Solution d'intelligence artificielle pour la documentation du code source de l'outil MM-AET

NewD

Soumission: 11 novembre 2024

Informations de contact:
Nom du représentant: Mounir SHAL
Email: Mounir.shal@uni.lu
Téléphone: 00352691130192
Adresse du siège: 104 Rue de mersch L8181 Kopstal

Lettre d'Introduction

NewD est un pionnier en intelligence artificielle, spécialisé dans la transformation numérique et l'optimisation de systèmes complexes. Nous avons pour mission d'automatiser les processus critiques en utilisant des technologies de pointe pour maximiser la performance et la précision.

Notre solution vise à répondre aux besoins du Ministère de l'Économie en automatisant la documentation technique et fonctionnelle du code legacy de l'outil MM-AET, garantissant une documentation de haute qualité et facilitant les évolutions futures.

Nous nous engageons à fournir une solution innovante et conforme aux normes de documentation internationales, avec un accent particulier sur la robustesse et le respect des délais et spécifications techniques.

1 Résumé Exécutif

1.1 Compréhension du Besoin

Le Ministère de l'Économie a exprimé un besoin urgent pour la documentation exhaustive du code Java 8 de l'application MM-AET, un système crucial pour la gestion des autorisations et aides financières. En raison de l'absence de documentation actuelle, l'évolution du système est freinée par des contraintes de compréhension.

1.2 Solution Proposée

NewD propose une solution fondée sur une intelligence artificielle avancée capable d'analyser le code de manière syntaxique et sémantique. Notre approche génère des descriptions détaillées de classes et fonctions, produit des diagrammes UML, et fournit des rapports de qualité et complexité avec une proposition d'amélioration. Notre solution permet une documentation automatisée et compréhensive, facilitant la maintenance et la gestion du code, tout en réduisant les coûts et le temps de développement pour les futures évolutions.

2 Présentation de NewD

2.1 Expérience

NewD possède une expertise éprouvée en intelligence artificielle, particulièrement dans l'analyse et la documentation de systèmes complexes, incluant des applications legacy. Notre savoir-faire s'étend à la fois aux entreprises privées et aux administrations publiques, où nous avons développé des solutions sur mesure pour documenter, optimiser, et moderniser des infrastructures logicielles. Grâce à notre maîtrise des technologies IA, nous sommes en mesure de créer et d'utiliser des modèles avancés qui répondent précisément

aux problématiques spécifiques de nos clients, améliorant ainsi l'efficacité, la sécurité, et la maintenabilité des systèmes documentés.

2.2 Certifications et Références

NewD s'engage à la fiabilité et à la précision de ses solutions, appuyé par un solide historique de collaborations avec des institutions de renom. Nos projets antérieurs incluent des initiatives de documentation et d'optimisation de code pour des organisations de grande envergure, dans le respect des exigences réglementaires et des meilleures pratiques du secteur. Ces expériences confirment notre capacité à fournir des services de haute qualité et adaptés aux standards industriels.

2.3 Expertise de l'Équipe

Notre équipe est composée de professionnels expérimentés en rétro-ingénierie, analyse de code et intelligence artificielle. Avec une expertise poussée dans la documentation de systèmes développés en Java, nos spécialistes sont qualifiés pour relever les défis complexes inhérents aux grands systèmes informatiques. Chacun de nos membres apporte une expérience et des compétences ciblées qui assurent la rigueur et la pertinence de notre travail. Vous trouverez les CVs de l'équipe, détaillant les qualifications et projets antérieurs, en annexe.

3 Analyse des Besoins et Compréhension de l'Appel d'Offre

3.1 Exigences Clés

La mission consiste à documenter un code source de 700 000 lignes en Java, avec des attentes spécifiques en matière de qualité et de structuration de la documentation. Les principales exigences incluent :

- **Production de Documentation Complète** : Génération de documentation technique couvrant la structure, les fonctionnalités et les flux de contrôle du code.
- **Création de Diagrammes UML** : Modélisation des relations entre classes, des processus et des modules à l'aide de diagrammes UML clairs et structurés, facilitant ainsi une compréhension visuelle de l'architecture.
- **Analyse de la Qualité et de la Complexité du Code** : Évaluation des normes de qualité du code, détection des vulnérabilités et analyse de la complexité pour identifier des points d'optimisation.

- **Recommandations d'Optimisation** : Propositions de modifications et d'améliorations pour renforcer la maintenabilité, la performance et la sécurité du code.

3.2 Objectifs

Notre objectif est de fournir une documentation complète et intelligible, qui :

- **Clarifie les Flux de Contrôle et Dépendances** : Détaille les enchaînements logiques et les interactions entre modules pour une compréhension approfondie de l'architecture de l'application.
- **Facilite la Maintenance et les Mises à Jour** : Enrichit le code de commentaires explicatifs détaillant les fonctionnalités principales, les classes, et les méthodes, pour permettre une maintenance plus fluide et des interventions rapides.
- **Optimise la Qualité du Code** : Identifie et corrige les vulnérabilités et les zones d'amélioration potentielles, tout en veillant à la conformité avec les bonnes pratiques de développement.
- **Améliore la Clarté et l'Accessibilité** : Fournit des supports visuels et des explications claires pour aider les équipes techniques à saisir rapidement les aspects essentiels de l'application, garantissant une collaboration efficace et un transfert de connaissances optimisé.

4 Solution Proposée

4.1 Objectifs et Raisonnement de la Solution

L'objectif de cette solution est de fournir une documentation complète, intelligente et facile à maintenir pour le code Java de l'application MM-AET. En combinant un modèle d'intelligence artificielle (IA) sur mesure avec des outils de documentation éprouvés, cette approche maximise l'efficacité et la rapidité de la documentation tout en apportant une forte valeur ajoutée au ministère de l'Économie. La documentation générée sera intuitive, évolutive et accessible, facilitant les mises à jour futures et optimisant la qualité du code.

4.2 Structure de la Solution et Valeur Ajoutée

La solution repose sur trois composants principaux qui interagiront dans un même logiciel centralisé :

- **Modèle IA Spécialisé :** Ce modèle, spécifiquement entraîné pour l'analyse de code Java (Basé sur le modéle préentrainé de **Codex, GPT et DeepCode(Snyk)** , est capable de :

  - *Analyser le code* de manière syntaxique et sémantique pour générer une documentation précise et contextuelle.
  - *Suivre les flux de contrôle* et les dépendances du code, assurant une traçabilité complète.
  - *Proposer des suggestions d'amélioration* pour optimiser la qualité et la maintenabilité du code.

- **Intégration d'Outils de Documentation Performants :**
  - *Javadoc* pour générer automatiquement la documentation des classes et méthodes Java.
  - *PlantUML* pour créer des diagrammes UML illustrant la structure et les interactions au sein du code.
  - *SonarQube* pour analyser la qualité du code, détecter les vulnérabilités et fournir des recommandations automatiques.
  - *Doxygen* pour une documentation multilingue enrichissant le contenu pour des équipes internationales.

- **Module de Documentation Dynamique :** Basé sur l'IA et les outils ci-dessus, ce module produit une documentation centralisée et évolutive, comprenant :
  - *Annotations Automatiques* dans le code pour une compréhension rapide des fonctions, classes et variables.
  - *Diagrammes UML Automatisés* pour visualiser les relations entre modules et composants.
  - *Glossaire Technique* pour clarifier les termes et concepts techniques.

4.3 Approche Détaillée par Étapes

4.3.1 Étude Préliminaire et Analyse des Besoins :

- Analyse des spécifications des technologies utilisées (Struts, JSP, jQuery, Vue.js) et des bases de données (DB2).
- Planification de l'intégration des outils de documentation et des exigences du modèle IA pour répondre aux besoins du projet.

4.3.2 Développement du Modèle IA et Intégration des Outils :

- *Entraînement et Adaptation du Modèle* : Utilisation de techniques NLP pour interpréter le code Java et générer automatiquement des suggestions de documentation et d'optimisation.

- *Implémentation des Outils* : Mise en place de Javadoc, PlantUML, SonarQube et Doxygen pour documenter, analyser et visualiser le code de façon efficace.

4.3.3 Documentation Automatisée et Enrichissement :

- *Génération de Commentaires Automatiques* : Insertion automatique de commentaires explicatifs sur les fonctions, classes et variables.
- *Création de Diagrammes UML et Architecturaux* : Visualisation de l'architecture du code pour un meilleur suivi.
- *Analyse de Qualité avec SonarQube* : Identification des vulnérabilités et suggestions d'optimisation basées sur des standards de qualité.

4.3.4 Rapport de Qualité et Suggestions d'Amélioration :

- *Évaluation de la Complexité et de la Qualité du Code* avec SonarQube, en détectant les redondances et en proposant des opportunités d'optimisation via le modèle IA.

4.4 Outils et Algorithmes Utilisés

Les principaux outils et technologies intégrés dans la solution sont :

- **Modèle IA Sur Mesure** : Entraîné spécifiquement pour l'analyse de code Java, il détecte les flux de contrôle et propose des améliorations.
- **Javadoc** : Génère la documentation des classes et méthodes.
- **PlantUML** : Crée des diagrammes UML.
- **SonarQube** : Analyse la qualité du code et identifie les vulnérabilités.
- **Doxygen** : Produit une documentation multilingue pour des rapports plus complets.

4.5 Valeur Ajoutée et Bénéfices

Cette solution offre une documentation claire, précise et évolutive, avec les bénéfices suivants :

- **Documentation Précise et Maintenable** : Les annotations et rapports facilitent les évolutions futures de l'application.
- **Optimisation de la Qualité** : Grâce à l'analyse combinée de l'IA et de SonarQube, des suggestions d'améliorations concrètes sont proposées.

- **Efficacité et Gain de Temps** : La génération automatique de documentation libère du temps pour les développeurs, qui peuvent se concentrer sur d'autres tâches.

4.6 Livrables

Les livrables de cette solution incluent :

- **Documentation Technique Complète** : Explications détaillées des fonctions, classes, et modules, avec diagrammes UML pour visualiser l'architecture.
- **Annotations Directes dans le Code Source** : Ajout de commentaires explicatifs et de recommandations directement dans le code.
- **Rapport de Qualité et Suggestions d'Amélioration** : Évaluation de la complexité et de la qualité du code avec recommandations spécifiques.
- **Glossaire Technique** : Définitions des termes et concepts utilisés dans l'outil MM-AET.
- **Documentation des Méthodologies et Outils** : Explications sur les méthodologies et outils utilisés pour assurer la pérennité de la solution.

4.7 Synthése

En intégrant un modèle IA spécialisé et des outils de documentation performants dans un un écosystème cohérent et performant pour la gestion du code et de la documentation. Cette solution fournit une documentation optimisée, complète et évolutive, en phase avec les exigences de qualité et de maintenabilité du ministère de l'Économie. Notre modèle, conçu spécifiquement pour analyser le code Java, générera des suggestions pertinentes de documentation et d'optimisation. Il sera à la fois précis et parfaitement adapté aux besoins uniques du projet. Forts de notre expertise en prompt engineering, nous savons exploiter pleinement chaque outil pour maximiser leur efficacité, garantissant ainsi une solution fluide qui génère automatiquement des documentations claires, améliore la qualité du code et offre une visibilité complète sur l'architecture et les flux de travail, tout en renforçant l'efficacité globale du projet.

5 Démarche de Réalisation et Plan de Travail

Phase	Durée Estimée	Activités Clés	Livrables Principaux
Phase 1 : Préparation	1 semaine	Analyse des spécifications, collecte des besoins	Cahier des charges détaillé
Phase 2 : Développement IA et intégration des outils	3 semaines	Développement du modèle IA, intégration des outils	Modèle IA fonctionnel, outils intégrés
Phase 3 : Documentation automatisée	2 semaines	Génération des commentaires, création des diagrammes UML	Documentation générée, diagrammes UML
Phase 4 : Validation et tests	2 semaines	Tests de la solution, validation de la qualité du code	Rapport de tests, validation des livrables
Phase 5 : Déploiement et formation	2 semaines	Déploiement, formation des équipes techniques	Solution déployée, support de formation
Phase 6 : Support et Maintenance	6 mois	Support post-livraison, mise à jour continue de la documentation	Documentation mise à jour

Table 1: Plan de travail ajusté sur une période de 3 mois

Délai Total : 3 mois

5.1 Suivi

- Réunions hebdomadaires de suivi du projet
- Rapports mensuels pour assurer une visibilité sur l'avancement du projet

6 Budget et Tarification

Le budget estimé pour la documentation du code source de l'application MM-AET est calculé sur une base de tarification par tâche. Les tarifs appliqués tiennent compte de la nature des activités spécifiques et des compétences nécessaires pour chaque étape du projet.

Tâche	Rôle	Tarif	Jours
1. Gestion de Projet			
Planification et Suivi	Chef de Projet	1500 €/j	20
Réunion de Coordination	Chef de Projet	1500 €/j	10
2. Analyse de Code et Documentation			
Analyse Code Java	Analyste Rétro-Ingénierie	1000 €/j	25
Documentation Code	Analyste Rétro-Ingénierie	1000 €/j	15
3. IA pour Analyse Syntaxique et Sémantique			
Développement Algorithmes IA	Développeur IA	1000 €/j	20
Intégration Modèles IA	Développeur IA	1000 €/j	15
Test et Validation des Modèles	Développeur IA	1000 €/j	10
4. Création et Validation de Diagrammes UML			
Modélisation UML	Consultant UML	1000 €/j	15
Révision et Validation Diagrammes	Consultant UML	1000 €/j	10
5. Validation et Contrôle Qualité			
Vérification Documentation	Analyste Rétro-Ingénierie	1000 €/j	10
Tests Qualité IA	Développeur IA	1000 €/j	5
Revue Finale et Approbation	Chef de Projet	1500 €/j	5
Total Estimé			
			150 jours

Offre Forfaitaire : Estimation totale à 210,000 €, avec facturation selon les jours effectivement prestés.

7 Engagement Qualité et Suivi

7.1 Assurance Qualité

Nous assurons un processus de vérification continue tout au long du projet afin de garantir la précision, la conformité et la qualité des livrables. Chaque étape du processus de documentation est rigoureusement contrôlée pour répondre aux normes les plus strictes et aux exigences spécifiées, assurant ainsi une documentation fiable et de haute qualité.

7.2 Plan de Tests et Validation

Un plan de tests détaillé est mis en place à chaque étape du projet pour valider la fiabilité et la cohérence de la documentation générée. Cela inclut des vérifications approfondies des algorithmes IA, des diagrammes UML, et des annotations, ainsi que des tests de qualité sur le code pour garantir une documentation précise et complète. Nous adoptons une

approche itérative, permettant des ajustements en fonction des retours et des résultats des tests.

7.3 Support Post-Livraison

Nous offrons un support post-livraison de 6 mois, durant lequel nous assurons la maintenance et les mises à jour régulières de la documentation en fonction des évolutions du code et des retours d'expérience. Cette période inclut une assistance pour résoudre toute question technique, apporter des modifications nécessaires et garantir que la documentation reste à jour et conforme aux exigences du projet.

Conclusion

En conclusion, notre solution hybride, alliant intelligence artificielle et outils éprouvés, permet de créer une documentation complète et de haute qualité pour l'application MM-AET. Grâce à l'intégration de modèles IA spécialisés et d'outils comme Javadoc, PlantUML, SonarQube et Doxygen, nous assurons une documentation précise, évolutive et facile à maintenir.

Nous garantissons la qualité de la documentation avec un processus de vérification continue, un plan de tests rigoureux à chaque étape et un suivi post-livraison de 6 mois pour les mises à jour et la maintenance. Cela permet de répondre aux besoins de documentation évolutive du projet, tout en garantissant la fiabilité, la traçabilité et l'optimisation du code à long terme.

En adoptant cette solution, nous offrons une gestion efficace du projet, une meilleure compréhension du code pour les équipes techniques, et une base solide pour la pérennité et l'amélioration continue de l'application MM-AET.