Une image contenant extérieur, volant, cerf-volant, bleu

Description générée automatiquement

**Documentation technique**

ToDo & Co

Version 1.0

00/00/0000

1. Sommaire

[1. Sommaire 1](#_Toc180675931)

[2. Contexte 2](#_Toc180675932)

[3. Problématique 3](#_Toc180675933)

[a) Migration technique d’un projet client 3](#_Toc180675934)

[b) Correction des anomalies sur un projet existant 3](#_Toc180675935)

[c) Nouvelle version de l’application « INGENEO » 3](#_Toc180675936)

[4. Missions confiées 4](#_Toc180675937)

[5. Bilan 5](#_Toc180675938)

1. Contexte

Je m'appelle Karol ZIELINSKI, j'ai 31 ans, et je suis actuellement en alternance dans le cadre de ma reconversion professionnelle vers le métier de développeur d'applications spécialisées en PHP et Symfony. Après plusieurs années d'expérience dans un autre secteur, j'ai décidé d'entamer cette nouvelle voie pour obtenir mon diplôme et évoluer dans un domaine qui me passionne. Cette alternance s'inscrit dans mon parcours de formation, et il est essentiel pour valider mes compétences et acquérir une expérience concrète.

L'organisation qui m'accueille pour cette alternance est Septeo Solutions Experts-Comptables, une entreprise qui propose des solutions logicielles dédiées aux professionnels de l'expertise comptable. Leur principal produit, INGENEO, offre un ensemble d'outils permettant aux experts-comptables de gérer efficacement et en toute simplicité les processus comptables, avec des fonctionnalités telles que la collecte automatique des factures, la reconnaissance et l'imputation des écritures comptables, ainsi que l'intégration fluide avec des logiciels de production comptable. En plus de faciliter la gestion comptable, l'entreprise permet aux experts de valoriser leur expertise auprès de leurs clients grâce à des modules comme les indicateurs de performance personnalisés et un espace client dédié, qui inclut des fonctionnalités telles que la gestion des documents (GED), la facturation et le suivi de la conformité à la facturation électronique.

J'ai choisi cette entreprise pour mon stage car j'y travaille déjà depuis janvier 2019 et j'ai eu l'opportunité de signer un contrat d'alternance avec eux à partir de janvier 2023, dans le cadre de mon projet de reconversion. Cette entreprise me permet de concilier l'acquisition de compétences professionnelles tout en poursuivant mon évolution interne.

Les objectifs que je me suis fixés pour ce stage sont principalement axés sur la montée en compétence dans le développement d'applications PHP/Symfony. Mon but est d'acquérir une expertise plus pointue dans ce domaine pour gagner rapidement en autonomie et en expérience, tout en participant activement aux projets de développement de l'entreprise.

1. Problématique
2. Migration technique d’un projet client

*Comment assurer une migration fluide d’un projet client vers une version technologique plus récente sans impact sur les fonctionnalités existantes et les performances ?*

La migration d’un projet client est une tâche délicate qui présente souvent des défis en termes de compatibilité et de continuité de service. En tant que développeur, un des enjeux est de garantir que la migration vers une version plus récente du Framework ne crée pas de rupture dans les fonctionnalités. Cette problématique implique de minimiser les risques de régressions et d’assurer que le système continue à répondre aux besoins fonctionnels et de performance du client.

1. Correction des anomalies sur un projet existant

*Comment optimiser le processus de correction des anomalies dans un projet existant pour garantir la stabilité et la qualité de service du produit ?*

Les projets existants peuvent accumuler des anomalies en raison de divers facteurs (changements de code, mises à jour non documentées, évolutions fonctionnelles, etc.). En tant que développeur, il est crucial de mettre en place une méthodologie efficace de gestion des anomalies qui permet de corriger rapidement les bugs sans créer de nouveaux dysfonctionnements. Ce processus peut inclure des tests unitaires, des revues de code et une analyse approfondie pour améliorer la robustesse et la maintenabilité du projet.

1. Nouvelle version de l’application « INGENEO »

*Comment structurer efficacement le développement collaboratif de la nouvelle version de l’application « INGENEO » pour garantir un suivi optimal et une qualité de livraison ?*

Le développement d’une nouvelle version de l’application « INGENEO » nécessite une organisation minutieuse et une communication constante entre les membres de l’équipe. Afin de faciliter le suivi de l’avancement et de structurer le travail, plusieurs pratiques et outils ont été mis en place : des réunions quotidiennes pour synchroniser les tâches et suivre l’état d’avancement de chacun, des sprints de deux semaines pour favoriser la réactivité et le feedback, ainsi que l’utilisation d’Azure DevOps pour la gestion des tâches et des versions. Le dépôt est organisé avec deux branches principales : une branche « develop » sur laquelle tous les développeurs collaborent, et une branche « main » représentant la version en production. Ce cadre méthodologique permet à l’équipe d’assurer une gestion structurée et d’améliorer la qualité des livraisons finales.

1. Missions confiées
2. Missions pour la Problématique 1 : Migration technique d’un projet client

Pour la problématique de migration technique, les missions avaient pour but de garantir une transition fluide vers une version technologique plus récente sans affecter les fonctionnalités en place. Voici les étapes et missions principales :

1. **Analyse des besoins de migration** :

**Objectif** : Comprendre les spécificités de l’infrastructure actuelle du projet et définir les besoins pour la nouvelle version.

**Tâches** :

* Étudier la configuration actuelle du projet client, ses dépendances et ses particularités (compatibilité, modules spécifiques).

Pour commencer, j’ai étudié la configuration actuelle du projet client en installant le projet initial sur mon ordinateur. J’ai récupéré le projet sur GitHub avec la commande suivante :   
  
***« git clone https://adresse-du-repository.git »***

Je me suis rendu dans le composer.json pour voir les packages d’installés et connaitre les versions utilisées pour chaque dépendances et la version de PHP et de Symfony.

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Dans le fichier composer.json, j’ai pu retrouver les informations dont j’avais besoin pour connaitre les versions actuelles du projet et pouvoir préparer la suite de la migration :   
  
php en version inférieur ou égale à la 5.5.9  
symfony en version 3.1

Et également les librairies utilisées :  
  
doctrine/orm en version 2.5  
doctrine/doctrine-bundle en version 1.6   
doctrine/doctrine-cache-bundle en version 1.2

1. **Préparation de l’environnement de migration** :

**Objectif : Créer un environnement de test représentatif pour effectuer les essais de migration sans impacter l’environnement de production.**

**Tâches :**

* **Configurer un environnement de préproduction similaire à la production.**

Pour configurer un environnement de préproduction qui serait similaire à la production, j’ai demandé en interne aux DevOps avec la confirmation du Directeur Technique pour qui me mette en place un serveur sur lequel je pourrais travailler qui serait en ligne et que chaque modification sur mon projet local soit testée également en production grâce à deux branch git que j’utiliserais pour mon travail. Donc en local une branch « features/ » et une autre branch « main » qui servira quant à elle, a la branch principal du projet en préproduction.

* Mettre en place une base de données de test pour simuler les opérations en production.

J’ai mis en place une base de données de teste avec des DataFxitures pour pouvoir simuler chaques fonctions du projet. Pour ce faire j’ai installé une librairie « fakerphp/faker » qui permet de remplir des éléments aléatoires en base de données. Voici une capture d’écran de comment a été mise en place la base de données :

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

On peut voir la capture d’écran que j’utilise la librairie « faker » par exemple pour remplir le contenu d’une tâche, j’ai utilisé « $faker->text() » qui me génère un texte aléatoire pour le contenu d’une tâche.  
  
Lorsque tout est en place pour remplir chaque tables de la base de données, il ne reste plus qu’à lancer la commande « php bin/console doctrine:fixtures:load ».  
  
Voici le résultat final :   
  
Une image contenant texte, Police, nombre, logiciel

Description générée automatiquement

Les tâches sont bien présentes et on retrouve bien le contenu des tâches avec la librairie « faker ».

* Installer et configurer les dépendances nécessaires, notamment les nouvelles versions des outils ou bibliothèques PHP/Symfony.

Aux vues de toutes les montées en version à faire et de la taille du projet et de ses dépendances, j’ai décidé de partir directement sur un projet vierge avec la version de PHP 8.2 et de Symfony 6.4 et ensuite d’utiliser ensuite d’installer les librairies utilisées avec les versions compatibles.

Une image contenant texte, capture d’écran

Description générée automatiquement

1. **Adaptation du code aux nouvelles versions** :

**Objectif** : Mettre à jour le code pour qu’il soit compatible avec les nouvelles versions des outils et frameworks.

**Tâches** :

* Identifier et modifier les fonctions obsolètes ou incompatibles avec la nouvelle version de Symfony ou des packages utilisés.

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquementUne image contenant texte, capture d’écran, logiciel

Description générée automatiquement  
Sachant que la version Symfony initiale était la version 3.1 les annotations étaient utilisées dans l’ensemble du code mais que sur la version 6.4 de Symfony et de la version 8 de PHP l’utilisation des attribues a été recommandé, j’ai dû modifier les controllers, les entitées pour bien mettre à jour le projet.  
  
  
Voici les captures d’écrans d’une entité du passage des annotations en attribut :  
  
Symfony 3.1 : Symfony 6.4 :

1. **Tests de régression et validation** :

**Objectif** : Vérifier que la migration ne crée pas de nouvelles anomalies et que les fonctionnalités restent intactes.

**Tâches** :

* Effectuer des tests de régression complets pour valider le bon fonctionnement des fonctionnalités principales.

*Mise en place de test unitaire et fonctionnel pour valider le bon fonctionnement*

* Réaliser des tests de performance pour garantir que les temps de réponse restent conformes.

*Capture de test de performance avec le profiler*

1. **Documentation et mise en production** :

**Objectif** : Documenter les étapes de la migration et les configurations pour faciliter la maintenance future.

**Tâches** :

* Rédiger un guide de migration pour les futurs développements et mises à jour.

*Expliquer comment j’ai mis en place la documentation de la migration pour les futurs développeurs*

1. Missions pour la Problématique 2 : Correction des anomalies sur un projet existant

Les missions autour de la correction des anomalies visaient à optimiser le processus de détection et de correction des bugs pour assurer la stabilité et la qualité de service. Voici les étapes de cette mission :

1. **Identification et priorisation des anomalies** :

**Objectif** : Dresser un inventaire des anomalies en fonction de leur criticité pour le produit et l’expérience utilisateur.

**Tâches** :

* + - Effectuer une analyse des retours utilisateurs et des rapports d’erreurs pour identifier les problèmes récurrents.

*Capture d’écran de retour de client sur le fonctionnement et erreurs identifiées – création de ticket*

* + - Classifier les anomalies par niveau de priorité (bloquants, majeurs, mineurs).

*Selon la criticité des anomalies, choisir la bonne catégorie dans le ticket pour les traiter en priorité*

1. **Analyse et reproduction des bugs** :

**Objectif** : Reproduire les erreurs signalées pour mieux comprendre leurs causes.

**Tâches** :

* Utiliser les logs et outils de debug pour identifier les chemins causant des erreurs.

*Trouver des erreurs et les analyser pour identifier le problème.*

1. **Correction des anomalies** :

**Objectif** : Résoudre les problèmes identifiés en apportant des modifications de code tout en évitant les régressions.

**Tâches** :

* Corriger les erreurs identifiées en respectant les bonnes pratiques et les standards de l’équipe.

1. Missions pour la Problématique 3 : Nouvelle version de l’application « INGENEO »
2. **Mise en place et utilisation des réunions quotidiennes** :

**Objectif** : Assurer une communication continue entre les membres de l’équipe pour coordonner l’avancement du projet.

**Tâches** :

* Participer à des réunions de type "stand-up" chaque matin pour partager l’avancement des tâches, identifier les obstacles et coordonner les priorités.
* Récapituler les objectifs de la journée, détailler les tâches en cours et discuter des points bloquants pour permettre une résolution rapide.

1. **Suivi des tâches et gestion des versions via Azure DevOps** :

**Objectif** : Utiliser un outil de gestion de projet pour suivre l’avancement des tâches, assigner les responsabilités et documenter les progrès.

**Tâches** :

* Utiliser Azure DevOps pour créer et assigner des tickets de tâches, en les organisant par ordre de priorité et par phase de développement.

1. **Gestion du dépôt de code et de la stratégie de branches** :

**Objectif** : Garantir une gestion ordonnée du code et des versions pour réduire les risques d’erreurs en production.

**Tâches** :

* Contribuer au code en utilisant deux branches principales sur Azure DevOps : la branche « develop » pour le développement collaboratif, et la branche « main » pour la version de production.

1. Bilan