



Procédure de développement

INTALIO

intalio

Références

Type du document	Procédure de travail
Centre de Production	Intalio France
Client	Commissariat Général au Développement Durable
Projet	NOVAe
Référence	INTALIO_CGDD_Procédure de développement_001

Table des modifications

Historique					
#	Date	Activité	Intervenant	Vérifié par	Validé par
1.0	31/05/2024	Version Initiale	Khaled NABIL	Marwa JEMLI	Samer NABOULSI

Table des matières

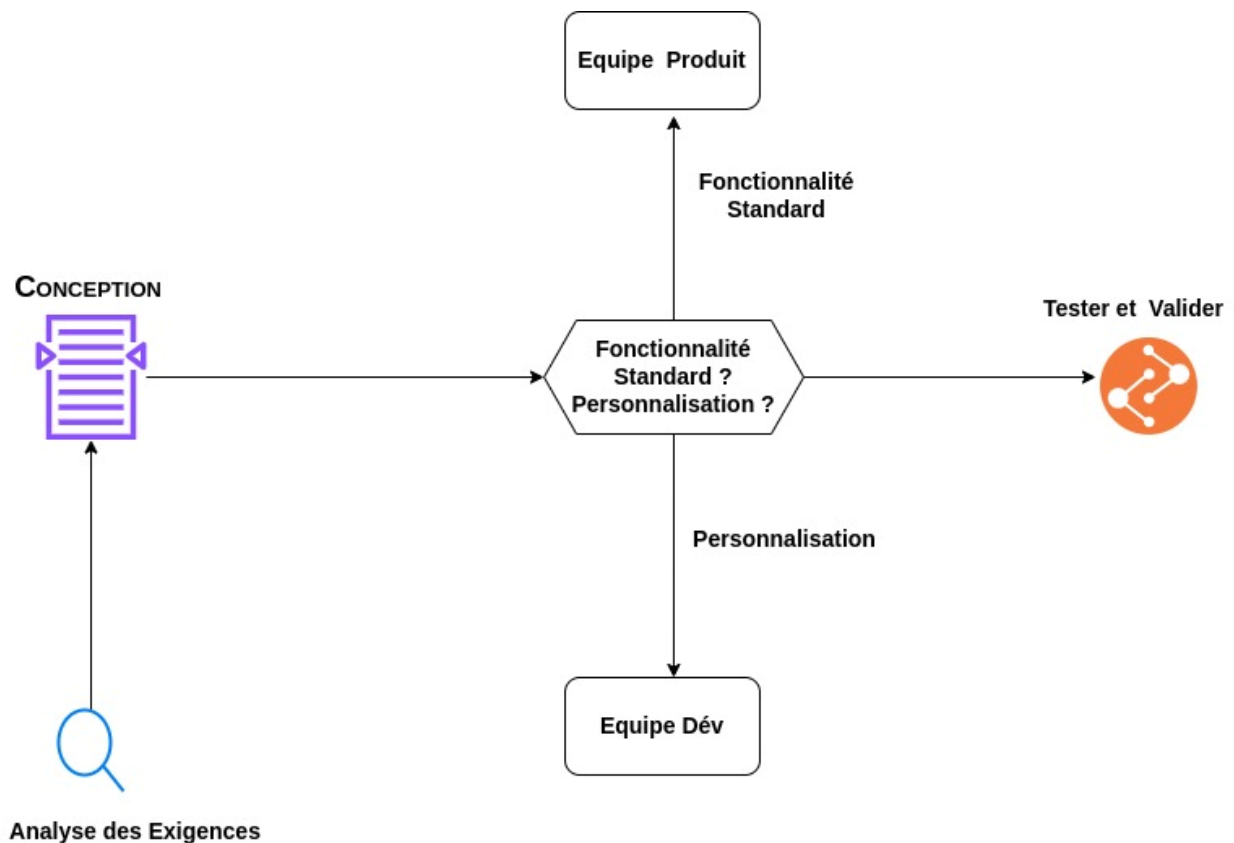
Introduction	4
1 Développement du Produit Standard	5
1.1 Procédure de Mise à Jour de la Version du Produit	5
1.1.1 Publication d'une Nouvelle Version	5
1.1.2 Analyse de la Compatibilité	5
1.1.3 Tests Internes	6
1.1.4 Planification de la Mise à Jour	6
1.1.5 Intégration des développements spécifiques dans la nouvelle version produit	6
1.1.6 Notification aux Clients	6
1.1.7 Tests Post-Mise à Jour	6
2 Personnalisation en Fonction des Exigences du Client	7
3 Cycle de Vie du Développement de l'Application Portail Pétitionnaire pour le Client CGDD	8
3.1 Analyse des Exigences	8
3.2 Conception	8
3.3 Analyse des Impacts et Gestion des Risques	8
3.4 Développement	8
3.5 Tests Internes	8
3.6 Déploiement	9
4 Gestion des Exigences et des Anomalies pour le Produit Standard et les Personnalisations	9
4.1 Analyse des Exigences	9
4.2 Développement et Test	9
4.3 Gestion des Anomalies	9
5 Cycle de Vie des Environnements	9
5.1 Environnement de Développement	10
5.2 Environnement de Test	10
5.3 Environnement de Recette	10
5.4 Environnement de Production	10
6 Cycle de Vie du Développement Backend et Frontend	10
6.1 Analyse des Besoins	10
6.2 Conception	11
6.3 Développement Backend	11
6.4 Développement Frontend	11
6.5 Tests Backend et Frontend	11
6.6 Déploiement	12
6.7 Gestion des Bogues et des Améliorations	12

7	Outils et Technologies Utilisés	12
7.1	Portail Pétitionnaire et Portail des Publications	12
7.2	Extension (Backend)	12
7.3	IAM	13
7.4	CTS	13
7.5	Viewer	13
	Conclusion	13

INTRODUCTION

L'objectif de ce document est de fournir une vue d'ensemble complète de notre processus de développement logiciel. En détaillant les pratiques et les étapes que nous suivons pour gérer les bogues et intégrer de nouvelles fonctionnalités, des améliorations et des corrections, nous visons à garantir la qualité, la fiabilité et la satisfaction des clients. Ce document met également en lumière la manière dont nous adaptons notre approche de développement pour répondre aux besoins uniques de chaque client.

Chaque demande est traitée de manière différente en fonction du type de demande, qu'il s'agisse d'un comportement standard dans le produit, d'une personnalisation nécessaire à un comportement existant du produit ou d'une nouvelle fonctionnalité personnalisée requise.



1 DÉVELOPPEMENT DU PRODUIT STANDARD

Notre produit standard suit un cycle de vie de développement logiciel bien défini pour sa création, son amélioration continue via la mise à disposition de nouvelles versions par produit, et ses mises à jour de sécurité. Les principales étapes incluent :

- **Analyse des Besoins** : Compréhension approfondie des exigences fonctionnelles et non fonctionnelles des client.
- **Conception** : Création d'une architecture logicielle robuste et de spécifications techniques détaillées.
- **Développement** : Utilisation de technologies telles que .NET, Angular, Java et JavaScript pour implémenter les fonctionnalités.
- **Tests** : Validation du produit à travers des tests unitaires, d'intégration et de système pour garantir sa qualité et sa stabilité.
- **Déploiement** : Mise en production après une évaluation rigoureuse et une validation finale.

Mises à Jour : Les mises à jour de sécurité et les nouvelles fonctionnalités sont analysées et intégrées par l'équipe produit, avec une évaluation des impacts potentiels sur les fonctionnalités existantes.

1.1 PROCÉDURE DE MISE À JOUR DE LA VERSION DU PRODUIT

1.1.1 Publication d'une Nouvelle Version

Lorsqu'une nouvelle version du produit est publiée, l'équipe produit fournit un guide et une liste des modifications effectuées, y compris les nouvelles fonctionnalités, les corrections de bugs et les nouvelles APIs mises à disposition.

1.1.2 Analyse de la Compatibilité

L'équipe du projet analyse si cette nouvelle version fonctionnera correctement avec les personnalisations déjà existantes. Cela inclut une évaluation approfondie des impacts possibles sur les personnalisations en place du projet.

1.1.3 Tests Internes

Des tests internes sont effectués pour vérifier le bon fonctionnement de la nouvelle version avec les personnalisations existantes. Cela comprend des tests d'intégration et fonctionnels 

1.1.4 Planification de la Mise à Jour

En collaboration avec le client, la décision de mettre à jour la dernière version disponible est étudiée en fonction :

- Du périmètre de la nouvelle version : la nouvelle version peut contenir des fonctionnalités demandées par le client et également de nouvelles fonctionnalités produit
- De l'impact de la nouvelle version sur les fonctionnalités spécifique et existante chez le client.

Un plan de mise à jour est établi, tenant compte des spécificités de leurs environnements et des éventuelles personnalisations. Les mises à jour peuvent être programmées pour minimiser les interruptions de service.

1.1.5 Intégration des développements spécifiques dans la nouvelle version produit

Cette étape consiste à fusionner les modifications et les développements spécifiques réalisés pour le produit dans la nouvelle version. Cela peut inclure la mise à jour des configurations existantes, l'adaptation des fonctionnalités spécifique ou même la mise à jour des APIs utilisées.

1.1.6 Notification aux Clients

Une fois la décision de mettre à jour la nouvelle version du produit a été actée, les clients sont informés de la disponibilité de la nouvelle version et de la planification de la mise à jour via des notifications par email.

1.1.7 Tests Post-Mise à Jour

Après la mise à jour, des tests post-déploiement sont effectués par l'équipe projet pour garantir que le système fonctionne correctement et que toutes les personnalisations sont intactes.

2 PERSONNALISATION EN FONCTION DES EXIGENCES DU CLIENT

Nous offrons des services de personnalisation pour répondre aux besoins spécifiques de chaque client. Les personnalisations couvrent différents aspects du besoin :

- **Personnalisation des produits INTALIO :** configuration spécifiques, personnalisation graphique, modules spécifiques, modèle de données adapté, etc...
- **Interfaçage avec les systèmes d'information du client :** APIs spécifiques, Web services et connecteurs dédiés, etc...
- **Migration et intégration de données :** Extraction, transformation et intégration des données depuis des sources de données différentes.



Les étapes impliquent :

- **Recueil des Exigences du Client :** Collaboration étroite avec le client pour comprendre ses besoins uniques.
- **Conception de la Personnalisation :** Adaptation de l'architecture et des fonctionnalités pour répondre aux exigences spécifiques du client. 
- **Analyse des Impacts des Personnalisations :** Avant de procéder à la mise à jour de la nouvelle version du produit, une analyse des impacts des personnalisations existantes est réalisée. Cette étape comprend l'évaluation des interactions entre les personnalisations et les modifications apportées dans la nouvelle version. Les résultats de cette analyse sont partagés avec le client pour décider s'il accepte les risques associés à la mise à jour
- **Développement de la Personnalisation :** Utilisation des technologies pertinentes pour implémenter les fonctionnalités personnalisées.
- **Tests et Validation :** Vérification rigoureuse pour garantir que les personnalisations répondent aux attentes du client. 
- **Déploiement Personnalisé :** Intégration des personnalisations dans l'environnement de production après une validation finale.

3 CYCLE DE VIE DU DÉVELOPPEMENT DE L'APPLICATION PORTAIL PÉTITIONNAIRE POUR LE CLIENT CGDD

Pour le client CGDD, nous utilisons des produits standard tels que IAM, CTS et Viewer, ainsi qu'une application spécifique développée sur mesure appelée Portail Pétitionnaire. L'application Portail Pétitionnaire est connectée via un middleware, qui assure l'intégration avec les produits standard. Les étapes du cycle de vie de développement sont décrites ci-dessous.

3.1 ANALYSE DES EXIGENCES

- Compréhension des besoins spécifiques du client.
- Documentation des exigences fonctionnelles et non fonctionnelles.

3.2 CONCEPTION

- Conception de l'architecture de l'application et des spécifications techniques.
- Développement de L'application intermédiaire backend pour assurer une intégration fluide entre le Portail Pétitionnaire et les produits standard.



3.3 ANALYSE DES IMPACTS ET GESTION DES RISQUES

- Évaluation des interactions potentielles entre les nouvelles fonctionnalités de l'application Portail Pétitionnaire, les produits standard existants et l'application intermédiaire backend.
- Partage des risques identifiés avec le client CGDD pour une décision éclairée sur l'acceptation ou l'atténuation de ces risques.

3.4 DÉVELOPPEMENT

- Implémentation des fonctionnalités de l'application en utilisant des technologies telles que .NET, Angular, Java et JavaScript.

3.5 TESTS INTERNES

- Tests unitaires pour valider les composants individuels de l'application.
- Tests d'intégration pour vérifier l'interaction entre l'application et le middleware, ainsi qu'avec les produits standard.
- Tests dans un environnement interne simulant l'environnement client pour s'assurer de la compatibilité et des performances.



3.6 DÉPLOIEMENT

- Déploiement de l'application dans l'environnement de recette pour validation finale.
- Mise en production après validation.

4 GESTION DES EXIGENCES ET DES ANOMALIES POUR LE PRODUIT STANDARD ET LES PERSONNALISATIONS

4.1 ANALYSE DES EXIGENCES

Analyse des nouvelles exigences du client pour déterminer si elles nécessitent une mise à jour produit pour ajouter une nouvelle fonctionnalité ou si elles peuvent être prises en charge par l'équipe de développement en tant que personnalisation.

4.2 DÉVELOPPEMENT ET TEST

- Implémentation des nouvelles fonctionnalités ou des personnalisations.
- Tests unitaires, d'intégration et de système pour valider les changements.
- Tests sur plusieurs environnements (développement, test, recette) pour s'assurer de la stabilité et de la compatibilité.

4.3 GESTION DES ANOMALIES

- Identification et signalement des bogues via un système de suivi.
- Analyse pour déterminer si le bogue est lié au produit standard ou à une personnalisation. Si le bogue est lié au produit, un ticket avec l'équipe produit sera ouvert. Ce ticket contiendra toutes les informations nécessaires à la reproduction et résolution du bug (logs, scénario de test..)
- Correction des bogues et validation des correctifs à travers des tests rigoureux.
- Re-tests pour s'assurer qu'aucun nouveau problème n'a été introduit.

5 CYCLE DE VIE DES ENVIRONNEMENTS

5.1 ENVIRONNEMENT DE DÉVELOPPEMENT

- **Objectif** : Développement initial et tests unitaires.
- **Accès** : Accès restreint aux développeurs.
- **Tests** : Tests unitaires pour valider les composants individuels du logiciel.

5.2 ENVIRONNEMENT DE TEST

- **Objectif** : Tests d'intégration et de système.
- **Accès** : Accès aux testeurs et aux développeurs et ingénieurs DevOps.
- **Tests** : Tests d'intégration pour vérifier l'interaction entre les composants et les systèmes.

5.3 ENVIRONNEMENT DE RECETTE

- **Objectif** : Validation finale avant déploiement en production.
- **Accès** : Accès contrôlé aux parties prenantes clés pour la validation.
- **Tests** : Tests de système pour valider l'ensemble du système dans un environnement proche de la production.



5.4 ENVIRONNEMENT DE PRODUCTION

- **Objectif** : Utilisation en conditions réelles par les utilisateurs finaux.
- **Accès** : Accès limité aux équipes de support pour la maintenance et la surveillance.

6 CYCLE DE VIE DU DÉVELOPPEMENT BACKEND ET FRONTEND



Pour garantir le bon déroulement du développement du backend et du frontend, nous suivons un cycle de vie structuré, comprenant les étapes suivantes :

6.1 ANALYSE DES BESOINS

- Compréhension approfondie des exigences fonctionnelles et non fonctionnelles spécifiques au backend et au frontend.

6.2 CONCEPTION

- Conception de l'architecture du backend et du frontend, y compris la définition des API, des bases de données et des interfaces utilisateur.
- Définition des spécifications techniques détaillées pour orienter le développement.

6.3 DÉVELOPPEMENT BACKEND

- Implémentation des fonctionnalités côté serveur en utilisant des technologies telles que .Net.
- Mise en place des bases de données et des services nécessaires pour prendre en charge les fonctionnalités du système.
- Conception et développement des API pour permettre la communication entre le frontend et le backend.
- Optimisation des performances et de la sécurité des applications serveur.
- Intégration des services tiers et des API externes lorsque nécessaire.
- Maintenance et mise à jour du code backend pour s'assurer de la compatibilité et des améliorations continues.

6.4 Développement FRONTEND

Création des interfaces utilisateur en utilisant des technologies telles que Angular, HTML, CSS Javascript.

- Intégration des conceptions graphiques et des fonctionnalités interactives pour une expérience utilisateur optimale.
- Utilisation de la bibliothèque spécifique DSFR (Design Système de l'État Français) pour la gestion du CSS, assurant une cohérence visuelle et une conformité aux standards gouvernementaux.

6.5 TESTS BACKEND ET FRONTEND

- Tests unitaires pour valider les composants individuels du backend et du frontend.

- Tests d'intégration pour vérifier l'interaction entre le backend et le frontend, ainsi qu'avec d'autres composants du système.
- Tests de système pour évaluer les performances globales du système, y compris la communication entre le backend et le frontend.

6.6 DÉPLOIEMENT

- Déploiement du backend et du frontend sur des environnements de test et de recette pour validation.
- Mise en production du backend et du frontend après validation finale.

6.7 GESTION DES BOGUES ET DES AMÉLIORATIONS

- Identification et correction des bogues dans le backend et le frontend, suivie de tests rigoureux pour s'assurer de la résolution.
- Intégration des améliorations et des nouvelles fonctionnalités en fonction des besoins des utilisateurs et des exigences du système.

En intégrant ce cycle de vie du développement backend et frontend, nous nous assurons de la qualité, de la fiabilité et de la performance de l'ensemble du système logiciel.

7 OUTILS ET TECHNOLOGIES UTILISÉS

Nous utilisons une gamme d'outils et de technologies modernes pour soutenir notre processus de développement, notamment comme suit.

7.1 PORTAIL PÉTITIONNAIRE ET PORTAIL DES PUBLICATIONS

- **Technologie :** Angular
- **IDE :** Visual Studio Code
- **Système de Gestion de Versions :** Git

7.2 EXTENSION (BACKEND)

- **Technologie :** .NET

- **IDE** : Visual Studio
- **Système de Gestion de Versions** : Git

7.3 IAM

- **Technologie** : .NET
- **IDE** : Visual Studio
- **Système de Gestion de Versions** : Git

7.4 CTS

- **Technologie** : .NET
- **IDE** : Visual Studio
- **Système de Gestion de Versions** : Git

7.5 VIEWER

- **Technologie** : Java
- **IDE** : IntelliJ IDEA
- **Système de Gestion de Versions** : Git

CONCLUSION

En suivant ces pratiques et en utilisant ces technologies de pointe, nous sommes en mesure de fournir des solutions logicielles de haute qualité qui répondent aux besoins variés de nos clients, tout en garantissant une expérience utilisateur exceptionnelle.

Cette version améliorée du document met en lumière notre engagement envers la qualité, l'innovation et la satisfaction client, tout en fournissant une description détaillée de notre processus de développement et des technologies que nous utilisons.