Mastère Avancé

Ingénierie et Qualité du Software

tech université technologique

tech université technologique

Mastère Avancé
Ingénierie et Qualité du Software

- Modalité: **en ligne**
- Durée: **2 ans**
- Qualification: **TECH Université Technologique**
- Intensité: **16h/semaine**
- Horaire: **à votre rythme**
- Examens: **en ligne**

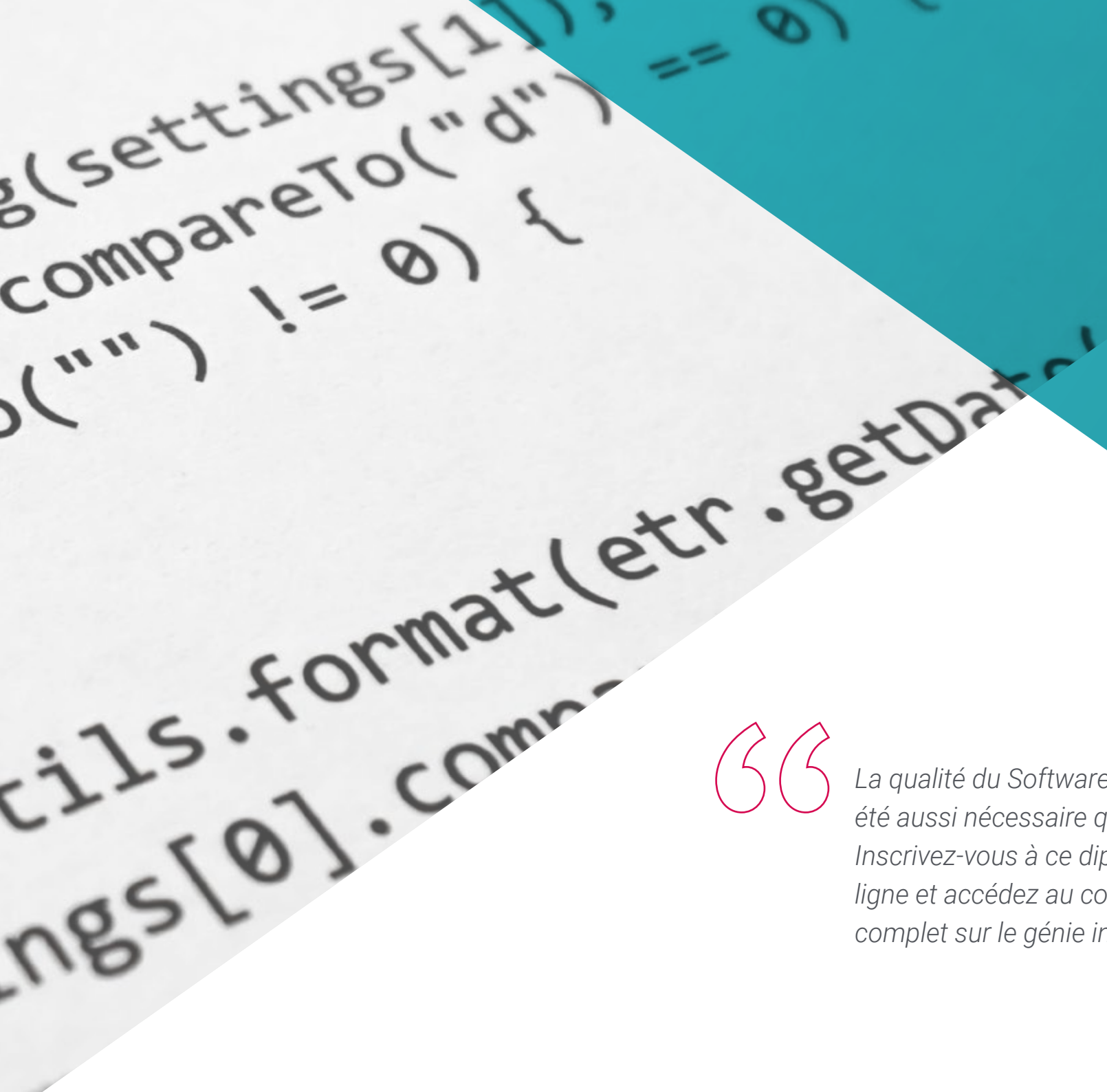
Accès au site web: www.techtitute.com/fr/informatique/mastere-avance/mastere-avance-ingenierie-qualite-software

Sommaire

01

Présentation

Le développement de la technologie et les progrès des systèmes informatiques ont créé une forte demande de la part de l'industrie pour des professionnels capables de gérer parfaitement l'Ingénierie Software, depuis les outils les plus sophistiqués et précis pour sa conception et sa mise en œuvre, jusqu'aux protocoles de sécurité qui garantissent un accès inviolable à leurs données. Pour cette raison, et dans le but d'offrir aux spécialistes la possibilité de se mettre à jour avec les informations les plus pointues sur l'ingénierie appliquée à ce domaine, TECH a développé ce diplôme multidisciplinaire et 100% en ligne. Il s'agit d'un programme conçu par des experts qui réunit, en une seule formation, 3 000 heures du meilleur contenu sur les systèmes informatiques et la qualité des Softwares, et qui aidera le diplômé à améliorer ses compétences informatiques de manière immédiate et spécifique.



La qualité du Software n'a jamais été aussi nécessaire qu'aujourd'hui. Inscrivez-vous à ce diplôme en ligne et accédez au contenu le plus complet sur le génie informatique"

L'ingénierie informatique a connu une croissance exponentielle ces dernières années en raison de l'évolution de la technologie et des outils numériques, notamment pour tout ce qui concerne le web et sa convivialité. C'est pourquoi, aujourd'hui, le développement de Softwares pour différentes fonctions est à l'ordre du jour et le catalogue de programmes s'étoffe. Cependant, cette quantité n'est pas toujours synonyme de qualité, c'est pourquoi on trouve souvent des applications qui ne font pas leur travail, qui renvoient des erreurs ou qui portent gravement atteinte à la sécurité des entreprises. C'est pourquoi la demande d'ingénieurs informaticiens spécialisés dans ce domaine est en augmentation.

C'est pourquoi TECH a décidé de concevoir ce Mastère Avancé en Ingénierie et Qualité du Software, un programme multidisciplinaire conçu par des experts du domaine et conçu de telle sorte que le diplômé pourra trouver tous les outils nécessaires pour mettre à jour ses connaissances de manière complète et sur la base des dernières évolutions du secteur. Il s'agit d'une formation qui combine théorie et pratique en 20 modules dans lesquels l'ingénierie software et la qualité des projets de systèmes informatiques sont étudiées en profondeur.

Tout au long des 24 mois que dure ce programme 100% en ligne, l'ingénieur aura accès au meilleur syllabus qui lui permettra d'améliorer ses compétences en matière de standardisation des bases de données et de découplage entre les composants du système, ainsi que d'élargir ses connaissances en matière d'architectures évolutives, de métriques de qualité et de travail collaboratif.

En outre, vous aurez accès à une classe virtuelle moderne et de pointe où vous trouverez tous les outils qui vous permettront de tirer le meilleur parti de ce diplôme, y compris des centaines d'heures de support supplémentaire dans différents formats. Tout ce contenu peut être téléchargé sur n'importe quel appareil doté d'une connexion internet, ce qui vous garantit de pouvoir le consulter quand vous le souhaitez et quand vous en avez besoin.

Ce **Mastère Avancé en Ingénierie et Qualité du Software** contient le programme le éducatif plus complet et le plus à jour du marché Ses principales caractéristiques sont:

- Le développement d'études de cas présentées par des experts en ingénierie
- Le contenu graphique, schématique et éminemment pratique du programme fournit des informations scientifiques et pratiques sur les disciplines essentielles à la pratique professionnelle
- Les exercices pratiques où le processus d'auto-évaluation peut être réalisé afin d'améliorer l'apprentissage
- Son accent particulier sur les méthodologies innovantes en conception et formation de Software
- Les cours théoriques, questions à l'expert, forums de discussion sur des sujets controversés et travail de réflexion individuel
- La possibilité d'accéder aux contenus depuis n'importe quel appareil fixe ou portable doté d'une connexion internet

Vous aurez accès aux exercices HTML et à leurs réponses, afin de mettre en pratique vos connaissances et la théorie développée tout au long de la programmation"

Grâce au module dédié au DevOps, vous disposerez des connaissances les plus étendues et les plus complètes pour accélérer le cycle de développement des Softwares et assurer une livraison continue de haute qualité"

Son corps enseignant comprend des professionnels du domaine de l'Ingénierie qui qui apportent leur expérience professionnelle, à ce programme, ainsi que des spécialistes reconnus par des sociétés de référence et des universités prestigieuses.

Son contenu multimédia, développé avec les dernières technologies éducatives, permettra au professionnel un apprentissage situé et contextuel, c'est-à-dire un environnement simulé qui fournira un étude immersif programmé pour s'entraîner dans des situations réelles.

La conception de ce programme est basée sur l'apprentissage par Problèmes. Ainsi l'apprenant devra essayer de résoudre les différentes situations de pratique professionnelle qui se présentent à lui tout au long du Certificat. Pour ce faire, le professionnel aura l'aide d'un système vidéo interactif innovant créé par des experts reconnus.

Avec cette qualification, vous serez en mesure de mettre en place votre propre projet de développement de Software et d'appliquer les tests unitaires de stress et d'endurance les plus sophistiqués et innovants pour vérifier sa qualité.

Plongez dans le Test Driven Development et obtenez une vision large et spécialisée de la conception et du développement de Softwares pilotés par les tests.

02

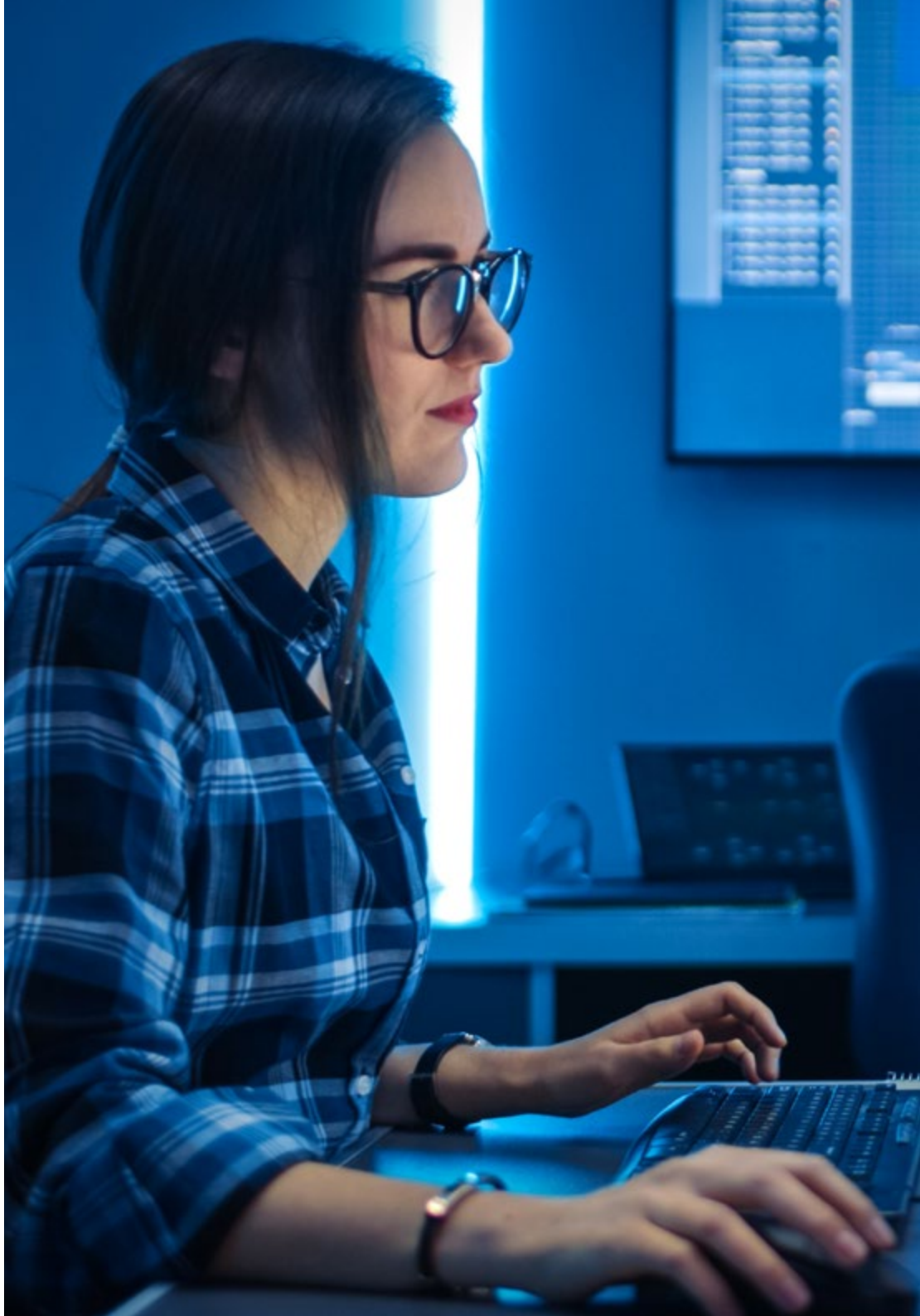
Objectifs

L'ingénierie informatique est un secteur en constante évolution. C'est pourquoi TECH a développé cette qualification, non seulement dans le but de fournir au spécialiste une connaissance large et actualisée de sa profession, mais aussi pour lui fournir une connaissance détaillée des outils qui lui permettront de rester à jour après avoir suivi ce Mastère Avancé. En outre, le meilleur support théorique, pratique et audiovisuel sera mis à votre disposition afin de faire de ce programme une expérience académique dynamique et hautement responsabilisante.

Si votre objectif est de devenir un spécialiste en Ingénierie et Qualité du Software, ce Grand Master vous fournira tout ce dont vous avez besoin pour dépasser vos attentes professionnelles avec une garantie totale de succès"

Objectifs généraux

- Développer les critères, les tâches et les méthodologies avancées pour comprendre la pertinence d'un travail orienté vers la qualité
- Analyser les facteurs clés de la qualité d'un projet Software
- Développer les aspects réglementaires pertinents
- Mettre en œuvre des processus DevOps et des systèmes pour l'assurance qualité
- Réduire la dette technique des projets avec une approche Qualité au lieu d'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 Software
- Défendre les propositions économiques des projets sur la base de la Qualité
- Acquérir de nouvelles connaissances en Ingénierie du Software et des Systèmes Informatiques
- Acquérir les nouvelles compétences en termes de nouvelles technologies, des derniers développements Software
- Traiter les données générées par les activités de l'Ingénierie du Software et des Systèmes Informatiques



Objectifs spécifiques

Module 1. Qualité du Software. Niveaux de développement TRL

- Développer de manière claire et concise les éléments de la qualité des Softwares
- Appliquer les modèles et les normes en fonction du système, du produit et du processus Software
- 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 et international)
- Examiner les niveaux de maturité TRL et les adapter aux différentes parties du projet Software à traiter
- Acquérir la capacité d'abstraction pour appliquer un ou plusieurs critères d'éléments et de niveaux de qualité Softwarele
- Distinguer les cas d'application des normes et des niveaux de maturité dans un cas réel de projet 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
- Analysez 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 Softwares
- 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/CEI 15504
- Déterminer les détails du CMMI
- Apprendre les clés de l'intégration continue, les référentiels et leurs répercussions sur une équipe de développement Software
- Établir la pertinence de l'intégration de référentiels pour les projets Softwares Apprenez à les créer avec TFS
- Analyser les différents types de tests fondamentaux, tels que les tests de charge, unitaires, de *Stress* et d'endurance
- Assimiler l'importance de l'évolutivité des Softwares dans la conception et le développement des systèmes d'information

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

- Déterminer en quoi consiste la méthodologie *Waterfall*
- Approfondir la méthodologie *SCRUM*
- Établir les différences entre *Waterfall* et *SCRUM*
- Préciser les différences entre les méthodologies *Waterfall* et *SCRUM* et la façon dont le client les perçoit
- Examiner le Panel Kanban
- Approcher le même projet avec *WaterFall* y *SCRUM*
- Mise en place d'un projet hybride

Module 5. TDD (*Test Driven Developement*). Conception du Software pilotée par les tests

- Apprendre l'application pratique du TDD et ses possibilités dans les tests futurs d'un projet Software
- 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 Software évolutif, adapté aux besoins réels
- Anticipez les erreurs possibles et évitez-les dès le processus de conception
- Justifier l'utilisation des différents modèles de mise en œuvre

Module 7. DevOps et intégration continue. Solutions pratiques avancées en matière de développement du Software

- Identifier les étapes du cycle de développement et de livraison du Software adaptées à des cas particuliers
- Concevoir un processus de livraison de Softwares utilisant l'intégration continue
- Construire et mettre en œuvre l'intégration et le déploiement continus sur la base de votre conception précédente
- Établir des points de contrôle de qualité automatiques sur chaque livraison de Software
- Maintenir un processus de livraison de Softwares 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 Software et après la livraison

Module 8. Conception de bases de données (BD). Normalisation et performance. Qualité du Software

- Évaluer l'utilisation du Modèle Entité-Relation pour la pré-conception d'une base de données
- Appliquer 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 les cas de simulation réels proposés pour l'apprentissage continu de la conception, de la normalisation et des performances 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 du Software

- Développer le concept d'architecture Softwarele et ses caractéristiques
- Déterminer les différents types d'évolutivité dans l'architecture Softwarele
- Analyser les différents niveaux qui peuvent se produire dans l'évolutivité du Web
- Acquérir une connaissance spécialisée du concept, des étapes et des modèles du cycle de vie des Softwares
- Déterminer l'impact d'une architecture sur le cycle de vie du Software, avec ses avantages, ses limites et ses outils de support
- Réaliser les cas de simulation réels proposés, comme un apprentissage continu de l'architecture et du cycle de vie des Softwares
- Évaluer, dans les cas de simulation, dans quelle mesure la conception de l'architecture peut être réalisable ou inutile

Module 10. Critères de Qualité ISO/IEC 9126. Mesures de la 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 Software pour répondre aux évaluations convenues
- Examiner les attributs internes et externes à prendre en compte dans la qualité d'un projet Software
- Distinguer les métriques en fonction du type de programmation (structurée, orientée objet, en couches, etc.)
- Réaliser des cas réels de simulation, comme apprentissage continu de la 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

Module 11. Méthodologies, développement et qualité en Ingénierie du Software

- Connaître les bases de l'Ingénierie du Software, ainsi que l'ensemble des règles ou principes éthiques et la responsabilité professionnelle pendant et après le développement
- Comprendre le processus de développement de Software sous les différents modèles de programmation et le paradigme de la programmation orientée objet
- Comprendre les différents types de modélisation d'applications et les modèles de conception dans le langage unifié de modélisation (UML)
- Acquérir les connaissances nécessaires à l'application correcte des méthodologies agiles dans le développement de software, y compris SCRUM
- Connaître la méthodologie de développement Lean pour discriminer les activités qui n'apportent pas de valeur ajoutée au processus, afin d'obtenir un Software de meilleure qualité

Module 12. Gestion de projets Software

- Connaître les concepts fondamentaux de la gestion de projet et le cycle de vie de la gestion de projet
- Comprendre les différentes étapes de la gestion de projet telles que l'initiation, la planification, la gestion des *stakeholders* et le scoping
- Apprenez à élaborer un calendrier pour la gestion du temps, le développement du budget et la réponse aux risques
- Comprendre le fonctionnement de la gestion de la qualité dans les projets, y compris la planification, l'assurance, le contrôle, les concepts statistiques et les outils disponibles
- Comprendre le fonctionnement des processus de passation de marchés, d'exécution, de suivi, de contrôle et de clôture d'un projet
- Acquérir les connaissances essentielles liées à la responsabilité professionnelle découlant de la gestion de projet

Module 13. Plateformes de développement de Software

- Comprendre les différentes plateformes de développement de Software
- Acquérir les connaissances nécessaires au développement des Applications et d'interfaces graphiques dans les langages Java et .NET
- Connaître les techniques nécessaires pour déboguer et tester les développements réalisés
- Apprendre les environnements de développement des Applications mobiles Android les processus de débogage et de publication
- Comprendre le développement des Applications basées sur le cloud et déterminer les procédures correctes pour leur mise en œuvre
- Maîtriser les concepts, services et outils de base de la plateforme Google Clouds

Module 14. Informatique client Web

- Assimiler le processus de création de contenu web à travers le langage de balisage HTML
- Comprendre les procédures et les techniques permettant d'améliorer l'apparence d'un document écrit en HTML
- Connaître l'évolution du langage JavaScript
- Acquérir les compétences nécessaires au développement d'applications web côté client
- Développez des applications aux structures complexes, en utilisant les différentes procédures, fonctions et objets qui composent JavaScript
- Apprenez à utiliser l'interface de programmation DOM pour les documents HTML et XML afin de modifier leur structure, leur style et leur contenu
- Comprendre l'utilisation du flux événementiel et des *Listeners,* ainsi que l'utilisation des systèmes modernes de *Toolkit* et d'alignement
- Connaître le concept d'utilisabilité du web, ses avantages, ses principes, ses méthodes et ses techniques pour rendre un site web utilisable par l'utilisateur
- Établir des connaissances sur l'accessibilité du Web, son importance dans les plateformes numériques actuelles, les méthodologies, les normes, les standards et déterminer les échelles de conformité

Module 15. Informatique du serveur web

- Comprendre les concepts de base, intermédiaires et avancés du langage PHP pour la mise en œuvre d'applications côté serveur
- Acquérir les connaissances nécessaires à la modélisation des données, de leurs relations, des clés et des normalisations
- Comprendre la construction du modèle logique de données, la spécification des tables, colonnes, clés et dépendances, ainsi; que les connaissances nécessaires à la manipulation physique des données, les types de fichiers, les modes d'accès et leur organisation
- Apprenez à intégrer des applications développées en PHP avec les bases de données MariaDB et MySQL
- Maîtriser le processus d'interaction avec les clients, grâce à l'utilisation de: formulaires, Cookies et gestion des sessions
- Comprendre l'architecture Softwarele MVC (Model View Controller View) qui sépare les données, l'interface utilisateur et la logique de contrôle d'une application en trois composants distincts
- Acquérir les compétences pour l'utilisation des services web, en utilisant XML, SOA et REST

Module 16. Gestion de la sécurité

- Comprendre le processus de sécurité de l'information, ses implications sur la confidentialité, l'intégrité, la disponibilité et les coûts économiques
- Apprendre l'utilisation de bonnes pratiques de sécurité dans la gestion des services informatiques
- Acquérir les connaissances nécessaires à la certification adéquate des processus de sécurité
- Comprendre les mécanismes et les méthodes d'authentification pour le contrôle d'accès, ainsi que le processus d'audit d'accès
- Comprendre les programmes de gestion de la sécurité, la gestion des risques et la conception des politiques de sécurité
- Apprendre les plans de continuité des activités, leurs phases et le processus de maintenance
- Comprendre les procédures pour une protection adéquate de l'entreprise par le biais de réseaux DMZ, l'utilisation de systèmes de détection d'intrusion et d'autres méthodologies

Module 17. Sécurité du Software

- Comprendre les problèmes liés à la sécurité des Softwares, leurs vulnérabilités et la manière dont ils sont classés
- Connaître les principes de conception, les méthodologies et les normes en matière de sécurité du software
- Comprendre l'application de la sécurité dans les différentes phases du cycle de vie du Softwares
- Acquérir les connaissances nécessaires pour le codage sécurisé des Cycle de vie et les techniques de validation
- Assimiler les méthodologies et les processus permettant de garantir la sécurité lors du développement et de la fourniture de services en nuage
- Comprendre les principes fondamentaux de la cryptologie et les différentes techniques de cryptage actuellement disponibles

Module 18. Administration de serveurs Web

- Comprendre le concept, le fonctionnement, l'architecture, les ressources et le contenu d'un serveur web
- Comprendre le fonctionnement, la structure et la manipulation du protocole HTTP
- Assimiler le concept d'architectures distribuées sur plusieurs serveurs
- Maîtriser le fonctionnement d'un serveur d'application et d'un serveur proxy
- Analyser les différents serveurs web qui sont en vogue sur le marché actuel
- Comprendre le processus de statistiques d'utilisation et d'équilibrage des charges sur les serveurs web
- Acquérir les connaissances nécessaires pour l'installation, l'administration, la configuration et la sécurité du serveur web Microsoft Internet Information Services (IIS) ainsi que, du serveur web libre Apache

Module 19. Contrôles de sécurité

- Acquérir les connaissances nécessaires à la bonne exécution du processus de contrôle et d'audit informatique interne
- Comprendre les processus à mettre en œuvre pour l'audit de sécurité des systèmes et des réseaux
- Comprendre les différents outils d'aide, les méthodologies et l'analyse ultérieure lors des audits de sécurité sur internet et les appareils mobiles
- Apprendre les propriétés et les facteurs d'influence qui conditionnent les risques commerciaux et déterminer la mise en œuvre correcte d'une gestion des risques appropriée
- Connaître les mesures d'atténuation des risques, ainsi que les méthodologies de mise en œuvre d'un système de gestion de la sécurité de l'information et les réglementations et normes à utiliser
- Comprendre les procédures de réalisation de l'audit de sécurité, sa traçabilité et la présentation des résultats

Module 20. Sécurité des applications en ligne

- Acquérir les connaissances nécessaires pour évaluer et détecter les vulnérabilités des applications en ligne
- Comprendre les politiques et les normes de sécurité à appliquer aux applications en ligne
- Apprendre les procédures à utiliser, lors du développement d'applications web et leur validation ultérieure par des analyses et des tests de sécurité
- Apprendre les mesures de sécurité pour le déploiement et la production d'applications web
- Comprendre les concepts, fonctions et technologies à appliquer dans la sécurité des services web, ainsi que les tests de sécurité et les mesures de protection
- Assimiler les procédures du *Hacking* éthique, de l'analyse du malware et du forensics
- Connaître les mesures d'atténuation et de confinement des incidents pour les services web
- Intégrer les techniques de meilleures pratiques pour le développement et la mise en œuvre d'applications en ligne

03

Compétences

Manipuler à la perfection les outils de conception, de planification, de gestion et de développement de Software est une tâche très complexe, notamment en raison du nombre de processus impliqués. Toutefois, ce cours de Mastère Avancé fournira aux diplômés toutes les informations dont ils ont besoin pour se perfectionner dans la maîtrise des outils de ce domaine. Ainsi, vous serez en mesure de mener à bien vos tâches et de réaliser vos projets avec les résultats les plus prometteurs et de grande qualité dans le domaine de l'ingénierie informatique.

La spécialisation dans ce domaine vous permettra de développer des compétences de leadership spécifiques pour mener des projets de gestion de Software, une compétence très appréciée par les sociétés d'ingénierie informatique"

Compétences générales

- Réduire la dette technique des projets avec une approche Qualité au lieu d'une approche basée sur l'économie et les délais courts
- Mesurer et quantifier la qualité d'un projet Software
- Effectuer correctement le *Test-DrivenDevelopment* (TDD) afin d'améliorer les normes de qualité du Software
- Justifier la budgétisation des projets axés sur la qualité
- Développer des normes, modèles et standards 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
- Concevoir, gérer et mettre en œuvre des projets de génie Software et de systèmes informatiques

Vous pourrez apprendre en détail les principales bases de données et accéder à des simulations de projets réels pour leur conception appliquée à des entreprises de différents secteurs"

Compétences spécifiques

- Évaluer un système Software en fonction du degré d'avancement du processus du projet
- Aborder ces points de fiabilité, de métrique et d'assurance dans les projets Softwares de manière correcte et stratégique
- Aborder le processus de décision de la méthodologie à utiliser dans le projet
- Maîtriser les aspects réglementaires essentiels à la création d'un Software
- Développer le *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 Software 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
- Refactor et traiter la gestion et la coordination des données
- Comprendre les différents types de modélisation d'applications et les modèles de conception dans le langage unifié de modélisation (UML)
- Comprendre le fonctionnement de la gestion de la qualité dans les projets, y compris la planification, l'assurance, le contrôle, les concepts statistiques et les outils disponibles
- Utiliser les connaissances nécessaires au développement d'applications et d'interfaces graphiques dans les langages Java et .NET
- Comprendre les procédures et les techniques permettant d'améliorer l'apparence d'un document écrit en HTML
- Maîtriser le processus d'interaction avec les clients , grâce à l'utilisation de formulaires, *Cookies* et gestion des sessions
- Comprendre les mécanismes et les méthodes d'authentification pour le contrôle d'accès, ainsi que le processus d'audit d'accès
- Comprendre l'application de la sécurité dans les différentes phases du cycle de vie du Softwares
- Comprendre le concept, le fonctionnement, l'architecture, les ressources et le contenu d'un serveur web
- Comprendre les différents outils d'aide, les méthodologies et l'analyse ultérieure lors des audits de sécurité sur internet et les appareils mobiles
- Comprendre les politiques et les normes de sécurité à appliquer aux applications en ligne

04

Direction de la formation

La direction et l'enseignement de ce Mastère Avancé en Ingénierie et Qualité du Software sont assurés par une équipe d'experts en ingénierie ayant de nombreuses années d'expérience dans la gestion et le développement de projets techniques et spécialisés. Leur parcours professionnel apporte à ce diplôme un plus de qualité qui se traduira par une meilleure contextualisation du contenu par le diplômé, ainsi que par la mise en œuvre à l'expérience académique d'études de cas réels et simulés, mais toujours dans le but d'offrir un programme 100% en ligne, dynamique, avant-gardiste et basé sur la réalité immédiate du secteur.

L'équipe d'ingénieurs chargée de l'enseignement de ce Mastère Avancé sera à votre disposition pour vous guider et vous aider à résoudre les questions ou les doutes que vous pourriez avoir sur le programme ou la profession"

Direction

M. Molina Molina, Jerónimo

- IA Engineer & Software Architect. NASSAT-Internet par satellite en Mouvement
- Consultant Senior Hexa Ingenieros. Introducteur d'Intelligence Artificielle (ML et CV)
- Expert en solutions basées sur l'intelligence artificielle dans les domaines de la *Computer Vision,* ML/DL y NLP
- Actuellement, étudiant les possibilités d'application de *Transformers* et de *Reinforcement Learning* dans un projet de recherche personnel
- Expert universitaire en Création et Développement d'Entreprise Bancaixa-FUNDEUN Alicante
- Ingénieur en informatique. Université d'Alicante
- Master en Intelligence Artificielle. Université Catolica de Avila
- MBA-Executive. Forum du Campus Européen des Affaires

Professeurs

M. Martinez Calvo, Francisco Javier

- Architecte-Analyste organique et fonctionnel
- Consultant technique-IT
- Développement et soutien au projet médical européen, intégration du FNMT, du PPG et du PCL dans HexaIngenieros
- Formateur Visual Studio, SqlServer, CCNA (routeurs et commutateurs Cisco), programmation web PHP et .NET dans plusieurs centres (Salesianos, Maforem, Dreamsoft)
- Ingénieur Technique Industriel, spécialisation en électricité, électronique industrielle
- Master Cibernos en .NET. MCAD
- Master Eidos en Programmation Avancée. Niveau Expert
- Master WEB. Certifications Dreamweaver, Fireworks, Flash et ActionScript, versions MX

M. Tenrero Moran, Marcos

- DevOps Engineer-Allot Communications
- Application Lifecycle Management & DevOps-Meta4 Spain. Cegid
- Ingénieur en automatisation QA-Meta4 Espagne Cegid
- Diplômé en ingénierie informatique de l'université Rey Juan Carlos
- Développement d'applications professionnelles pour Android-Universidad 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

Dr Acebes Tamargo, Patricia

- Département des Opérations, travaille avec Elasticsearch et Kivana. Sirt
- Chercheuse Ligne Facteur Humain et Applications IA. Centre Technologique du CTIC
- Chercheuse Ligne d'Unité Commerciale. Centre Technologique du CTIC
- Département "Santé Numérique et Vieillissement Actif" Centre Technologique du CTIC
- Département Data Science. Centre Technologique du CTIC
- Doctorante en Ingénierie Informatique
- Licence en Économie Université d'Oviedo
- Études pour un Master en Analyse de Données UCJC
- Études pour un Master en Intelligence Artificielle(ia). UNED
- Études de Mathématiques et d'Ingénierie des TIC à l'UNED
- Master en Blockchain, Smart Contracts et Cryptocurrencies. Université d'Alcala
- Postgraduate en Ingénierie Blockchain. EADA
- Master en Économie et Outils d'Analyse Économique
- Master en Fiscalité Collège d'économistes

Mme Rodriguez Miguez, Candida

- CoFondateur et City Leader du réseau Galicia AI (association espagnole d'IA)
- Streamer académique sur YouTube
- Projet SISAP pour SERGAS, fonctionnalité web auto vaccination COVID Cita Internet. INDRA Production S.L.
- OSAL Aixiña. Collaboration en matière de TFM à distance
- Enseignement d'une Session d'Introduction à l'Intelligence Artificielle WordPress Galicia
- Ingénieur informaticien spécialisé dans les Softwares ESEI Ourense. Université de Vigo
- Master en Ingénierie Informatique, spécialisé dans le développement de grands systèmes Softwares. ESEI Ourense. Université de VigoUniversité de Vigo
- Cycle Supérieur en Gestion Commerciale et Marketing Centre privé de Formation Professionnelle de Novacaixagalicia Ourense
- Cycle Supérieur en Administration de Systèmes Informatiques Centre privé de Formation Professionnelle de Novacaixagalicia Ourense

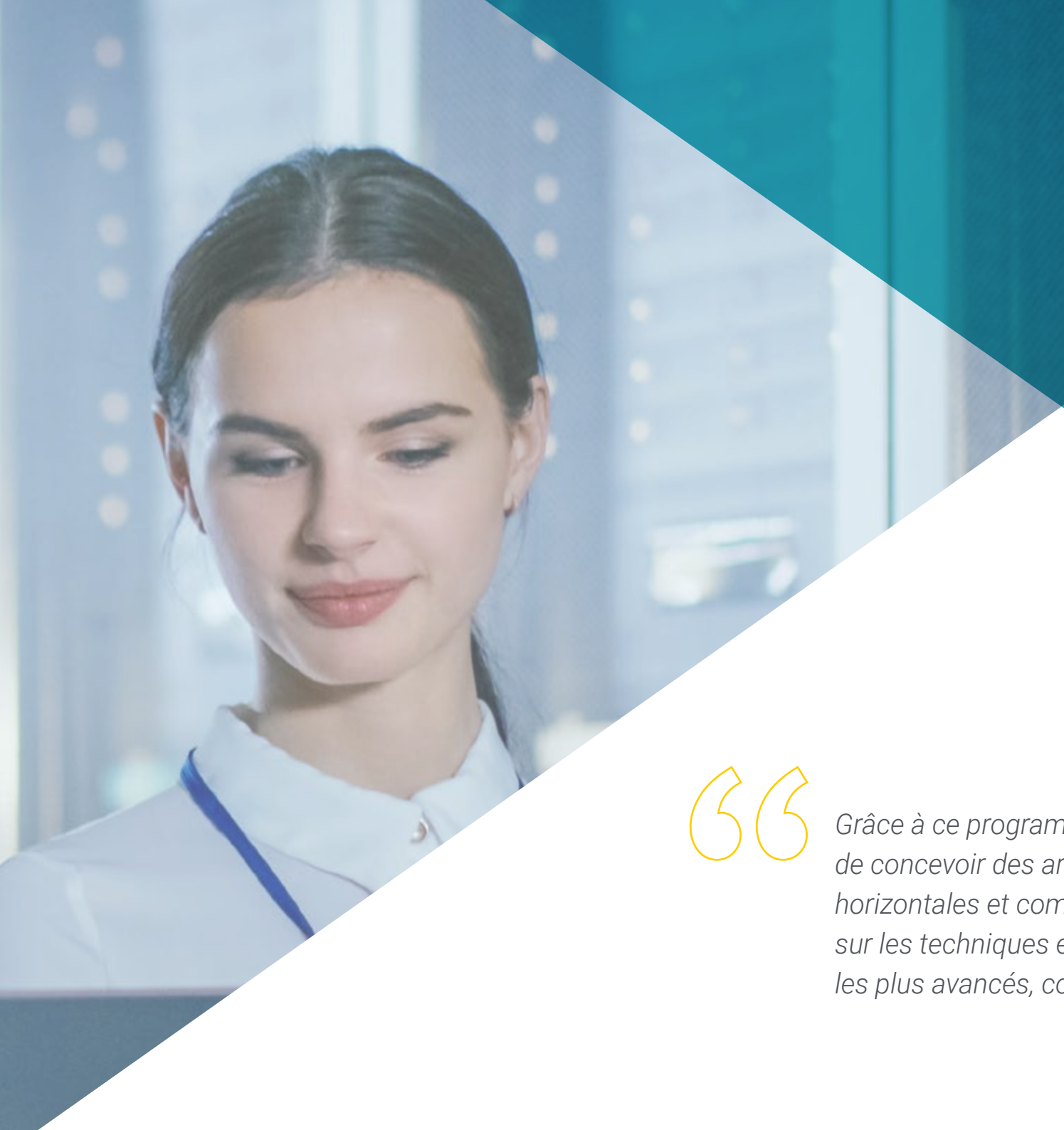
M. Pi Morell, Oriol

- Product Owner pour Hosting and Mail. CDMON
- Analyste Fonctionnel et Ingénieur Software dans différentes organisations telles que Fihoca, Atmira, CapGemini
- Enseignant de différents cours tels que BPM à CapGemini, ORACLE Forms CapGemini, Business Processes Atmira
- Diplôme en Ingénierie 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 d'enseignement
- Postgraduate, Postgraduate en Design Patterns. Université Oberta de Catalunya

05

Structure et contenu

Pour le développement de ce diplôme, TECH a basé la structure de son contenu sur trois piliers fondamentaux: les informations les plus récentes et les plus complètes du secteur de l'ingénierie informatique spécialisée dans le domaine du Software, les recommandations de l'équipe enseignante et la méthodologie pédagogique innovante de *Relearning.* En outre, dans son engagement à offrir un programme adapté et personnalisé aux besoins académiques de chaque diplômé, non seulement dans la conception des horaires, mais aussi dans le niveau d'approfondissement, les étudiants trouveront des centaines d'heures de support supplémentaire dans la classe virtuelle avec laquelle ils pourront approfondir les aspects qu'ils considèrent les plus pertinents pour leur pratique professionnelle.



Grâce à ce programme, vous serez en mesure de concevoir des architectures verticales, horizontales et combinées évolutives, basées sur les techniques et protocoles informatiques les plus avancés, complets et actuels"

Module 1. Qualité du Software. Niveaux de développement TRL

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

1.2. Éléments influençant la qualité des Softwares (II). Coûts Associés
- 1.2.1. Qualité du Software. Éléments d'influence
- 1.2.2. Qualité du Software. Idées fausses
- 1.2.3. Qualité du Software. Coûts Associés

1.3. Modèles de qualité des Softwares (I). Gestion des connaissances
- 1.3.1. Modèles de Qualité générales
  - 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 Softwares (III). Qualité des données, des processus et des modèles SEI
- 1.4.1. Modèle de qualité des données
- 1.4.2. Modélisation des processus Softwares
- 1.4.3. *Software & Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. Modèles SEI
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. Normes ISO de qualité des Softwares (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 du Software (II). Exigences et évaluation
- 1.6.1. Normes sur les 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. Niveaux 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 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 industrielle de R&D&I
- 1.9.4. Exemple d'une entreprise commune laboratoire-ingénieur

1.10. Qualité du Software. Principaux détails
- 1.10.1. Détails méthodologiques
- 1.10.2. Détails techniques
- 1.10.3. Détails sur la gestion des projets Softwares
  - 1.10.3.1. Qualité des systèmes informatiques
  - 1.10.3.2. Qualité des produits Softwares
  - 1.10.3.3. Qualité des processus Softwares

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

2.1. Gestion de projets
- 2.1.1. Gestion de projets dans la qualité des Softwares
- 2.1.2. Gestion de projet. Avantages
- 2.1.3. Gestion de projet. Typologie

2.2. Méthodologie dans la gestion de projets
- 2.2.1. Méthodologie dans la gestion de projets
- 2.2.2. Méthodologies de projets. 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 d'un 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

2.7. *MUST* du Projet
- 2.7.1. *MUST* du Projet
- 2.7.2. Identification des *MUST* du projet
- 2.7.3. Identification des points d'exécution pour la livraison 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 Software
- 2.9.1. Architecte du projet. Aspects techniques
- 2.9.2. Responsables techniques
- 2.9.3. Construction du projet Software
- 2.9.4. Évaluation de la qualité du code, Sonar

2.10. Prestations 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 Softwares
- 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. Règlementation ISO 15504
  - 3.2.3.1. Objectifs
  - 3.2.3.2. Contexte
  - 3.2.3.3. Étapes

3.3. Règlementation 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. 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. Administration de Projets

3.5. Gestion des changements et des référentiels
- 3.5.1. Gestion des changements Softwares
  - 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 des versions
  - 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'équipe
- 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. Test bêta
- 3.7.4. Mise en œuvre et maintenance

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

3.9. Test de stress et d'endurance de l'unité
- 3.9.1. Motivation pour les tests unitaires
- 3.9.2. Outils de *Unit Testing*
- 3.9.3. Motivation pour les tests de stress
- 3.9.4. Test à l'aide de *StressTesting*
- 3.9.5. Motivation pour les tests d'endurance
- 3.9.6. Test à l'aide de *StressTesting*

3.10. L'évolutivité. Conception de Softwares évolutifs
- 3.10.1. L'évolutivité et l'architecture Softwarele
- 3.10.2. L'indépendance entre les couches
- 3.10.3. Couplage entre les couches Modèles d'architecture

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 du Software
- 4.1.3. Méthodologie *Waterfall* Exemples

4.2. Méthodologie Agile
- 4.2.1. Méthodologie Agile
- 4.2.2. Méthodologie Agile Assurance qualité du Software
- 4.2.3. Méthodologie Agile Exemples

4.3. Méthodologie SCRUM
- 4.3.1. Méthodologie SCRUM
- 4.3.2. Manifeste SCRUM
- 4.3.3. Applications du 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. Les phases d'un projet
- 4.5.2. Vision dans un projet *Waterfall*
- 4.5.3. Produits livrables à prendre en compte

4.6. Gestion de projet en SCRUM
- 4.6.1. Les phases d'un projet SCRUM
- 4.6.2. Vision dans un projet SCRUM
- 4.6.3. Livrables à considérer

4.7. Waterfall vs. SCRUM. Comparaison
- 4.7.1. Approche par projet pilote
- 4.7.2. Projet appliquant la méthode *Waterfall*. Exemple
- 4.7.3. Projet appliquant SCRUM. Exemple

4.8. Aperçu du Client
- 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 d'un projet
- 4.10.3. Livrables à considérer

Module 5. TDD (*Test Driven Developement*). Conception du Software pilotée 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. Typologies
  - 5.2.2.1. Tests unitaires
  - 5.2.2.2. Test d'intégration
  - 5.2.2.3. Tests *End To End*
- 5.2.3. Vérification d'examen 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 de pré-faillite 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 de la problématique
- 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 du Software
- 6.1.3. DevOps. Les 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 méthodologies
- 6.3.1. Identification du problème
- 6.3.2. Mise en œuvre dans une entreprise
- 6.3.3. Paramètres de mise en œuvre

6.4. Cycle de livraison des Softwares
- 6.4.1. Méthodes de conception
- 6.4.2. Conventions
- 6.4.3. Les artefacts du futur

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* de code
- 6.5.4. Développement de Softwares 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 de la cible
- 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. Comment 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. Les bonnes pratiques L'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 du Software

7.1. Flux de livraison des Softwares
- 7.1.1. Identification des acteurs et des artefacts
- 7.1.2. Conception du flux de livraison des Softwares
- 7.1.3. Flux de livraison des Softwares. Exigences inter-étapes

7.2. Automatisation des processus
- 7.2.1. Intégration continue
- 7.2.2. Déploiement continu
- 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 dans 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. Sécurité dans la gestion des artefacts

7.6. Déploiement continu
- 7.6.1. Déploiement continu sous forme de conteneurs
- 7.6.2. Déploiement continu avec PaaS

7.7. Amélioration de l'exécution de *Pipeline*: nalyse 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 des conteneurs
- 7.8.1. Vulnérabilités des conteneurs
- 7.8.2. Scannage d'images
- 7.8.3. Rapports et alertes périodiques

Module 8. Conception de Bases de Données (BD). Normalisation 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 actuellement utilisées
  - 8.1.2.1. Relationnels
  - 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 1
  - 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. Echantillon en représentation de tableau

8.4. Base de données (DB) Normalisation (I). Considérations sur la qualité des Softwares
- 8.4.1. DB Standardisation et qualité
- 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 (DB). 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: mise à jour de la vue
  - 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. Performance 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 d'un projet réel pour la conception du DB (I)
- 8.8.1. Aperçu du projet (Entreprise A)
- 8.8.2. Application de 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 de la base de données dans la qualité des Softwares
- 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 Softwares. Recommandations d'utilisation

Module 9. Conception d'architectures évolutives. L'architecture dans le cycle de vie du Software

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. Arquitecturas 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 DES DESCRIPTIFS)

9.4. Cycle de vie du Software (I). Étapes
- 9.4.1. Cycle de vie d'un Software
  - 9.4.1.1. Étapes 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 Softwares
- 9.5.1. Modèle en cascade
- 9.5.2. Modèle répétitif
- 9.5.3. Modèle en spirale
- 9.5.4. Modèle de Big Bang

9.6. Cycle de vie du Software (II). Automatisation
- 9.6.1. Cycles de vie du développement Software. Solutions
  - 9.6.1.1. Intégration et développement continus (CI/CD)
  - 9.6.1.2. Méthodologie *Agile*
  - 9.6.1.3. Tendances futures
- 9.6.2. Exemples pratiques
- 9.6.3. Exemples pratiques

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

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

9.9. Simulation d'un projet réel de conception d'architecture Softwarele (II)
- 9.9.1. Aperçu du projet (entreprise B)
- 9.9.2. Application de la conception de l'architecture Softwarele
- 9.9.3. Exercices proposés
- 9.9.4. Exercices proposés Feedback

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

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

10.1. Critères de qualité Normes ISO, IEC 9126
- 10.1.1. Critères de qualité
- 10.1.2. Qualité du Software. Justification Normes ISO, IEC 9126
- 10.1.3. La mesure de la qualité des Softwares comme indicateur clé

10.2. Critères qualité du Software. Caractéristiques
- 10.2.1. Fiabilité
- 10.2.2. Fonctionnalité
- 10.2.3. Efficacité
- 10.2.4. Utilisabilité
- 10.2.5. Maintenabilité
- 10.2.6. Portabilité

10.3. Normes ISO, IEC 9126 R+D Présentation
- 10.3.1. Description de la Normes ISO, IEC 9126
- 10.3.2. Fonctionnalité
- 10.3.3. Fiabilité
- 10.3.4. Utilisabilité
- 10.3.5. Maintenabilité
- 10.3.6. Portabilité
- 10.3.7. Qualité de l'utilisation
- 10.3.8. Mesures de la Qualité du Software
- 10.3.9. Métriques de Qualité dans ISO ISO 9126

10.4. Norme ISO, CEI 9126 (II). Modèles 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 Softwares (I). Éléments
- 10.5.1. Mesure
- 10.5.2. Métriques
- 10.5.3. Sommaire
  - 10.5.3.1. Types d'indicateurs
- 10.5.4. Mesures et modèles
- 10.5.5. Portée de la métrologie des Softwares
- 10.5.6. Classification des métriques Softwareles

10.6. Mesure de la qualité des Softwares (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 Softwares
- 10.7.1. Indicateur unique en tant que scoreur global
- 10.7.2. Développement, justification et application des indicateurs
- 10.7.3. Exemple d'application. Besoin de connaître le détail

10.8. Simulation d'un projet réel pour la mesure de la qualité (I)
- 10.8.1. Description générale 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*

Module 11. Méthodologies, développement et Qualité en Ingénierie du Software

11.1. Développement Software guidé par le modèle
- 11.1.1. Le besoin de
- 11.1.2. Modélisation d'objets
- 11.1.3. UML
- 11.1.4. Outils de CASE

11.2. Modélisation d'applications et patrons de conception
- 11.2.1. Modélisation avancée des exigences
- 11.2.2. Modélisation statique avancée
- 11.2.3. Modélisation dynamique avancée
- 11.2.4. Modélisation des composants
- 11.2.5. Introduction aux modèles de conception avec UML
- 11.2.6. *Adapter*
- 11.2.7. *Factory*
- 11.2.8. Singleton
- 11.2.9. *Strategy*
- 11.2.10. *Composite*
- 11.2.11. Facade
- 11.2.12. *Observer*

11.3. Ingénierie guidée par le modèle
- 11.3.1. Introduction
- 11.3.2. Métamodélisation des systèmes
- 11.3.3. MDA
- 11.3.4. DSL
- 11.3.5. Raffinements de modèles avec OCL
- 11.3.6. Transformations de modèles

11.4. Ontologie dans l'Ingénierie du Software
- 11.4.1. Introduction
- 11.4.2. Ontologies dans l'Ingénierie du Software
- 11.4.3. Application des Ontologies dans l'Ingénierie du Software

Module 12. Gestion de projets Software

12.1. La gestion des *Stakeholders* et de la sensibilisation
- 12.1.1. Identifier les parties prenantes
- 12.1.2. Développer un plan de gestion des parties prenantes
- 12.1.3. Gérer l'engagement des parties prenantes
- 12.1.4. Contrôler l'engagement des parties prenantes
- 12.1.5. L'objectif du projet
- 12.1.6. La gestion de la portée et son plan
- 12.1.7. Recueillir les besoins
- 12.1.8. Définir l'énoncé de la portée
- 12.1.9. Créer le WBS
- 12.1.10. Vérifier et contrôler la portée

12.2. L'élaboration du calendrier
- 12.2.1. La gestion du temps et son plan
- 12.2.2. Définir les activités
- 12.2.3. Établissement de la séquence des activités
- 12.2.4. Estimation des ressources des activités
- 12.2.5. Estimation des ressources pour les activités
- 12.2.6. Développement du calendrier et calcul du chemin critique
- 12.2.7. Contrôle des horaires

12.3. Élaboration du budget et réponse aux risques
- 12.3.1. Estimation des coûts
- 12.3.2. Élaboration du budget et de la courbe en S
- 12.3.3. Contrôle des coûts et méthode de la valeur acquise
- 12.3.4. Concepts de risque
- 12.3.5. Comment faire une analyse de risque
- 12.3.6. L'élaboration du plan de réponse

12.4. La communication et les ressources humaines
- 12.4.1. Planification de la gestion des communications
- 12.4.2. Analyse des besoins de communication
- 12.4.3. Technologie des communications
- 12.4.4. Modèle de communication
- 12.4.5. Méthodes de communication
- 12.4.6. Plan gestion des communications
- 12.4.7. Gestion des communications
- 12.4.8. La gestion des ressources humaines
- 12.4.9. Principaux acteurs et leurs rôles dans les projets
- 12.4.10. Types d'organisations
- 12.4.11. Organisation du projet
- 12.4.12. L'équipe de travail

12.5. L'approvisionnement
- 12.5.1. Le processus d'acquisitions
- 12.5.2. Planification
- 12.5.3. Recherche de fournisseurs et appels d'offres
- 12.5.4. Attribution du contrat
- 12.5.5. Administration des contrats
- 12.5.6. Contrats
- 12.5.7. Types de contrats
- 12.5.8. Négociation de contrats

12.6. Exécution, suivi et contrôle et clôture
- 12.6.1. Groupes de processus
- 12.6.2. L'exécution des projets
- 12.6.3. Suivi et contrôle des projets
- 12.6.4. Clôture du projet

12.7. Responsabilité professionnelle
- 12.7.1. Responsabilité professionnelle
- 12.7.2. Caractéristiques de la responsabilité sociale et professionnelle
- 12.7.3. Code d'éthique du chef de projet
- 12.7.4. Responsabilité vs PMP®
- 12.7.5. Exemples de responsabilité
- 12.7.6. Avantages de la professionnalisation

Module 13. Plateformes de développement de Software

13.1. Introduction au développement d'application
- 13.1.1. Applications de bureau
- 13.1.2. Langage de programmation
- 13.1.3. Environnements de développement intégrés
- 13.1.4. Applications web
- 13.1.5. Applications mobiles
- 13.1.6. Applications en nuage

13.2. Développement des Applications et d'interfaces graphiques en Java
- 13.2.1. Environnements de développement intégrés pour Java
- 13.2.2. Principaux IDE de Java
- 13.2.3. Introduction à la plateforme de développement Éclipse
- 13.2.4. Introduction à la plateforme de développement Éclipse
- 13.2.5. Modèle contrôleur-vue pour les interfaces utilisateur graphiques
- 13.2.6. Réaliser une interface utilisateur graphique dans Éclipse
- 13.2.7. Réaliser une interface utilisateur graphique dans NetBeans

13.3. Débogage et test dans Java
- 13.3.1. Test et débogage des programmes Java
- 13.3.2. Débogage dans Eclipse
- 13.3.3. Débogage dans NetBeans

13.4. développement des Applications et d'interfaces graphiques en .NET
- 13.4.1. Net Framework
- 13.4.2. Composants de la plate-forme de développement .NET
- 13.4.3. Visual Studio .NET
- 13.4.4. Outils GUI .NET
- 13.4.5. L'interface graphique avec Windows Presentation Foundation
- 13.4.6. Déboguer et compiler une application WPF

13.5. Programmation pour les réseaux .NET
- 13.5.1. Introduction à la programmation réseau .NET
- 13.5.2. Demandes et réponses dans .NET
- 13.5.3. Utilisation des protocoles d'application dans .NET
- 13.5.4. Sécurité dans programmation réseau .NET

13.6. Environnements de développement des Applications mobiles
- 13.6.1. Applications mobiles
- 13.6.2. Applications mobiles Android
- 13.6.3. Étapes du développement d'Android
- 13.6.4. L'IDE Android Studio

13.7. Développer des applications dans l'environnement Android Studio
- 13.7.1. Installer et démarrer Android Studio
- 13.7.2. Exécution d'une application Android
- 13.7.3. Développer l'interface graphique dans Android Studio
- 13.7.4. Lancement d'activités dans Android Studio

13.8. Débogage et publication des applications Android
- 13.8.1. Déboguer une application dans Android Studio
- 13.8.2. Stockage des applications dans Android Studio
- 13.8.3. Publier une application sur Google Play

13.9. Développer des applications pour le cloud
- 13.9.1. *Cloud computing*
- 13.9.2. Niveaux du *Cloud*: SaaS, PaaS, IaaS
- 13.9.3. Principales plateformes de développement en nuage
- 13.9.4. Références bibliographiques

13.10. Introduction à Google Cloud Platform
- 13.10.1. Notions de base de Google Cloud Platform
- 13.10.2. Service Google Cloud Platform
- 13.10.3. Outils de Google Cloud Platform

Module 14. Informatique client Web

14.1. Informatique du client Web
- 14.1.1. Structure d'un document
- 14.1.2. Couleur
- 14.1.3. Texte
- 14.1.4. Liens hypertextes
- 14.1.5. Images
- 14.1.6. Listes
- 14.1.7. Tables
- 14.1.8. Cadres (*Frames*)
- 14.1.9. Formulaires
- 14.1.10. Éléments spécifiques aux technologies mobiles
- 14.1.11. Éléments désaffectés

14.2. Fiches de style Web (CSS)
- 14.2.1. Éléments et structure d'une fiche de style
  - 14.2.1.1. Création de fiches de style
  - 14.2.1.2. Application des modèles. Sélecteurs
  - 14.2.1.3. Héritage de style et cascade
  - 14.2.1.4. Mise en forme des pages à l'aide de modèles
  - 14.2.1.5. Mise en forme des pages à l'aide de modèles. Le modèle de boîte
- 14.2.2. Concevoir des styles pour différents appareils
- 14.2.3. Types de feuilles de style: statiques et dynamiques Pseudo-classes
- 14.2.4. Bonnes pratiques dans l'utilisation des fiches de style

14.3. Introduction et histoire de JavaScript
- 14.3.1. Introduction
- 14.3.2. Histoire de JavaScript
- 14.3.3. Environnement de développement que nous allons utiliser

14.4. Notions de base de la programmation web
- 14.4.1. Syntaxe de base de JavaScript
- 14.4.2. Types de données et opérateurs primitifs
- 14.4.3. Variables et champs d'application
- 14.4.4. Chaînes de texte et *Template Literals*
- 14.4.5. Nombres et booléens
- 14.4.6. Comparaisons

14.5. Structures complexes en JavaScript
- 14.5.1. Vecteurs ou *Arrays* et objets
- 14.5.2. Conjonctifs
- 14.5.3. Cartes
- 14.5.4. Disjonctions
- 14.5.5. Boucles

14.6. Fonctions et objets
- 14.6.1. Définition et invocation des fonctions
- 14.6.2. Arguments
- 14.6.3. Fonctions des flèches
- 14.6.4. Fonctions de rappel ou *Callback*
- 14.6.5. Fonctions d'ordre supérieur
- 14.6.6. Objets littéraux
- 14.6.7. L'objet *This*
- 14.6.8. Objets en tant qu'espaces de noms: l'objet *Math* et l'objet *Date*

14.7. Le modèle d'objet de document (DOM)
- 14.7.1. Qu'est-ce que le DOM?
- 14.7.2. Un peu d'histoire
- 14.7.3. Navigation et récupération d'éléments
- 14.7.4. Un DOM virtuel avec JSDOM
- 14.7.5. Sélecteurs de requêtes ou *Query Selectors*
- 14.7.6. Navigation dans les propriétés
- 14.7.7. Attribution d'attributs aux éléments
- 14.7.8. Création et modification de nœuds
- 14.7.9. Mise à jour du style des éléments du DOM

14.8. Développement web moderne
- 14.8.1. Flux piloté par les événements et *Listeners*
- 14.8.2. *Toolkits* web modernes et systèmes d'alignement
- 14.8.3. Mode écrit de JavaScript
- 14.8.4. Un peu plus sur les fonctions
- 14.8.5. Fonctions asynchrones et promesses
- 14.8.6. *Fermetures*
- 14.8.7. Programmation fonctionnelle
- 14.8.8. POO en JavaScript

14.9. Utilisabilité du Web
- 14.9.1. Introduction à l'utilisabilité
- 14.9.2. Définition de l'utilisabilité
- 14.9.3. Importance d'une conception web centrée sur l'utilisateur
- 14.9.4. Différences entre l'accessibilité et la facilité d'utilisation
- 14.9.5. Avantages et problèmes liés à la combinaison de l'accessibilité et de la facilité d'utilisation
- 14.9.6. Avantages et difficultés de la mise en œuvre de sites web utilisables
- 14.9.7. Méthodes d'utilisabilité
- 14.9.8. Analyse des besoins des utilisateurs
- 14.9.9. Principes de conception Prototypage orienté vers l'utilisateur
- 14.9.10. Directives pour la création de sites web utilisables
  - 14.9.10.1. Directives d'utilisabilité selon Jakob Nielsen
  - 14.9.10.2. Directives d'utilisabilité selon Bruce Tognazzini
- 14.9.11. Évaluation de la convivialité

14.10. Accessibilité du Web
- 14.10.1. Introduction
- 14.10.2. Définition de l'accessibilité du Web
- 14.10.3. Types de handicaps
  - 14.10.3.1. Handicaps temporaires ou permanents
  - 14.10.3.2. Déficiences visuelles
  - 14.10.3.3. Déficiences auditives
  - 14.10.3.4. Handicaps moteurs
  - 14.10.3.5. Handicaps neurologiques ou cognitifs
  - 14.10.3.6. Difficultés liées au vieillissement
  - 14.10.3.7. Limitations environnementales
  - 14.10.3.8. Obstacles à l'accès au web
- 14.10.4. Aides techniques et produits d'assistance pour surmonter les obstacles
  - 14.10.4.1. Aides pour les aveugles
  - 14.10.4.2. Aides pour les personnes atteintes de basse vision
  - 14.10.4.3. Aides pour les personnes atteintes de daltonisme
  - 14.10.4.4. Aides pour les malentendants
  - 14.10.4.5. Aides pour les personnes souffrant d'un handicap moteur
  - 14.10.4.6. Aides pour les personnes souffrant de handicaps cognitifs et neurologiques
- 14.10.5. Avantages et difficultés de la mise en œuvre de l'accessibilité du Web
- 14.10.6. Réglementations et normes en matière d'accessibilité du Web
- 14.10.7. Organismes de réglementation de l'accessibilité du Web
- 14.10.8. Comparaison des normes et des standards
- 14.10.9. Lignes directrices pour le respect des réglementations et des normes
  - 14.10.9.1. Description des principales orientations (images, liens vidéo, etc.)
  - 14.10.9.2. Directives pour une navigation accessible
    - 14.10.9.2.1. Perceptibilité
    - 14.10.9.2.2. Exploitabilité
    - 14.10.9.2.3. Compréhensibilité
    - 14.10.9.2.4. Résistance
- 14.10.10. Description du processus de conformité de l'accessibilité du Web
- 14.10.11. Niveaux de conformité
- 14.10.12. Critères de conformité
- 14.10.13. Exigences de conformité
- 14.10.14. Méthodologie d'évaluation de l'accessibilité des sites web

Module 15. Informatique du serveur web

15.1. Introduction à la programmation côté serveur: PHP
- 15.1.1. Concepts de base de la programmation côté serveur
- 15.1.2. Syntaxe PHP de base
- 15.1.3. Générer du contenu HTML avec PHP
- 15.1.4. Environnements de développement et de test: XAMPP

15.2. PHP avancé
- 15.2.1. Structures de contrôle avec PHP
- 15.2.2. Fonctions en PHP
- 15.2.3. Gestion de *Arrays* en PHP
- 15.2.4. Manipulation des chaînes en PHP
- 15.2.5. Orientation objet en PHP

15.3. Modèles de données
- 15.3.1. Concept de données. Cycle de vie des données
- 15.3.2. Types de données
  - 15.3.2.1. Principes de base
  - 15.3.2.2. Enregistrements
  - 15.3.2.3. Dynamiques

15.4. Modèle relationnel
- 15.4.1. Description
- 15.4.2. Entités et types d'entités
- 15.4.3. Éléments de données. Attributs
- 15.4.4. Relations: types, sous-types, cardinalité
- 15.4.5. Clés. Types de clés
- 15.4.6. Normalisation Formes normales

15.5. Construction du modèle logique de données
- 15.5.1. Spécification des tables
- 15.5.2. Définition des colonnes
- 15.5.3. Spécification clés
- 15.5.4. Conversion en formes normales Dépendances

15.6. Le modèle physique de données. Fichiers de données
- 15.6.1. Description des fichiers de données
- 15.6.2. Types de fichiers
- 15.6.3. Modes d'accès
- 15.6.4. Organisation des fichiers

15.7. Accès aux bases de données depuis PHP
- 15.7.1. Introduction à MariaDB
- 15.7.2. Travailler avec une base de données MariaDB: le langage SQL
- 15.7.3. Accéder à la base de données MariaDB depuis PHP
- 15.7.4. Introduction à MySql
- 15.7.5. Travailler avec une base de données MySql: le langage SQL
- 15.7.6. Accéder à la base de données MySql desde PHP

15.8. Interaction avec le client à partir de PHP
- 15.8.1. Formulaires PHP
- 15.8.2. Cookies
- 15.8.3. Traitement des sessions

15.9. Architecture d'applications Web
- 15.9.1. Le modèle Modèle-Vue-Contrôleur Modèle
- 15.9.2. Contrôle
- 15.9.3. Modèle
- 15.9.4. Voir

15.10. Introduction aux services Web
- 15.10.1. Introduction à XML
- 15.10.2. Architectures orientées services (SOA): Services Web
- 15.10.3. Création de services web SOAP et REST
- 15.10.4. Le protocole SOAP
- 15.10.5. Le protocole REST

Module 16. Gestion de la sécurité

16.1. La sécurité de l'information
- 16.1.1. Introduction
- 16.1.2. La sécurité des informations implique la confidentialité, l'intégrité et la disponibilité
- 16.1.3. La sécurité est une question économique
- 16.1.4. La sécurité est un processus
- 16.1.5. La classification des informations
- 16.1.6. La sécurité de l'information implique la gestion des risques
- 16.1.7. La sécurité s'articule avec les contrôles de sécurité
- 16.1.8. La sécurité est à la fois physique et logique
- 16.1.9. La sécurité implique des personnes

16.2. Le professionnel de la sécurité de l'information
- 16.2.1. Introduction
- 16.2.2. La sécurité de l'information en tant que profession
- 16.2.3. Les certifications ISC2
- 16.2.4. La norme ISO 27001
- 16.2.5. Bonnes pratiques de sécurité dans la gestion des services informatiques
- 16.2.6. Modèles de maturité de la sécurité de l'information
- 16.2.7. Autres certifications, normes et ressources professionnelles

16.3. Contrôle d'accès
- 16.3.1. Introduction
- 16.3.2. Exigences en matière de contrôle d'accès
- 16.3.3. Mécanismes d'authentification
- 16.3.4. Méthodes d'autorisation
- 16.3.5. Comptabilité et audit des accès
- 16.3.6. Technologies Triple A

16.4. Programmes, processus et politiques de sécurité de l'information
- 16.4.1. Introduction
- 16.4.2. Programmes de gestion de la sécurité
- 16.4.3. La gestion des risques
- 16.4.4. Conception de la politique de sécurité

16.5. Plans de continuité des activités
- 16.5.1. Introduction aux PCA
- 16.5.2. Phase I et II
- 16.5.3. Phase III et IV
- 16.5.4. Maintenance du PCA

16.6. Procédures pour la protection adéquate de l'entreprise
- 16.6.1. Réseaux DMZ
- 16.6.2. Systèmes de détection d'intrusion
- 16.6.3. Listes de contrôle d'accès
- 16.6.4. Apprendre de l'agresseur *Honeypot*

16.7. Architectures de sécurité Prévention
- 16.7.1. Aperçu général. Activités et modèle de superposition
- 16.7.2. Défense du périmètre (Firewalls, WAFs, IPS, etc.)
- 16.7.3. Défense des points d'extrémité (équipements, serveurs et services)

16.8. Architectures de sécurité Détection
- 16.8.1. Aperçu de la détection et de la surveillance
- 16.8.2. Logs, rupture de trafic crypté, enregistrement et *Siems*
- 16.8.3. Alertes et renseignements

16.9. Architectures de sécurité Réaction
- 16.9.1. Réaction Produits, services et ressources
- 16.9.2. Gestion des incidents
- 16.9.3. CERTS y CSIRTs

16.10. Architectures de sécurité Récupération
- 16.10.1. Résilience, concepts, exigences commerciales et normes
- 16.10.2. Solutions informatiques de résilience
- 16.10.3. Gestion de crise et gouvernance

Module 17. Sécurité du Software

17.1. Questions relatives à la sécurité des Softwares
- 17.1.1. Introduction au problème de la sécurité des Softwares
- 17.1.2. Vulnérabilité et leur classification
- 17.1.3. Principes de conception de la sécurité des Softwares
- 17.1.4. Références

17.2. Principes de conception de la sécurité des Softwares
- 17.2.1. Introduction
- 17.2.2. Principes de conception de la sécurité des logiciels
- 17.2.3. Types de S-SDLC
- 17.2.4. Sécurité des Softwares dans les phases S-SDLC
- 17.2.5. Méthodologies et normes
- 17.2.6. Références

17.3. Sécurité dans le cycle de vie du Software dans les phases d'exigences et de conception
- 17.3.1. Introduction
- 17.3.2. Modélisation des attaques
- 17.3.3. Cas d'abus
- 17.3.4. Ingénierie des exigences de sécurité
- 17.3.5. Analyse des risques Architectural
- 17.3.6. Modèles de conception
- 17.3.7. Références

17.4. Sécurité du cycle de vie des Softwares dans les phases de codage, de test et d'exploitation
- 17.4.1. Introduction
- 17.4.2. Tests de sécurité fondés sur le risque
- 17.4.3. Examen du code
- 17.4.4. Test de pénétration
- 17.4.5. Opérations de sécurité
- 17.4.6. Examen externe
- 17.4.7. Références

17.5. Applications de codage sécurisé I
- 17.5.1. Introduction
- 17.5.2. Pratiques de codage sécurisées
- 17.5.3. Traitement et validation des entrées
- 17.5.4. Débordement de mémoire
- 17.5.5. Références

17.6. Applications de codage sécurisé II
- 17.6.1. Introduction
- 17.6.2. *Integers Overflows*, erreurs de troncature et problèmes de conversion de type entre entiers
- 17.6.3. Erreurs et exceptions
- 17.6.4. Vie privée et confidentialité
- 17.6.5. Programmes privilégiés
- 17.6.6. Références

17.7. Développement et sécurité du cloud
- 17.7.1. Sécurité du développement; méthodologie et pratique
- 17.7.2. Modèles PaaS, PaaS et SaaS
- 17.7.3. Sécurité dans le nuage et pour les services en nuage

17.8. Cryptage
- 17.8.1. Principes fondamentaux de la communication
- 17.8.2. Cryptage symétrique et asymétrique
- 17.8.3. Cryptage au repos et en transit

17.9. Orchestration et automatisation de la sécurité (SOAR)
- 17.9.1. Complexité du traitement manuel: nécessité d'automatiser les tâches
- 17.9.2. Produits et services
- 17.9.3. Architecture du SOAR

17.10. Sécurité dans le télétravail
- 17.10.1. Besoin et scénarios
- 17.10.2. Produits et services
- 17.10.3. Sécurité dans le télétravail

Module 18. Administration de serveurs Web

18.1. Introduction aux serveurs web
- 18.1.1. Qu'est-ce qu'un serveur web?
- 18.1.2. Architecture et fonctionnement d'un serveur web
- 18.1.3. Ressources et contenus dans un serveur web
- 18.1.4. Serveurs d'application
- 18.1.5. Serveurs Proxy
- 18.1.6. Principaux serveurs web sur le marché
- 18.1.7. Statistiques d'utilisation du serveur Web
- 18.1.8. Sécurité des serveurs Web
- 18.1.9. Équilibrage des charges dans les serveurs web
- 18.1.10. Références

18.2. Gestion du protocole HTTP
- 18.2.1. Fonctionnement et structure
- 18.2.2. Description des demandes ou *request methods*
- 18.2.3. Codes d'état
- 18.2.4. En-têtes
- 18.2.5. Codage du contenu Pages de code

  Effectuer des requêtes HTTP sur Internet en utilisant un proxy, *Livehttpheaders* ou une méthode similaire, en analysant le protocole utilisé

18.3. Description des architectures distribuées sur plusieurs serveurs
- 18.3.1. Modèle à 3 couches
- 18.3.2. Modèle à 3 couches
- 18.3.3. Partage de la charge
- 18.3.4. Magasins de l'État de la session
- 18.3.5. Magasins de cache

18.4. Internet Information Services (IIS)
- 18.4.1. Qu'est-ce que IIS?
- 18.4.2. Histoire et évolution de l'IIS
- 18.4.3. Principaux avantages et caractéristiques de IIS7 et au-delà
- 18.4.4. Architecture IIS7 et supérieure

18.5. Installation, administration et configuration de IIS
- 18.5.1. Préambule
- 18.5.2. Installation d' Internet Information Services (IIS)
- 18.5.3. Outils d'administration d'IIS
- 18.5.4. Créer, configurer et administrer des sites Web
- 18.5.5. Installation et gestion des extensions IIS

18.6. Sécurité avancée dans IIS
- 18.6.1. Préambule
- 18.6.2. Authentification, autorisation et contrôle d'accès à IIS
- 18.6.3. Configuration d'un site Web sécurisé sur IIS avec SSL
- 18.6.4. Politiques de sécurité mises en œuvre dans IIS 18.x

18.7. Introduction à Apache
- 18.7.1. Qu'est-ce qu'Apache?
- 18.7.2. Principaux avantages d'Apache
- 18.7.3. Caractéristiques principales d'Apache
- 18.7.4. Architecture

18.8. Installation et configuration d'Apache
- 18.8.1. Installation initiale d'Apache
- 18.8.2. Configuration d'Apache

18.9. Installer et configurer les différents modules Apache
- 18.9.1. Installation des modules Apache
- 18.9.2. Types de modules
- 18.9.3. Configuration sécurisée d'Apache

18.10. Sécurité avancée
- 18.10.1. Authentification, autorisation et contrôle d'accès
- 18.10.2. Méthodes d'authentification
- 18.10.3. Configuration sécurisée d'Apache avec SSL

Module 19. Contrôles de sécurité

19.1. Introduction aux systèmes d'information et à leur contrôle
- 19.1.1. Introduction aux systèmes d'information et au rôle de contrôle informatique
- 19.1.2. Définitions de l'audit informatique et du contrôle interne informatique
- 19.1.3. Fonctions et objectifs des contrôles informatiques
- 19.1.4. Différences entre le contrôle interne et le contrôle informatique

19.2. Contrôles internes des systèmes d'information
- 19.2.1. Organigramme fonctionnel d'un centre de traitement des données
- 19.2.2. Classification de la contrôles des systèmes d'information
- 19.2.3. La règle d'or

19.3. Le processus et les phases de contrôle des systèmes d'information
- 19.3.1. Évaluation des risques (RRA) et autres méthodologies d'audit informatique
- 19.3.2. Réalisation d'un audit des systèmes d'information. Phases de contrôle
- 19.3.3. Compétences fondamentales de l'auditeur de systèmes d'information

19.4. Contrôle de la sécurité technique des systèmes et des réseaux
- 19.4.1. Contrôles techniques de sécurité. Tests d'intrusion. Concepts préliminaires
- 19.4.2. Contrôles de sécurité des systèmes Outils de soutien
- 19.4.3. Contrôles de sécurité des Réseaux Outils de soutien

19.5. Contrôle de la sécurité technique des Internet et des dispositifs mobiles
- 19.5.1. Contrôles de sécurité Internet. Outils de soutien
- 19.5.2. Contrôle de la sécurité des des Dispositifs mobiles. Outils de soutien
- 19.5.3. Annexe 1. Structure du rapport exécutif et du rapport technique
- 19.5.4. Annexe 2. Inventaire des outils
- 19.5.5. Annexe 3. Méthodologies

19.6. Système de gestion de la sécurité de l'information
- 19.6.1. Sécurité des SI: propriétés et facteurs d'influence
- 19.6.2. Risque d'entreprise et gestion des risques: mise en œuvre des contrôles
- 19.6.3. Système de gestion de la sécurité de l'information (SGSI): concept et facteurs critiques de succès
- 19.6.4. ISMS - Modèle PDCA
- 19.6.5. ISMS ISO-IEC 27001: Contexte organisationnel
- 19.6.6. Contexte organisationnel
- 19.6.7. Leadership
- 19.6.8. Planification
- 19.6.9. Soutien
- 19.6.10. Opération
- 19.6.11. Évaluation des performances
- 19.6.12. Amélioration
- 19.6.13. Annexe à l'ISO 27001/ISO-IEC 27002: Objectifs et contrôles
- 19.6.14. Contrôle du SGSI

19.7. Réalisation de l'audit
- 19.7.1. Procédures
- 19.7.2. Techniques

19.8. Traçabilité
- 19.8.1. Méthodologies
- 19.8.2. Analyse

19.9. Garde
- 19.9.1. Techniques
- 19.9.2. Résultats

19.10. Rapports et preuves
- 19.10.1. Types de rapports
- 19.10.2. Analyse des données
- 19.10.3. Présentation des preuves

Module 20. Sécurité des applications en ligne

20.1. Vulnérabilités et problèmes de sécurité dans les applications en ligne
- 20.1.1. Introduction à la sécurité des applications en ligne
- 20.1.2. Failles de sécurité dans la conception des applications web
- 20.1.3. Vulnérabilités de sécurité dans la mise en œuvre des applications web
- 20.1.4. Vulnérabilité de la sécurité dans le déploiement des applications web
- 20.1.5. Listes officielles de failles de sécurité

20.2. Politiques et normes pour la sécurité des applications en ligne
- 20.2.1. Piliers de la sécurité des applications en ligne
- 20.2.2. Politique de sécurité
- 20.2.3. Système de gestion de la sécurité de l'information
- 20.2.4. Cycle de vie du développement Software sécurisé
- 20.2.5. Normes de sécurité des applications

20.3. Sécurité dans la conception des applications web
- 20.3.1. Introduction à la sécurité des applications en Web
- 20.3.2. Sécurité dans la conception des applications web

20.4. Tester la sécurité en ligne et la protection des applications web
- 20.4.1. Analyse et test de la sécurité des applications Web
- 20.4.2. Sécurité du déploiement et de la production des applications Web

20.5. Sécurité des services Web
- 20.5.1. Introduction à la sécurité des services web
- 20.5.2. Fonctions et technologies de sécurité des services Web

20.6. Tester la sécurité et la protection en ligne des services web
- 20.6.1. Évaluation de la sécurité des services web
- 20.6.2. Protection en ligne. *Firewalls* et *Gateways* XML

20.7. *Hacking* étique, malware et *Forensic*
- 20.7.1. Piratage éthique
- 20.7.2. Analyse des malwares
- 20.7.3. Analyse médico-légale

20.8. Meilleures pratiques pour assurer la sécurité des applications
- 20.8.1. Manuel de bonnes pratiques pour le développement d'applications en ligne
- 20.8.2. Manuel de bonnes pratiques pour la mise en œuvre des applications en ligne

20.9. Erreurs courantes qui compromettent la sécurité des applications
- 20.9.1. Erreurs courantes de développement
- 20.9.2. Erreurs courantes en matière d'hébergement
- 20.9.3. Erreurs courantes dans la production

En passant ce Mastère Avancé, vous ferez non seulement un pas décisif vers l'élargissement de vos connaissances en ingénierie informatique spécialisée dans les Softwares, mais aussi vers une carrière professionnelle prospère et réussie"

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

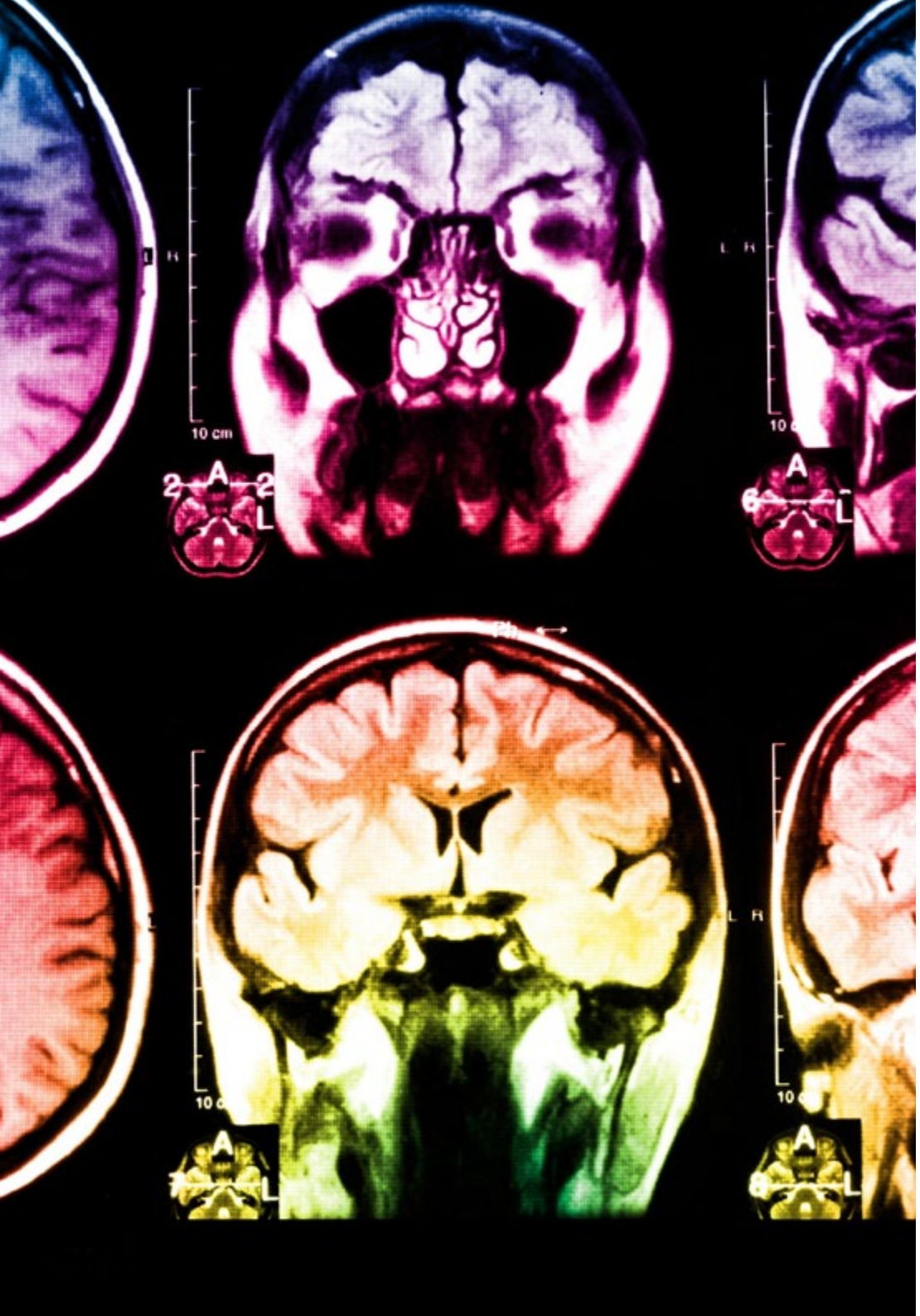
TECH combine efficacement la méthodologie des Études de Cas avec un système d'apprentissage 100% en ligne basé sur la répétition, qui associe différents éléments didactiques dans chaque leçon.

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.

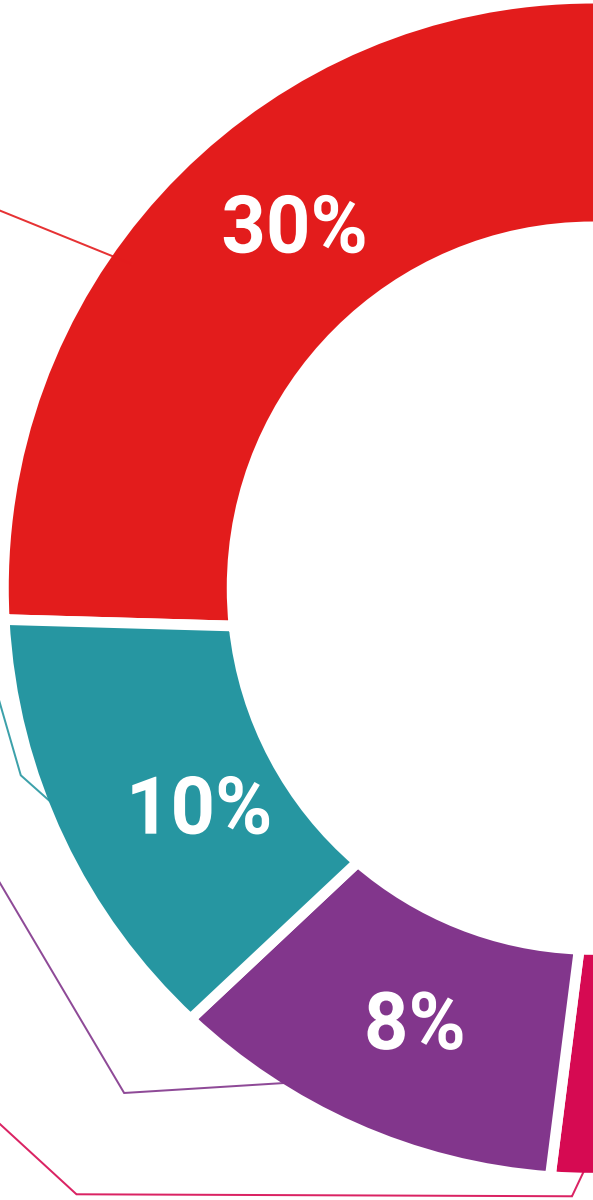
La méthode "Learning from an Expert" renforce les connaissances et la mémoire, et donne confiance dans les futures décisions difficiles.

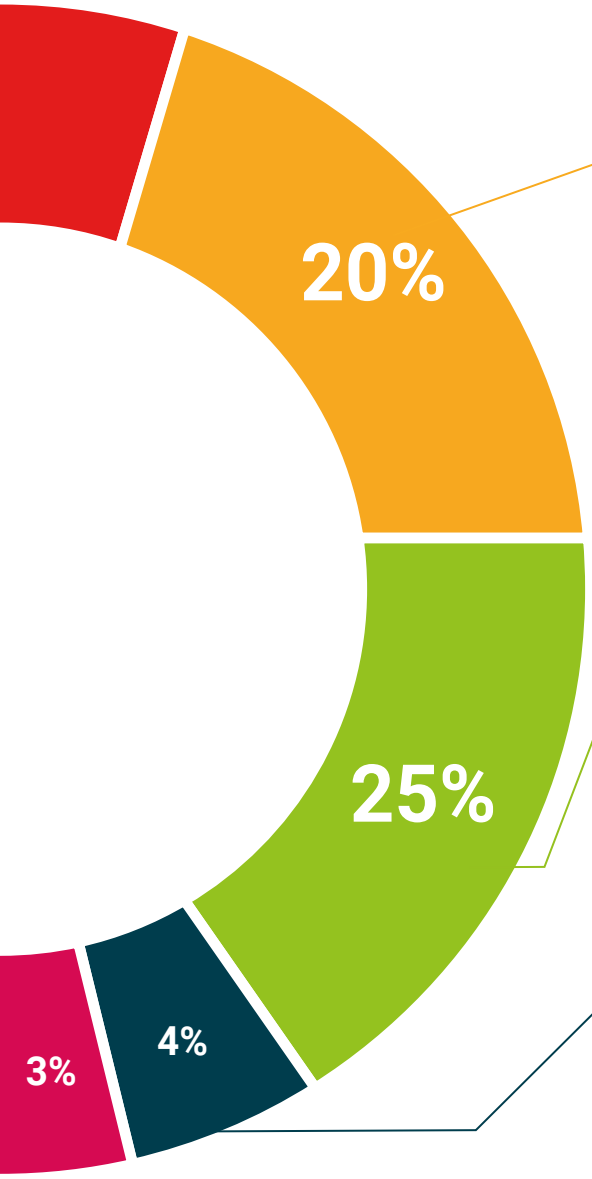
Pratiques en compétences et aptitudes

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 Avancé en Ingénierie et 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 Avancé délivré par TECH Université Technologique.

Terminez ce programme avec succès et recevez votre Mastère Avancé sans avoir à vous soucier des contraintes de déplacements ou des formalités administratives”

Ce **Mastère Avancé en Ingénierie et 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 Avancé** délivré par **TECH Université Technologique**.

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

Diplôme: **Mastère Avancé en Ingénierie et Qualité du Software**

N.º d'heures Officielles: **3.000 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 AVANCÉ

en

Ingénierie et Qualité du Software

Il s'agit d'un diplôme spécialisé octroyé par cette Université d'une durée de 3.000 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 Avancé en Ingénierie et Qualité du Software

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 Development). Conception du Software pilotée 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 du Software	150	OB
1º	Conception de Bases de Données (BD). Normalisation et performance. Qualité du Software	150	OB
1º	Conception d'architectures évolutives. L'architecture dans le cycle de vie du Software	150	OB

Cours	Matière	Heures	Type
2º	Critères de Qualité ISO, IEC 9126. Mesures de la Qualité du Software	150	OB
2º	Méthodologies, développement et Qualité en Ingénierie du Software	150	OB
2º	Gestion de projets Software	150	OB
2º	Plateformes de développement de Software	150	OB
2º	Informatique client Web	150	OB
2º	Informatique du serveur web	150	OB
2º	Gestion de la sécurité	150	OB
2º	Sécurité du Software	150	OB
2º	Administration de serveurs Web	150	OB
2º	Contrôles de sécurité	150	OB
2º	Sécurité des applications en ligne	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 Avancé
Ingénierie et Qualité du Software

- Modalité: **en ligne**
- Durée: **2 ans**
- Qualification: **TECH Université Technologique**
- Intensité: **16h/semaine**
- Horaire: **à votre rythme**
- Examens: **en ligne**

Mastère Avancé

Ingénierie et Qualité du Software

tech université technologique