensibles doit être assurée de façon optimale, ce qui implique que des mesures appropriées doivent être prises, afin de garantir la protection des données.

Fonctionnalités principales.

1. Connexion et Authentification : Permet aux gestionnaires et aux chauffeurs d'accéder à l'application de manière sécurisée.
2. Attribution des Trajets : Permet aux gestionnaires d'attribuer les trajets aux chauffeurs, et recevoir des notifications à propos des trajets attribués à ces derniers.
3. Gestion des Clients : Permet la gestion complète des fiches clients et des informations client pendant les trajets.
4. Suivi des Chauffeurs : Permet le suivi en temps réel des chauffeurs connectés à l'application.
5. Gestion des Horaires et des Étapes : Facilite la planification des horaires et des étapes pour chaque trajet.
6. Création des Fiches Chauffeurs : Permet la création et la gestion des fiches chauffeurs avec attributions des identifiants.

Mise en place de la méthodologie devops.

Dans cette partie, nous allons aborder les outils que nous avons utilisé pour la mise en place de l’approche DevOps, également nous argumenterons nos choix. Les intitulés des sections, qui vont suivront, sont des reformulations des intitulés des compétences listé dans le référentiel « Certification DevOps ».

Optimisation du Processus de Développement.

Gestion du Versionnement, Collaboration, Assurance Qualité et Intégration Continue

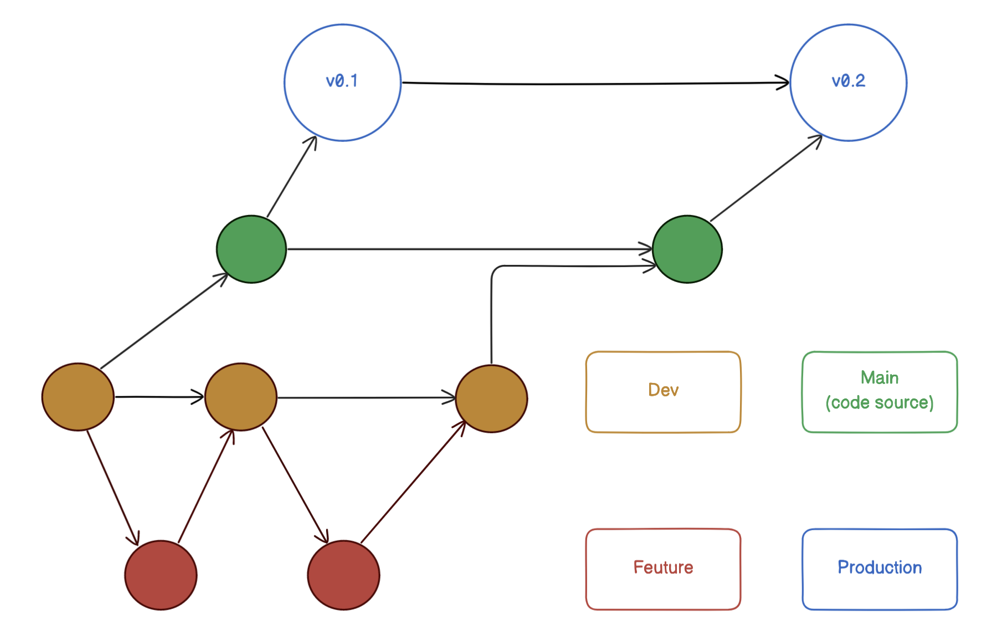


Fig. 2

Pour assurer un versionnement de notre code, nous avons décidé d’utiliser Git et GitHub, ce choix vient du fait que tout au long du parcours d’apprentissage, nous avons utilisé cet outils; également, dans le future, au moment de l’intégration des nouveaux collaborateur, nous sommes certains qu’ils sauront utiliser cet outils.

Également nous avons fait le choix d’utiliser plusieurs repository, pour cette solution. C’est à dire, nous allons utiliser un repository pour le back-end, un autre pour le front-end et un autre pour le code de l’application mobile des chauffeurs. Nous pensons que cette approche, nous convient le plus, car l’application de standardiste et l’application de chauffeurs ne sont pas similaire et nous risquons pas nous retrouver face à la question « Combien de fois tel ou tel composant est-il utilisé? ». Aussi dans le future quand ce projet prendre l’ampleur et aura plus de collaborateur que maintenant, il sera plus facile d’attribuer les permission d’accès.

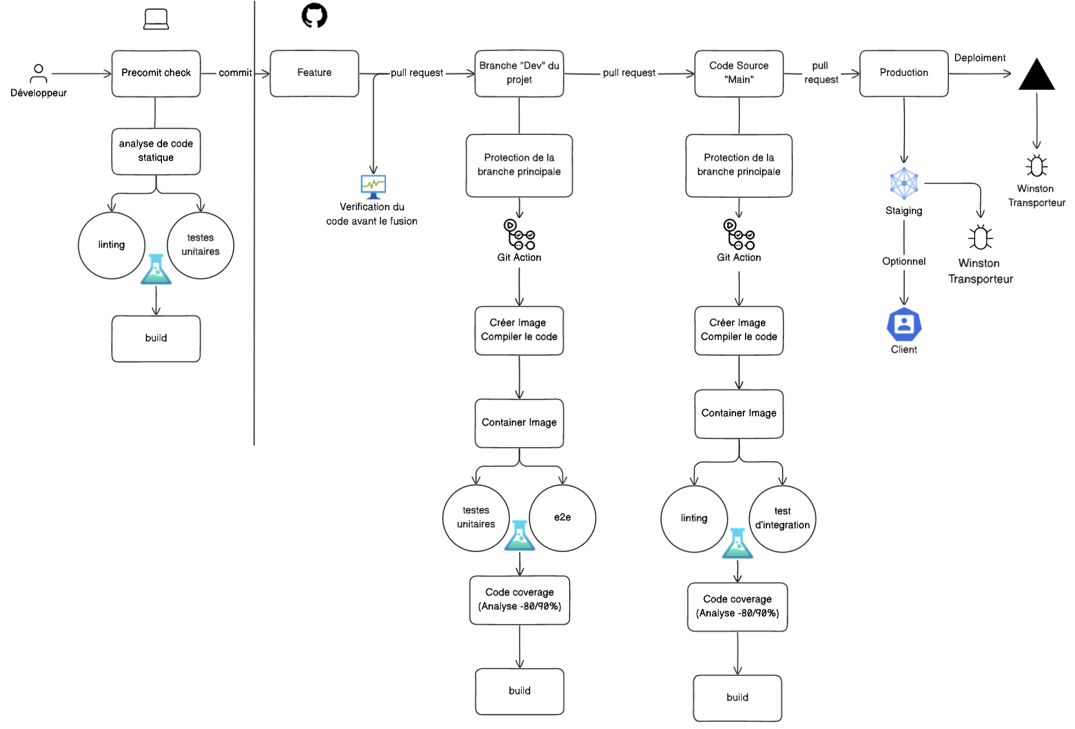


Fig. 3

Dans la figure 2, nous avons voulu illustrer le modèle de gestion de branche que nous avons adopté. Il était décidé d’utiliser trois branches principales, « Dev », « Main » et « Production ». Ces branches sont permanentes, alors que la branche « Feauture » va être utilisé pour l’intégration de nouvelles fonctionnalités ou de correctifs, par conséquent ce ne sera pas une branché permanente, et elle va être supprimé, après sa fusion avec la branche « Dev ».

A son tour la branche « Dev », va passer plusieurs batteries de testes automatisé, tels que testes unitaires et testes e2e, avant d’être fusionné avec la branche « Main ». Pour les testes unitaire nous souhaitons utiliser Jest. Et pour les testes end-to-end nous avons décidé d’intégrer Playwright dans notre projet, pour vérifier le bon fonctionnement, en simulant le comportement d’un potentiel utilisateur. Car après avoir fait des testes unitaires, pour s’assurer de bonne fonctionnement du code, il nous voir si l’utilisateur ne rencontre pas de bugs au moment de l’utilisation.

La branche « Main » aura plus au moins le même fonctionnement que « Dev », seul différence, c’est que c’est à partir d’elle que nous allons envoyer le code dans la production. Aussi les deux branches auront quelques différences dans les règles que nous avons appliqué aux branches.

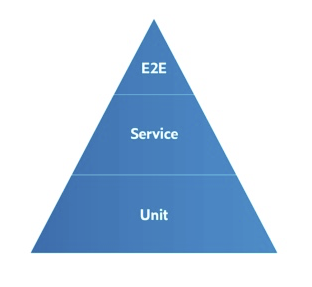


Fig. 4

Pour

La branche « Production », est une branche qui ne peut recevoir le code que depuis la branche « Main », à son tour « Main » doit passer des testes d’intégrations et de linting pour envoyer le code vers la production. De ce fait pour faire des testes d’intégrations nous avons choisi d’utiliser, comme pour les testes unitaires, Jest, un outil qui nous est, maintenant familier, car nous l’avons utilisé en cours. Pour les testes statiques, nous avons choisi d’utiliser ESLint, du fait de sa flexibilité et sa facilité d’initialisation. Comme cet outils peut être intégré dans les pipelines de buis et les workflows d’intégration continuer pour automatiser la vérification du code à chaque modification. Nous avons décidé d’utiliser cet outils pendant les testes de la branche « Main ». Mais aussi cet outils est employé dans le cadre de precomit check.

Pour mieux expliquer nos propos, nous avons décidés de joindre l’illustration de pipeline (fig. 3), que nous avons choisi. Dans cette figure nous pouvons voir, plus en détail, certaines étapes de CI et CD.

Nous avons fait le choix de faire des precomit checks, autrement écrit, avant de envoyer le travail vers le GitHub, nous allons faire les testes à l’issue desquels nous allons pouvoir envoyer notre travail dans la branche « Feuture », uniquement si les testes et le build sont passés sans erreurs.

Une fois le code est sur GitHub, nous allons revoir le code pour valider le pull & request, éventuellement cette tache devrai être assuré par une personne qui validera le « Feuture », après l’échange avec les collaborateur. Ensuite, dans la branche « Dev » Git Actions prend le relais. Pour l’automatisation des testes.

1. Conception d'une Architecture Micro-services avec des Outils et Bibliothèques Adaptées
2. Implémentation d'un Système de Veille Technologique pour l'Optimisation des Décisions Techniques
3. Sensibilisation et Familiarisation aux Méthodes d'Organisation et de Production DevOps

Partie 2.

Partie 3.

Conclusion

Bibliographie

A successful Git branching model. (s. d.). nvie.com. https://nvie.com/posts/a-successful-git-branching-model/

Atlassian. (s. d.). Principes des microservices | Atlassian. https://www.atlassian.com/fr/microservices

Benefits of a DevOps environment | MuleSoft. (s. d.). MuleSoft. https://www.mulesoft.com/resources/api/devops-environment-benefits

Bogdan Stashchuk. (2021a, novembre 4). Back-end development and APIs - FreeCodeCamp tutorial [Vidéo]. YouTube. https://www.youtube.com/watch?v=hHLmb3OD7Mo

Bogdan Stashchuk. (2021b, novembre 4). Back-end development and APIs - FreeCodeCamp tutorial [Vidéo]. YouTube. https://www.youtube.com/watch?v=hHLmb3OD7Mo

Catoire, L. (2023, 16 mai). Quelle architecture de projet choisir entre micro-services et monolithe modulaire ? Efficience IT. https://www.itefficience.com/article/quelle-architecture-de-projet-choisir-entre-micro-service-ou-monolithe-modulaire

DevOps Journey. (2023, 17 octobre). How to design a modern CI/CD pipeline [Vidéo]. YouTube. https://www.youtube.com/watch?v=KnSBNd3b0qI

Losoviz, L. (2023, 12 septembre). Mono-repo vs multi-repo : avantages et inconvénients des stratégies de dépôt de code. Kinsta®. https://kinsta.com/fr/blog/mono-repo-vs-multi-repo/

McKendrick, J. (2017, 20 avril). Pourquoi les microservices ne sont peut-être pas faits pour tout le monde. ZDNetFR. https://www.zdnet.fr/amp/actualites/pourquoi-les-microservices-ne-sont-peut-etre-pas-faits-pour-tout-le-monde-39851458.htm

Monolithic vs. Microservices : Why decoupled and headless architectures are the future | Contentstack. (s. d.). https://www.contentstack.com/cms-guides/monolithic-vs-microservices-cms-architectures

Wallen, J. (2022, 7 novembre). How to build a docker image and upload it to Docker Hub. TechRepublic. https://www.techrepublic.com/article/how-to-build-a-docker-image-and-upload-it-to-docker-hub/

What Is an API Gateway ? (s. d.). Nginx. https://www.nginx.com/learn/api-gateway/

1. Les principes du DevOps sont fondés sur la collaboration, l'automatisation, la livraison continue, la surveillance active, l'infrastructure en tant que code et l’agilité [↑](#footnote-ref-1)
2. Annexe 4 - le document que les chauffeurs doivent remplir à chaque fin de course pour la facturation ultérieur. [↑](#footnote-ref-2)