

Table des matières

Table des figures	ii
Introduction générale	1
1 Présentation de l'entreprise d'accueil	2
1.1 Orange Business	2
1.2 L'équipe WAT	2
1.3 Organigramme	3
2 Cadre du stage	5
2.1 Intégration continue et déploiement continue	5
2.1.1 Intégration Continue	5
2.1.2 Déploiement continue	6
2.2 Contexte	6
2.3 Objectifs	6
2.4 Planification	7
2.5 Méthode de travail	8
3 Choix technologiques	9
3.1 Technologie back-end	9
3.1.1 Symfony	9
3.1.2 PHPQA	10
3.1.3 Codeception	11
3.1.4 MariaDB	13
3.2 Technologie back-end	14
3.2.1 AdminLTE	14
3.2.2 ChartJS	14
3.2.3 ReactJS	15
4 Réalisation	16
4.1 Introduction	16
4.2 Création d'un utilisateur	16
4.3 Authentification	17
4.4 API de récupération des fichiers de rapports	17

4.5	Dashboard	17
4.5.1	Récapitulatif de l'état d'un projet	17
4.5.2	Diagrammes	18
4.5.3	Tableau de builds	19
4.6	Difficultés rencontrées	21
4.6.1	Les entités	21
4.6.2	Les merges requests	21
4.6.3	Single responsibility	21
4.6.4	Adaptation aux outils de qualité de code	21
webographiques		21

Table des figures

1.1	Equipe WAT	3
1.2	Organigramme	4
3.1	Symfony	9
3.2	PHPQA	10
3.3	Codeception	11
3.4	Docker	12
3.5	Docker	13
3.6	AdminLTE	14
3.7	ChartJS	14
3.8	React	15
4.1	Authenfication	17
4.2	Recaptulatif	18
4.3	Diagramme - builds	18
4.4	Diagramme - outils	19
4.5	Tableau - build	20
4.6	Repartition - outils	20

Introduction générale

Dans le domaine du développement informatique, la montée en version d'une application est une chose qui intervient très souvent face aux besoins des clients qui ne cessent de s'accroître. Pour s'adapter aux changements, le code d'une application se doit d'être maintenable, testable et invulnérable face aux failles de sécurité.

Des outils de qualité de code il en existe des tonnes(citez quelques uns), qui en plus de pouvoir établir des règles de bonnes pratiques, génèrent des rapports sur l'état d'un projet.

Dans le cadre de ce stage, l'objectif est de récupérer tous ces rapports et de les visualiser de manière globale à travers une plateforme web.

Chapitre 1

Présentation de l'entreprise d'accueil

1.1 Orange Business

Orange Business (anciennement Orange Business Services) est une filiale et une marque commerciale du groupe Orange qui fournit, en France et dans le monde, des services de communication intégrée aux entreprises dans les domaines du cloud computing, des télécommunications, des communications unifiées et de la collaboration.

Orange Business regroupe les activités business to business des différentes filiales françaises et internationales du groupe Orange.[1]

Orange Business est une identité commerciale du groupe Orange qui a été constituée le 1er juin 2006 (sous le nom d'Orange Business Services)², afin d'offrir une marque unique des services de télécommunication (données, téléphonie et convergence) et informatiques pour les entreprises. Elle opère dans 220 pays et territoires et emploie plus de 21 000 personnes réparties sur 166 pays³.

Orange Business est le résultat de la consolidation et du regroupement sous une seule marque, d'entités issues de trois entreprises aujourd'hui dissoutes : France Telecom, Equant et Wanadoo².

Le 16 février 2023, Christel Heydemann, la directrice générale du groupe Orange, annonce qu'Orange Business Services se nommera désormais tout simplement Orange Business.[1]

Les lignes qui suivront, seront consacrées à Orange Business Nantes.

1.2 L'équipe WAT

En fonction des technologies utilisées, Orange Business Nantes est répartie en plusieurs équipes (WAT, BIM, PIX, TED et TM).

L'équipe WAT(Web Applications Team) est spécialisée dans le pilotage et le développement d'applications Web basées sur les technologies PHP, dont le framework Symfony, ou des CMS, dont

drupal et Wordpress.

L'équipe est composée de :

- **Chefs de projets** : responsables du bon déroulement des projets.
- **Experts techniques** : ils ont un rôle transverse dans l'équipe en intervenant sur plusieurs projets. Leurs activités principales : support technique aux projets, avant-vente, mise en place des environnements techniques, de la CI/CD, veille techno, ...
- **Développeurs** : codent les fonctionnalités souhaitées sur les projets.



FIGURE 1.1 – Equipe WAT

1.3 Organigramme

L'illustration ci-dessous permet de résumer en une image les paragraphes ci-dessus décrivant l'entreprise.

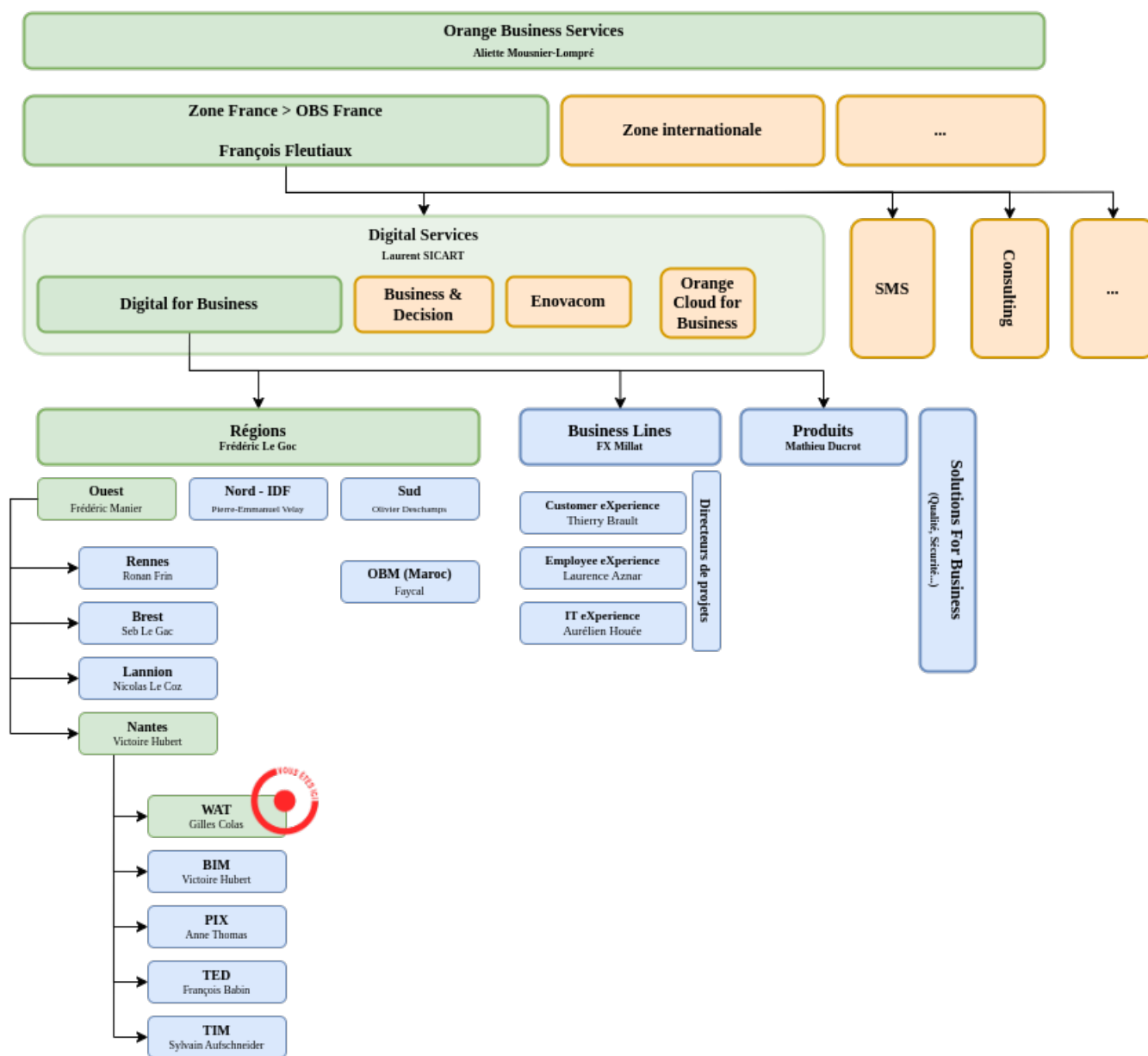


FIGURE 1.2 – Organigramme

Chapitre 2

Cadre du stage

2.1 Intégration continue et déploiement continue

2.1.1 Intégration Continue

L'intégration continue (IC) est une pratique de développement logiciel qui vise à améliorer la qualité, la fiabilité et l'efficacité du processus de développement en automatisant la construction, les tests et la validation du code à chaque modification (les commits effectués par les développeurs, par exemple).

Le principal but de cette pratique est de détecter les problèmes d'intégration au plus tôt lors du développement.

L'intérêt de l'intégration continue réside dans plusieurs avantages clés :

- **Détection précoce des problèmes** : L'IC permet de détecter rapidement les erreurs et les problèmes de compatibilité en exécutant des tests automatisés à chaque fois qu'une modification de code est soumise. Cela aide à identifier les erreurs dès qu'elles sont introduites, facilitant ainsi leur correction avant qu'elles ne deviennent des problèmes plus graves.
- **Rapidité et réduction des retards** : En automatisant la compilation, les tests et les déploiements, l'IC accélère le processus de développement. Les développeurs n'ont pas besoin d'attendre de longues périodes avant de vérifier si leur code fonctionne correctement, ce qui réduit les retards et favorise une itération plus rapide.
- **Amélioration de la collaboration** : L'IC favorise la collaboration entre les membres de l'équipe en fournissant un environnement intégré où chaque développeur peut voir les modifications apportées par les autres membres de l'équipe et s'assurer qu'elles s'intègrent correctement. Cela réduit les conflits d'intégration et facilite le travail d'équipe.

2.1.2 Déploiement continue

Le déploiement continu (DC) est une pratique de développement logiciel qui vise à automatiser et à simplifier le processus de mise en production des nouvelles fonctionnalités, des correctifs et des modifications de code. Contrairement à l'intégration continue qui se concentre sur l'automatisation des tests et de la validation à chaque modification de code, le déploiement continu va plus loin en automatisant également le déploiement des modifications validées dans un environnement de production.

L'objectif principal du déploiement continu est de réduire le temps entre la validation d'une modification de code et sa disponibilité pour les utilisateurs finaux. Cela permet de livrer plus rapidement de nouvelles fonctionnalités, des améliorations et des correctifs, ce qui améliore l'expérience utilisateur et accroît l'agilité de l'équipe de développement.

2.2 Contexte

Pour automatiser l'intégration de ses applications, l'équipe WAT faisait appel à Jenkins et SonarQube. Ces outils, en plus de gérer les différents stage de pipeline, avait la possibilité d'afficher les rapports d'analyse de code (grâce aux artefacts générés après l'intégrations), avec des diagramme permettant de visualiser l'évolution du nombre d'erreur par outil en fonction des builds.

Bien que Jenkins offre des fonctionnalités intéressantes en terme d'intégration, il peut être difficile à maintenir et difficile à configurer. Ainsi, depuis Juin 2023 la direction technique de l'entreprise à décider vers Gitlab CI/CD.

Le code source des projets étant hébergé sur Gitlab, alors ces projets peuvent bénéficier (grâce à un fichier de configuration) de Gitlab CI/CD sans aucune installation particulière.

Des runners partagés sont mis à disposition afin d'exécuter les jobs. Il n'est pas nécessaire de déclarer un slave afin de lancer nos pipelines comme on le faisait avec Jenkins.

Gitlab CI/CD n'offre pas cette possibilité d'afficher les rapports d'analyse de code de manière détaillée (Diagramme, courbe d'évolution, etc) comme le fait Jenkins. C'est ainsi qu'entre en scène les missions de ce stage.

2.3 Objectifs

L'objectif de ce stage est de développer une application Web en PHP Symfony pour améliorer les pipelines d'intégration continue (avec Gitlab CI/CD). Cette application sera une alternative à la fonctionnalité qu'offrait Jenkins pour l'affichage des rapports de qualité de code à travers un dashboard. Les fonctionnalités principales de cette application seront :

- Recueillir les données des outils de qualités de code via des API

- Historisation de ces données par projet.
- Consulter ces données sur des dashboards.
- Être alertés lorsque les seuils de qualités ne sont pas atteints.
- La réalisation d'un backoffice depuis lequel les administrateurs auront la possibilité de gérer les projets sur lesquels des développeurs seront amenés à collaborer. Aussi, à travers ce backoffice, il sera possible de créer d'autres utilisateurs qui auront la possibilité d'effectuer ces mêmes actions.
- Définir les tendances pour chaque outil de qualité de code (indentation du code, respect de la nomenclature des variables, etc.). En effet, cela consiste à indiquer s'il y a une régression ou une amélioration après un certain nombre de commits effectués par les développeurs.
- Définir une tendance globale pour avoir un aperçu sur l'état de chaque projet.

2.4 Planification

Que l'on soit chef d'équipe ou responsable de sa propre activité, bien planifier ses projets est essentiel pour être efficace. Cela permet d'organiser son temps dépendamment du travail à réaliser, et de garantir son efficacité sur le long terme. Ainsi la planification d'une application passe par plusieurs étapes (qui peut dépendre d'une application à une autre) telle que :

- Étape n° 1 : **analyse fonctionnelle et définition des objectifs** : Cette partie consistera à rechercher et à caractériser les fonctions offertes par notre application pour satisfaire les différents besoins du client.
- Étape n° 2 : **conception détaillée** : La phase de conception détaillée donne lieu à la rédaction du cahier des charges opérationnel. C'est elle qui précisera les différents éléments de dimensionnement du projet.[2] C'est aussi dans cette étape qu'entre la modélisation du système avec des langages de modélisations comme UML.
- Étape n° 3 : **Développement du projet** : C'est dans cette partie qu'entre en jeu la partie technique de l'application. Cette étape exige la maîtrise d'au moins un langage de programmation.
- Étape n° 4 : **Phase de tests** : C'est l'ensemble des tests (tests unitaires, tests fonctionnels et tests d'acceptances) qui permettront de retrouver les erreurs moins évidentes qui n'ont pas été détectées pendant la phase de développement.
- Étape n° 5 : **Recette** : Permet de s'assurer que l'application développée correspond bien aux exigences fixées par le client.

- Étape n° 6 : **Mise en production** : Déploiement de l'application.
- Étape n° 7 : **Maintenance** : On entend par maintenance, l'ensemble des modifications mises en place après la mise en œuvre de l'application en production afin de corriger les bogues, améliorer les performances ou encore l'adapter à une modification de son environnement.[2]

2.5 Méthode de travail

En premier lieu, les besoins sont définis par le client à travers les tickets sur JIRA ou les issues sur GitLab. Une fois que ces tickets sont réalisés par le développeur que je suis au sein de l'équipe, une merge request est créé pour permettre à un expert technique d'intervenir afin de vérifier la qualité de code produit par le développeur.

Après vérification par l'expert technique, celui-ci peut approuver les fonctionnalités développer en cas de respect des bonnes pratiques et de cohérence entre le ticket (ou issue) et la fonctionnalité. Dans le cas contraire, un retour est fait au développeur pour lui signifier les problèmes liés à sa fonctionnalité.

Ce processus est effectué de manière itérative jusqu'à la réalisation de chaque ticket. Enfin, toutes les deux semaines, une réunion de suivi de stage est réalisée entre les parties prenantes(client, chef de projet, tuteur de stage et stagiaire développeur) du stage pour faire des points sur l'avancement du stage, des difficultés rencontrées et s'assurer que les besoins du client sont bien respectés à travers parfois une démonstration.

Chapitre 3

Choix technologiques

3.1 Technologie back-end

3.1.1 Symfony



FIGURE 3.1 – Symfony

Symfony est un framework de développement web open-source écrit en PHP. Il fournit un ensemble de composants modulaires et d'outils qui facilitent la création et la maintenance d'applications web complexes et évolutives. Symfony suit les principes de conception du modèle MVC (Modèle-Vue-Contrôleur), ce qui favorise la séparation des préoccupations et la structuration claire du code.

Les caractéristiques principales de Symfony incluent :

- **Composants réutilisables** : Symfony est basé sur un ensemble de composants indépendants, tels que le gestionnaire de dépendances, le système de routage, le composant de for-

mulaire, le composant de sécurité, etc. Ces composants peuvent être utilisés de manière modulaire dans différentes applications.

- **Productivité** : Symfony propose des outils et des générateurs de code qui accélèrent le développement. Il offre également une interface en ligne de commande (CLI) pour effectuer des tâches courantes telles que la génération de code, la gestion de bases de données et les tests.
- **Flexibilité** : Les développeurs peuvent personnaliser et configurer les composants selon les besoins spécifiques de leur projet. Symfony permet également l'intégration de bundles tiers pour ajouter des fonctionnalités supplémentaires.
- **Qualité du code** : Symfony encourage les bonnes pratiques de codage, le test unitaire et fournit des outils pour garantir la qualité du code. Cela facilite la maintenance à long terme des applications.
- **Sécurité** : Symfony offre des fonctionnalités de sécurité intégrées, telles que la gestion des autorisations, la protection contre les vulnérabilités courantes et la prévention des attaques.
- **Documentation** : Symfony dispose d'une documentation complète et détaillée, ainsi que d'une communauté active qui partage des ressources et des solutions aux problèmes courants.
- **Support à long terme** : Symfony suit des cycles de maintenance à long terme, ce qui garantit que les applications développées avec ce framework restent à jour et sécurisées.

Symfony est largement utilisé pour développer des applications web de toutes tailles, des sites web simples aux applications d'entreprise complexes. De nombreuses entreprises et projets renommés utilisent Symfony pour sa fiabilité, sa robustesse et sa capacité à accélérer le développement tout en maintenant des normes élevées de qualité et de sécurité.

3.1.2 PHPQA

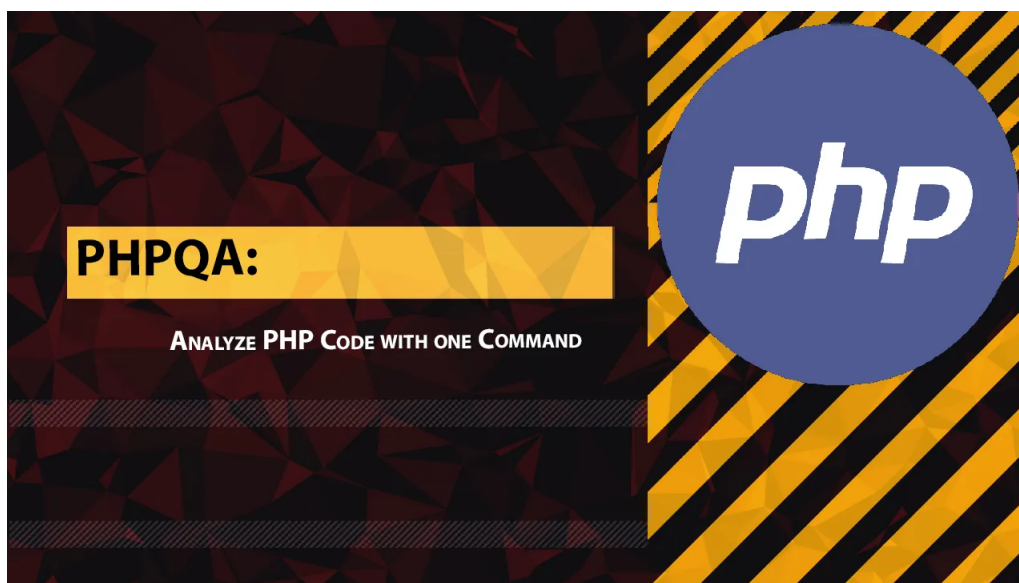


FIGURE 3.2 – PHPQA
[3]

PHPQA est un ensemble d'outils et de techniques destinés à l'analyse statique et dynamique du code PHP, ainsi qu'à l'amélioration de sa qualité et de sa performance. Il vise à aider les développeurs à identifier les problèmes potentiels dans le code, à maintenir de bonnes pratiques de développement et à optimiser les performances de leurs applications.

Les principales catégories d'outils et de pratiques incluses dans PHPQA sont les suivantes :

- **Analyse statique du code** : Ces outils examinent le code source PHP sans l'exécuter et identifient des erreurs de syntaxe, des problèmes de sécurité, des antipatterns (mauvaises pratiques) et d'autres problèmes potentiels. Des exemples d'outils d'analyse statique populaires sont **PHPStan**, et **PHP CodeSniffer**.
- **Mesure de la couverture de code** : Ces outils mesurent quelle partie du code est couverte par les tests automatisés, aidant ainsi à identifier les zones non testées et à garantir une meilleure couverture.
- **Profiler et analyseurs de performances** : Ces outils permettent de surveiller et d'analyser les performances de l'application en identifiant les goulots d'étranglement, les requêtes lentes et d'autres problèmes de performance. Des outils populaires incluent Xdebug et Blackfire.
- **Documentation du code** : La documentation claire et précise du code est importante pour la compréhension et la maintenance du projet. Des outils comme **PHPDocumentor** aident à générer une documentation automatique à partir des commentaires du code source.
- **Amélioration du style de code** : Les outils comme **PHP CS Fixer** aident à maintenir un style de code cohérent en automatisant la correction des espaces, des indentations et d'autres conventions de codage.

En utilisant PHPQA, les développeurs peuvent identifier rapidement les problèmes potentiels dans leur code, améliorer sa qualité, accélérer le processus de développement et optimiser les performances de leurs applications PHP. Cela contribue à créer des applications plus robustes, fiables et performantes.

3.1.3 Codeception



FIGURE 3.3 – Codeception
[4]

Basé sur PHPUnit, Codeception est un framework de test automatisé pour les applications web. Il est principalement utilisé pour effectuer des tests unitaires, fonctionnels et d'acceptation dans le contexte du développement web. Codeception simplifie la création et l'exécution de tests automatisés en fournissant des fonctionnalités spécifiques pour l'émulation d'actions utilisateur, la validation du comportement de l'application.

Aussi, il rassemble et partage les meilleures pratiques et solutions pour tester les applications web PHP. Grâce à un ensemble flexible de modules inclus, les tests sont faciles à écrire, à utiliser et à maintenir [5].

En plus d'offrir les fonctionnalités de PHPUnit, codeception offre de modules intéressants permettant de faire facilement des tests d'acceptance et fonctionnels.

Docker

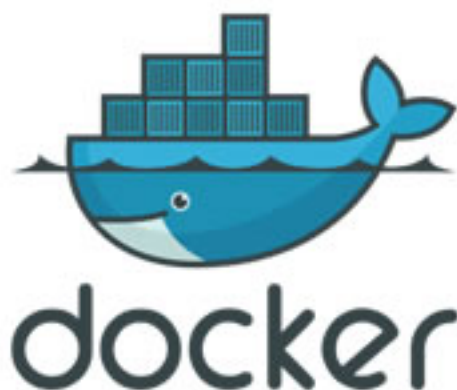


FIGURE 3.4 – Docker
[6]

Docker est une plate-forme open-source qui permet de développer, déployer et exécuter des applications dans des conteneurs légers et portables. Les conteneurs sont des environnements logiciels isolés qui incluent tout ce dont une application a besoin pour s'exécuter, y compris le code, les bibliothèques, les dépendances et les configurations.

Docker permet aux développeurs d'être sur la même longueur d'onde en terme de version de logiciels utilisées lors de la phase de développement. Ce qui peut être très utile pour éviter de potentiels problèmes lors du déploiements d'une application.

3.1.4 MariaDB



FIGURE 3.5 – Docker
[7]

MariaDB est un système de gestion de base de données open-source et relationnel (SGBDR) qui est conçu pour être une alternative compatible avec MySQL. Il a été créé par certains des développeurs originaux de MySQL en réponse à des préoccupations concernant la direction et la gouvernance de MySQL après son acquisition par Oracle Corporation.

MariaDB partage une grande partie de la syntaxe et des fonctionnalités avec MySQL, ce qui signifie que les applications écrites pour MySQL peuvent généralement être utilisées avec MariaDB sans modifications majeures. Cependant, MariaDB propose également des améliorations et des fonctionnalités spécifiques qui en font une option attrayante pour de nombreux utilisateurs.

Les avantages principaux qu'on peut tirer de MariaDB sont :

- **Performance** : MariaDB inclut des optimisations et des améliorations de performances par rapport à MySQL, notamment dans les domaines de la vitesse d'exécution des requêtes et de la gestion de la mémoire.[8]
- **Stockage** : MariaDB prend en charge divers moteurs de stockage, y compris InnoDB (le moteur de stockage par défaut), Aria, MyISAM et d'autres.[8]
- **Haute disponibilité** : MariaDB propose des fonctionnalités de réplication et de clustering pour garantir une haute disponibilité des données et une tolérance aux pannes, etc.[8]

3.2 Technologie back-end

3.2.1 AdminLTE



FIGURE 3.6 – AdminLTE
[9]

AdminLTE est un template open-source de conception d'interface utilisateur (UI) et d'administration pour les applications web. Il fournit une collection de composants, de styles CSS et de scripts JavaScript préconstruits pour créer rapidement des interfaces d'administration élégantes et conviviales.

L'avantage avec Admin Lte, c'est qu'il permet aux développeurs avoir d'un gain de temps énorme et de se concentrer sur la logique métier de l'application, du fait des composants réutilisables mis à leur disposition.

3.2.2 ChartJS



FIGURE 3.7 – ChartJS
[10]

ChartJS est une bibliothèque JavaScript open source gratuite pour la visualisation de données. Créée par le développeur web Nick Downie en 2013, la bibliothèque est maintenant maintenue par la communauté et est la deuxième bibliothèque de graphiques JS la plus populaire sur GitHub par le nombre d'étoiles après D3.js. Chart.js est rendu dans un canvas HTML5. Elle est disponible sous la licence MIT.[11]

Pour réaliser les diagrammes d'évolutions et les courbes de tendance, ChartJS a été un élément essentiel, voir indispensable.

3.2.3 ReactJS

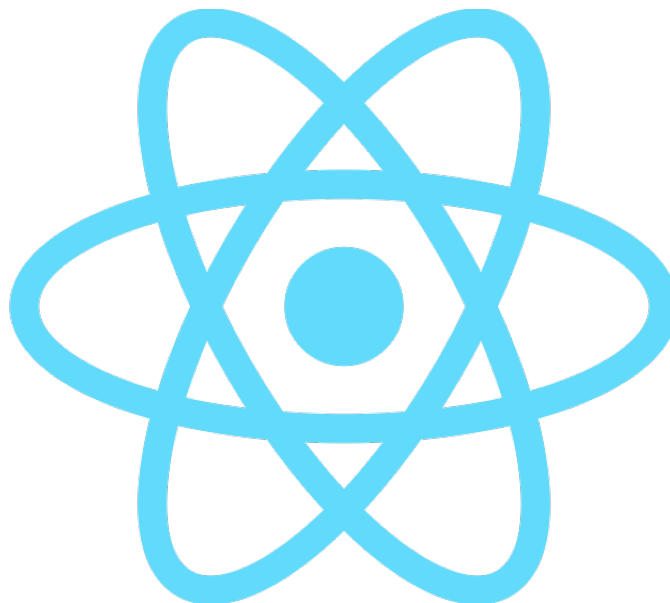


FIGURE 3.8 – React
[12]

React JS est une bibliothèque javascript développée par Facebook depuis 2013 [13]. Il offre la possibilité de créer des composants réactifs grâce aux hooks. React a pour avantage d'être utilisé en fonction de l'état d'un projet (from scratch ou en plein milieu). Pour un projet démarré à partir de zéro, il est possible de créer une application full-react, dont la mise en place est juste effectuée par une commande (**create-react-app**).

Dans le cas de notre application il a été utilisé (grâce à Symfony UX) pour rendre le tableau des bluids plus réactif à chaque clique d'un utilisateur.

Chapitre 4

Réalisation

4.1 Introduction

Dans ce chapitre, nous passerons en revue la réalisation des termes évoqués tout au long des lignes précédentes. Nous aborderons également les difficultés techniques rencontrées durant le stage.

4.2 Création d'un utilisateur

La gestion des utilisateurs est une fonctionnalité indispensable dans la création d'un back office. Dans le système mis en place, pour que les données des utilisateurs soient prises en compte, plusieurs phases de vérifications doivent être effectuées. Lorsque l'utilisateur renseigne ses informations, la phase de validation de données est déclenchée. Si les données ne sont pas valides alors un ensemble de message d'erreurs est renvoyé à ce dernier.

Dans le cas contraire, un mail de vérification est envoyé à l'utilisateur pour confirmer son adresse mail.

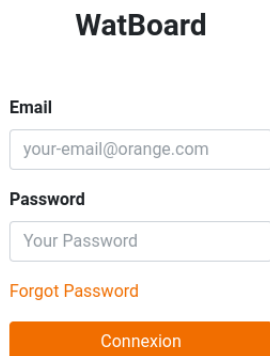
Tout ce processus peut être initié par deux méthodes : **Commande symfony** et le **back office**.

Depuis une commande Symfony : L'accès au back office requiert le compte d'un utilisateur pré-enregistré dans la base de données. Pour un projet initialisé, il n'existe aucun utilisateur créé dans la base de données ce qui peut être problématique. Ainsi la création d'une commande symfony de créer le compte d'un utilisateur sans passer par une interface graphique.

Depuis le back-office : En plus d'une commande Symfony, une interface graphique permet à un administrateur.

4.3 Authentification

L'authentification est une couche de sécurité mise en place pour accéder aux fonctionnalités réservées aux administrateurs. Pour y accéder, l'utilisateur doit avoir un compte déjà créé par un administrateur au préalable.



The image shows a login interface for 'WatBoard'. At the top, the title 'WatBoard' is centered. Below it, there are two input fields: one for 'Email' with the placeholder text 'your-email@orange.com', and one for 'Password' with the placeholder text 'Your Password'. Below the password field is a link labeled 'Forgot Password' in orange text. At the bottom, there is an orange button with the text 'Connexion' in white.

FIGURE 4.1 – Authentification

4.4 API de récupération des fichiers de rapports

Les rapports de qualité de code, tests, sécurités sont générés depuis GitLab CI/CD. Ces rapports sont appelés artefacts dans le contexte d'intégration continue. Un pipeline regroupe un ensemble de stage destiné à faire des tâches spécifiques (appelée jobs). Ainsi, dans notre système d'intégration continue mise en place, nous avons plusieurs stages permettant d'analyser le code écrit par les développeurs tels que : Quality, tests, security, etc. En fonction de chaque stage, il existe des outils appropriés à effectuer les tâches qu'il faut. Par exemple PHPQA pour l'analyse de la qualité de code, Codeception pour les tests, Trivy pour la vérification de la sécurité des packages.

À la fin de chaque stage, les fichiers de rapports sont générés. Ces fichiers de rapports sont envoyés par API à notre application qui aura pour tâches :

- Stocker les données récupérer dans ces fichiers et les stocker dans une base de données.
- Afficher les diagrammes à travers ces données stockées.

4.5 Dashboard

4.5.1 Récapitulatif de l'état d'un projet

Le récapitulatif global d'un projet consiste à avoir un aperçu global sur l'état d'un projet. On pourrait y voir les informations suivantes :

- **Le nombre de builds** : Correspond au nombre de fois qu'un projet a été déployé dans le pipeline.
- **Le nombre d'erreurs** : Correspond au nombre d'erreurs repéré dans tous les rapports.
- **Le nombre de warnings** : Correspond au nombre d'avertissements repéré dans tous les rapports.
- **La tendance** : Est une fonctionnalité qui consiste à analyser l'état d'un projet sur certain nombre de builds (cinq, dix, ou plus). En fonction de ces analyses, on peut en déduire plusieurs états :
 - **Soleil** : Lorsqu'on constate une évolution de la qualité de code sur les dix derniers builds.
 - **Nuageux** : Lorsqu'on constate une évolution sur les cinq derniers builds.
 - **Pluie** : Lorsque les deux cas ne sont pas respectés.

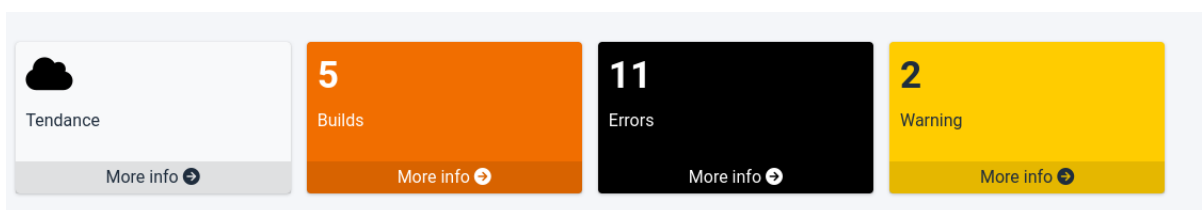


FIGURE 4.2 – Recaptulatif

4.5.2 Diagrammes

Build

Le diagramme de builds est un graphique permettant d'afficher l'évolution d'un projet en fonction des issues (erreurs + warnings) récupérés dans les fichiers de rapports. Un coup d'œil sur ce diagramme permet de savoir si un projet est en régression ou non.

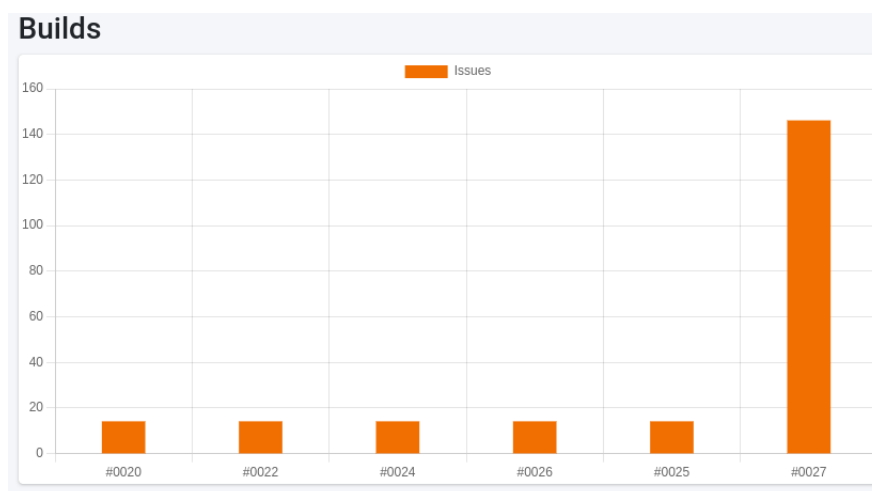


FIGURE 4.3 – Diagramme - builds

Outils

En plus de la tendance, le diagramme d'un outil permet de mettre en lumière l'évolution d'un projet vis à vis de l'utilisation cet outil à travers un diagramme en lignes.

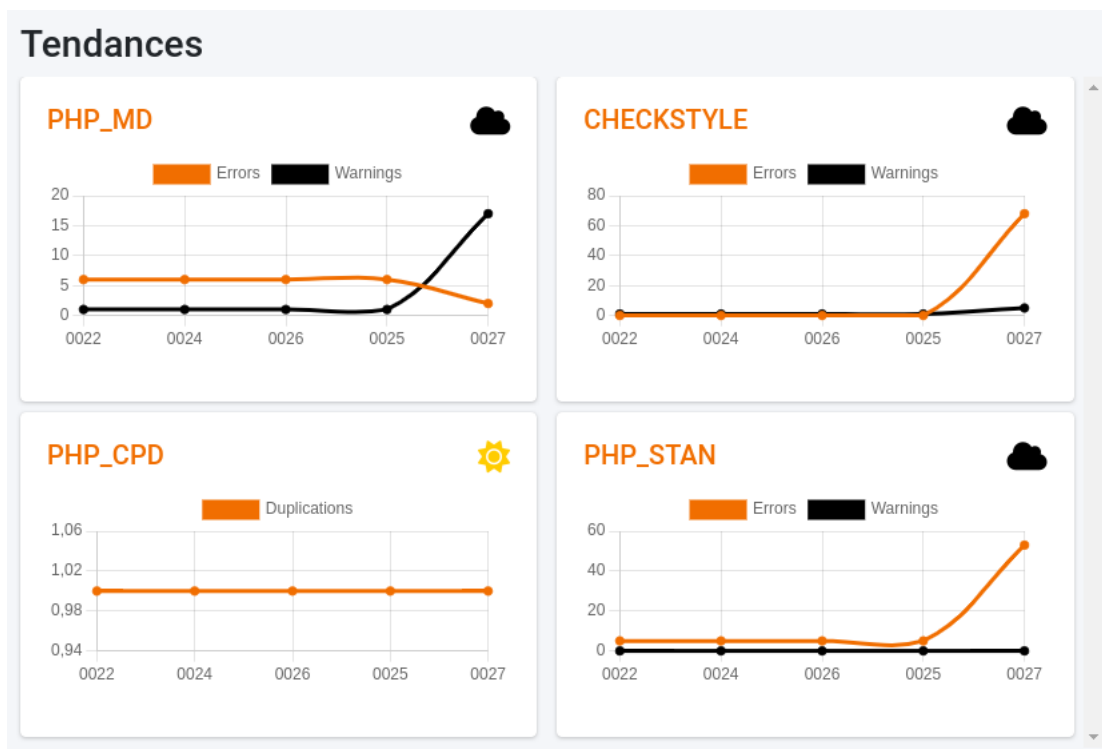


FIGURE 4.4 – Diagramme - outils

4.5.3 Tableau de builds

Le tableau de build permet d'afficher l'historique des builds d'un projet et leurs issues. Il est possible d'interagir avec ce tableau pour afficher la répartition de ces issues par outils de qualité de code.

Liste des builds		
Version ↕	Issues ↕	Date ↕
#0027	146	15-08-2023 - 19:01:09
#0026	14	15-08-2023 - 18:22:38
#0025	14	15-08-2023 - 18:22:48
#0024	14	15-08-2023 - 18:22:26
#0022	14	10-08-2023 - 10:32:46
<div><< < 1 2 > >> 5 ▾</div>		

FIGURE 4.5 – Tableau - build

Un clique sur une ligne du tableau permet d'afficher un digramme en camembert montrant la répartition des issues à travers les outils utilisés.

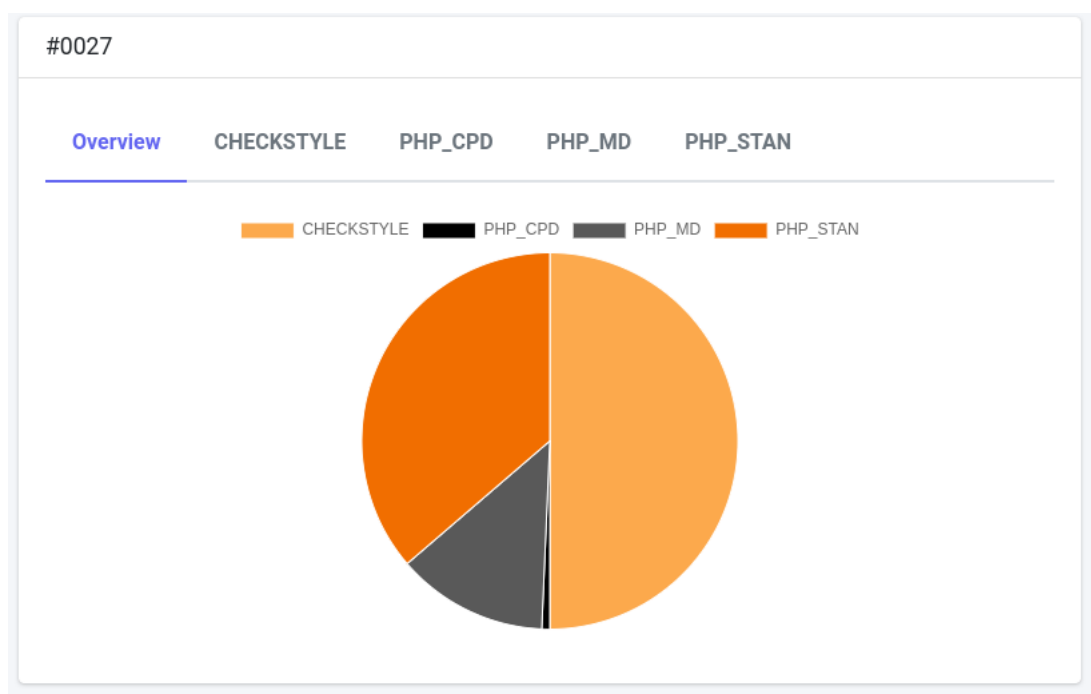


FIGURE 4.6 – Repartition - outils

Bibliographie

- [1] <https://fr.wikipedia.org/>. Orange business. https://fr.wikipedia.org/wiki/Orange_Business.
- [2] Samantha Mur. Comment gerer un projet informatique? <https://www.appvizer.fr/magazine/operations/gestion-de-projet/gestion-projet-informatique>. 18/12/2021.
- [3] PHPQA. <https://i.morioh.com/2023/06/17/ebfc7e41.webp>.
- [4] Codeception. <https://blog.eleven-labs.com/imgs/posts/2017-03-09-retour-experience-codeception/codeception-logo.png>.
- [5] Codeception. <https://codeception.com/>.
- [6] Docker. <https://www.1min30.com/wp-content/uploads/2018/05/logo-docker-1.jpg>.
- [7] MariaDB. <https://e7.pngegg.com/pngimages/468/466/png-clipart-mariadb-mysql-amazon-relational-database-service-fork-marine-mammal-mammal-thumbnail.png>.
- [8] OpenAI. <https://chat.openai.com/>.
- [9] AdminLTE. <https://seeklogo.com/images/a/adminlte-logo-f861652b55-seeklogo.com.png>.
- [10] ChartJS. <https://www.chartjs.org/img/chartjs-logo.svg>.
- [11] ChartJS. <https://fr.wikipedia.org/wiki/chart.js>.
- [12] <https://fr.wikipedia.org/wiki/react>.
- [13] React. <https://fr.wikipedia.org/wiki/react>.