EduGroupe
Accompagner pour réussir

Qualité des applications

Référence : **DEQUAL001** Durée : **3 jours** Certification : **Aucune**

CONNAISSANCES PREALABLES

• Disposer d'une première expérience de développement et maîtriser un langage de programmation (C#, Java ou C++).

PROFIL DES STAGIAIRES

• Analystes. • Architectes. • Chefs de projets. • Développeurs.

OBJECTIFS

• Connaître les bonnes pratiques d'écriture d'un code incluant la maintenance de l'application. • Connaître les outils nécessaires à la fabrique logicielle pour produire des livrables de qualité. • Appréhender l'offre des outils de tests de performance et de charge. • Appréhender les outils et phases de mise en œuvre d'une intégration continue. • Savoir utiliser Git pour gérer les codes sources.

CERTIFICATION PREPAREE

Aucune

METHODES PEDAGOGIQUES

- Mise à disposition d'un poste de travail par stagiaire
- Remise d'une documentation pédagogique papier ou numérique pendant le stage
- La formation est constituée d'apports théoriques, d'exercices pratiques, de réflexions et de retours d'expérience
- Le suivi de cette formation donne lieu à la signature d'une feuille d'émargement

FORMATEUR

Consultant-Formateur expert Développement

METHODE D'EVALUATION DES ACQUIS

- Auto-évaluation des acquis par le stagiaire via un questionnaire
- Attestation de fin de stage adressée avec la facture

CONTENU DU COURS

Généralités
- Qu'est-ce que la qualité
- Le coût de la non-qualité
- La qualité du code
- La qualité des livrables
- La qualité de la documentation

Génie logiciel
- Couplage fort / couplage faible
- Structuration modulaire des programmes
- Les cycles de vie (en V, en spirale)
- Notion de méthode agile
- Développement piloté par les tests (TDD)
- Développement piloté par le comportement (BDD)

Les tests
- Qu'est-ce que le test ?
- Plan de test, jeux de test
- Tests fonctionnels et non fonctionnels
- Test sen boite blanche vs boite noire
- Tests automatisés : avantages et limites
- Tests unitaires, d'intégration, de recette
- Tests d'IHM : client lourd, client léger
- Test de charge, de performances, de mise à l'échelle
- Exemple avec JUnit et Selenium

Git
- Principe de Git
- Commit
- Branch, merge et gestion des conflits
- Travailler en local ou à distance

- Gestion des versions du code et de la BdD
- Règles pour travailler ensemble
- Les différents outils : GitHub, GitLab, ...
- Utilisation de Jenkins
- Intégration de Maven
- Présentation des alternatives

Jenkins

- Intégration continue
- Déploiement continu