

Plan d'Assurance Qualité du Projet

Coach Virtuel pour formation interactive avec IA généraliste

État : Première version

Rédigé par : BAANNI Zaineb, IDRISSE Mohamed,
OUGAS Fadoua

Version : 1.0

Date de dernière mise à jour : 15 décembre 2025

Diffusion : Équipe Technique, Encadrants,
Superviseurs

Validation

Organisme	Fonction	Nom	Date
EMSI Marrakech	Encadrant	D.ESSABAR Driss	17/12/2025

Historique du document

Date	Nature de la modification	Réalisé par	Version
07/12/2025	Début de la rédaction de ce document	Équipe projet	0.1.0
13/12/2025	Finalisation de l'ébauche du document	Équipe projet	0.2.0
15/12/2025	Première version du document	Équipe projet	1.0

Table des matières

1	Objet du document	1
1.1	Introduction	1
1.2	Objectifs du PAQ	2
1.3	Cadre d'application	2
1.4	Technologies utilisées	3
1.5	Parties prenantes et supervision	3
1.6	Terminologie et abréviations	4
2	Organisation des ressources humaines	5
2.1	Rôle des différents intervenants	5
2.2	Outils utilisés pour la coordination	6
3	Qualité au niveau du processus de développement	7
3.1	Méthodologie adoptée : SCRUM	7
3.2	Phases du projet et livrables associés	7
3.2.1	Phase 1 : Analyse de besoins et conception	7
3.2.2	Phase 2 : Développement Back-End et Front-End (Phase 1)	8
3.2.3	Phase 3 : Développement Mobile, IA et Intégration	8
3.2.4	Phase 4 : Tests, Optimisation et Documentation	9
3.3	Gestion des sprints et suivi des tâches	9
3.3.1	Sprint 1 : Analyse de besoins et conception	9
3.3.2	Sprint 2 : Développement Back-End et Front-End (Phase 1)	10
3.3.3	Sprint 3 : Développement Mobile, IA et Intégration	11
3.3.4	Sprint 4 : Tests, Optimisation et Documentation	12

3.4	Suivi des tâches et progrès	14
4	Documentation	15
4.1	Règles de gestion et de structuration des documents	15
4.2	Documents de gestion de projet	16
4.3	Documentation technique et livrables de réalisation	17
5	Gestion des modifications	20
5.1	Origine et typologie des modifications	20
5.1.1	Origine des modifications	20
5.1.2	Typologie des modifications	21
5.2	Procédures pour l’approbation et l’implémentation des changements	23
6	Méthodes, outils et normes	26
6.1	Méthodes et méthodologie retenues	26
6.1.1	Méthodologie SCRUM	26
6.1.2	Gestion des changements	26
6.2	Règles et normes devant être appliquées	27
6.2.1	Normes de développement	27
6.2.2	Normes de sécurité	27
6.2.3	Normes de performance	28
6.2.4	Normes de gestion documentaire	29
6.2.5	Suivi des versions et des modifications	29
7	Gestion des risques	31
7.1	Identification des risques projet	31
7.1.1	Retards dans le développement	31
7.1.2	Problèmes de performance	31
7.1.3	Problèmes de compatibilité entre technologies	32
7.1.4	Risques liés à l’intégration IA	32
7.1.5	Changements de scope (scope creep)	32
7.1.6	Risques liés à la gestion des équipes	33

7.2	Plans d'atténuation des risques	33
7.2.1	Retards dans le développement	33
7.2.2	Problèmes de performance	34
7.2.3	Problèmes de compatibilité entre technologies	34
7.2.4	Risques liés à l'intégration IA	35
7.2.5	Changements de scope (scope creep)	35
7.2.6	Risques liés à la gestion des équipes	35
7.3	Suivi des risques	36
7.3.1	Suivi via Azure DevOps	36
7.3.2	Réévaluation périodique des risques	36
7.3.3	Reporting des risques	37

Conclusion	38
-------------------	-----------

Partie 1

Objet du document

1.1 Introduction

Le Plan d'Assurance Qualité (PAQ) du projet “**Coach virtuel pour formation professionnelle interactive avec IA générative**” définit les processus, méthodes et responsabilités nécessaires pour garantir la qualité du développement, de la mise en œuvre et de la maintenance de la plateforme. Ce document sert de référence pour toutes les parties prenantes et permet d'assurer la cohérence, la fiabilité et la traçabilité des livrables tout au long du cycle de vie du projet.

- **But principal** : Garantir la qualité technique, pédagogique et fonctionnelle du coach virtuel IA.
- **Intégration de la qualité** : La qualité est appliquée à toutes les étapes : conception, développement, tests, déploiement et suivi post-livraison.
- **Référence académique et professionnelle** : Le PAQ s'appuie sur les normes reconnues en ingénierie logicielle (ISO 9001, bonnes pratiques Scrum, CI/CD) et sur les standards pédagogiques applicables aux plateformes d'apprentissage interactives.
- **Amélioration continue** : Les indicateurs de performance et les revues périodiques permettent de détecter les écarts et de proposer des actions correctives.

1.2 Objectifs du PAQ

L'objectif principal de ce Plan d'Assurance Qualité Projet (PAQP) est de définir et de mettre en œuvre des stratégies et des pratiques rigoureuses pour garantir la qualité de la Plateforme de Formation Interactive par IA Générative (Coach Virtuel) à chaque étape de son cycle de vie, de la conception à la mise en production, en passant par les phases de test et de suivi post livraison.

Ce plan vise à atteindre les objectifs suivants :

- **Une couverture exhaustive des tests :** Toutes les fonctionnalités de l'application seront soumises à des tests rigoureux pour vérifier leur bon fonctionnement dans des conditions réelles et garantir qu'elles répondent aux exigences fonctionnelles (interactions avec le coach virtuel, gestion des parcours d'apprentissage) et non fonctionnelles (fiabilité, performance).
- **Des livrables conformes aux exigences de performance et de sécurité :** L'application devra offrir une expérience utilisateur fluide, un temps de réponse rapide et une sécurité renforcée pour protéger les données des utilisateurs et des apprenants.
- **Une gestion proactive des risques et des modifications :** Les risques identifiés seront anticipés et traités à l'aide de plans d'atténuation précis. Toute modification du projet sera gérée de manière structurée pour éviter toute dérive des délais ou des fonctionnalités.
- **Une communication fluide et continue au sein de l'équipe projet :** Des processus de communication efficaces favoriseront la collaboration entre les équipes de développement, de test, et de gestion de projet, garantissant que tous les membres soient alignés sur les objectifs et les attentes du projet.

Ce PAQP constitue le cadre de référence pour assurer le maintien de la qualité tout au long du développement de l'application et garantir que le produit final répond aux attentes des utilisateurs et des parties prenantes.

1.3 Cadre d'application

Le projet vise à développer une plateforme de formation professionnelle interactive intégrant un coach virtuel IA.

- **Objectif principal :** Offrir une interface intuitive et complète pour les utilisateurs finaux, avec des recommandations personnalisées et un suivi pédagogique automatisé.
- **Utilisateurs cibles :**

- **Apprenants** : Accès aux parcours de formation, consultation de contenus interactifs, suivi des progrès, interaction avec le coach virtuel.
- **Formateurs / Experts pédagogiques** : Supervision des parcours, validation des contenus et suivi des performances des apprenants.
- **Administrateurs** : Gestion des utilisateurs, supervision de la plateforme et analyse statistique pour optimisation continue.
- **Périmètre fonctionnel** :
 - Pour les apprenants : accès aux modules, suivi et communication avec le coach.
 - Pour les formateurs : gestion des contenus, suivi des apprenants, analyse des indicateurs.
 - Pour les administrateurs : gestion des comptes, supervision globale, rapports détaillés.

1.4 Technologies utilisées

- **Backend** : Développé avec Spring Boot, qui fournit une structure robuste et modulaire pour la création d'APIs REST sécurisées, facilitant ainsi l'interaction entre le client et la base de données.
- **Frontend Web** : Développé avec Angular, une plateforme moderne et performante pour créer des interfaces utilisateur dynamiques et interactives adaptées à différents appareils.
- **Mobile** : Développé avec Flutter, une technologie multiplateforme qui permet de développer une application mobile performante et homogène pour les systèmes Android et iOS.
- **Base de données** : Stockage des données avec PostgreSQL, un système de gestion de base de données relationnel fiable et performant, idéal pour gérer efficacement les informations sur les utilisateurs, les parcours de formation et les interactions avec le coach virtuel.
- **Modèle IA** : Ollama pour la génération de contenu interactif et l'assistance personnalisée aux apprenants.

1.5 Parties prenantes et supervision

Le projet est supervisé par Mme SBAI Hanae et Mme BOUSQAOU Halima, et sera évalué par M. ESSABBAR Driss. Ces parties prenantes assureront la qualité du développement et la conformité du projet tout au long de sa mise en œuvre.

1.6 Terminologie et abréviations

- **PAQ (Plan d'Assurance Qualité)** : Document définissant les processus, outils et standards adoptés pour garantir la qualité du projet à chaque étape de son cycle de vie.
- **PAQP (Plan d'Assurance Qualité Projet)** : Plan spécifique au projet définissant les stratégies et pratiques pour garantir la qualité de la plateforme.
- **CDP (Chef de Projet)** : Responsable de la coordination globale, de la gestion des ressources, et de la livraison des objectifs du projet.
- **RQ (Responsable Qualité)** : Personne en charge de superviser les tests et de valider la conformité des livrables avec les exigences du projet.
- **SCRUM** : Méthodologie Agile structurée en sprints, permettant d'organiser et de planifier le développement itératif et collaboratif du projet.
- **API** : Interface de Programmation Applicative, facilitant la communication entre le backend (Spring Boot) et le frontend (Angular).
- **REST** : Architectural style pour les services web utilisant les méthodes HTTP (GET, POST, PUT, DELETE).
- **CRUD** : Opérations fondamentales sur les données dans une base de données : Create, Read, Update, Delete.
- **MVC (Model-View-Controller)** : Pattern architectural séparant les données, la présentation et la logique métier.
- **SQL** : Langage de requête structuré pour interroger et manipuler les bases de données.
- **Redis** : Système de stockage clé-valeur en mémoire utilisé pour la mise en cache.

Partie 2

Organisation des ressources humaines

2.1 Rôle des différents intervenants

- **BAANNI Zaineb** : Chef de l'équipe et Développeuse Front-End Web avec Angular. Responsable de la gestion de l'équipe, de la planification des sprints, et de la répartition des tâches entre les membres de l'équipe. Elle supervise également la création de la documentation technique et fonctionnelle. Elle est également responsable de l'intégration Back-End et Front-End en utilisant des outils comme Selenium pour effectuer des tests d'intégration et garantir la cohérence entre les différentes parties de l'application.
- **OUGAS Fadoua** : Développeuse Back-End (Spring Boot). En charge de la conception et de la mise en œuvre des API REST, ainsi que de l'optimisation des performances du back-end. Elle est également responsable des tests de qualité, notamment l'intégration de SonarQube et JMeter pour l'analyse des performances et la gestion des tests d'intégration avec Postman et JUnit.
- **IDRISSI Mohamed** : Développeur Mobile (Flutter). Développe et optimise l'application mobile, en veillant à l'amélioration de la performance, de l'UI/UX, ainsi que de la version Android de l'application. Il utilise également JMeter pour effectuer des tests de performance sur l'application mobile. Il est également responsable de la partie IA du projet.

2.2 Outils utilisés pour la coordination

- **Azure DevOps** : Plateforme de gestion de projet pour la planification des sprints, le suivi des tâches, la gestion du backlog et la génération de rapports d'avancement.
- **GitHub** : Plateforme de gestion de versions basée sur Git pour le contrôle de version du code source, la gestion des branches, des commits et des revues de code (pull requests).
- **Google Meet** : Outil de visioconférence et de communication en temps réel pour les réunions techniques et la coordination d'équipe.
- **WhatsApp** : Application de messagerie pour la coordination quotidienne de l'équipe, les notifications urgentes et les échanges informels.

Partie 3

Qualité au niveau du processus de développement

3.1 Méthodologie adoptée : SCRUM

Chaque sprint dure deux semaines et se termine par un livrable évalué pour ajustement et amélioration continue.

- **Sprints** : Planification initiale, suivi des tâches et rétrospective.
- **Équipe SCRUM** : Développeurs, Product Owner et SCRUM Master.

3.2 Phases du projet et livrables associés

3.2.1 Phase 1 : Analyse de besoins et conception

Période : 23 octobre - 5 novembre 2025 (14 jours)

Objectifs :

- Comprendre les besoins fonctionnels et techniques du projet
- Concevoir l'architecture globale du système (Front-End, Back-End, Mobile, IA)
- Préparer les spécifications techniques détaillées
- Mettre en place l'environnement de développement
- Créer le backlog produit priorisé

Livrables :

- Cahier des charges fonctionnel complet
- Architecture technique complète (diagrammes UML)
- Maquettes UI/UX pour toutes les interfaces
- Schéma de base de données (modèle entité-relation)
- Spécifications API REST détaillées
- Backlog produit priorisé avec User Stories
- Environnement de développement configuré (GitHub, outils de collaboration)
- Planification détaillée des sprints

3.2.2 Phase 2 : Développement Back-End et Front-End (Phase 1)

Période : 6 novembre - 19 novembre 2025 (14 jours)

Objectifs :

- Développer les fonctionnalités core du Back-End (Spring Boot)
- Développer les interfaces principales du Front-End (Angular)
- Mettre en place l'authentification et l'autorisation (JWT)
- Intégrer les premières API entre Front-End et Back-End
- Initialiser l'application mobile (Flutter) avec les écrans de base

Livrables :

- Back-End fonctionnel avec authentification JWT
- Front-End avec authentification et interfaces principales (Login, Signup, Dashboard, Profil)
- Application mobile avec écrans de base (connexion, inscription, accueil)
- Intégration Front-Back opérationnelle
- Tests unitaires Back-End (JUnit)
- Documentation technique des APIs (Swagger/OpenAPI)
- Configuration SonarQube pour l'analyse de code
- Tests de performance baseline avec JMeter

3.2.3 Phase 3 : Développement Mobile, IA et Intégration

Période : 20 novembre - 3 décembre 2025 (14 jours)

Objectifs :

- Finaliser le développement de l'application mobile complète
- Intégrer le modèle IA (Ollama) dans tous les composants
- Compléter les fonctionnalités avancées Front-End et Back-End
- Assurer l'intégration complète entre tous les composants
- Optimiser les performances et la qualité du code

Livrables :

- Application mobile complète et fonctionnelle (tous les écrans utilisateur)
- Intégration IA opérationnelle (Front, Back, Mobile) - Chat IA, recommandations personnalisées
- Toutes les fonctionnalités principales implémentées (Espace Formateur, Espace Admin, gestion des contenus)
- Intégration complète entre tous les composants
- Tests de performance effectués (JMeter)
- Tests d'intégration avec Selenium (flux principaux)
- Tests unitaires complets (couverture > 80%)
- Application prête pour les tests finaux

3.2.4 Phase 4 : Tests, Optimisation et Documentation

Période : 4 décembre - 16 décembre 2025 (13 jours)

Objectifs :

- Effectuer des tests complets de l'application (E2E, intégration, performance)
- Corriger tous les bugs identifiés
- Optimiser les performances finales
- Rédiger la documentation complète (technique et fonctionnelle)
- Préparer la livraison finale du projet

Livrables :

- Application complètement testée et validée
- Documentation technique complète (architecture, APIs, déploiement)
- Documentation fonctionnelle complète (Manuel utilisateur, Guide formateur, Guide administrateur, FAQ)
- Application optimisée et prête pour la production
- Rapport de tests complet (Selenium, JMeter, SonarQube, Postman, JUnit)
- Guide d'installation et de déploiement
- Présentation finale du projet
- Rapport final avec synthèse des objectifs, performances, défis et recommandations

3.3 Gestion des sprints et suivi des tâches

3.3.1 Sprint 1 : Analyse de besoins et conception

Durée : 23 octobre - 5 novembre 2025 (2 semaines)

Tâches par membre :

BAANNI Zaineb (Chef d'équipe)

- Gestion de projet : Planification détaillée des sprints, création du backlog produit, mise en place des outils de suivi (Azure DevOps/Trello), organisation des réunions quotidiennes (Daily Standup)
- Analyse Front-End : Analyse des besoins utilisateur (User Stories), conception de l'architecture Angular, définition des composants et services, maquettage des interfaces utilisateur (Figma/Adobe XD), spécification des routes et navigation
- Documentation : Rédaction du cahier des charges fonctionnel, documentation de l'architecture Front-End, création des templates de documentation technique

OUGAS Fadoua (Back-End)

- Analyse Back-End : Analyse des besoins API et services, conception de l'architecture Spring Boot, modélisation de la base de données (schéma ER), définition des endpoints REST, spécification des entités JPA
- Conception technique : Architecture microservices ou monolithique, définition des couches (Controller, Service, Repository), spécification de la sécurité (JWT, Spring Security), planification de l'intégration avec l'IA

IDRISSI Mohamed (Mobile & IA)

- Analyse Mobile : Analyse des besoins spécifiques mobile, conception de l'architecture Flutter, définition des écrans et navigation, spécification des fonctionnalités offline
- Analyse IA : Analyse des besoins en intelligence artificielle, recherche et sélection du modèle IA approprié (Ollama), conception de l'architecture d'intégration IA, définition des cas d'usage IA (chat, recommandations, etc.), spécification des APIs d'IA nécessaires

3.3.2 Sprint 2 : Développement Back-End et Front-End (Phase 1)

Durée : 6 novembre - 19 novembre 2025 (2 semaines)

Tâches par membre :

BAANNI Zaineb (Front-End Angular)

- Développement Front-End Core : Configuration du projet Angular, création de la structure de modules, implémentation du système de routing, développement des

composants de base (Header, Footer, Layout)

- Authentification & Autorisation : Page de connexion (Login), page d'inscription (Signup), service d'authentification, guards de routes (AuthGuard, RoleGuard), gestion des tokens JWT
- Interfaces Utilisateur : Dashboard utilisateur, page de profil utilisateur, liste des cours, page d'accueil (Landing page)
- Intégration API (Phase 1) : Configuration des services HTTP, intégration des endpoints d'authentification, gestion des erreurs et intercepteurs, tests d'intégration avec Postman

OUGAS Fadoua (Back-End Spring Boot)

- Configuration Back-End : Initialisation du projet Spring Boot, configuration de la base de données (JPA/Hibernate), configuration de Spring Security, configuration CORS pour le Front-End
- Authentification & Autorisation : Implémentation de l'authentification JWT, création des entités User, Role, endpoints d'authentification (login, register, refresh token), gestion des rôles et permissions
- API Core : CRUD Utilisateurs, CRUD Formations, CRUD Modules et Cours, gestion des ressources pédagogiques
- Tests Back-End : Tests unitaires avec JUnit, tests d'intégration avec Postman, configuration SonarQube pour l'analyse de code, tests de performance avec JMeter (baseline)

IDRISSI Mohamed (Mobile Flutter)

- Configuration Mobile : Initialisation du projet Flutter, configuration de l'architecture (Provider/Bloc), configuration de la navigation, mise en place des services HTTP
- Interfaces Mobile (Phase 1) : Écran de connexion, écran d'inscription, écran d'accueil, navigation principale

3.3.3 Sprint 3 : Développement Mobile, IA et Intégration

Durée : 20 novembre - 3 décembre 2025 (2 semaines)

Tâches par membre :

BAANNI Zaineb (Front-End Angular)

- Interfaces Avancées : Espace Formateur (Trainer Space), Espace Administrateur (Admin Space), gestion des contenus pédagogiques, système de chat et notifications, tableaux de bord avancés
- Intégration IA Front-End : Interface de chat avec l'IA, affichage des recommandations IA, intégration des suggestions de parcours
- Tests d'Intégration : Tests d'intégration avec Selenium, tests E2E des flux principaux, validation de l'intégration Front-Back, tests avec Postman pour toutes les API

OUGAS Fadoua (Back-End Spring Boot)

- API Avancées : API de gestion des apprenants, API de statistiques et analytics, API de notifications, API de support/tickets, API de recherche
- Intégration IA Back-End : Service d'intégration avec le modèle IA, API pour le chat IA, API pour les recommandations personnalisées, gestion des requêtes IA asynchrones
- Optimisation & Performance : Optimisation des requêtes base de données, mise en cache (Redis si nécessaire), optimisation des endpoints critiques, tests de performance avec JMeter
- Qualité Code : Analyse SonarQube et correction des issues, tests unitaires complets (couverture > 80%), tests d'intégration API complets

IDRISSI Mohamed (Mobile Flutter & IA)

- Développement Mobile Complet : Tous les écrans utilisateur, gestion des cours et contenus, système de notifications push, mode offline (cache local), optimisation UI/UX
- Intégration IA Mobile : Chat IA dans l'application mobile, recommandations personnalisées, synchronisation avec le Back-End IA
- Intégration Back-End Mobile : Intégration complète des API, gestion de l'authentification mobile, synchronisation des données
- Tests Performance Mobile : Tests de performance avec JMeter, optimisation des performances, tests sur différentes versions Android

3.3.4 Sprint 4 : Tests, Optimisation et Documentation

Durée : 4 décembre - 16 décembre 2025 (2 semaines)

Tâches par membre :

BAANNI Zaineb (Chef d'équipe)

- Tests d'Intégration Complets : Tests E2E avec Selenium (tous les scénarios), tests d'intégration Front-Back complets, tests de régression, validation des fonctionnalités critiques
- Documentation Technique : Documentation technique complète, guide d'installation et déploiement, documentation des APIs (Swagger/OpenAPI), guide utilisateur
- Documentation Fonctionnelle : Manuel utilisateur, guide formateur, guide administrateur, FAQ
- Gestion de Projet : Revue finale des sprints, préparation de la présentation finale, coordination des livrables, validation finale avec les parties prenantes

OUGAS Fadoua (Back-End)

- Tests Qualité Complets : Tests unitaires finaux (couverture maximale), tests d'intégration API complets, tests de charge avec JMeter, analyse SonarQube finale, correction de tous les bugs critiques
- Optimisation Back-End : Optimisation des performances, optimisation des requêtes SQL, gestion de la scalabilité, sécurisation finale (audit sécurité)
- Documentation Back-End : Documentation technique des APIs, guide de déploiement Back-End, documentation de l'architecture, guide de maintenance

IDRISSI Mohamed (Mobile & IA)

- Tests Mobile Complets : Tests sur différentes versions Android, tests de performance finaux (JMeter), tests d'utilisation réelle, correction des bugs mobile
- Optimisation Mobile : Optimisation des performances, optimisation de l'UI/UX, réduction de la taille de l'application, optimisation de la consommation batterie
- Tests & Optimisation IA : Tests de l'intégration IA, validation de la qualité des réponses IA, optimisation des performances IA, tests de charge sur les endpoints IA
- Documentation Mobile & IA : Guide d'utilisation mobile, documentation technique Flutter, documentation de l'intégration IA, guide de déploiement mobile

3.4 Suivi des tâches et progrès

Le suivi des tâches et des progrès du projet **Coach Virtuel pour formation interactive avec IA générative** est réalisé à l'aide d'Azure DevOps, un outil de gestion de projet permettant une gestion détaillée des sprints et des tâches. Chaque fonctionnalité est découpée en tickets représentant des tâches spécifiques. Les membres de l'équipe sont responsables de l'achèvement de leurs tâches dans les délais impartis. Les progrès sont suivis à travers un tableau de gestion visuel où les tickets évoluent entre les statuts "À faire", "En cours", et "Terminé".

- **Réunion quotidienne (Daily Standup)** : L'équipe se réunit chaque jour pour discuter des progrès réalisés, des obstacles rencontrés et des actions à entreprendre. Cela permet de maintenir une communication fluide et d'ajuster les priorités en fonction des besoins. Les réunions sont organisées par la chef d'équipe BAANNI Zaineb et durent environ 15 minutes.
- **Sprint Planning** : Une réunion de planification est organisée au début de chaque sprint pour définir les tâches à accomplir et les objectifs à atteindre. Cette réunion permet à l'équipe de se concentrer sur les priorités du sprint tout en ayant une vision claire de ce qui doit être livré à la fin. La répartition des tâches est effectuée selon les compétences de chaque membre : Front-End Angular (BAANNI Zaineb), Back-End Spring Boot (OUGAS Fadoua), et Mobile Flutter avec IA (IDRISSI Mohamed).
- **Sprint Review et Sprint Retrospective** : À la fin de chaque sprint, l'équipe se réunit pour revoir les livrables réalisés et s'assurer qu'ils répondent aux critères de qualité. Lors de la Sprint Review, les livrables sont validés et présentés aux parties prenantes (Mme SBAI Hanae, Mme BOUSQAOU Halima, et M. ESSABBAR Driss). Ensuite, lors de la Sprint Retrospective, l'équipe réfléchit aux améliorations possibles dans le processus de développement pour augmenter l'efficacité et la qualité dans les sprints suivants.

Partie 4

Documentation

4.1 Règles de gestion et de structuration des documents

La gestion documentaire du projet **Coach Virtuel pour formation interactive avec IA générative** s'appuie sur des standards rigoureux de qualité, afin de garantir une traçabilité complète, une cohérence technique et une collaboration optimisée entre les différents acteurs du projet. Cette démarche vise à assurer la pérennité et la reproductibilité des livrables, tout en facilitant la maintenance future du système.

- **Uniformité et standardisation des formats** : Tous les documents sont structurés selon un gabarit uniforme incluant titres hiérarchisés, numérotation logique et styles de paragraphes homogènes.
- **Versionnement rigoureux** : Chaque document est versionné (ex. V1.0, V1.1, V2.0) avec un historique détaillé des modifications.
- **Centralisation et sécurité** : L'ensemble des documents est hébergé dans un espace collaboratif sécurisé (**Azure DevOps**) avec des droits d'accès différenciés selon le rôle.
- **Traçabilité complète** : Chaque modification majeure est consignée avec la date, l'auteur et la nature de la modification.
- **Collaboration et communication** : La communication entre les membres est assurée via **Google meet** et **WhatsApp**, avec suivi des commentaires et revues dans Azure DevOps et GitHub.

4.2 Documents de gestion de projet

1. Diagramme de Gantt :

- Permet de visualiser les tâches, les jalons et la répartition des responsabilités sur les différents sprints du projet.
- Utilisé pour suivre l'avancement des différentes phases et pour s'assurer que le projet progresse selon le calendrier prévu.
- Le diagramme couvre la période complète du projet (23 octobre - 16 décembre 2025) et illustre les 4 sprints avec leurs dépendances et leurs jalons critiques.

2. Roadmap :

- Plan stratégique indiquant les objectifs principaux, les délais et les étapes clés à franchir pour garantir la réussite du projet.
- Identifie les modules critiques (Front-End Angular, Back-End Spring Boot, Mobile Flutter, Intégration IA) et les risques associés.
- Définit les priorités fonctionnelles et techniques pour chaque phase du projet.

3. Plan d'Assurance Qualité (PAQ) :

- Document garantissant le respect des normes de qualité et la validation des livrables à chaque phase du projet.
- Il inclut les processus de test (unitaires, intégration, E2E, performance), la gestion des risques et la conformité aux exigences.
- Définit les standards de qualité, les méthodologies de test (JUnit, Selenium, JMeter, SonarQube, Postman) et les critères de validation pour chaque livrable.

4. Rapports d'Avancement :

- Synthèses périodiques détaillant l'état d'avancement des tâches, les défis rencontrés, ainsi que les ajustements nécessaires pour respecter les délais et les objectifs du projet.
- Produits à la fin de chaque sprint et lors des points d'avancement hebdomadaires.
- Incluent l'état des fonctionnalités, les anomalies identifiées, les mesures correctives mises en place et les indicateurs de performance (burn-down chart, vélocité de l'équipe).

5. Répartition des Tâches (Azure DevOps) :

- Organisation des tâches par sprint, avec assignation des responsables et suivi de leur état (to-do, in-progress, done).
- Azure DevOps permet de suivre en temps réel l'avancement de chaque tâche et d'assurer une gestion efficace des ressources.

- Chaque fonctionnalité est découpée en tickets représentant des tâches spécifiques, avec assignation claire selon les compétences : Front-End Angular (BAANNI Zaineb), Back-End Spring Boot (OUGAS Fadoua), et Mobile Flutter avec IA (IDRISSI Mohamed).
- Le tableau de bord visuel permet de visualiser rapidement l'état d'avancement global du projet et d'identifier les éventuels goulots d'étranglement.

4.3 Documentation technique et livrables de réalisation

Ces documents concernent les aspects techniques et les livrables produits durant la réalisation du projet **Coach Virtuel pour formation interactive avec IA générative** :

1. Diagrammes UML (StarUML) :

- **Diagrammes de classes** : Représentent la structure du système, montrant les entités (User, Formation, Module, Course, Quiz, etc.), leurs attributs et leurs relations. Ces diagrammes couvrent les modèles de données pour le Front-End Angular, le Back-End Spring Boot et l'application mobile Flutter.
- **Diagrammes de séquences** : Illustrent les interactions entre les composants du système lors de l'exécution des processus (authentification, gestion des cours, interaction avec l'IA, notifications, etc.). Ils montrent les flux de communication entre le Front-End, le Back-End, la base de données PostgreSQL et le service IA.
- **Diagrammes d'activités** : Décrivent le déroulement des processus métiers et les flux de travail au sein de l'application (parcours d'apprentissage, génération de contenu par IA, évaluation des apprenants, gestion des contenus pédagogiques).

2. Rapports de tests :

- **SonarQube** : Analyse de la qualité du code, incluant l'identification des bugs, la couverture des tests et les vulnérabilités potentielles. Les rapports couvrent le code Back-End (Spring Boot) et permettent d'assurer une qualité de code optimale avec une couverture de tests supérieure à 80%.
- **JMeter** : Résultats des tests de charge pour évaluer la performance et la scalabilité du backend du système. Les tests incluent également l'évaluation des performances de l'application mobile Flutter et des endpoints d'intégration IA.
- **Selenium** : Résultats des tests automatisés des interfaces web (Front-End Angular) pour s'assurer de leur bon fonctionnement et de leur réactivité. Les tests E2E couvrent les principaux flux utilisateur (authentification, navigation, ges-

tion des cours, interaction avec le coach IA).

- **Postman** : Tests des API développées avec Spring Boot pour vérifier leur bon fonctionnement et leur conformité aux spécifications. Les collections Postman incluent tous les endpoints REST (authentification, CRUD formations, gestion des utilisateurs, intégration IA, etc.).
- **Tests IA** : Validation de la qualité des réponses du modèle Ollama, tests de performance des endpoints IA, et évaluation de la pertinence des recommandations personnalisées générées par le coach virtuel.

3. Documentation technique :

- **Spécifications des API REST utilisées** : Détail des endpoints, des méthodes HTTP (GET, POST, PUT, DELETE), des réponses attendues et des erreurs possibles. Documentation complète avec Swagger/OpenAPI incluant les schémas de données, les codes de statut HTTP et les exemples de requêtes/réponses.
- **Description de l'architecture technique** : Explication de la structure du projet, des choix de conception et des technologies utilisées (Angular pour le Front-End, Spring Boot pour le Back-End, Flutter pour le mobile, PostgreSQL pour la base de données, Ollama pour l'IA). Documentation des patterns architecturaux adoptés (MVC, Repository, Service Layer) et des décisions techniques justifiées.

4. Documentation IA et modèles :

- **Modèles IA** : Documentation du modèle Ollama utilisé, de sa configuration et de son intégration dans la plateforme.
- **Pipeline de génération** : Description du processus de génération de contenu par IA (exercices, quiz, recommandations, réponses du chat).
- **Journal de performance IA** : Suivi des métriques de performance du modèle IA (temps de réponse, qualité des réponses, taux de satisfaction utilisateur).

5. Documentation utilisateur :

- **Manuel utilisateur** : Guide détaillé pour aider les trois types d'utilisateurs (apprenants, formateurs, administrateurs) à naviguer et à utiliser les différentes fonctionnalités de l'application. Inclut des captures d'écran, des tutoriels pas à pas et des FAQ pour chaque profil utilisateur.
- **Guide formateur** : Documentation spécifique pour les formateurs expliquant comment créer et gérer les contenus pédagogiques, suivre les apprenants, utiliser l'assistant IA pour générer du contenu, et analyser les statistiques.
- **Guide administrateur** : Documentation pour les administrateurs couvrant la gestion des utilisateurs, la supervision de la plateforme, la configuration de

l'IA, et l'analyse des données globales.

6. Rapport final :

- **Synthèse globale du projet :** Résumé des objectifs atteints, des défis rencontrés durant le développement et des recommandations pour de futures améliorations ou évolutions du projet.
- Inclut une analyse des performances techniques (temps de réponse, scalabilité, qualité du code), une évaluation de l'intégration IA, et des perspectives d'évolution de la plateforme.

Partie 5

Gestion des modifications

5.1 Origine et typologie des modifications

Les modifications dans le projet **Coach Virtuel pour formation interactive avec IA générative** peuvent avoir différentes origines, et leur gestion est cruciale pour s'assurer que le projet respecte les objectifs fixés, les délais et les normes de qualité. Les modifications peuvent être classées en fonction de leur origine et de leur typologie.

5.1.1 Origine des modifications

1. Retour des encadrants :

- Des ajustements peuvent être proposés par les superviseurs du projet (Mme SBAI Hanae, Mme BOUSQAoui Halima, et M. ESSABBAR Driss), en fonction de l'avancement, des nouvelles exigences ou des ajustements nécessaires pour répondre aux attentes du projet.
- Ces retours peuvent concerner l'architecture technique, les fonctionnalités pédagogiques, l'intégration IA, ou les aspects de qualité et de sécurité.

2. Bugs et anomalies :

- Les problèmes techniques identifiés lors des différentes phases de tests (tests unitaires avec JUnit, tests d'intégration avec Selenium, tests de performance avec JMeter, tests fonctionnels, ou tests d'intégration IA) peuvent entraîner des modifications pour résoudre ces problèmes et améliorer la stabilité du système.

- Les bugs peuvent être détectés dans le Front-End Angular, le Back-End Spring Boot, l'application mobile Flutter, ou dans l'intégration avec le modèle IA OllaMA.

3. **Feedback des utilisateurs :**

- Des retours des utilisateurs (apprenants, formateurs, administrateurs) peuvent être collectés durant les tests utilisateurs. Ces retours peuvent concerner l'ergonomie de l'interface (Front-End Angular ou application mobile Flutter), les fonctionnalités manquantes, ou la correction de bugs identifiés par les utilisateurs.
- Les retours peuvent également porter sur l'expérience d'interaction avec le coach virtuel IA, la pertinence des recommandations personnalisées, ou l'utilité des fonctionnalités pédagogiques.

4. **Changements de priorité ou de stratégie :**

- Si les priorités du projet évoluent (par exemple, ajout de nouvelles fonctionnalités pédagogiques, révision du design de l'application, amélioration de l'intégration IA, ou ajustement en fonction des besoins du marché de la formation professionnelle), des modifications peuvent être nécessaires pour s'adapter à ces changements.
- Ces changements peuvent affecter la planification des sprints, la répartition des tâches entre les membres de l'équipe, ou les choix techniques (technologies, architecture, modèles IA).

5. **Améliorations techniques :**

- Des modifications peuvent aussi viser l'optimisation des performances de l'application, telles que l'amélioration de la gestion des parcours d'apprentissage en temps réel, l'optimisation de l'interface utilisateur (Front-End Angular et application mobile Flutter), l'optimisation des requêtes base de données PostgreSQL, ou encore l'amélioration de l'authentification des utilisateurs pour garantir la sécurité et la fluidité de l'expérience utilisateur.
- Les améliorations peuvent également concerner l'optimisation des performances du modèle IA (OllaMA), la réduction des temps de réponse des endpoints IA, ou l'amélioration de la qualité des réponses générées par le coach virtuel.

5.1.2 **Typologie des modifications**

Les modifications dans le projet **Coach Virtuel pour formation interactive avec IA générative** sont classées en fonction des types suivants :

1. **Fonctionnelles :**

- Ces modifications concernent l'ajout, la modification ou la suppression de fonctionnalités spécifiques à l'application. Par exemple, l'ajout d'une fonctionnalité de génération de contenu pédagogique par IA, la modification du système de recommandations personnalisées, l'ajout de nouvelles fonctionnalités de suivi des apprenants, ou l'amélioration de l'interface de chat avec le coach virtuel.
- Les modifications fonctionnelles peuvent toucher les trois interfaces : Front-End Angular (espaces utilisateur, formateur, administrateur), Back-End Spring Boot (nouvelles API, nouveaux services), ou application mobile Flutter (nouvelles fonctionnalités mobiles).

2. Techniques :

- Ce type de modification implique des changements dans l'architecture du projet, l'optimisation des performances, ou l'amélioration de la sécurité. Par exemple, la mise à jour de la base de données PostgreSQL pour améliorer la gestion des utilisateurs et des parcours d'apprentissage, l'optimisation de l'authentification JWT, l'amélioration de l'intégration avec le modèle IA Ollama, ou l'optimisation des requêtes API pour réduire les temps de réponse.
- Les modifications techniques peuvent également inclure la refactorisation du code, l'amélioration de la scalabilité du système, ou la mise à jour des dépendances et des frameworks (Angular, Spring Boot, Flutter).

3. Documentaires :

- Ces modifications concernent la mise à jour des documents techniques relatifs au projet, tels que les diagrammes UML (StarUML), les rapports de tests (SonarQube, JMeter, Selenium, Postman), la documentation technique (spécifications API REST, architecture), la documentation IA (modèles, pipeline de génération), ou la documentation utilisateur (manuels pour apprenants, formateurs, administrateurs).
- Ces mises à jour permettent de garder une trace des évolutions et de garantir que les informations sont à jour pour tous les membres de l'équipe et les parties prenantes.

4. Planification :

- Ce type de modification concerne le réajustement du calendrier ou des sprints du projet, en fonction des modifications à apporter ou des retards éventuels. Cela peut inclure la reprogrammation de certaines étapes du développement ou la modification des objectifs d'un sprint pour mieux répondre aux besoins.
- Les ajustements de planification sont gérés via Azure DevOps et sont discutés lors des réunions de Sprint Planning et de Sprint Retrospective, avec validation par la chef d'équipe BAANNI Zaineb et communication aux encadrants.

5.2 Procédures pour l’approbation et l’implémentation des changements

Pour assurer que chaque modification est bien documentée, analysée, et implémentée sans perturber la bonne marche du projet **Coach Virtuel pour formation interactive avec IA générative**, une procédure claire est mise en place. Voici les étapes de gestion des modifications :

1. Identification de la modification :

- Chaque modification est d’abord enregistrée dans Azure DevOps sous forme de ticket.
- Le ticket doit inclure les informations suivantes : description détaillée de la modification, origine, impact sur le projet, priorité et ressources nécessaires pour la mise en œuvre.
- Le ticket est catégorisé selon la typologie (fonctionnelle, technique, documentaire, planification) et assigné à la personne responsable selon le domaine concerné (Front-End Angular pour BAANNI Zaineb, Back-End Spring Boot pour OUGAS Fadoua, Mobile Flutter et IA pour IDRISI Mohamed).

2. Analyse et évaluation :

- Une réunion avec l’équipe projet est organisée pour analyser l’impact de la modification sur les objectifs du projet, sur les fonctionnalités existantes et sur la planification globale (Diagramme de Gantt).
- Une évaluation des risques et des ressources nécessaires est réalisée pour déterminer la faisabilité de la modification. Cette évaluation prend en compte l’impact sur les trois composants principaux (Front-End Angular, Back-End Spring Boot, application mobile Flutter) et sur l’intégration IA (modèle Ollama).
- L’équipe projet discute ensuite de la modification et décide si elle doit être validée ou rejetée en fonction de son coût, de son impact et de sa faisabilité. La décision est prise en concertation avec la chef d’équipe BAANNI Zaineb.

3. Approbation des modifications :

- Les modifications sont validées par les superviseurs (Mme SBAI Hanae, Mme BOUSQAoui Halima, et M. ESSABBAR Driss) selon leur impact et leur criticité.
- En cas de modifications urgentes (bugs critiques, problèmes de sécurité, anomalies bloquantes), celles-ci peuvent être traitées rapidement avec l’accord des responsables et de la chef d’équipe.
- Les modifications majeures nécessitent une validation formelle lors des réunions de Sprint Review ou lors de points d’avancement avec les encadrants.

4. Mise en œuvre de la modification :

- Une fois approuvée, la modification est assignée à un membre de l'équipe via Azure DevOps. Les tâches sont alors priorisées selon leur urgence et leur impact.
- L'équipe de développement met en œuvre les modifications nécessaires en suivant les étapes de développement et de test :
 - **Front-End Angular (BAANNI Zaineb)** : Développement des composants, services, et intégration avec les API.
 - **Back-End Spring Boot (OUGAS Fadoua)** : Développement des API REST, services métier, et gestion de la base de données PostgreSQL.
 - **Mobile Flutter et IA (IDRISSI Mohamed)** : Développement des écrans mobiles, intégration des API, et intégration avec le modèle IA Ollama.
- Le code est versionné via GitHub avec des branches dédiées pour chaque modification, permettant la revue de code avant intégration.

5. Tests et validation :

- Des tests sont réalisés pour vérifier l'impact de la modification sur l'ensemble du système :
 - **Tests unitaires** : Avec JUnit pour le Back-End Spring Boot, garantissant le bon fonctionnement des méthodes et services modifiés.
 - **Tests d'intégration** : Avec Selenium pour le Front-End Angular (tests E2E) et avec Postman pour les API REST, vérifiant l'intégration entre les différents composants.
 - **Tests fonctionnels** : Validation des fonctionnalités modifiées selon les spécifications.
 - **Tests IA** : Validation de l'impact sur l'intégration avec le modèle Ollama et vérification de la qualité des réponses générées.
- En cas de détection d'anomalies, des corrections sont apportées et les tests sont relancés jusqu'à validation complète.
- Des tests de performance (JMeter) et de sécurité sont également réalisés pour s'assurer que la modification n'affecte pas la stabilité et la sécurité du projet. L'analyse SonarQube est effectuée pour maintenir la qualité du code.

6. Mise à jour de la documentation :

- Une fois la modification implémentée et validée, la documentation technique est mise à jour pour refléter les changements :
 - **Diagrammes UML (StarUML)** : Mise à jour des diagrammes de classes, de séquences et d'activités si nécessaire.
 - **Rapports de tests** : Mise à jour des rapports SonarQube, JMeter, Selenium, Postman, et tests IA.

- **Documentation technique** : Mise à jour des spécifications API REST (Swagger/OpenAPI), de la documentation d'architecture, et de la documentation IA (modèles, pipeline de génération).
- La documentation utilisateur peut également être mise à jour si la modification concerne l'interface ou les fonctionnalités visibles pour l'utilisateur final (manuels pour apprenants, formateurs, administrateurs).
- La documentation est versionnée et accessible via Azure DevOps pour garantir la traçabilité des modifications.

7. Suivi des modifications :

- Un suivi est effectué pour s'assurer que la modification a bien été intégrée dans le produit final et que l'impact attendu est atteint. Le suivi est réalisé via Azure DevOps avec mise à jour du statut des tickets.
- Une évaluation des retours utilisateurs (apprenants, formateurs, administrateurs) est réalisée pour déterminer si des ajustements supplémentaires sont nécessaires, notamment concernant l'expérience d'utilisation et l'interaction avec le coach virtuel IA.
- Les métriques de performance sont surveillées pour s'assurer que la modification n'a pas dégradé les performances du système (temps de réponse, scalabilité, qualité des réponses IA).
- Lors de la Sprint Retrospective, l'équipe évalue l'efficacité de la gestion des modifications et propose des améliorations pour les prochains sprints.

Partie 6

Méthodes, outils et normes

6.1 Méthodes et méthodologie retenues

Pour garantir la réussite du projet **Coach Virtuel pour formation interactive avec IA générative**, nous avons opté pour la méthodologie SCRUM, qui est une approche Agile. Cette méthodologie nous permet de diviser le projet en plusieurs phases appelées sprints, facilitant ainsi une adaptation rapide aux besoins changeants du projet tout en assurant une communication continue et fluide entre les membres de l'équipe et les parties prenantes.

6.1.1 Méthodologie SCRUM

Nous avons choisi SCRUM en raison de sa flexibilité et de son efficacité dans la gestion de projets logiciels. Elle repose sur un processus itératif et incrémental, où le travail est divisé en sprints de 2 semaines. À la fin de chaque sprint, un livrable fonctionnel et testable est produit, permettant ainsi une évaluation régulière de l'avancement du projet. Les détails de l'organisation des sprints et des activités SCRUM sont présentés dans la Partie 3 de ce document.

6.1.2 Gestion des changements

Les modifications au projet sont gérées par le biais de tickets dans Azure DevOps, où chaque modification est soigneusement détaillée, évaluée, puis approuvée avant d'être

implémentée dans le projet. Cette approche permet une gestion transparente des changements et garantit que chaque ajustement est effectué de manière structurée et conforme aux objectifs du projet. Les procédures détaillées de gestion des modifications sont présentées dans la Partie 5 de ce document.

6.2 Règles et normes devant être appliquées

Pour assurer la réussite du projet **Coach Virtuel pour formation interactive avec IA générative** et garantir qu'il respecte les bonnes pratiques, plusieurs règles et normes sont mises en place à chaque étape du développement :

6.2.1 Normes de développement

- **Respect des principes SOLID** : Nous veillons à structurer notre code de manière claire et réutilisable en appliquant les principes SOLID, garantissant ainsi une architecture solide et évolutive. Ces principes sont appliqués à tous les composants du projet : Back-End Spring Boot, Front-End Angular, et application mobile Flutter.
- **Clean Code** : L'objectif est de maintenir un code lisible et facile à maintenir en suivant les principes du Clean Code. Cela permet de garantir la qualité et la simplicité du code sur le long terme, facilitant la maintenance et l'évolution de la plateforme.
- **Tests unitaires et d'intégration** : Tous les modules du projet sont couverts par des tests unitaires. Des tests d'intégration sont également réalisés pour vérifier l'interopérabilité des différents composants du système (Front-End Angular, Back-End Spring Boot, application mobile Flutter, intégration IA avec Ollama) et assurer leur bon fonctionnement ensemble. Les tests unitaires Back-End sont réalisés avec JUnit, tandis que les tests d'intégration utilisent Selenium pour le Front-End et Postman pour les API REST.
- **Analyse de qualité du code** : L'utilisation de SonarQube permet d'analyser en continu la qualité du code, d'identifier les bugs potentiels, les vulnérabilités de sécurité et de maintenir une couverture de tests supérieure à 80%.

6.2.2 Normes de sécurité

- **Authentification et autorisation** : L'application respecte les meilleures pratiques de sécurité, en particulier pour la gestion des comptes utilisateurs (apprenants, formateurs, administrateurs), avec des mécanismes de cryptage des données sensibles telles que les mots de passe. L'authentification est gérée via JWT (JSON

Web Tokens) avec Spring Security pour le Back-End, et des guards de routes pour le Front-End Angular et l'application mobile Flutter.

- **Sécurisation des API :** Les API REST développées avec Spring Boot sont sécurisées pour protéger les données des utilisateurs, des parcours de formation et des interactions avec le coach virtuel IA. Les endpoints sont protégés par des mécanismes d'autorisation basés sur les rôles.
- **Tests de sécurité :** Des tests de sécurité réguliers sont effectués pour identifier et corriger toute vulnérabilité dans l'application, garantissant ainsi une protection maximale des données. L'analyse SonarQube inclut également la détection des vulnérabilités de sécurité.
- **Sécurisation de l'intégration IA :** Les interactions avec le modèle IA Ollama sont sécurisées pour protéger les données des utilisateurs et garantir la confidentialité des échanges avec le coach virtuel.

6.2.3 Normes de performance

- **Tests de performance avec JMeter :** Tous les services sont soumis à des tests de performance pour évaluer leur capacité à supporter des charges spécifiques et leur temps de réponse sous différents scénarios d'utilisation. Les tests incluent :
 - Tests de charge sur les API REST du Back-End Spring Boot
 - Tests de performance de l'application mobile Flutter
 - Tests de performance des endpoints d'intégration IA (chat, recommandations personnalisées)
 - Évaluation de la scalabilité de la base de données PostgreSQL
- **Optimisation du code :** Le code est régulièrement optimisé pour garantir des temps de réponse rapides et une gestion efficace des ressources, contribuant ainsi à une expérience utilisateur fluide. L'optimisation concerne :
 - L'optimisation des requêtes base de données PostgreSQL
 - L'optimisation des performances du Front-End Angular (lazy loading, code splitting)
 - L'optimisation de l'application mobile Flutter (réduction de la taille, amélioration de la consommation batterie)
 - L'optimisation des temps de réponse du modèle IA Ollama
- **Monitoring des performances :** Un suivi continu des métriques de performance est effectué pour détecter les goulots d'étranglement et optimiser les composants critiques du système.

6.2.4 Normes de gestion documentaire

- **Documentations complètes** : Chaque étape du projet est accompagnée d'une documentation détaillée et à jour, comprenant :
 - Diagrammes UML (StarUML) : diagrammes de classes, de séquences et d'activités
 - Rapports de tests : SonarQube, JMeter, Selenium, Postman, tests IA
 - Documentation technique : spécifications API REST (Swagger/OpenAPI), architecture technique, documentation de l'intégration IA
 - Documentation utilisateur : manuels pour apprenants, formateurs et administrateurs
- **Mise à jour continue** : La documentation est régulièrement mise à jour pour refléter l'évolution du projet, incluant des modifications dans les fonctionnalités, l'architecture, les tests ou l'intégration IA. Tous les documents sont versionnés et centralisés dans Azure DevOps.
- **Traçabilité** : Chaque modification majeure est consignée avec la date, l'auteur et la nature de la modification, garantissant une traçabilité complète de l'évolution du projet.

6.2.5 Suivi des versions et des modifications

- **Contrôle de version** : Le code source est géré via GitHub avec des pratiques strictes pour le contrôle des branches et des commits, assurant une gestion claire des versions du code. Les pratiques incluent :
 - Utilisation de branches dédiées pour chaque fonctionnalité ou correction
 - Revues de code (pull requests) avant intégration dans la branche principale
 - Messages de commit descriptifs et cohérents
 - Gestion des tags pour les versions majeures du projet
- **Gestion des tickets et des changements** : Toute modification ou ajout de fonctionnalité est documenté et suivi à travers des tickets dans Azure DevOps, garantissant que chaque changement est correctement validé et implémenté. Les tickets incluent :
 - Description détaillée de la modification
 - Impact sur les différents composants (Front-End Angular, Back-End Spring Boot, Mobile Flutter, IA)
 - Priorité et criticité
 - Assignation selon les compétences (BAANNI Zaineb pour Front-End, OUGAS Fadoua pour Back-End, IDRISSE Mohamed pour Mobile et IA)

- Suivi de l'état d'avancement (to-do, in-progress, done)
- **Intégration continue et déploiement continu (CI/CD)** : Les workflows CI/CD sont configurés pour automatiser les tests et le déploiement, garantissant que chaque modification est testée avant intégration dans le code principal.

Partie 7

Gestion des risques

7.1 Identification des risques projet

L'identification et la gestion proactive des risques sont essentielles pour assurer le bon déroulement du projet **Coach Virtuel pour formation interactive avec IA générative**. Les principaux risques identifiés sont les suivants :

7.1.1 Retards dans le développement

- **Description** : Des retards peuvent survenir en raison de problèmes techniques, de dépendances externes (notamment l'intégration avec le modèle IA Ollama), d'absences temporaires de membres de l'équipe ou d'une mauvaise estimation des tâches. La complexité de l'intégration entre les trois composants principaux (Front-End Angular, Back-End Spring Boot, application mobile Flutter) et l'IA peut également entraîner des retards imprévus.
- **Impact potentiel** : Ces retards peuvent affecter la qualité des livrables, retarder la mise en production des fonctionnalités, impacter les délais de livraison du projet et compromettre la validation académique du projet.

7.1.2 Problèmes de performance

- **Description** : Des problèmes de performance peuvent survenir, notamment des lenteurs dues à une surcharge serveur, des temps de réponse trop longs pour les

interactions avec le coach virtuel IA, des problèmes de performance de l'application mobile Flutter, ou une mauvaise optimisation du code. Les requêtes complexes sur la base de données PostgreSQL et les temps de réponse du modèle IA Ollama peuvent également affecter les performances globales.

- **Impact potentiel :** Ces problèmes peuvent nuire à l'expérience utilisateur (apprenants, formateurs, administrateurs), réduire la satisfaction des utilisateurs finaux, affecter la stabilité du système et compromettre l'efficacité pédagogique de la plateforme.

7.1.3 Problèmes de compatibilité entre technologies

- **Description :** L'interopérabilité entre les différentes technologies utilisées, telles que l'intégration entre Angular pour le Front-End et Spring Boot pour le Back-End, l'intégration de l'application mobile Flutter avec les API REST, ou l'intégration du modèle IA Ollama avec les différents composants, peut entraîner des problèmes techniques imprévus. Les différences de versions, les incompatibilités d'API ou les problèmes de communication entre les services peuvent également survenir.
- **Impact potentiel :** Les dysfonctionnements ou erreurs d'intégration peuvent retarder la livraison du projet, compromettre l'expérience utilisateur, affecter la cohérence des données entre les différentes interfaces et impacter la qualité des interactions avec le coach virtuel IA.

7.1.4 Risques liés à l'intégration IA

- **Description :** L'intégration du modèle IA Ollama peut présenter des défis spécifiques, notamment des temps de réponse variables, une qualité des réponses générées qui peut ne pas toujours répondre aux attentes pédagogiques, des problèmes de disponibilité du service IA, ou des difficultés d'optimisation des coûts d'utilisation de l'IA. La génération de contenu pédagogique par IA peut également nécessiter des ajustements et des validations supplémentaires.
- **Impact potentiel :** Ces problèmes peuvent affecter la qualité pédagogique de la plateforme, réduire la confiance des utilisateurs dans le coach virtuel, impacter l'expérience d'apprentissage et compromettre l'objectif principal du projet.

7.1.5 Changements de scope (scope creep)

- **Description :** L'ajout de nouvelles fonctionnalités pédagogiques, des demandes non prévues en cours de projet, ou des ajustements liés à l'intégration IA peuvent entraîner des écarts par rapport à la planification initiale. Les retours des enca-

drants (Mme SBAI Hanae, Mme BOUSQAoui Halima, M. ESSABBAR Driss) ou des utilisateurs peuvent également générer des demandes de modifications non anticipées.

- **Impact potentiel** : Cela peut entraîner des retards supplémentaires, des surcoûts en termes de ressources humaines, une complexité accrue du projet, rendant difficile la gestion des délais et des objectifs initiaux, et compromettre la livraison dans les temps impartis.

7.1.6 Risques liés à la gestion des équipes

- **Description** : Des problèmes internes à l'équipe, tels qu'un manque de communication, l'absence d'un membre clé (BAANNI Zaineb pour le Front-End, OUGAS Fadoua pour le Back-End, IDRISSE Mohamed pour le Mobile et l'IA), des divergences sur la répartition des tâches, ou des difficultés de coordination entre les différents domaines techniques peuvent affecter la coordination et la productivité de l'équipe.
- **Impact potentiel** : Ces problèmes peuvent réduire la performance de l'équipe, entraîner des tâches incomplètes ou mal réalisées, compromettre ainsi la qualité du projet et affecter l'intégration entre les différents composants (Front-End, Back-End, Mobile, IA).

7.2 Plans d'atténuation des risques

Afin de minimiser les impacts des risques identifiés dans le projet **Coach Virtuel pour formation interactive avec IA générative**, plusieurs plans d'atténuation ont été définis :

7.2.1 Retards dans le développement

- **Plan d'atténuation** :
 - **Gestion proactive des ressources** : Réévaluation régulière des priorités et répartition flexible des tâches en fonction de l'avancement du projet. Les tâches critiques seront priorisées et l'équipe devra s'adapter pour pallier à toute ressource manquante. La chef d'équipe BAANNI Zaineb coordonne la répartition des tâches selon les compétences de chaque membre.
 - **Réunions régulières de suivi** : Des réunions quotidiennes (daily standups) seront organisées pour évaluer l'avancement et ajuster le planning des sprints. Les obstacles sont identifiés rapidement et des solutions sont proposées en temps réel.

- **Gestion des dépendances** : Les dépendances entre les tâches Front-End, Back-End, Mobile et IA sont identifiées en amont et planifiées pour éviter les blocages. L'intégration avec le modèle IA Ollama est testée dès les premières phases pour anticiper les problèmes.

7.2.2 Problèmes de performance

- **Plan d'atténuation** :
 - **Tests de performance réguliers** : Mise en place de tests de charge et de performance à l'aide de JMeter afin de valider la stabilité de l'application sous diverses charges. Les tests incluent les API REST du Back-End Spring Boot, l'application mobile Flutter, et les endpoints d'intégration IA.
 - **Optimisation continue** : L'optimisation du code et la révision des processus backend seront effectuées de manière itérative tout au long du projet. L'optimisation concerne également les requêtes PostgreSQL, les performances du Front-End Angular, et les temps de réponse du modèle IA Ollama.
 - **Monitoring et alertes** : Des outils de monitoring des performances seront utilisés pour détecter les anomalies dès leur apparition, permettant une intervention rapide. L'analyse SonarQube permet également de détecter les problèmes de performance dans le code.

7.2.3 Problèmes de compatibilité entre technologies

- **Plan d'atténuation** :
 - **Tests d'intégration fréquents** : Des tests d'intégration seront réalisés dès l'ajout de nouvelles fonctionnalités, notamment pour assurer la compatibilité entre le Front-End Angular et le Back-End Spring Boot, entre l'application mobile Flutter et les API REST, et entre tous les composants et le modèle IA Ollama. Les tests utilisent Selenium pour le Front-End et Postman pour les API.
 - **Revue technique** : Des revues de code régulières permettront de détecter les incohérences et de garantir la bonne communication entre les différents modules du projet. Les revues sont effectuées via GitHub avec des pull requests avant intégration.
 - **Documentation des interfaces** : La documentation des API REST avec Swagger/OpenAPI permet de garantir la cohérence des interfaces entre les différents composants et facilite l'intégration.

7.2.4 Risques liés à l'intégration IA

- **Plan d'atténuation :**

- **Tests spécifiques IA :** Des tests dédiés sont effectués pour valider la qualité des réponses du modèle Ollama, les temps de réponse, et la pertinence des recommandations personnalisées. Des tests de charge sont également réalisés sur les endpoints IA.
- **Gestion des erreurs IA :** Des mécanismes de fallback et de gestion d'erreurs sont implémentés pour gérer les cas où le service IA n'est pas disponible ou génère des réponses inappropriées.
- **Validation pédagogique :** Les contenus générés par l'IA sont validés par les formateurs et les encadrants pour garantir leur qualité pédagogique avant d'être proposés aux apprenants.
- **Optimisation des coûts :** L'utilisation du modèle IA est optimisée pour réduire les coûts tout en maintenant la qualité des réponses, notamment par la mise en cache des réponses fréquentes.

7.2.5 Changements de scope (scope creep)

- **Plan d'atténuation :**

- **Gestion stricte des demandes :** Tout ajout de fonctionnalité devra être validé et approuvé par la chef d'équipe BAANNI Zaineb et les encadrants (Mme SBAI Hanae, Mme BOUSQAOU Halima, M. ESSABBAR Driss). Il sera intégré dans les sprints futurs, et toute modification non planifiée sera examinée avant d'entraîner un réajustement des délais.
- **Revue des exigences :** Une revue continue des exigences et du scope sera effectuée pour éviter toute dérive par rapport aux objectifs et délais initiaux. Les modifications sont documentées dans Azure DevOps avec évaluation de l'impact.
- **Priorisation :** Les nouvelles demandes sont priorisées selon leur impact sur les objectifs du projet et leur faisabilité technique, en tenant compte des ressources disponibles et des délais restants.

7.2.6 Risques liés à la gestion des équipes

- **Plan d'atténuation :**

- **Renforcement de la communication :** Mise en place de canaux de communication clairs (Google meet pour la communication quotidienne et technique, WhatsApp pour les notifications urgentes, Azure DevOps pour le suivi des

tâches). Des réunions régulières sont organisées pour maintenir la cohésion de l'équipe.

- **Rotation des tâches et soutien mutuel** : Pour pallier l'absence ou la surcharge de travail d'un membre de l'équipe, une gestion souple des tâches et un soutien croisé entre les développeurs seront encouragés. Chaque membre a une connaissance de base des autres domaines pour assurer la continuité en cas d'absence.
- **Documentation partagée** : Toutes les connaissances techniques sont documentées et partagées via Azure DevOps et GitHub, permettant à n'importe quel membre de l'équipe de prendre le relais si nécessaire.

7.3 Suivi des risques

Le suivi des risques sera effectué de manière continue tout au long du projet, avec une attention particulière aux risques critiques. Les actions de suivi seront les suivantes :

7.3.1 Suivi via Azure DevOps

- Chaque risque sera enregistré sous forme de ticket dans Azure DevOps, où il pourra être suivi, mis à jour et évalué régulièrement. Les tickets incluent la description du risque, son niveau de criticité, les actions d'atténuation prévues et leur statut d'avancement.
- Un responsable sera désigné pour chaque risque identifié afin de garantir une gestion efficace. Des actions correctives seront définies et mises en œuvre en fonction de l'évolution du projet. La chef d'équipe BAANNI Zaineb supervise la gestion globale des risques.
- Les risques sont catégorisés selon leur impact sur les différents composants du projet (Front-End Angular, Back-End Spring Boot, Mobile Flutter, Intégration IA) pour faciliter leur suivi et leur résolution.

7.3.2 Réévaluation périodique des risques

- Une évaluation formelle des risques sera réalisée à la fin de chaque sprint lors des réunions de Sprint Retrospective. Cela permettra de réajuster les plans d'atténuation si nécessaire et d'identifier de nouveaux risques potentiels qui pourraient survenir au cours du projet.
- Les risques liés à l'intégration IA sont évalués spécifiquement lors des tests d'intégration IA, avec un suivi des métriques de performance et de qualité des réponses générées.

- Les risques techniques sont réévalués après chaque phase d'intégration majeure entre les composants (Front-End/Back-End, Mobile/Back-End, Intégration IA).

7.3.3 Reporting des risques

- Les risques seront régulièrement présentés lors des Sprint Reviews ainsi que dans les rapports d'avancement pour garantir une gestion transparente et proactive. Les rapports sont partagés avec les encadrants (Mme SBAI Hanae, Mme BOUSQAoui Halima, M. ESSABBAR Driss).
- Les risques majeurs ou critiques feront l'objet d'un rapport détaillé, incluant les actions prises pour y faire face, afin de garantir que l'équipe est bien informée et que les mesures nécessaires sont prises pour les atténuer.
- Un tableau de bord des risques est maintenu dans Azure DevOps, permettant une visualisation rapide de l'état de tous les risques identifiés et de leur évolution au fil du projet.

Conclusion

Le Plan d'Assurance Qualité du projet **Coach Virtuel pour formation interactive avec IA générative** a pour objectif de garantir la livraison d'un produit final de haute qualité, conforme aux attentes des utilisateurs (apprenants, formateurs et administrateurs) et répondant aux exigences pédagogiques et techniques définies. En adoptant une méthodologie SCRUM structurée en sprints de deux semaines et en utilisant des outils performants tels qu'Azure DevOps, JMeter, Selenium, Postman, SonarQube et GitHub, nous assurons une gestion rigoureuse des risques, une qualité constante du code, ainsi qu'une optimisation continue des performances.

Ce plan d'assurance qualité inclut également une gestion efficace des modifications, un suivi régulier de l'avancement du projet, et des procédures adaptées à la spécificité du projet, notamment l'intégration du modèle d'intelligence artificielle Ollama et l'architecture multi-plateformes (Front-End Angular, Back-End Spring Boot, application mobile Flutter). Il constitue le cadre de référence pour assurer le maintien de la qualité tout au long du développement de la plateforme et garantir que le produit final répond aux attentes des utilisateurs, des parties prenantes et aux objectifs académiques du projet.