Mastère Spécialisé
Qualité du Software

tech université technologique

tech université technologique

Mastère Spécialisé
Qualité du Software

- Modalité: **en ligne**
- Durée: **12 mois**
- Qualification: **TECH Université Technologique**
- Intensité: **16h/semaine**
- Horaire: **à votre rythme**
- Examens: **en ligne**

Accès au site web: www.techtitute.com/fr/informatique/master/master-qualite-software

Sommaire

01

Présentation

La croissance rapide du secteur et les exigences actuelles du marché ont conduit à un niveau élevé de dette technique dans les projets logiciels. En raison de l'impérieuse nécessité de refléter des réponses rapides aux exigences du client ou de l'entreprise, sans évaluer ou spécifier les détails de la qualité du système. C'est là que se reflète la nécessité de prendre en compte l'évolutivité du projet tout au long de son cycle de vie, ce qui exige des connaissances informatiques axées sur la qualité à partir d'une *top-down*. Ce programme développe les critères, les tâches et les méthodologies avancées pour comprendre la pertinence d'un travail orienté vers la nécessité de mettre en œuvre des politiques de qualité dans les *Software Factories*. Son étude sera entièrement en ligne et d'une durée de 12 mois, ajustée à la méthodologie mise en œuvre par la plus grande université numérique du monde.

Spécialisez-vous dans la Qualité du Software d'un point de vue technique et de gestion; obtenez votre diplôme en 12 mois et faites la différence dans votre environnement professionnel"

Le concept de Dette Technique, actuellement appliqué par un grand nombre de sociétés et d'administrations avec leurs fournisseurs, reflète la manière improvisée dont les projets ont été développés. Cela génère un nouveau coût implicite lié au fait de devoir refaire un projet parce qu'on a adopté une solution rapide et simple au lieu de ce qui devrait être une approche évolutive dans l'évolution du projet.

Depuis quelques années, les projets sont développés très rapidement, dans le but de les conclure avec le client sur la base de critères de prix et de délais, au lieu d'adopter une démarche qualité. Ces décisions ont maintenant des répercussions sur de nombreux fournisseurs et clients.

Ce Mastère Spécialisé permettra au professionnel de l'informatique d'analyser les critères sous-jacents de la Qualité du Software, à tous les niveaux. Des critères tels que la standardisation des bases de données, le découplage entre les composants d'un système d'information, les architectures évolutives, les métriques, la documentation, tant fonctionnelle que technique. Outre les méthodologies de gestion et de développement de projets et d'autres méthodes visant à garantir la qualité, comme les techniques de travail collaboratif, notamment le Pair Programming, qui permettent de faire résider les connaissances dans l'entreprise et non dans les personnes.

La grande majorité des masters de ce type se concentrent sur une technologie, un langage ou un outil. Ce programme est unique en ce sens qu'il sensibilise le professionnel à l'importance de la qualité des logiciels, en réduisant la dette technique des projets par une approche de qualité plutôt que par une approche basée sur l'économie et les délais courts; il dote l'étudiant de connaissances spécialisées, de sorte que la budgétisation des projets puisse être justifiée.

Pour rendre cela possible, TECH Université Technologie a réuni un groupe d'experts dans ce domaine qui transmettront les connaissances et l'expérience les plus récentes. Grâce à un campus virtuel moderne avec des contenus théoriques et pratiques, distribués sous différents formats. Il y aura 10 modules divisés en différents sujets et sous-thèmes qui permettront d'apprendre en 12 mois en utilisant la méthodologie Relearning, qui facilite la mémorisation et l'apprentissage de manière agile et efficace.

Le **Mastère Spécialisé en Qualité du Software** contient le programme éducatif le plus complet et le plus actualisé du marché. Ses principales caractéristiques sont:

- Le développement d'études de cas présentées par des experts en Développement de Logiciels
- Les contenus graphiques, schématiques et éminemment pratiques avec lesquels ils sont conçus fournissent des informations scientifiques et sanitaires essentielles à la pratique professionnelle
- Les exercices pratiques où le processus d'auto-évaluation peut être réalisé pour améliorer l'apprentissage
- Il met l'accent sur les méthodologies innovantes
- Cours théoriques, questions à l'expert et travail de réflexion individuel
- La possibilité d'accéder aux contenus depuis n'importe quel appareil fixe ou portable doté d'une connexion internet

Le Mastère Spécialisé en Qualité du Software analyse les critères qui sous-tendent le sujet à tous les niveaux. Élargissez votre niveau d'expertise. Inscrivez-vous maintenant"

Développez les critères, les tâches et les méthodologies avancées pour comprendre la pertinence d'un travail axé sur la qualité, et apportez des solutions efficaces à votre entreprise ou à votre client"

Un programme axé sur la sensibilisation à l'importance de la qualité des logiciels et à la nécessité de mettre en œuvre des politiques de qualité dans les software Factories.

Apprendre d'une manière pratique et flexible. Partagez votre quotidien avec cette formation 100% en ligne exclusive à TECH Université Technologique.

Le programme comprend, dans son corps enseignant, des professionnels du secteur qui apportent à cette formation l'expérience de leur travail, ainsi que des spécialistes reconnus de grandes sociétés et d'universités prestigieuses.

Grâce à son contenu multimédia développé avec les dernières technologies éducatives, les spécialistes bénéficieront d'un apprentissage situé et contextuel, ainsi, ils se formeront dans un environnement simulé qui leur permettra d'apprendre en immersion et de s'entrainer dans des situations réelles.

La conception de ce programme est axée sur l'Apprentissage par les Problèmes, grâce auquel le professionnel doit essayer de résoudre les différentes situations de la pratique professionnelle qui se présentent tout au long du Mastère Spécialisé. Pour ce faire, l'étudiant sera assisté d'un innovant système de vidéos interactives, créé par des experts reconnus.

02

Objectifs

Le Mastère Spécialisé en Qualité du Software donne aux étudiants une vision claire et spécialisée de l'importance de la qualité dans les processus de développement de logiciels. Ainsi que les outils les plus avancés pour la mise en œuvre des processus DevOps et des systèmes d'assurance qualité. En bref, il fournira des connaissances théoriques et pratiques larges et spécialisées permettant de comprendre le développement de projets dans une perspective moderne et efficace.

Vous pourrez accéder facilement à tous les contenus quand vous le souhaitez. Depuis votre ordinateur ou votre appareil préféré. Vous pouvez également les télécharger pour votre prochaine consultation"

Objectifs généraux

- Développez les critères, les tâches et les méthodologies avancées pour comprendre la pertinence d'un travail axé sur la qualité
- Analyser les facteurs clés de la qualité d'un projet logiciel
- Développer les aspects réglementaires pertinents
- Mise en œuvre de processus DevOps et de systèmes pour l'assurance qualité
- Réduire la dette technique des projets avec une approche de qualité plutôt qu'une approche basée sur l'économie et les délais courts
- Fournir à l'étudiant le savoir-faire pour être capable de mesurer et de quantifier la qualité d'un projet logiciel
- Défendre les propositions économiques des projets sur la base de la qualité

Objectifs spécifiques

Module 1. Qualité du Software. Niveaux de développement TRL

- Développer de manière claire et concise les éléments qui englobent la qualité des logiciels
- Appliquer les modèles et les normes en fonction du système, du produit et du processus logiciel
- Approfondir les normes de qualité ISO appliquées tant en général que dans des parties spécifiques
- Appliquer les normes en fonction de la portée de l'environnement (local, national, international)
- Examiner les niveaux de maturité TRL et les adapter aux différentes parties du projet logiciel à traiter
- Acquérir une capacité d'abstraction pour appliquer un ou plusieurs critères d'éléments et de niveaux de qualité des logiciels
- Distinguer les cas d'application des normes et des niveaux de maturité dans un projet de cas réel simulé

Module 2. Développement de Projets Software. Documentation fonctionnelle et technique

- Déterminer l'influence de la gestion de projet sur la qualité
- Développer les différentes phases d'un projet
- Différencier les concepts de qualité inhérents à la documentation fonctionnelle et technique
- Analyser la phase de collecte des besoins, la phase d'analyse, la gestion de l'équipe et la phase de construction
- Établir les différentes méthodologies de gestion de projets logiciels
- Générer des critères pour décider quelle est la méthodologie la plus appropriée en fonction du type de projet

Module 3. *Testing* de Software. Automatisation des tests

- Établir les différences entre la qualité du produit, la qualité du processus et la qualité d'utilisation
- Comprendre la norme ISO/IEC 15504
- Déterminer les détails du CMMI
- Pour connaître les clés de l'intégration continue, les référentiels et leurs répercussions sur une équipe de développement logiciel
- Établir la pertinence de l'intégration de référentiels pour les projets logiciels Apprenez à les créer avec TFS
- Assimiler l'importance de l'évolutivité des logiciels dans la conception et le développement des systèmes d'information

Module 4. Méthodologies de Gestion de Projets de Software. Méthodologies *Waterfall* par rapport aux méthodologies agiles

- Déterminer en quoi consiste la méthodologie Waterfall
- Approfondissement de la Méthodologie Scrum
- Établir les Différences entre Waterfall et Scrum
- Pour préciser les différences entre les méthodologies Waterfall et Scrum et comment le client le voit
- Examen du Panel Kanban
- Mise en place d'un même projet avec Waterfall et Scrum
- Mise en place d'un projet hybride

Module 5. TDD (*Test Driven Developement*). Conception de Logiciels pilotés par les tests

- Apprendre l'application pratique du TDD et ses possibilités pour tester un projet de logiciel à l'avenir
- Compléter les cas de simulation réels proposés, comme un apprentissage continu de ce concept TDD
- Analyser, dans les cas de simulation, dans quelle mesure les tests peuvent réussir ou échouer d'un point de vue constructif
- Déterminer les alternatives au TDD, en effectuant une analyse comparative entre elles

Module 6. DevOps. Gestion de Qualité du Software

- Analyser les défauts d'un processus traditionnel
- Évaluer les solutions possibles et choisir la plus appropriée
- Comprendre les besoins de l'entreprise et leur impact sur la mise en œuvre
- Évaluer les coûts des améliorations à mettre en œuvre
- Développer un cycle de vie logiciel évolutif, adapté aux besoins réels
- Anticipez les erreurs possibles et évitez-les dès le processus de conception
- Justifier l'utilisation de différents modèles de mise en œuvre

Module 7. DevOps et Intégration Continue. Solutions pratiques avancées en matière de Développement de Software

- Identifier les étapes du cycle de développement et de livraison du logiciel adaptées à des cas particuliers
- Concevoir un processus de livraison de logiciels utilisant l'intégration continue
- Construire et mettre en œuvre l'intégration et le déploiement continus sur la base de sa conception précédente
- Établir des points de contrôle de qualité automatiques sur chaque livraison de logiciel
- Maintenir un processus de livraison de logiciels automatisé et robuste
- Adapter les besoins futurs au processus d'intégration et de déploiement continus
- Analyser et anticiper les vulnérabilités de sécurité pendant le processus de livraison du logiciel et après la livraison du logiciel

Module 8. Conception de Bases de Données (DB). Standardisation et Performance. Qualité du Software

- Évaluer l'utilisation du modèle entité-relation pour la conception préliminaire d'une base de données
- Appliquez une entité, un attribut, une clé, etc., pour une meilleure intégrité des données
- Évaluer les dépendances, les formes et les règles de la normalisation des bases de données
- Se spécialiser dans l'exploitation d'un système d'entrepôt de données OLAP, en développant et en utilisant des tables de faits et de dimensions
- Déterminer les points clés pour les performances de la base de données
- Réaliser des cas de simulation réels proposés comme expérience d'apprentissage continu en matière de conception, de normalisation et de performance des bases de données
- Établir dans les cas de simulation, les options à résoudre dans la création de la base de données d'un point de vue constructif

Module 9. Conception d'Architectures Évolutives. L'Architecture dans le Cycle de Vie des Logiciels

- Développer le concept d'architecture logicielle et ses caractéristiques
- Déterminer les différents types d'évolutivité dans l'architecture logicielle
- Analyser les différents niveaux qui peuvent intervenir dans l'évolutivité du Web
- Acquérir des connaissances spécialisées sur le concept, les étapes et les modèles du cycle de vie des logiciels
- Déterminer l'impact d'une architecture sur le cycle de vie du logiciel, avec ses avantages, ses limites et les outils de soutien
- Réaliser les cas de simulation réels proposés, en tant qu'apprentissage continu de l'architecture et du cycle de vie des logiciels
- Évaluer, dans les cas de simulation, dans quelle mesure ils peuvent rendre la conception de l'architecture réalisable ou inutile

Module 10. Critères de Qualité ISO/IEC 9126. Mesures de Qualité du Software

- Développer le concept de critères de qualité et les aspects pertinents
- Examiner la norme ISO/IEC 9126, ses principaux aspects et ses indicateurs
- Analyser les différentes métriques d'un projet logiciel pour répondre aux évaluations convenues
- Examiner les attributs internes et externes à prendre en compte dans la qualité d'un projet logiciel
- Distinguer les métriques en fonction du type de programmation (structurée, orientée objet, en couches, etc.)
- Réalisation de cas réels de simulation comme processus d'apprentissage continu en matière de mesure de la qualité
- Voir dans les cas de simulation dans quelle mesure elle est réalisable ou inutile, c'est-à-dire d'un point de vue constructif des auteurs

Mettez en valeur votre profil professionnel grâce à cette formation exclusive. Obtenez votre diplôme en 12 mois et de manière pratique avec la méthodologie que seul TECH Université Technologique peut vous offrir"

03

Compétences

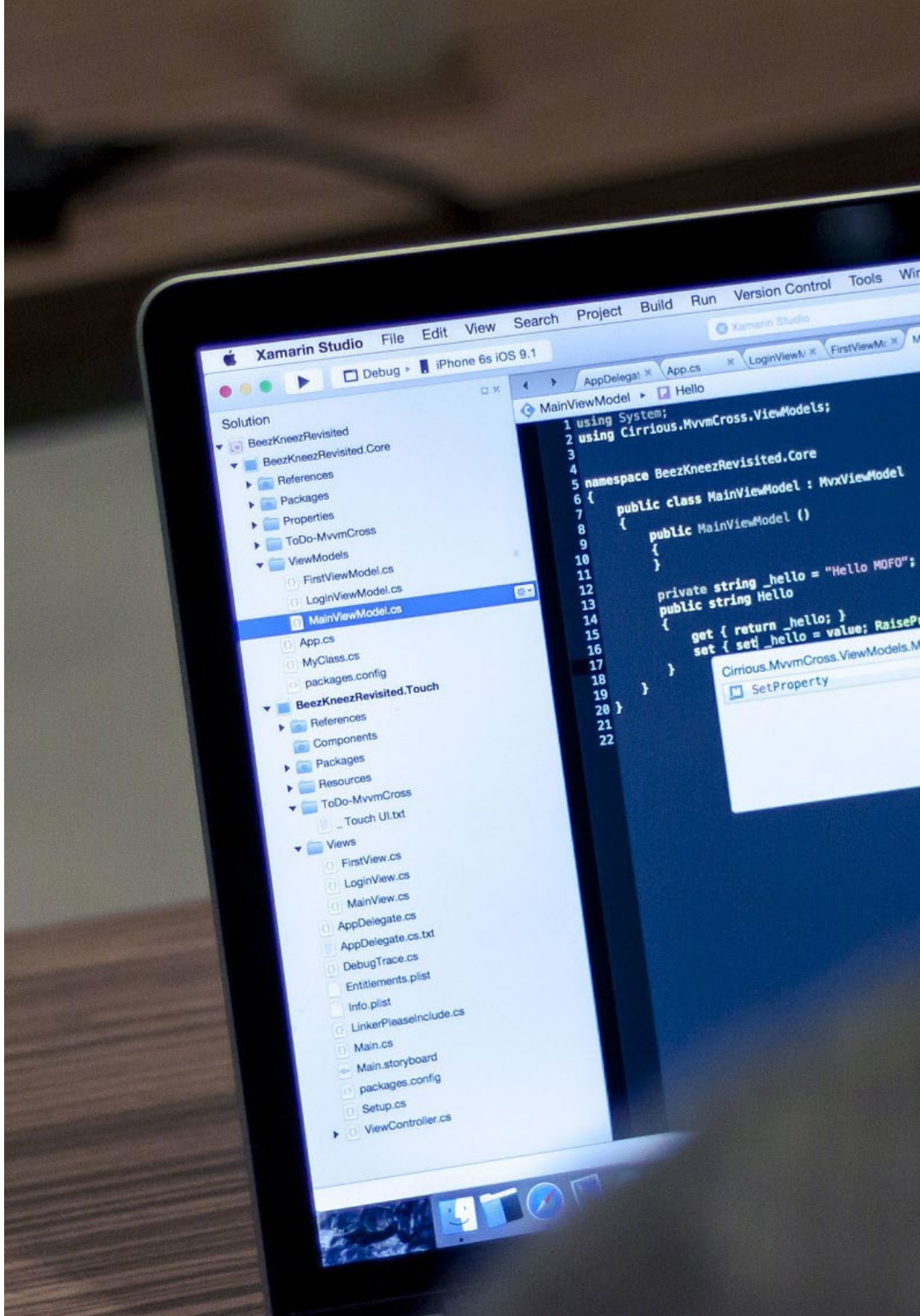
Les diplômés de ce Mastère Spécialisé en Qualité du Software maîtriseront le sujet d'un point de vue à la fois technique et de gestion. Ils seront capables de développer l'approche d'un projet, ainsi que son exécution et de développer une architecture durable, efficace et de qualité dans les projets logiciels qui leur sont présentés. Pour ce faire, le corps enseignant a mis toute son expérience personnelle dans l'élaboration d'une multitude de cas pratiques, qui serviront de contextualisation et d'évolution face à cette "dette technique" qui ne sera plus présente.

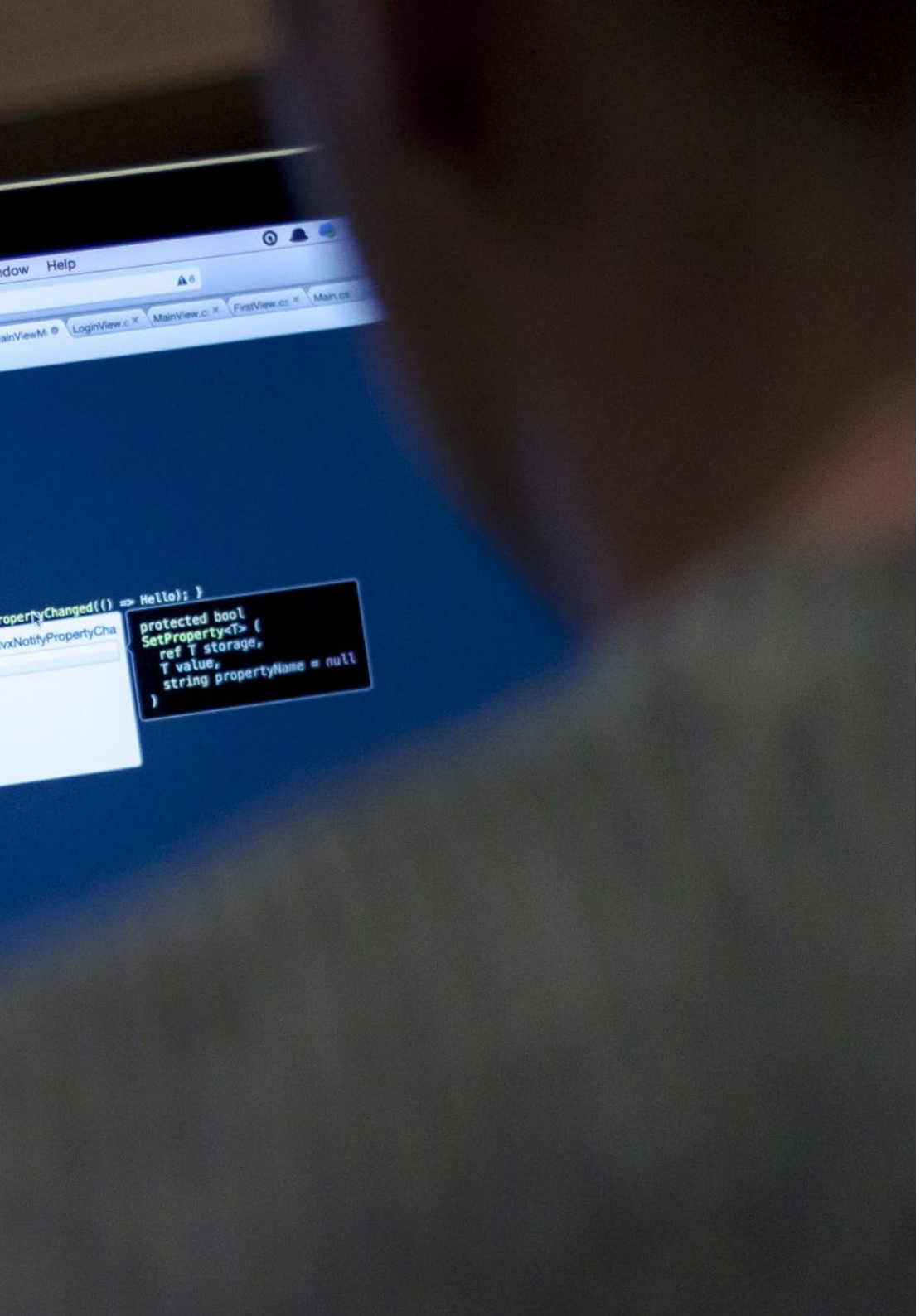
Un professionnel de l'informatique soucieux de la qualité est une valeur montante pour les sociétés de conseil en logiciels et les grandes entreprises. Inscrivez-vous dès maintenant à ce Mastère Spécialisé en Qualité du Software"

Compétences générales

- Réduire la dette technique des projets avec une approche de qualité plutôt qu'une approche basée sur l'économie et les délais courts
- Mesurer et quantifier la qualité d'un projet logiciel
- Exécuter correctement le TDD, afin d'élever les normes de qualité des logiciels
- Justifier la budgétisation des projets axés sur la qualité
- Développer des standards, modèles et normes de qualité
- Examiner les différentes évaluations de la maturité technologique
- Réduire les risques et assurer la maintenance et le contrôle des versions ultérieures
- Maîtriser les phases en lesquelles se décompose un projet

Améliorez vos compétences et découvrez les possibilités infinies de croissance professionnelle qui s'ouvrent avec cette nouvelle expérience"





Compétences spécifiques

- Évaluer un système logiciel en fonction du degré d'avancement du processus du projet
- Aborder ces points de fiabilité, de métrique et d'assurance dans les projets logiciels de manière correcte et stratégique
- Aborder le processus de décision sur la méthodologie à utiliser dans le projet
- Maîtriser les aspects réglementaires essentiels à la création de software
- Développer les *Testing* de manière automatique
- Établir une communication adéquate avec le client, en comprenant la manière dont il perçoit le projet en fonction de la méthodologie appliquée
- Élaborer la liste des exigences de test
- Effectuer l'abstraction, la division en tests plus unitaires et éliminer ce qui ne s'applique pas au bon déroulement des tests du projet logiciel à réaliser
- Mettre à jour la liste des exigences de test de manière mesurée et correcte
- Adapter la culture DevOps aux besoins de l'entreprise
- Développer les dernières pratiques et outils en matière d'intégration et de déploiement continus
- Refactoring et traitement de la gestion et de la coordination des données

04

Direction de la formation

Des enseignants experts, dotés d'un vaste programme d'études dans le domaine des solutions informatiques et du développement et de la recherche de logiciels, dirigent ce Mastère Spécialisé afin de fournir les outils et les connaissances nécessaires au futur diplômé axé sur la qualité dans les processus de développement de logiciels et les outils les plus avancés pour mettre en œuvre les processus DevOps et les systèmes d'assurance qualité. Cette équipe de professionnels guidera l'étudiant à tout moment, pour atteindre les objectifs à distance puisqu'il s'agit d'un programme purement en ligne et suivant la méthodologie la plus avant-gardiste mise en œuvre par TECH.

Des enseignants spécialisés s'engagent à vous fournir le meilleur contenu et à faire de votre processus d'apprentissage une expérience agile et dynamique. Clarifier vos doutes et vous accompagner tout au long du chemin"

Directeur invité international

Fort d'une carrière professionnelle de plus de 30 ans dans le secteur technologique, Daniel St. John est un prestigieux **Ingénieur Informaticien** hautement spécialisé dans la **Qualité des Logiciels**. Dans cette même ligne, il s'est imposé comme un véritable leader dans ce domaine grâce à son approche pragmatique basée sur l'amélioration continue et l'innovation.

Tout au long de sa carrière, il a fait partie d'institutions internationales de référence telles que **General Electric Healthcare** dans l'Illinois. Ainsi, son travail s'est concentré sur l'optimisation des **infrastructures numériques** des organisations dans le but d'améliorer significativement **l'expérience utilisateur**. Grâce à cela, de nombreux patients ont bénéficié de soins plus personnalisés et plus souples, avec un accès plus rapide aux résultats cliniques et au suivi de leur santé. Parallèlement, il a mis en œuvre des solutions technologiques qui ont permis aux professionnels de **prendre des décisions stratégiques** mieux informées sur la base de grands volumes de données.

Il a également combiné ce travail avec la création de projets technologiques de pointe pour maximiser l'efficacité des processus opérationnels des institutions. À cet égard, il a dirigé la **transformation numérique** de nombreuses entreprises appartenant à différents secteurs d'activité. Il a ainsi mis en œuvre des outils émergents tels que **l'Intelligence Artificielle,** le ***Big Data*** ou le ***Machine Learning*** pour automatiser des tâches quotidiennes complexes. En conséquence, ces organisations ont pu s'adapter aux tendances du marché avec immédiateté et assurer leur pérennité sur le long terme.

Il convient de noter que Daniel St. John a participé en tant qu'orateur à plusieurs conférences scientifiques à l'échelle mondiale. Il a ainsi partagé ses vastes connaissances dans des domaines tels que l'adoption de **Méthodologies Agiles,** les **Tests d'Applications** pour assurer la fiabilité des systèmes ou la mise en œuvre de techniques innovantes de ***Blockchain*** qui garantissent la protection des données confidentielles.

M. St. John, Daniel

- Directeur de l'Ingénierie Logicielle chez General Electric Healthcare of Wisconsin, États-Unis
- Chef de l'Ingénierie Logicielle chez Siemens Healthineers, Illinois
- Directeur du Génie Logiciel chez Natus Medical Incorporated, Illinois
- Ingénieur Logiciel Senior chez WMS Gaming of Chicago
- Ingénieur Logiciel Senior chez Siemens Medical Solutions, Illinois
- Master en Stratégie et Analyse des Données de la Lake Forest Graduate School of Management
- Diplôme en Sciences Informatiques de l'Université du Wisconsin-Parkside
- Membre du Conseil Consultatif de l'Institut de Technologie de l'Illinois
- Certifications en : Python pour la Science des Données, Intelligence Artificielle et Développement, SAFe SCRUM et Gestion de Projet

Grâce à TECH, vous pourrez apprendre avec les meilleurs professionnels du monde"

Direction

M. Molina Molina, Jerónimo

- Ingénieur IA et Software Architect. NASSAT-Internet Satellite in Motion
- Consultant Sr. En Hexa Ingénieurs. Introducteur de l'Intelligence Artificielle (ML et CV)
- Expert en solutions basées sur l'intelligence artificielle, dans les domaines de Computer Vision, ML/DL et NLP Étudie actuellement les possibilités d'application de Transformers et de l'apprentissage par renforcement dans le cadre d'un projet de recherche personnel
- Expert Universitaire en Création et Développement d'Entreprises. Bancaixa–FUNDEUN Alicante
- Ingénieur en Informatique. Université d'Alicante
- Master en Intelligence Artificielle. Université Catholique de Avila
- MBA-Executive. Forum Européen Campus Entrepreneurial

Professeurs

M. Pi Morell, Oriol

- Product Owner de Hosting et courrier. CDMON
- Analyste Fonctionnel et Software Engineer dans différentes organisations telles que Fihoca, Atmira, CapGemini
- Enseignant de différents Cours tels que BPM à CapGemini, ORACLE Forms CapGemini, de Processus d'affaires Atmira
- Diplôme d'Ingénieur Technique en Gestion Informatique de l'Université Autonome de Madrid
- Master en Intelligence Artificielle
- Master en Direction et Administration des Entreprises. MBA
- Master en Direction des Systèmes d'Information Expérience Enseignante
- Postgraduate, Postgraduate Modèles de conception. Université Oberta de Catalogne

M. Tenrero Morán, Marcos

- DevOps Engineer–Allot Communications
- Application Lifecycle Management & DevOps - Meta4 Espagne. Cegid
- Ingénieur Automation QA-Meta4 Espagne. Cegid
- Diplômé en Ingénierie de ordinateur de l'Université Rey Juan Carlos
- Développement d'applications professionnelles pour Android-Université Galileo (Guatemala)
- Développement de Services cloud (nodeJs, JavaScript, HTML5) - UPM
- Intégration Continue avec Jenkins - Meta4. Cegid
- Développement Web avec Angular-CLI (4), Ionic et nodeJS. Meta4 - Université Rey Juan Carlos

Mme Martínez Cerrato, Yésica

- Technicien en produits de sécurité électronique chez Securitas Security Espagne
- Analyste de l'intelligence des Affaires à Ricopia Technologies (Alcalá de Henares) Diplôme en Ingénierie Électronique des Communications à l'École Polytechnique Supérieure, Université d'Alcalá
- Responsable de la formation des nouveaux arrivants aux logiciels de gestion commerciale (CRM, ERP, INTRANET), produit et procédures chez Ricopia Technologies (Alcalá de Henares)
- Responsable de la formation de nouveaux boursiers intégrés dans les Classes d'informatique à l'Université d'Alcalá
- Gestionnaire de projets dans le domaine de l'Intégration des Grands Comptes dans les Postes et Télégraphes (Madrid)
- Technicien Informatique-Responsable classes informatiques OTEC, Université d'Alcalá (Alcalá de Henares)
- Professeur de cours d'Informatique à Asociación ASALUMA (Alcalá de Henares)
- Bourse de formation en Informatique à l'OTEC, Université d'Alcalá (Alcalá de Henares)

Dr Peralta Martín-Palomino, Arturo

- CEO y CTO de Prometeus Global Solutions
- CTO chez Korporate Technologies
- CTO de AI Shephers GmbH
- Docteur en ingénierie de informatique de l'Université de Castilla La Mancha
- Doctorat en économie, commerce et finances de l'Université Camilo José Cela. Prix du Doctorat Extraordinaire
- Docteur en Psychologie par l'Université de Castilla la Mancha
- Master en technologies avancées de l'information de l'Université de Castilla la Mancha
- Master MBA+E (Master en Administration des Affaires et Ingénierie Organisationnelle) de l'Université de Castilla la Mancha
- Professeur associé, enseignant en licence et en Master d'Ingénierie Informatique à l'Université de Castilla la Mancha
- Enseignant dans le Master en Big Data et Data Science à l'Université Internationale de Valence
- Enseignant du Master en Industrie 4.0 et le Master en Design Industriel et Développement de produits
- Membre du Groupe de Recherche SMILE à l'Université de Castilla la Mancha.

Notre équipe d'enseignants vous apportera toutes ses connaissances afin que vous soyez au courant des dernières informations sur le sujet"

05

Structure et contenu

La structure et le contenu de ce Mastère Spécialisé ont été développés pour couvrir les sujets les plus importants pour le développement d'un Qualité du Software. Composé de 10 modules d'enseignement, allant du développement de projets logiciels, de la documentation fonctionnelle et technique, du test *test Driven Developement* et des différentes méthodologies, à la mise en oeuvre de solutions pratiques avancées avec DevOps et l'intégration continue, le tout basé sur la qualité du logiciel. Le large contenu multimédia, sélectionné avec rigueur par les enseignants experts, sera d'un grand soutien pour alléger la charge de travail et servir de matériel de référence pour les futures consultations.

Les études de cas, basées sur la réalité, vous aideront à ancrer et contextualiser toute la théorie apprise au cours du programme"

Module 1. Qualité du Software. Niveaux de développement TRL

1.1. Éléments influençant la qualité des logiciels (I). La dette technique
- 1.1.1. La dette technique. Causes et conséquences
- 1.1.2. Qualité des logiciels. Principes généraux
- 1.1.3. Qualité des logiciels avec et sans principes
  - 1.1.3.1. Conséquences
  - 1.1.3.2. La nécessité de l'application des principes de qualité dans les logiciels
- 1.1.4. Qualité des logiciels. Typologie
- 1.1.5. La qualité des logiciels. Caractéristiques spécifiques

1.2. Éléments influençant la qualité des logiciels (II). Coûts associés
- 1.2.1. Qualité des logiciels. Éléments d'influence
- 1.2.2. Qualité des logiciels. Idées fausses
- 1.2.3. Qualité des logiciels. Coûts associés

1.3. Modèles de qualité des logiciels (I). Gestion des connaissances
- 1.3.1. Modèles de qualité générale
  - 1.3.1.1. Gestion de la qualité totale
  - 1.3.1.2. Modèle Européen d'Excellence Commerciale (EFQM)
  - 1.3.1.3. Modèle Six-sigma
- 1.3.2. Modèles de Gestion des Connaissances
  - 1.3.2.1. Modèle Dyba
  - 1.3.2.2. Modèle SEKS
- 1.3.3. Expérience du paradigme Factory et QIP
- 1.3.4. Modèles de qualité d'usage (25010)

1.4. Modèles de qualité des logiciels (III). Qualité des données, des processus et des modèles SEI
- 1.4.1. Modèle de qualité des données
- 1.4.2. Modélisation des processus logiciels
- 1.4.3. *Software & Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. Modèles de la DIE
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. Normes ISO de qualité des logiciels (I). Analyse des normes
- 1.5.1. Normes ISO 9000
  - 1.5.1.1. Normes ISO 9000
  - 1.5.1.2. Famille de normes de qualité ISO (9000)
- 1.5.2. Autres normes ISO relatives à la qualité
- 1.5.3. Normes de modélisation de la qualité (ISO 2501)
- 1.5.4. Normes de mesure de la qualité (ISO 2502n)

1.6. Normes ISO de qualité des logiciels (II). Exigences et évaluation
- 1.6.1. Normes d'exigences de qualité (2503n)
- 1.6.2. Normes sur l'évaluation de la qualité (2504n)
- 1.6.3. ISO/IEC 24744:2007

1.7. Niveaux de développement TRL (I). Niveaux 1 à 4
- 1.7.1. Niveau TRL
- 1.7.2. Niveau 1: principes de base
- 1.7.3. Niveau 2: concept et/ou application
- 1.7.4. Niveau 3: fonction analytique critique
- 1.7.5. Niveau 4: validation des composants dans un environnement de laboratoire

1.8. Niveaux de développement TRL (II). Niveaux de 5 à 9
- 1.8.1. Niveau 5: validation du composant dans un environnement pertinent
- 1.8.2. Niveau 6: modèle de système/sous-système
- 1.8.3. Niveau 7: démonstration en environnement réel
- 1.8.4. Niveau 8: système complet et certifié
- 1.8.5. Niveau 9: succès dans un environnement réel

1.9. Niveaux de développement TRL. Utilisations
- 1.9.1. Exemple d'une entreprise avec un environnement de laboratoire
- 1.9.2. Exemple d'une entreprise de R&D&I
- 1.9.3. Exemple d'une entreprise de R&D&I industriel
- 1.9.4. Exemple d'une entreprise commune laboratoire-ingénierie

1.10. Qualité des logiciels. Principaux détails

1.10.1. Détails méthodologiques
1.10.2. Détails techniques
1.10.3. Détails sur la gestion des projets logiciels
1.10.3.1. Qualité Systèmes d'information
1.10.3.2. Qualité des systèmes informatiques
1.10.3.3. Qualité des produits logiciels

Module 2. Développement de Projets Software. Documentation Fonctionnelle et Technique

2.1. Gestion de projets
2.1.1. Gestion de projet en matière de qualité des logiciels
2.1.2. Gestion de projets. Avantages
2.1.3. Gestion de projets. Typologie
2.2. Méthodologie de la gestion de projet
2.2.1. Méthodologie de la gestion de Projets
2.2.2. Méthodologie de la gestion de projet Typologie
2.2.3. Méthodologie dans la gestion de projets. Application
2.3. Phase d'identification des besoins
2.3.1. Identification des besoins du projet
2.3.2. Gestion des réunions de projet
2.3.3. Documentation à fournir
2.4. Modèle
2.4.1. Phase initiale
2.4.2. Phase d'analyse
2.4.3. Phase de construction
2.4.4. Phase de test
2.4.5. Livraison
2.5. Modèle de données à utiliser
2.5.1. Détermination du nouveau modèle de données
2.5.2. Identification du plan de migration des données
2.5.3. Ensemble de données
2.6. Impact sur d'autres projets
2.6.1. Impact d'un projet. Exemples
2.6.2. Risques liés au projet
2.6.3. Gestion des risques
2.7. *"Must"* du projet
2.7.1. *Must* du projet
2.7.2. Identification du *Must* du projet
2.7.3. Identification des points de mise en œuvre pour la réalisation d'un projet
2.8. L'équipe de construction du projet
2.8.1. Rôles à jouer en fonction du projet
2.8.2. Contact avec les RH pour le recrutement
2.8.3. Livrables et calendrier du projet
2.9. Aspects techniques d'un projet de logiciel
2.9.1. Architecte du projet. Aspects Techniques
2.9.2. Responsables techniques
2.9.3. Construction du projet logiciel
2.9.4. Évaluation de la qualité du code, sonar
2.10. Livrables du projet
2.10.1. Analyse fonctionnelle
2.10.2. Modèles de données
2.10.3. Diagrammes d'état
2.10.4. Documentation technique

Module 3. *Testing de Software.* Automatisation des Tests

3.1. Modèles de qualité des logiciels
- 3.1.1. Qualité du produit
- 3.1.2. Qualité du processus
- 3.1.3. Qualité de l'utilisation

3.2. Qualité du processus
- 3.2.1. Qualité du processus
- 3.2.2. Modèles de maturité
- 3.2.3. Norme ISO 15504
  - 3.2.3.1. Objectifs
  - 3.2.3.2. Contexte
  - 3.2.3.3. Étapes

3.3. Norme ISO/IEC 15504
- 3.3.1. Catégories de processus
- 3.3.2. Processus de développement. Exemple
- 3.3.3. Fragment de profil
- 3.3.4. Étapes

3.4. CMMI (*Capability Maturity Model Integration*)
- 3.4.1. CMMI. Intégration du modèle de maturité de la capacité
- 3.4.2. Modèles et zones. Typologie
- 3.4.3. Domaines de processus
- 3.4.4. Niveaux de capacité
- 3.4.5. Gestion des processus
- 3.4.6. Gestion de projet

3.5. Gestion des changements et des référentiels
- 3.5.1. Gestion des changements logiciels
  - 3.5.1.1. Élément de configuration. Intégration continue
  - 3.5.1.2. Lignes
  - 3.5.1.3. Organigrammes
  - 3.5.1.4. *Branches*
- 3.5.2. Référentiel
  - 3.5.2.1. Contrôle de la version
  - 3.5.2.2. Équipe de travail et utilisation du référentiel
  - 3.5.2.3. Intégration continue dans le référentiel

3.6. *Team Foundation Server* (TFS)
- 3.6.1. Installation et configuration
- 3.6.2. Création d'un projet d'équipement
- 3.6.3. Ajouter du contenu au contrôle de la source
- 3.6.4. *TFS on Cloud*

3.7. *Testing*
- 3.7.1. Motivation pour les tests
- 3.7.2. Test de vérification
- 3.7.3. Tests bêta
- 3.7.4. Mise en œuvre et maintenance

3.8. Essais de charge
- 3.8.1. *Load testing*
- 3.8.2. Tests avec *LoadView*
- 3.8.3. Tests avec K6 *Cloud*
- 3.8.4. Tests avec *Loader*

3.9. Tests unitaires, de stress et d'endurance
- 3.9.1. Raison d'être des tests unitaires
- 3.9.2. Outils de *Unit Testing*
- 3.9.3. Motivation des tests de résistance
- 3.9.4. Test en utilisant le *StressTesting*
- 3.9.5. Motivation pour les tests de résistance
- 3.9.6. Test à l'aide de *LoadRunner*

3.10. Évolutivité. Conception de logiciels évolutifs
- 3.10.1. Scalabilité et architecture logicielle
- 3.10.2. Indépendance entre les couches
- 3.10.3. Couplage entre les couches. Modèles architecturaux

Module 4. Méthodologies de Gestion de Projets de Software. Méthodologies *Waterfall* par rapport aux méthodologies agiles

4.1. Méthodologie *Waterfall*
- 4.1.1. Méthodologie *Waterfall*
- 4.1.2. Méthodologie *Waterfall* Assurance qualité des logiciels
- 4.1.3. Méthodologie *Waterfall*. Exemples

4.2. Méthodologie *Agile*
- 4.2.1. Méthodologie *Agile*
- 4.2.2. Méthodologie *Agile* Influence sur la qualité des logiciels
- 4.2.3. Méthodologie *Agile* Exemples

4.3. Méthodologie Scrum
- 4.3.1. Méthodologie Scrum
- 4.3.2. Manifeste de Scrum
- 4.3.3. Mise en œuvre de Scrum

4.4. Panel Kanban
- 4.4.1. Méthode Kanban
- 4.4.2. Panel Kanban
- 4.4.3. Panel Kanban. Exemples d'application

4.5. Gestion de projet en *Waterfall*
- 4.5.1. Phases d'un projet
- 4.5.2. Vision dans un projet *Waterfall*
- 4.5.3. Livrables à considérer

4.6. Gestion de projet en Scrum
- 4.6.1. Phases d'un projet Scrum
- 4.6.2. Vision dans un projet Scrum
- 4.6.3. Produits livrables à considérer

4.7. W*aterfall* vs. Comparaison de Scrum
- 4.7.1. Approche par projet pilote
- 4.7.2. Projet utilisant *Waterfall*. Exemple
- 4.7.3. Projet utilisant Scrum. Exemple

4.8. Aperçu des clients
- 4.8.1. Documents dans un *Waterfall*
- 4.8.2. Documents dans un Scrum
- 4.8.3. Comparaison

4.9. Structure Kanban
- 4.9.1. Histoires d'utilisateurs
- 4.9.2. *Backlog*
- 4.9.3. Analyse Kanban

4.10. Projets hybrides
- 4.10.1. Construction du projet
- 4.10.2. Gestion de projet
- 4.10.3. Produits livrables à considérer

Module 5. TDD (*Test Driven Developement*). Conception de Software Pilotés par les Tests

5.1. TDD. *Test Driven Development*
- 5.1.1. TDD. *Test Driven Development*
- 5.1.2. TDD. Influence du TDD sur la qualité
- 5.1.3. Conception et développement pilotés par les tests. Exemples

5.2. Cycle TDD
- 5.2.1. Choix d'une exigence
- 5.2.2. Test. Typologie
  - 5.2.2.1. Tests unitaires
  - 5.2.2.2. Test d'intégration
  - 5.2.2.3. Preuves *End To End*
- 5.2.3. Vérification des tests. Défaillances
- 5.2.4. Création de la mise en œuvre
- 5.2.5. Exécution de tests automatisés
- 5.2.6. Élimination des doubles emplois
- 5.2.7. Mise à jour de la liste des exigences
- 5.2.8. Répétition du cycle TDD
- 5.2.9. Cycle TDD. Exemple théoriques et pratiques

5.3. Stratégies de mise en œuvre du TDD
- 5.3.1. Mise en œuvre fictive
- 5.3.2. Mise en œuvre triangulaire
- 5.3.3. Mise en œuvre évidente

5.4. TDD. Utilisation. Avantages et inconvénients
- 5.4.1. Avantages de l'utilisation
- 5.4.2. Limites d'utilisation
- 5.4.3. Équilibre de la qualité dans la mise en œuvre

5.5. TDD. Bonnes pratiques
- 5.5.1. Règles TDD
- 5.5.2. Règle 1: Faites un test préalable qui échoue avant de coder en production
- 5.5.3. Règle 2: ne pas écrire plus d'un test unitaire
- 5.5.4. Règle 3: ne pas écrire plus de code que nécessaire
- 5.5.5. Erreurs et anti-modèles à éviter dans le TDD

5.6. Simulation d'un projet réel pour utiliser TDD (I)
- 5.6.1. Aperçu du projet (Entreprise A)
- 5.6.2. Application du TDD
- 5.6.3. Exercices proposés
- 5.6.4. Exercices *Feedback*

5.7. Simulation d'un projet réel pour utiliser le TDD (II)
- 5.7.1. Aperçu du projet (Entreprise B)
- 5.7.2. Application du TDD
- 5.7.3. Exercices Proposés
- 5.7.4. Exercices *Feedback*

5.8. Simulation d'un projet réel pour utiliser le TDD (III)
- 5.8.1. Aperçu du projet (Entreprise C)
- 5.8.2. Application du TDD
- 5.8.3. Exercices Proposés
- 5.8.4. Exercices *Feedback*

5.9. Alternatives au TDD. *Test Driven Development*
- 5.9.1. TCR (*Test Commit Revert*)
- 5.9.2. BDD (*Behavior Driven Development*)
- 5.9.3. ATDD (*Acceptance Test Driven Development*)
- 5.9.4. TDD. Comparaison théorique

5.10. TDD TCR, BDD et ATDD. Comparaison pratique
- 5.10.1. Définition du problème
- 5.10.2. Résoudre avec TCR
- 5.10.3. Résoudre avec BDD
- 5.10.4. Résoudre avec ATDD

Module 6. DevOps. Gestion de Qualité du Software

6.1. DevOps. Gestion de qualité du software
- 6.1.1. DevOps
- 6.1.2. DevOps et qualité des logiciels
- 6.1.3. DevOps. Avantages de la culture DevOps

6.2. DevOps. Relation avec Agile
- 6.2.1. Livraison accélérée
- 6.2.2. Qualité
- 6.2.3. Réduction des coûts

6.3. Mise en œuvre de DevOps
- 6.3.1. Identification des problèmes
- 6.3.2. Mise en œuvre dans une entreprise
- 6.3.3. Paramètres de mise en œuvre

6.4. Cycle de livraison des logiciels
- 6.4.1. Méthodes de conception
- 6.4.2. Conventions
- 6.4.3. Feuille de route

6.5. Développement d'un code sans bogues
- 6.5.1. Un code facile à maintenir
- 6.5.2. Modèles de développement
- 6.5.3. *Testing* du code
- 6.5.4. Développement de logiciels au niveau du code. Bonnes pratiques

6.6. Automatisation
- 6.6.1. Automatisation. Types de tests
- 6.6.2. Coût de l'automatisation et de la maintenance
- 6.6.3. Automatisation. Atténuer les erreurs

6.7. Déploiements
- 6.7.1. Évaluation des objectifs
- 6.7.2. Conception d'un processus automatique et adapté
- 6.7.3. Retour d'information et réactivité

6.8. Gestion des incidents
- 6.8.1. Préparation aux incidents
- 6.8.2. Analyse et résolution des incidents
- 6.8.3. Éviter les erreurs futures

6.9. Automatisation des déploiements
- 6.9.1. Préparation des déploiements automatisés
- 6.9.2. Évaluation automatique de l'état des processus
- 6.9.3. Métriques et capacité de retour en arrière

6.10. Bonnes pratiques. Évolution de DevOps
- 6.10.1. Guide des meilleures pratiques DevOps
- 6.10.2. DevOps. Méthodologie pour l'équipe
- 6.10.3. Éviter les niches

Module 7. DevOps et Intégration Continue. Solutions Pratiques Avancées en matière de Développement de Software

7.1. Flux de livraison des logiciels
- 7.1.1. Identification des acteurs et des artefacts
- 7.1.2. Conception du flux de livraison du logiciel
- 7.1.3. Flux de livraison du logiciel. exigences entre les étapes

7.2. Automatisation des processus
- 7.2.1. Intégration continue
- 7.2.2. Intégration continue
- 7.2.3. Configuration des environnements et gestion des secrets

7.3. Pipelines déclaratifs
- 7.3.1. Différences entre les pipelines traditionnels, de type code et déclaratifs
- 7.3.2. Pipelines déclaratifs
- 7.3.3. Pipelines déclaratifs dans Jenkins
- 7.3.4. Comparaison des fournisseurs d'intégration continue

7.4. Des portails de qualité et un retour d'information riche
- 7.4.1. Portes de qualité
- 7.4.2. Des normes de qualité avec des portes de qualité. Maintenance
- 7.4.3. Exigences commerciales sur les demandes d'intégration

7.5. Gestion des artefacts
- 7.5.1. Artefacts et cycle de vie
- 7.5.2. Systèmes de stockage et de gestion des artefacts
- 7.5.3. La sécurité dans la gestion des artefacts

7.6. Intégration continue
- 7.6.1. Déploiement continu sous forme de conteneurs
- 7.6.2. Déploiement continu avec PaaS
- 7.6.3. Déploiement continu d'applications mobiles

7.7. Amélioration de l'exécution du pipeline : analyse statique et *Git Hooks*
- 7.7.1. Analyse statique
- 7.7.2. Règles de style de code
- 7.7.3. *Git Hooks* et *tests* unitaires
- 7.7.4. L'impact des infrastructures

7.8. Vulnérabilités dans les conteneurs
- 7.8.1. Vulnérabilités dans les conteneurs
- 7.8.2. Balayage d'images
- 7.8.3. Rapports et alertes périodiques

Module 8. Conception de Bases de Données (DB). Standardisation et Performance. Qualité du Software

8.1. Conception de bases de données
- 8.1.1. Bases de données. Typologie
- 8.1.2. Bases de données utilisées actuellement
  - 8.1.2.1. Relationnel
  - 8.1.2.2. Clé-valeur
  - 8.1.2.3. Basé sur le réseau
- 8.1.3. Qualité des données

8.2. Conception d'un modèle entité-relation (I)
- 8.2.1. Modèle entité-relation. Qualité et documentation
- 8.2.2. Entités
  - 8.2.2.1. Entité forte
  - 8.2.2.2. Entité faible
- 8.2.3. Attributs
- 8.2.4. Ensemble de relations
  - 8.2.4.1. 1 a 1
  - 8.2.4.2. 1 à plusieurs
  - 8.2.4.3. De plusieurs à un
  - 8.2.4.4. Beaucoup à beaucoup
- 8.2.5. Clés
  - 8.2.5.1. Clé primaire
  - 8.2.5.2. Clé étrangère
  - 8.2.5.3. Clé primaire de l'entité faible
- 8.2.6. Restrictions
- 8.2.7. Cardinalité
- 8.2.8. Héritage
- 8.2.9. Agrégation

8.3. Modèle entité-relation (II). Outils
- 8.3.1. Modèle entité-relation. Outils
- 8.3.2. Modèle entité-relation. Exemple pratique
- 8.3.3. Modèle entité-relation réalisable
  - 8.3.3.1. Echantillon visuel
  - 8.3.3.2. Échantillon en représentation de tableau

8.4. Normalisation (I) des bases de données (DB). Considérations sur la qualité des logiciels
- 8.4.1. Normalisation et qualité des DB
- 8.4.2. Dépendances
  - 8.4.2.1. Dépendance fonctionnelle
  - 8.4.2.2. Propriétés de la dépendance fonctionnelle
  - 8.4.2.3. Propriétés inférées
- 8.4.3. Clés

8.5. Normalisation (II) de la base de données (BD). Formes normales et règles de Codd
- 8.5.1. Formes normales
  - 8.5.1.1. Première forme normale (1FN)
  - 8.5.1.2. Deuxième forme normale (2FN)
  - 8.5.1.3. Troisième forme normale (3FN)
  - 8.5.1.4. Forme normale de Boyce-Codd (BCNF)
  - 8.5.1.5. Quatrième forme normale (4FN)
  - 8.5.1.6. Cinquième forme normale (5FN)
- 8.5.2. Les règles de Codd
  - 8.5.2.1. Règle 1: Information
  - 8.5.2.2. Règle 2: accès garanti
  - 8.5.2.3. Règle 3: Traitement systématique des valeurs nulles
  - 8.5.2.4. Règle 4: description de la base de données
  - 8.5.2.5. Règle 5: Sous-langage intégral
  - 8.5.2.6. Règle 6: Voir la mise à jour
  - 8.5.2.7. Règle 7: Insertion et mise à jour
  - 8.5.2.8. Règle 8: indépendance physique
  - 8.5.2.9. Règle 9: indépendance logique
  - 8.5.2.10. Règle 10: indépendance de l'intégrité
    - 8.5.2.10.1. Règles d'intégrité
  - 8.5.2.11. Règle 11: distribution
  - 8.5.2.12. Règle 12: Non-subversion
- 8.5.3. Exemple pratique

8.6. Entrepôt de données / système OLAP
- 8.6.1. Entrepôt de données
- 8.6.2. Tableau des faits
- 8.6.3. Tableau des dimensions
- 8.6.4. Création du système OLAP. Outils

8.7. Performances des bases de données (DB)
- 8.7.1. Optimisation de l'index
- 8.7.2. Optimisation des requêtes
- 8.7.3. Partitionnement des tables

8.8. Simulation du projet réel pour la conception du DB (I)
- 8.8.1. Aperçu du projet (Entreprise A)
- 8.8.2. Application de la conception de bases de données
- 8.8.3. Exercices proposés
- 8.8.4. Exercices proposés. *Feedback*

8.9. Simulation d'un projet réel pour la conception de BD (II)
- 8.9.1. Aperçu du projet (Entreprise B)
- 8.9.2. Application de la conception de bases de données
- 8.9.3. Exercices proposés
- 8.9.4. Exercices proposés. *Feedback*

8.10. Pertinence de l'optimisation des bases de données dans la Qualité des Logiciels
- 8.10.1. Optimisation de la conception
- 8.10.2. Optimisation du code de requête
- 8.10.3. Optimisation du code des procédures stockées
- 8.10.4. Influence des *Triggers* sur la qualité des logiciels. Recommandations d'utilisation

Module 9. Conception d'Architectures Évolutives. L'Architecture dans le Cycle de Vie des Logiciels

9.1. Conception d'architectures évolutives(I)
- 9.1.1. Architectures évolutives
- 9.1.2. Principes d'une architecture évolutive
  - 9.1.2.1. Fiable
  - 9.1.2.2. Évolutif
  - 9.1.2.3. Maintenable
- 9.1.3. Types d'extensibilité
  - 9.1.3.1. Vertical
  - 9.1.3.2. Horizontal
  - 9.1.3.3. Combinaison

9.2. Architectures de DDD (*Domain-Driven Design*)
- 9.2.1. Le Modèle DDD. Orientation du domaine
- 9.2.2. Couches, répartition des responsabilités et modèles de conception
- 9.2.3. Le découplage comme base de la qualité

9.3. Conception d'architectures évolutives (II). Avantages, limites et stratégies de conception
- 9.3.1. Architecture évolutive. Bénéfices
- 9.3.2. Architecture évolutive. Limites
- 9.3.3. Stratégies pour le développement d'architectures évolutives (Tableau descriptif)

9.4. Cycle de qualité des logiciels (I). Étapes
- 9.4.1. Cycle de vie d'une software
  - 9.4.1.1. Phase de planification
  - 9.4.1.2. Phase d'analyse
  - 9.4.1.3. Phase de conception
  - 9.4.1.4. Phase de mise en œuvre
  - 9.4.1.5. Phase de test
  - 9.4.1.6. Phase d'installation/déploiement
  - 9.4.1.7. Phase d'utilisation et de maintenance

9.5. Modèles de cycle de vie des logiciels
- 9.5.1. Modèle en cascade
- 9.5.2. Modèle répétitif
- 9.5.3. Modèle en spirale
- 9.5.4. Modèle *Big Bang*

9.6. Cycle de vie des logiciels (II). Automatisation
- 9.6.1. Cycles de vie du développement logiciel. Solutions
  - 9.6.1.1. Intégration continue et développement continu (CI/CD)
  - 9.6.1.2. Méthodologies Agiles
  - 9.6.1.3. DevOps / opérations de production
- 9.6.2. Tendances futures
- 9.6.3. Exemples pratiques

9.7. L'architecture logicielle dans le cycle de vie du logiciel
- 9.7.1. Bénéfices
- 9.7.2. Limites
- 9.7.3. Outils

9.8. Simulation d'un projet réel pour la conception de d'architecture (I)
- 9.8.1. Aperçu du projet (Entreprise A)
- 9.8.2. Application de la conception de l'architecture logicielle
- 9.8.3. Exercices Proposés
- 9.8.4. Exercices Proposés. *Feedback*

9.9. Simulation d'un projet réel pour la conception de d'architecture (II)
- 9.9.1. Aperçu du projet (Entreprise B)
- 9.9.2. Application de la conception de l'architecture logicielle
- 9.9.3. Exercices Proposés
- 9.9.4. Exercices Proposés. *Feedback*

9.10. Simulation d'un projet réel pour la conception de d'architecture (III)
- 9.10.1. Aperçu du projet (Entreprise C)
- 9.10.2. Application de la conception de l'architecture logicielle
- 9.10.3. Exercices Proposés
- 9.10.4. Exercices Proposés. *Feedback*

Module 10. Critères de Qualité ISO, IEC 9126. Mesures de Qualité du Software

10.1. Critères de qualité Norme ISO, IEC 9126
- 10.1.1. Critères de qualité
- 10.1.2. Qualité des logiciels. Justification. Norme ISO, IEC 9126
- 10.1.3. La mesure de la qualité des logiciels comme indicateur clé

10.2. Critères de qualité des logiciels. Caractéristiques
- 10.2.1. Fiabilité
- 10.2.2. Fonctionnalité
- 10.2.3. Efficacité
- 10.2.4. Utilisabilité
- 10.2.5. Maintenance
- 10.2.6. Portabilité
- 10.2.7. Sécurité

10.3. Norme ISO, CEI 9126 (I). Présentation
- 10.3.1. Description de la Norme ISO, IEC 9126
- 10.3.2. Fonctionnalité
- 10.3.3. Fiabilité
- 10.3.4. Utilisabilité
- 10.3.5. Maintenance
- 10.3.6. Portabilité
- 10.3.7. Qualité de l'utilisation
- 10.3.8. Mesures de la qualité des logiciels
- 10.3.9. Les mesures de la qualité dans la norme ISO 9126

10.4. Norme ISO, CEI 9126 (II). Modèles de McCall et Boehm
- 10.4.1. Modèle de McCall: facteurs de qualité
- 10.4.2. Modèle Boehm
- 10.4.3. Niveau intermédiaire. Caractéristiques

10.5. Mesures de la qualité des logiciels (I). Éléments
- 10.5.1. Mesure
- 10.5.2. Métriques
- 10.5.3. Indicateur
  - 10.5.3.1. Types d'indicateurs
- 10.5.4. Mesures et modèles
- 10.5.5. Portée des métriques logicielles
- 10.5.6. Classification des métriques logicielles

10.6. Mesure de la qualité des logiciels (II). Pratique de la mesure
- 10.6.1. Collecte de données métriques
- 10.6.2. Mesure des attributs internes du produit
- 10.6.3. Mesure des attributs externes du produit
- 10.6.4. Mesure des ressources
- 10.6.5. Métriques pour les systèmes orientés objet

10.7. Conception d'un indicateur unique de qualité des logiciels
- 10.7.1. Indicateur unique en tant que scoreur global
- 10.7.2. Développement, justification et application des indicateurs
- 10.7.3. Exemples d'application. Besoin de connaître le détail

10.8. Simulation d'un projet réel pour la mesure de la qualité (I)
- 10.8.1. Aperçu du projet (Entreprise A)
- 10.8.2. Application de la mesure de la qualité
- 10.8.3. Exercices Proposés
- 10.8.4. Exercices Proposés. *Feedback*

10.9. Simulation d'un projet réel pour la mesure de la qualité (II)
- 10.9.1. Aperçu du projet (Entreprise B)
- 10.9.2. Application de la mesure de la qualité
- 10.9.3. Exercices Proposés
- 10.9.4. Exercices Proposés. *Feedback*

10.10. Simulation d'un projet réel pour la mesure de la qualité (III)
- 10.10.1. Aperçu du projet (Entreprise C)
- 10.10.2. Application de la mesure de la qualité
- 10.10.3. Exercices Proposés
- 10.10.4. Exercices Proposés. *Feedback*

Vous aurez accès à un contenu unique et spécialisé. Sélectionné par des enseignants experts pour un diplôme qui fera transcender votre profil professionnel"

06

Méthodologie

Ce programme de formation offre une manière différente d'apprendre. Notre méthodologie est développée à travers un mode d'apprentissage cyclique: ***le Relearning***.

Ce système d'enseignement est utilisé, par exemple, dans les écoles de médecine les plus prestigieuses du monde et a été considéré comme l'un des plus efficaces par des publications de premier plan telles que le ***New England Journal of Medicine.***

Découvrez Relearning, un système qui renonce à l'apprentissage linéaire conventionnel pour vous emmener à travers des systèmes d'enseignement cycliques: une façon d'apprendre qui s'est avérée extrêmement efficace, en particulier dans les matières qui exigent la mémorisation"

Étude de Cas pour mettre en contexte tout le contenu

Notre programme offre une méthode révolutionnaire de développement des compétences et des connaissances. Notre objectif est de renforcer les compétences dans un contexte changeant, compétitif et hautement exigeant.

Avec TECH, vous pouvez expérimenter une manière d'apprendre qui ébranle les fondations des universités traditionnelles du monde entier"

Vous bénéficierez d'un système d'apprentissage basé sur la répétition, avec un enseignement naturel et progressif sur l'ensemble du cursus.

L'étudiant apprendra, par des activités collaboratives et des cas réels, à résoudre des situations complexes dans des environnements commerciaux réels.

Une méthode d'apprentissage innovante et différente

Cette formation TECH est un programme d'enseignement intensif, créé de toutes pièces, qui propose les défis et les décisions les plus exigeants dans ce domaine, tant au niveau national qu'international. Grâce à cette méthodologie, l'épanouissement personnel et professionnel est stimulé, faisant ainsi un pas décisif vers la réussite. La méthode des cas, technique qui constitue la base de ce contenu, permet de suivre la réalité économique, sociale et professionnelle la plus actuelle.

> *Notre programme vous prépare à relever de nouveaux défis dans des environnements incertains et à réussir votre carrière"*

La méthode des cas est le système d'apprentissage le plus largement utilisé dans les meilleures écoles d'informatique du monde depuis qu'elles existent. Développée en 1912 pour que les étudiants en Droit n'apprennent pas seulement le droit sur la base d'un contenu théorique, la méthode des cas consiste à leur présenter des situations réelles complexes afin qu'ils prennent des décisions éclairées et des jugements de valeur sur la manière de les résoudre. En 1924, elle a été établie comme méthode d'enseignement standard à Harvard.

Dans une situation donnée, que doit faire un professionnel? C'est la question à laquelle nous sommes confrontés dans la méthode des cas, une méthode d'apprentissage orientée vers l'action. Tout au long du programme, les étudiants seront confrontés à de multiples cas réels. Ils devront intégrer toutes leurs connaissances, faire des recherches, argumenter et défendre leurs idées et leurs décisions.

Relearning Methodology

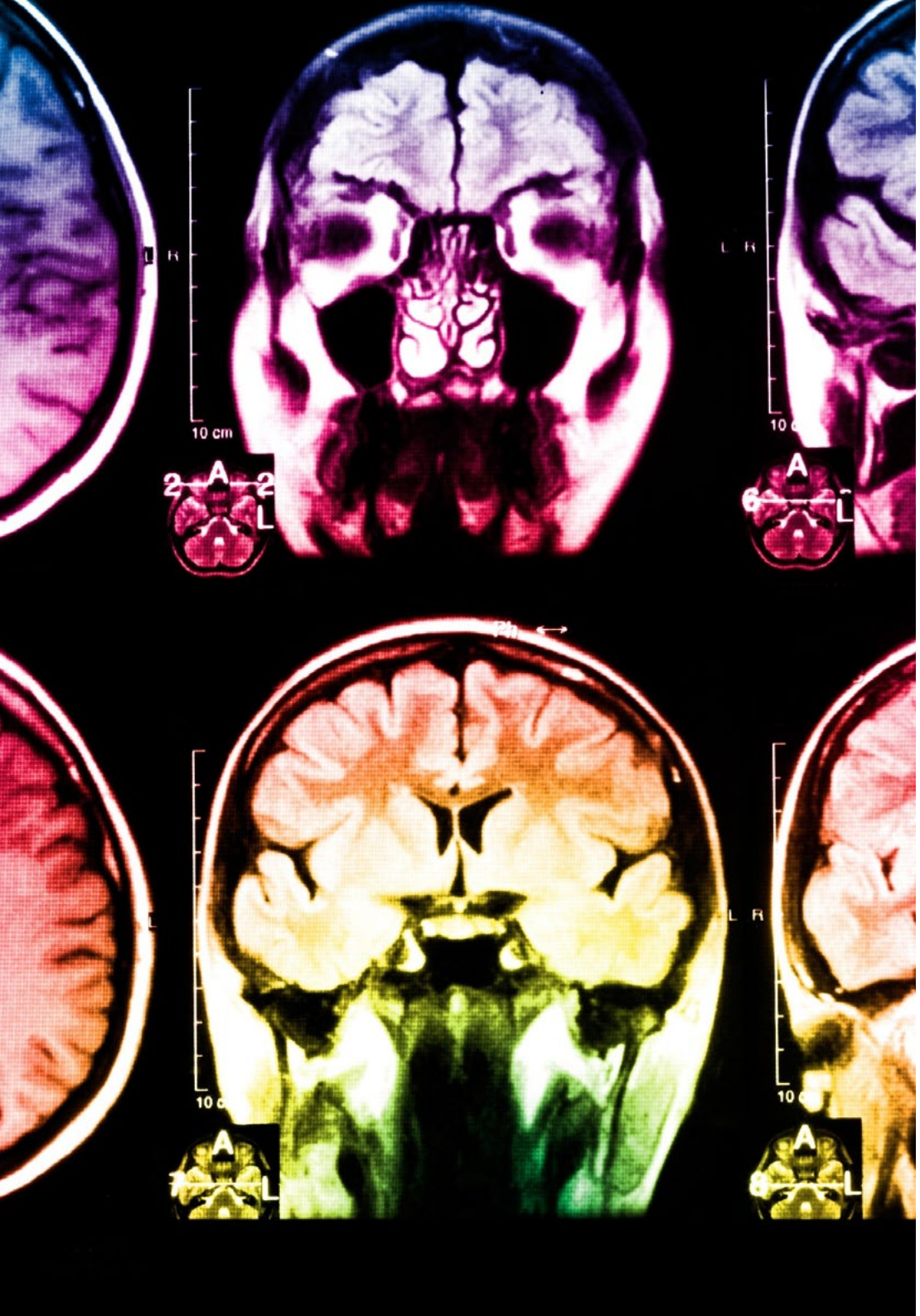
TECH combine efficacement la méthodologie des Études de Cas avec un système d'apprentissage 100% en ligne basé sur la répétition, qui associe différents éléments didactiques dans chaque leçon.

Nous enrichissons l'Étude de Cas avec la meilleure méthode d'enseignement 100% en ligne: le Relearning.

En 2019, nous avons obtenu les meilleurs résultats d'apprentissage de toutes les universités en ligne du monde.

À TECH, vous apprendrez avec une méthodologie de pointe conçue pour former les managers du futur. Cette méthode, à la pointe de la pédagogie mondiale, est appelée Relearning.

Notre université est la seule université autorisée à utiliser cette méthode qui a fait ses preuves. En 2019, nous avons réussi à améliorer les niveaux de satisfaction globale de nos étudiants (qualité de l'enseignement, qualité des supports, structure des cours, objectifs...) par rapport aux indicateurs de la meilleure université en ligne.



Dans notre programme, l'apprentissage n'est pas un processus linéaire, mais se déroule en spirale (apprendre, désapprendre, oublier et réapprendre). Par conséquent, chacun de ces éléments est combiné de manière concentrique. Cette méthodologie a permis de former plus de 650.000 diplômés universitaires avec un succès sans précédent dans des domaines aussi divers que la biochimie, la génétique, la chirurgie, le droit international, les compétences en gestion, les sciences du sport, la philosophie, le droit, l'ingénierie, le journalisme, l'histoire, les marchés financiers et les instruments. Tout cela dans un environnement très exigeant, avec un corps étudiant universitaire au profil socio-économique élevé et dont l'âge moyen est de 43,5 ans.

Le Relearning vous permettra d'apprendre avec moins d'efforts et plus de performance, en vous impliquant davantage dans votre formation, en développant un esprit critique, en défendant des arguments et en contrastant les opinions: une équation directe vers le succès.

À partir des dernières preuves scientifiques dans le domaine des neurosciences, non seulement nous savons comment organiser les informations, les idées, les images et les souvenirs, mais nous savons aussi que le lieu et le contexte dans lesquels nous avons appris quelque chose sont fondamentaux pour notre capacité à nous en souvenir et à le stocker dans l'hippocampe, pour le conserver dans notre mémoire à long terme.

De cette manière, et dans ce que l'on appelle Neurocognitive context-dependent e-learning, les différents éléments de notre programme sont reliés au contexte dans lequel le participant développe sa pratique professionnelle.

Ce programme offre le support matériel pédagogique, soigneusement préparé pour les professionnels:

Support d'étude

Tous les contenus didactiques sont créés par les spécialistes qui enseigneront le cours, spécifiquement pour le cours, afin que le développement didactique soit vraiment spécifique et concret.

Ces contenus sont ensuite appliqués au format audiovisuel, pour créer la méthode de travail TECH en ligne. Tout cela, avec les dernières techniques qui offrent des pièces de haute qualité dans chacun des matériaux qui sont mis à la disposition de l'étudiant.

Cours magistraux

Il existe des preuves scientifiques de l'utilité de l'observation par un tiers expert.

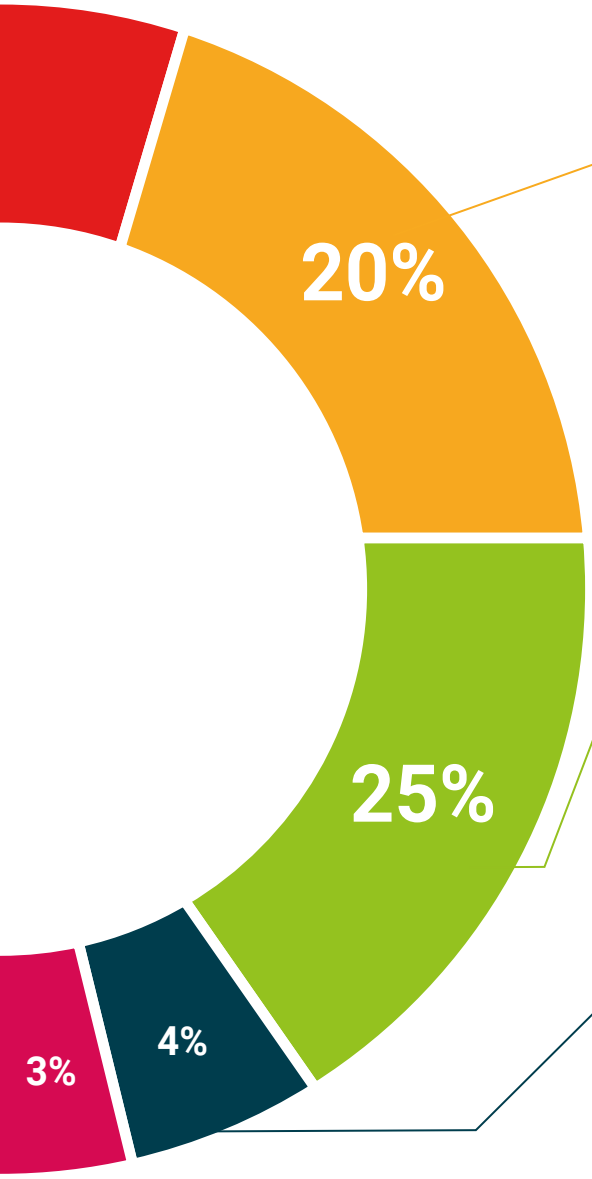
La méthode "Learning from an Expert" renforce les connaissances et la mémoire, et donne confiance dans les futures décisions difficiles.

Pratiques en compétences et aptitudes

Les étudiants réaliseront des activités visant à développer des compétences et des aptitudes spécifiques dans chaque domaine. Des activités pratiques et dynamiques pour acquérir et développer les compétences et aptitudes qu'un spécialiste doit développer dans le cadre de la mondialisation dans laquelle nous vivons.

Lectures complémentaires

Articles récents, documents de consensus et directives internationales, entre autres. Dans la bibliothèque virtuelle de TECH, l'étudiant aura accès à tout ce dont il a besoin pour compléter sa formation.



Case studies

Ils réaliseront une sélection des meilleures études de cas choisies spécifiquement pour ce diplôme. Des cas présentés, analysés et tutorés par les meilleurs spécialistes de la scène internationale.

Résumés interactifs

L'équipe TECH présente les contenus de manière attrayante et dynamique dans des pilules multimédia comprenant des audios, des vidéos, des images, des diagrammes et des cartes conceptuelles afin de renforcer les connaissances.

Ce système éducatif unique pour la présentation de contenu multimédia a été récompensé par Microsoft en tant que "European Success Story".

Testing & Retesting

Les connaissances de l'étudiant sont périodiquement évaluées et réévaluées tout au long du programme, par le biais d'activités et d'exercices d'évaluation et d'auto-évaluation, afin que l'étudiant puisse vérifier comment il atteint ses objectifs.

07

Diplôme

Le Mastère Spécialisé en Qualité du Software vous garantit, en plus de la formation la plus rigoureuse et la plus actuelle, l'accès à un diplôme universitaire de Mastère Spécialisé délivré par TECH Université Technologique.

Finalisez cette formation avec succès et recevez votre Mastère Spécialisé sans avoir à vous soucier des déplacements ou des démarches administratives”

Ce **Mastère Spécialisé en Qualité du Software** contient le programme le plus complet et le plus actuel du marché.

Après avoir réussi l'évaluation, l'étudiant recevra par courrier postal* avec accusé de réception son correspondant diplôme de **Mastère Spécialisé** délivré par **TECH Université Technologique**.

Le diplôme délivré par **TECH Université Technologique** indiquera la note obtenue lors du Mastère Spécialisé, et répond aux exigences communément demandées par les bourses d'emploi, les concours et les commissions d'évaluation des carrières professionnelles.

Diplôme: **Mastère Spécialisé en Qualité du Software**

N.º d'Heures Officielles: **1.500 h.**

tech université technologique

Délivre le présent

DIPLÔME

à

Mme/M. _____, avec nº d'identification _____

Pour avoir finalisé et accrédité avec succès le programme de

MASTÈRE SPÉCIALISÉ

en

Qualité du Software

Il s'agit d'un diplôme spécialisé octroyé par cette Université d'une durée de 1.500 heures, débutant le dd/mm/aaaa et finalisant le dd/mm/aaaa.

TECH est une Institution Privée d'Enseignement Supérieur reconnue par le Ministère de l'Enseignement Public depuis le 28 juin 2018.

Fait le 17 juin 2020

Pre Tere Guevara Navarro
Rectrice

Code Unique TECH: AFWOR23S techtitute.com/diplômes

Mastère Spécialisé en Qualité du Software

Types de matière	Heures
Obligatoire (OB)	1.500
Optionnelle (OP)	0
Stages Externes (ST)	0
Mémoire du Mastère (MDM)	0
	Total 1.500

Distribution Générale du Programme d'Études

Cours	Matière	Heures	Type
1º	Qualité du Software. Niveaux de développement TRL	150	OB
1º	Développement de Projets Software. Documentation Fonctionnelle et Technique	150	OB
1º	Testing de Software. Automatisation des Tests	150	OB
1º	Méthodologies de Gestion de Projets de Software. Méthodologies Waterfall par rapport aux méthodologies agiles	150	OB
1º	TDD (Test Driven Developement). Conception de Software Pilotés par les Tests	150	OB
1º	DevOps. Gestion de Qualité du Software	150	OB
1º	DevOps et Intégration Continue. Solutions Pratiques Avancées en matière de Développement de Software	150	OB
1º	Conception de Bases de Données (DB). Standardisation et Performance. Qualité du Software	150	OB
1º	Conception d'Architectures Évolutives. L'Architecture dans le Cycle de Vie des Logiciels	150	OB
1º	Critères de Qualité ISO, IEC 9126. Mesures de Qualité du Software	150	OB

Pre Tere Guevara Navarro
Rectrice

tech université technologique

*Si l'étudiant souhaite que son diplôme version papier possède l'Apostille de La Haye, TECH EDUCATION fera les démarches nécessaires pour son obtention moyennant un coût supplémentaire.

tech université technologique

Mastère Spécialisé
Qualité du Software

- Modalité: **en ligne**
- Durée: **12 mois**
- Qualification: **TECH Université Technologique**
- Intensité: **16h/semaine**
- Horaire: **à votre rythme**
- Examens: **en ligne**

Mastère Spécialisé
Qualité du Software

tech université technologique