Master Privato
Qualità del Software

tech università tecnologica

tech università tecnologica

Master Privato
Qualità del Software

- Modalità: **online**
- Durata: **12 mesi**
- Titolo: **TECH Università Tecnologica**
- Dedizione: **16 ore/settimana**
- Orario: **a scelta**
- Esami: **online**

Accesso al sito web: **www.techtitute.com/it/informatica/master/master-qualita-software**

Indice

01

Presentazione

La rapida crescita del settore e le attuali richieste del mercato hanno portato a un elevato livello di debito tecnico nei progetti software. A causa dell'imperiosa necessità di dare risposte rapide alle esigenze del cliente o dell'azienda, è stata spesso trascurata l'attenzione nel valutare o specificare i dettagli della qualità del sistema. È qui che si riflette la necessità di prendere in considerazione la scalabilità del progetto per tutto il suo ciclo di vita, che richiede conoscenze informatiche incentrate sulla qualità da un approccio *top-down.* Questo programma sviluppa i criteri, i compiti e le metodologie avanzate per comprendere la rilevanza di un lavoro orientato alla necessità di implementare politiche di qualità nelle *Software Factories.* Il percorso si svolgerà completamente online e durerà 12 mesi, seguendo la metodologia implementata dalla più grande università digitale del mondo.

Specializzati in Qualità del Software da una prospettiva tecnica e gestionale; ottieni la tua qualifica in 12 mesi e fai la differenza nel tuo ambiente professionale"

Il concetto di Debito Tecnico, attualmente applicato da un gran numero di aziende e amministrazioni con i loro fornitori, riflette il modo improvvisato in cui i progetti sono stati sviluppati. Questo genera un nuovo costo implicito dovuto al dover rifare un progetto per aver adottato una soluzione rapida e semplice, piuttosto che preferire un approccio scalabile nell'evoluzione del progetto.

Da qualche anno a questa parte, i progetti vengono sviluppati molto rapidamente, con l'obiettivo di chiuderli con il cliente in base a criteri di prezzo e scadenza, invece di adottare un approccio qualitativo. Queste decisioni si ripercuotono su molti fornitori e clienti.

Questo Master Privato consentirà ai professionisti dell'IT di analizzare i criteri alla base della Qualità del Software a tutti i livelli. Criteri come la standardizzazione dei database, il disaccoppiamento tra i componenti di un sistema informativo, le architetture scalabili, le metriche, la documentazione, sia funzionale che tecnica. Oltre alle metodologie di gestione e sviluppo dei progetti e ad altri metodi per garantire la qualità, come le tecniche di lavoro collaborativo, tra cui il *Pair Programming*, che fa sì che la conoscenza risieda nell'azienda e non nelle persone.

La maggior parte di questo tipo di Master Privato si concentra su una tecnologia, un linguaggio o uno strumento. Questo programma è unico nel suo genere in quanto rende il professionista consapevole dell'importanza della qualità del software, riducendo il debito tecnico dei progetti con un approccio di qualità piuttosto che con un approccio basato sull'economia e sulle scadenze brevi; fornisce allo studente conoscenze specialistiche, in modo da giustificare il budget del progetto.

Per questo TECH Università Tecnologica ha riunito un gruppo di esperti del settore che trasmetteranno le conoscenze e le esperienze più aggiornate. Attraverso un moderno campus virtuale con contenuti teorici e pratici, distribuiti in diversi formati. Il programma è composto da 10 moduli suddivisi in diversi argomenti e sottoargomenti che renderanno possibile l'apprendimento in 12 mesi, seguendo la metodologia *Relearning*, che facilita la memorizzazione e l'apprendimento in modo agile ed efficiente.

Questo **Master Privato in Qualità del Software** possiede il programma scientifico più completo e aggiornato del mercato. Le caratteristiche principali del corso sono:

- Sviluppo di casi di studio presentati da esperti in sviluppo di software
- Contenuti grafici, schematici ed eminentemente pratici che forniscono informazioni scientifiche e pratiche riguardo alle discipline essenziali per l'esercizio della professione
- Esercizi pratici che offrono un processo di autovalutazione per migliorare l'apprendimento
- Speciale enfasi sulle metodologie innovative
- Lezioni teoriche, domande all'esperto e lavori di riflessione individuale
- Contenuti disponibili da qualsiasi dispositivo fisso o mobile dotato di connessione a internet

Il Master Privato in Qualità del Software analizza i criteri alla base della materia a tutti i livelli. Amplia il tuo livello di competenza. Iscriviti subito"

Sviluppa i criteri, i compiti e le metodologie avanzate per comprendere la rilevanza del lavoro orientato alla qualità e fornisci soluzioni efficaci alla tua azienda o al tuo cliente"

Un programma incentrato sulla sensibilizzazione dell'importanza della qualità del software e sulla necessità di implementare politiche di qualità nelle Software Factories.

Impara in modo pratico e flessibile. Condividi il tuo tempo quotidiano con questo programma 100% online di TECH Università Tecnologica.

Il personale docente del programma comprende rinomati specialisti del settore che forniscono agli studenti le competenze necessarie a intraprendere un percorso di studio eccellente.

I contenuti multimediali, sviluppati in base alle ultime tecnologie educative, forniranno al professionista un apprendimento coinvolgente e localizzato, ovvero inserito in un contesto reale.

La creazione di questo programma è incentrata sull'Apprendimento Basato su Problemi, mediante il quale lo specialista deve cercare di risolvere le diverse situazioni che gli si presentano durante il corso. Sarai supportato da un innovativo sistema di video interattivi sviluppato da esperti rinomati.

02

Obiettivi

Il Master Privato in Qualità del Software fornisce agli studenti una visione chiara e specializzata dell'importanza della qualità nei processi di sviluppo del software. Oltre agli strumenti più avanzati per l'implementazione dei processi DevOps e dei sistemi per il controllo qualità. In breve, fornirà una conoscenza teorica e pratica ampia e specializzata per comprendere lo sviluppo dei progetti da una prospettiva moderna ed efficiente.

È possibile accedere facilmente a tutti i contenuti quando preferisci. Dal computer o dal dispositivo preferito. Potrai anche scaricarli per la tua prossima consultazione"

Obiettivi generali

- Sviluppare i criteri, i compiti e le metodologie avanzate per comprendere la rilevanza del lavoro orientato alla qualità
- Analizzare i fattori chiave della qualità di un progetto software
- Sviluppare gli aspetti normativi pertinenti
- Implementare i processi DevOps e i sistemi per il controllo della qualità
- Ridurre il debito tecnico dei progetti con un approccio di qualità piuttosto che con un approccio basato sull'economia e sulle scadenze brevi
- Fornire allo studente il know-how per essere in grado di misurare e quantificare la qualità di un progetto software
- Difendere le proposte economiche dei progetti sulla base della qualità

Obiettivi specifici

Modulo 1. Qualità del Software. Livelli di sviluppo TRL

- Sviluppare in modo chiaro e conciso gli elementi della qualità del software
- Applicare i modelli e gli standard in base al sistema, al prodotto e al processo software
- Approfondire le norme di qualità ISO applicate sia in generale che in parti specifiche
- Applicare gli standard in base all'ambito dell'ambiente (locale, nazionale, internazionale)
- Esaminare i livelli di maturità TRL e adattarli alle diverse parti del progetto software da trattare
- Acquisire la capacità di astrazione per applicare uno o più criteri di elementi e livelli di qualità del software
- Distinguere i casi di applicazione degli standard e dei livelli di maturità in un progetto reale simulato

Modulo 2. Sviluppo di Progetti Software. Documentazione funzionale e tecnica

- Determinare l'influenza della gestione dei progetti sulla qualità
- Sviluppare le diverse fasi di un progetto
- Distinguere i concetti di qualità inerenti alla documentazione funzionale e tecnica
- Analizzare la fase di raccolta dei requisiti, la fase di analisi, la gestione del team e la fase di costruzione
- Stabilire le diverse metodologie di gestione dei progetti software
- Generare criteri per decidere quale sia la metodologia più appropriata in base al tipo di progetto

Modulo 3. *Testing* di Software. Automazione dei test

- Stabilire le differenze tra qualità del prodotto, qualità del processo e qualità d'uso
- Comprendere lo standard ISO/IEC 15504
- Determinare i dettagli di CMMI
- Imparare le chiavi dell'integrazione continua, i repository e le ripercussioni che hanno su un team di sviluppo software
- Stabilire la rilevanza dell'incorporazione dei repository per i progetti software Imparare a crearli con TFS
- Assimilare l'importanza della scalabilità del software nella progettazione e nello sviluppo dei sistemi informativi

Modulo 4. Metodologie di Gestione dei Progetti Software. Metodologie *Waterfall* contro metodologie agili

- Determinare in cosa consiste la metodologia Waterfall
- Approfondire la metodologia Scrum
- Stabilire le differenze tra Waterfall e Scrum
- Specificare le differenze tra le metodologie Waterfall e Scrum e il modo in cui il cliente le percepisce
- Esaminare il Panel Kanban
- Approcciarsi allo stesso progetto con Waterfall e Scrum
- Impostare un progetto ibrido

Modulo 5. TDD (*Test Driven Development*). Progettazione del Software guidata dai test

- Conoscere l'applicazione pratica del TDD e le sue possibilità di testare un progetto software in futuro
- Completare i casi di simulazione reali proposti, come apprendimento continuo del concetto di TDD
- Analizzare, nei casi di simulazione, in che misura i test possono avere successo o fallire da un punto di vista costruttivo
- Determinare le alternative al TDD, effettuando un'analisi comparativa tra di esse

Modulo 6. DevOps. Gestione della Qualità del Software

- Analizzare le carenze di un processo tradizionale
- Valutare le possibili soluzioni e scegliere quella più adatta
- Comprendere le esigenze aziendali e il loro impatto sull'implementazione
- Valutare i costi dei miglioramenti da implementare
- Sviluppare un ciclo di vita del software evolutivo, adattato alle esigenze reali
- Anticipare i possibili errori ed evitarli durante il processo di progettazione
- Giustificare l'uso dei diversi modelli di implementazione

Modulo 7. DevOps e Integrazione Continua. Soluzioni pratiche avanzate nello sviluppo di software

- Identificare le fasi del ciclo di sviluppo e consegna del software adattate a casi particolari
- Progettare un processo di consegna del software utilizzando l'integrazione continua
- Costruire e implementare l'integrazione e il deployment continui sulla base del progetto precedente
- Stabilire punti di controllo automatici della qualità per ogni consegna di software
- Mantenere un processo di consegna del software automatizzato e robusto
- Adattare le esigenze future al processo di integrazione e distribuzione continua
- Analizzare e anticipare le vulnerabilità della sicurezza durante e dopo il processo di consegna del software

Modulo 8. Progettazione di Database (DB). Normalizzazione e Rendimento. Qualità del Software

- Valutare l'uso del modello entità-relazione per la progettazione preliminare di un database
- Applicare un'entità, un attributo, una chiave, ecc. per ottenere la migliore integrità dei dati
- Valutare le dipendenze, le forme e le regole della normalizzazione dei database
- Specializzarsi nella gestione di un sistema di data warehouse OLAP, nello sviluppo e nell'utilizzo di tabelle di dati e di dimensioni
- Determinare i punti chiave per le prestazioni del database
- Completare i casi di simulazione proposti nel mondo reale come esperienza di apprendimento continuo in materia di progettazione, normalizzazione e prestazioni dei database
- Stabilire nei casi di simulazione le opzioni da risolvere nella creazione del database da un punto di vista costruttivo

Modulo 9. Progettazione di Architetture Scalabili. L'Architettura nel Ciclo di Vita del Software

- Sviluppare il concetto di architettura del software e le sue caratteristiche
- Determinare i diversi tipi di scalabilità nell'architettura del software
- Analizzare i diversi livelli che possono verificarsi nella scalabilità del Web
- Acquisire una conoscenza specialistica del concetto, delle fasi e dei modelli del ciclo di vita del software
- Determinare l'impatto di un'architettura sul ciclo di vita del software, con i suoi vantaggi, limiti e strumenti di supporto
- Completare i casi di simulazione reali proposti, come apprendimento continuo dell'architettura e del ciclo di vita del software
- Valutare, nei casi di simulazione, in che misura possono rendere il progetto dell'architettura fattibile o non necessario

Modulo 10. Criteri di qualità ISO/IEC 9126. Metriche della Qualità del Software

- Sviluppare il concetto di criteri di qualità e gli aspetti rilevanti
- Esaminare lo standard ISO/IEC 9126, gli aspetti principali e gli indicatori
- Analizzare le diverse metriche di un progetto software per soddisfare le valutazioni concordate
- Esaminare gli attributi interni ed esterni da affrontare nella qualità di un progetto software
- Distinguere le metriche in base al tipo di programmazione (strutturata, orientata agli oggetti, a strati, ecc.)
- Completare casi di simulazione reali, come apprendimento continuo della misurazione della qualità
- Vedere nei casi di simulazione fino a che punto è fattibile o non necessario, cioè dal punto di vista costruttivo degli autori

Metti in risalto il tuo profilo professionale con questo programma di specializzazione esclusivo. Ottieni la tua qualifica in 12 mesi e in modo pratico con la metodologia che solo TECH Università Tecnologica può offrirti"

03

Competenze

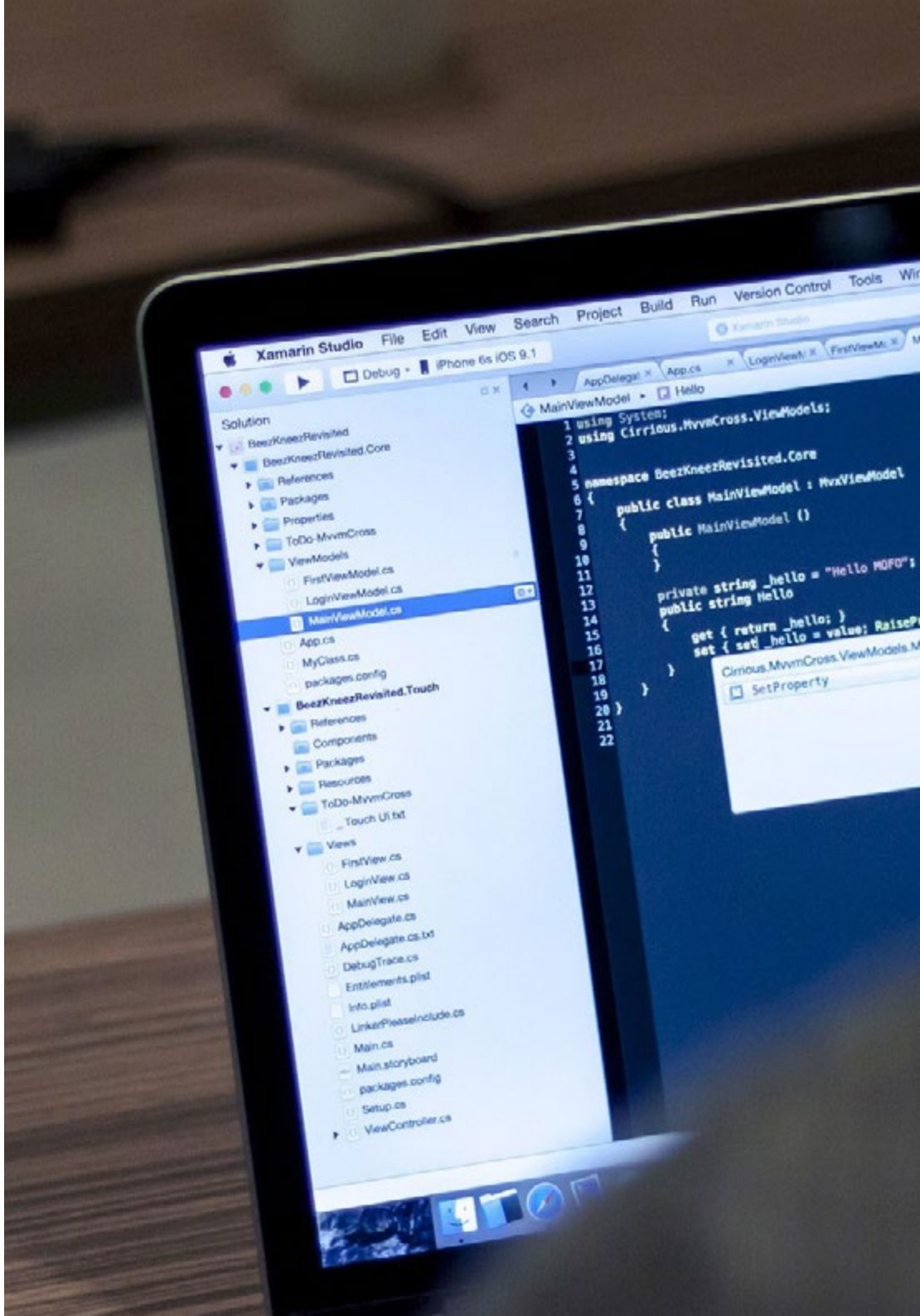
Chi porta a termine questo Master Privato in Qualità del Software padroneggerà la materia sia dal punto di vista tecnico che gestionale. Sarà in grado di sviluppare l'approccio a un progetto, nonché la sua esecuzione e di sviluppare un'architettura sostenibile, efficace e di qualità nei progetti software che gli verranno presentati. A tal fine, il personale docente ha messo a disposizione tutta la propria esperienza per elaborare una moltitudine di casi pratici, che serviranno a spiegare il contesto e l'evoluzione di tale "debito tecnico" che non sarà più presente.

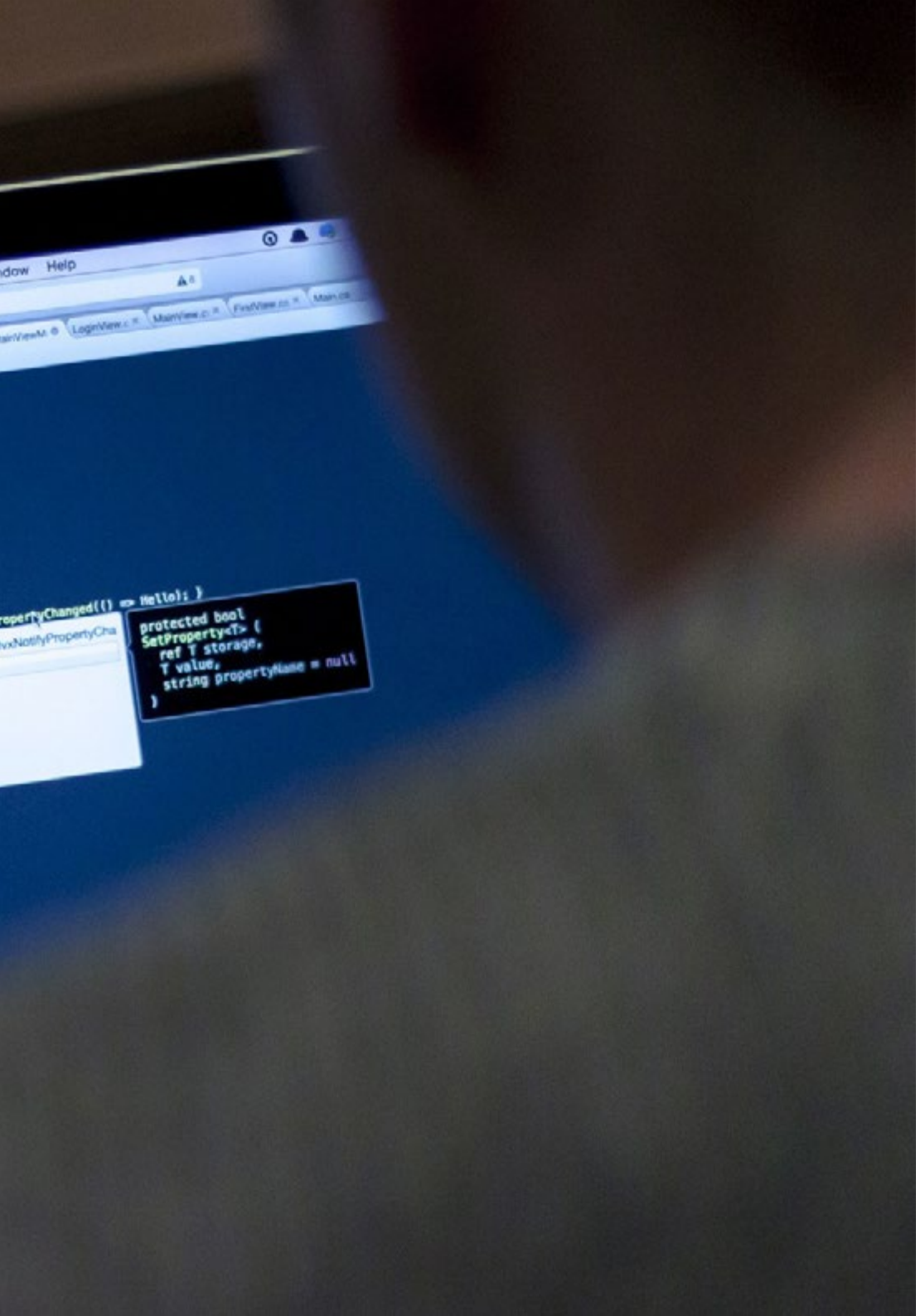
Un informatico orientato alla qualità è un valore in crescita per le società di consulenza software e le grandi aziende. Iscriviti ora a questo Master Privato in Qualità del Software"

Competenze generali

- Ridurre il debito tecnico dei progetti con un approccio di qualità piuttosto che con un approccio basato sull'economia e sulle scadenze brevi
- Misurare e quantificare la qualità di un progetto software
- Eseguire correttamente il TDD, per aumentare gli standard di qualità del software
- Giustificare il budget di progetti orientati alla qualità
- Sviluppare standard, modelli e norme di qualità
- Esaminare diverse valutazioni della maturità tecnologica
- Ridurre il rischio e garantire la manutenzione e il controllo delle release successive
- Padroneggiare le fasi in cui è suddiviso un progetto

Migliora le tue competenze e scopri le infinite possibilità di crescita professionale che si aprono con questa nuova esperienza"





Competenze specifiche

- Valutare un sistema software in termini di grado di avanzamento del processo di progetto
- Affrontare questi punti di affidabilità, metrica e garanzia nei progetti software in modo corretto e strategico
- Affrontare il processo di decisione della metodologia da utilizzare nel progetto
- Padroneggiare gli aspetti normativi essenziali per la creazione di software
- Sviluppare il *Testing* in modo automatico
- Stabilire una comunicazione adeguata con il cliente, comprendendo il modo in cui percepisce il progetto in base alla metodologia applicata
- Elaborare l'elenco dei requisiti di test
- Eseguire l'astrazione, la suddivisione in test più unitari ed eliminare ciò che non serve al buon svolgimento dei test del progetto software da realizzare
- Aggiornare l'elenco dei requisiti di prova in modo misurato e corretto
- Adattare la cultura DevOps alle esigenze aziendali
- Sviluppare le pratiche e gli strumenti più recenti per l'integrazione e il deployment continui
- Riformulare e affrontare la gestione e il coordinamento dei dati

04

Direzione del corso

Docenti esperti con un ampio curriculum nell'area delle soluzioni IT e dello sviluppo e ricerca software, guidano questo Master Privato per fornire gli strumenti e le conoscenze necessarie al professionista focalizzato sulla qualità nei processi di sviluppo software e sugli strumenti più avanzati per implementare processi DevOps e sistemi per l'assicurazione della qualità. Questo team di professionisti guiderà lo studente in ogni momento, per raggiungere gli obiettivi a distanza, trattandosi di un programma totalmente online e seguendo la metodologia più all'avanguardia implementata da TECH.

Insegnanti specializzati si impegnano a fornire i migliori contenuti e a rendere il processo di apprendimento un'esperienza agile e dinamica. Chiariranno i tuoi dubbi e ti accompagneranno lungo tutto il percorso"

Direzione

Dott. Molina Molina, Jerónimo

- IA Engineer & Software Architect. NASSAT-"Internet Satélite en Movimiento"
- Consulente presso "Sr. En Hexa Ingenieros" Introduttore di Intelligenza Artificiale (ML e CV)
- Esperto di soluzioni basate sull'Intelligenza Artificiale nei settori della Computer Vision, ML/DL e NLP Attualmente sta studiando le possibilità di applicazione di Transformers e Reinforcement Learning in un progetto di ricerca personale
- Esperto universitario in Creazione e Sviluppo di Imprese Bancaixa-FUNDEUN Alicante
- Ingegnere Informatico Università di Alicante
- Master in Intelligenza Artificiale Università Cattolica di Ávila
- MBA-Executive. Forum Europeo Campus Aziendale

Personale docente

Dott. Pi Morell, Oriol

- Product Owner di Hosting e posta CDMON
- Analista funzionale e Software Engineer in diverse organizzazioni come Fihoca, Atmira, CapGemini
- Insegnante di diversi corsi come BPM in CapGemini, ORACLE Forms CapGemini, Business Processes Atmira
- Laurea in Ingegneria Tecnica in Gestione Informatica presso l'Università Autonoma di Madrid
- Master in Intelligenza Artificiale
- Master in Direzione e Amministrazione d'Impresa MBA
- Master in Gestione dei Sistemi di Informazione con vasta esperienza di insegnamento
- Corso Post-laurea in Modelli di Design Università Politecnica della Catalogna

Dott. Tenrero Morán, Marcos

- DevOps Engineer-Allot Communications
- Application Lifecycle Management & DevOps-Meta4 Spain. Cegid
- Ingegnere dell'automazione, QA-Meta4 Spain. Cegid
- Laurea in Ingegneria dei Computer presso l'Università Rey Juan Carlos di Madrid
- Sviluppo di applicazioni professionali per Android-Università Galileo (Guatemala)
- Sviluppo di Servizi Cloud (nodeJs, JavaScript, HTML5)-UPM
- Integrazione Continua con Jenkins-Meta4. Cegid
- Sviluppatore Web con Angular-CLI (4), Ionic e nodeJS. Meta4-Università Rey Juan Carlos

Dott. Peralta Martín-Palomino, Arturo

- CEO e CTO presso Prometeus Global Solutions
- CTO presso Korporate Technologies
- CTO presso AI Shephers GmbH
- Dottore in Ingegneria Informatica presso l'Università di Castilla La Mancha
- Dottore in Economia Aziendale e Finanze presso l'Università Camilo José Cela Premio di Eccellenza del Dottorato
- Laurea in Psicologia presso l'università di CastillaLa Mancha
- Master in Tecnologie Informatiche Avanzate presso l'Università di Castiglia La Mancia
- Master MBA+E (Master in Amministrazione Aziendale e Ingegneria Organizzativa) presso l'Università di Castilla-La Mancha
- Professore associato con docenza nella Laurea triennale e Master in Ingegneria Informatica presso l'Università di Castiglia La Mancia
- Professore del Master in Big Data e Data Science presso l'Università Internazionale di Valencia
- Professore del Master in Industria 4.0 e Master in Disegno Industriale e Sviluppo di Prodotti
- Membro del Gruppo di Ricerca SMIL dell'Università di Castilla La Mancha

Dott.ssa Martínez Cerrato, Yésica

- Tecnico di prodotti di sicurezza elettronica presso Securitas Seguridad España
- Analista di Intelligenza Aziendale presso Ricopia Technologies (Alcalá de Henares) Laurea in Ingegneria Elettronica delle Comunicazioni presso la Scuola Politecnica Superiore dell'Università di Alcalá
- Responsabile delle nuove incorporazioni dei software di gestione commerciale (CRM, ERP, INTRANET), prodotti e procedure presso Ricopia Technologies (Alcalá de Henares)
- Responsabile dei nuovi tirocinanti incorporati alle Aule di Informatica dell'Università di Alcalá
- Responsabile di progetti nell'area dell'Integrazione di Grandi Account presso "Correos y Telégrafos" (Madrid)
- Tecnico Informatico-Responsabile delle aule informatiche OTEC presso l'Università di Alcalá (Alcalá de Henares)
- Professoressa di classi di Informatica presso l'Associazione ASALUMA (Alcalá de Henares)
- Tirocinio educativo come Tecnico Informatico presso OTEC, Università di Alcalá

Il nostro personale docente ti fornirà tutte le sue conoscenze in modo che tu sia aggiornato sulle ultime informazioni in materia"

05

Struttura e contenuti

La struttura e i contenuti di questo Master Privato sono stati sviluppati per coprire gli argomenti più importanti per lo sviluppo di Software di Qualità. Composto da 10 moduli didattici, che spaziano dallo sviluppo di progetti software, alla documentazione funzionale e tecnica, al *Test Driven Development* e alle diverse metodologie, fino all'implementazione di soluzioni pratiche avanzate con DevOps e l'integrazione continua, il tutto basato sul raggiungimento della qualità del software. L'ampio contenuto multimediale, rigorosamente selezionato da docenti esperti, sarà di grande supporto per alleggerire il carico didattico e servirà come materiale di riferimento per il futuro.

I casi pratici, basati sulla realtà, serviranno a rafforzare e contestualizzare tutta la teoria appresa durante il programma"

Modulo 1. Qualità del Software. Livelli di sviluppo TRL

1.1. Elementi che influenzano la qualità del software (I). Il debito tecnico
- 1.1.1. Il debito tecnico. Cause e conseguenze
- 1.1.2. Qualità del software. Principi generali
- 1.1.3. Software senza principi e con principi di qualità
  - 1.1.3.1. Conseguenze
  - 1.1.3.2. La necessità di applicare i principi della qualità nel software
- 1.1.4. Qualità del software. Tipologia
- 1.1.5. Qualità del software. Caratteristiche specifiche

1.2. Elementi che influenzano la qualità del software (II). Costi associati
- 1.2.1. Qualità del software. Elementi determinanti
- 1.2.2. Qualità del software. Idee sbagliate
- 1.2.3. Qualità del software. Costi associati

1.3. Modello di qualità del software (I). Gestione della conoscenza
- 1.3.1. Modelli di Qualità generali
  - 1.3.1.1. Gestione della qualità totale
  - 1.3.1.2. Modello Europeo di Eccellenza Aziendale (EFQM)
  - 1.3.1.3. Modello Six-Sigma
- 1.3.2. Modelli di Gestione della Conoscenza
  - 1.3.2.1. Modello Dyba
  - 1.3.2.2. Modello SEKS
- 1.3.3. Esperienza di fabbrica e paradigma QIP
- 1.3.4. Modelli di qualità d'uso (25010)

1.4. Modello di qualità del software (III). Qualità dei dati, dei processi e dei modelli SEI
- 1.4.1. Modello di qualità dei dati
- 1.4.2. Modello di processo del software
- 1.4.3. *Software & Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. Modelli del SEI
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. Standard ISO di qualità del software (I). Analisi degli Standard
- 1.5.1. Norme ISO 9000
  - 1.5.1.1. Norme ISO 9000
  - 1.5.1.2. Famiglia di standard di qualità ISO (9000)
- 1.5.2. Altri standard ISO relativi alla qualità
- 1.5.3. Standard di modellazione della qualità (ISO 2501)
- 1.5.4. Normativa di misurazione della qualità (ISO 2502n)

1.6. Standard ISO di qualità del software (II). Requisiti e valutazione
- 1.6.1. Standard dei requisiti di qualità (2503n)
- 1.6.2. Normativa sulla valutazione della qualità (2504n)
- 1.6.3. ISO/IEC 24744: 2007

1.7. Livelli di sviluppo TRL (I). Livelli da 1 a 4
- 1.7.1. Livelli TRL
- 1.7.2. Livello 1: principi di base
- 1.7.3. Livello 2: concetto e/o applicazione
- 1.7.4. Livello 3: funzione analitica critica
- 1.7.5. Livello 4: convalida dei componenti in ambiente di laboratorio

1.8. Livelli di sviluppo TRL (II). Livelli da 5 a 9
- 1.8.1. Livello 5: convalida del componente in un ambiente pertinente
- 1.8.2. Livello 6: modello di sistema/sottosistema
- 1.8.3. Livello 7: dimostrazione in ambiente reale
- 1.8.4. Livello 8: sistema completo e certificato
- 1.8.5. Livello 9: successo in un ambiente reale

1.9. Livelli di sviluppo TRL. Usi
- 1.9.1. Esempio di azienda con ambiente di laboratorio
- 1.9.2. Esempio di azienda di R&S&I
- 1.9.3. Esempio di azienda di R&S&I industriale
- 1.9.4. Esempio di joint venture laboratorio-ingegneria

1.10. Qualità del software. Dettagli principali
- 1.10.1. Dettagli metodologici
- 1.10.2. Dettagli tecnici
- 1.10.3. Dettagli sulla gestione dei progetti software
  - 1.10.3.1. Qualità dei sistemi informatici
  - 1.10.3.2. Qualità del prodotto software
  - 1.10.3.3. Qualità del processo software

Modulo 2. Sviluppo di Progetti Software. Documentazione funzionale e tecnica

2.1. Gestione dei progetti
- 2.1.1. Gestione di progetti sulla qualità del software
- 2.1.2. Gestione dei progetti Vantaggi
- 2.1.3. Gestione dei progetti Tipologia

2.2. Metodologia nella gestione di progetti
- 2.2.1. Metodologia nella gestione di progetti
- 2.2.2. Metodologie di progetto. Tipologia
- 2.2.3. Metodologie di gestione dei progetti. Applicazioni

2.3. Fase di identificazione dei requisiti
- 2.3.1. Identificazione dei requisiti del progetto
- 2.3.2. Gestione delle riunioni di progetto
- 2.3.3. Documentazione da fornire

2.4. Modello
- 2.4.1. Fase iniziale
- 2.4.2. Fase di analisi
- 2.4.3. Fase di costruzione
- 2.4.4. Fase di test
- 2.4.5. Consegna

2.5. Modello di dati da utilizzare
- 2.5.1. Determinazione del nuovo modello di dati
- 2.5.2. Identificazione del piano di migrazione dei dati
- 2.5.3. Set di dati

2.6. Impatto su altri progetti
- 2.6.1. Impatto di un progetto. Esempi
- 2.6.2. Rischi del progetto
- 2.6.3. Gestione del rischio

2.7. "*Must*" del progetto
- 2.7.1. *Must* del progetto
- 2.7.2. Identificazione dei *Must* del progetto
- 2.7.3. Identificazione dei punti di attuazione per la realizzazione di un progetto

2.8. Il team di costruzione del progetto
- 2.8.1. Ruoli da svolgere in base al progetto
- 2.8.2. Contatto con le risorse umane per il reclutamento
- 2.8.3. Consegna dei prodotti e calendario del progetto

2.9. Aspetti tecnici di un progetto software
- 2.9.1. Architetto del progetto. Aspetti tecnici
- 2.9.2. Leader tecnici
- 2.9.3. Costruzione del progetto software
- 2.9.4. Valutazione della qualità del codice, sonar

2.10. Prodotti da consegnare al progetto
- 2.10.1. Analisi funzionale
- 2.10.2. Modelli di dati
- 2.10.3. Diagrammi di stato
- 2.10.4. Documentazione tecnica

Modulo 3. *Testing di Software.* Automazione dei test

3.1. Modello di qualità del software
- 3.1.1. Qualità del prodotto
- 3.1.2. Qualità del processo
- 3.1.3. Qualità d'uso

3.2. Qualità del processo
- 3.2.1. Qualità del processo
- 3.2.2. Modelli di maturità

3.2.3. Normativa ISO 15504
3.2.3.1. Propositi
3.2.3.2. Contesto
3.2.3.3. Tappe
3.3. Normativa ISO/IEC 15504
3.3.1. Categorie di processo
3.3.2. Processo di sviluppo. Esempio
3.3.3. Frammento di profilo
3.3.4. Tappe
3.4. CMMI (*Capability Maturity Model Integration*)
3.4.1. CMMI. Integrazione dei modelli di maturità delle capacità
3.4.2. Modelli e aree. Tipologia
3.4.3. Aree di processo
3.4.4. Livelli di capacità
3.4.5. Gestione dei processi
3.4.6. Gestione dei progetti
3.5. Gestione delle modifiche e del repository
3.5.1. Gestione delle modifiche al software
3.5.1.1. Voce di configurazione. Integrazione continua
3.5.1.2. Linee
3.5.1.3. Diagrammi di flusso
3.5.1.4. *Rami*
3.5.2. Repository
3.5.2.1. Controllo delle versioni
3.5.2.2. Team di lavoro e utilizzo del repository
3.5.2.3. Integrazione continua nel repository
3.6. *Team Foundation Server* (TFS)
3.6.1. Installazione e configurazione
3.6.2. Creazione di un progetto di squadra
3.6.3. Aggiunta di contenuti al controllo del codice sorgente
3.6.4. *TFS on Cloud*
3.7. *Testing*
3.7.1. Motivazione per la realizzazione di test
3.7.2. Test di verifica
3.7.3. Test beta
3.7.4. Implementazione e manutenzione
3.8. Implementazione e manutenzione
3.8.1. *Load testing*
3.8.2. Test con *LoadView*
3.8.3. Test con K6 *Cloud*
3.8.4. Test con *Loader*
3.9. Test di unità, stress e resistenza
3.9.1. Motivazione dei test unitari
3.9.2. Strumenti per *Unit Testing*
3.9.3. Motivazione degli stress test
3.9.4. Test con *StressTesting*
3.9.5. Motivazione degli stress test
3.9.6. Test con *LoadRunner*
3.10. Scalabilità. Progettazione software scalabile
3.10.1. Scalabilità e architettura del software
3.10.2. Indipendenza tra gli strati
3.10.3. Accoppiamento tra gli strati. Modelli di architettura

Modulo 4. Metodologie di Gestione dei Progetti Software. Metodologie *Waterfall* contro metodologie agili

4.1. Metodologia *Waterfall*
4.1.1. Metodologia *Waterfall*
4.1.2. Metodologia *Waterfall* Influenza sulla qualità del software
4.1.3. Metodologia *Waterfall* Esempi
4.2. Metodologia *Agile*
4.2.1. Metodologia *Agile*
4.2.2. Metodologia *Agile*. Influenza sulla qualità del software
4.2.3. Metodologia *Agile*. Esempi

4.3. Metodologia Scrum
  4.3.1. Metodologia Scrum
  4.3.2. Manifesto Scrum
  4.3.3. Applicazione di Scrum
4.4. Panel Kanban
  4.4.1. Metodo Kanban
  4.4.2. Panel Kanban
  4.4.3. Panel Kanban. Esempi di applicazione
4.5. Gestione del progetto con *Waterfall*
  4.5.1. Fasi di un progetto
  4.5.2. Visione in un progetto *Waterfall*
  4.5.3. Prodotti da prendere in considerazione
4.6. Gestione del progetto con Scrum
  4.6.1. Fasi di un progetto Scrum
  4.6.2. Visione in un progetto Scrum
  4.6.3. Aspetti da considerare
4.7. *Waterfall* vs. Scrum, confronto
  4.7.1. Approccio al progetto pilota
  4.7.2. Il progetto applicando *Waterfall.* Esempio
  4.7.3. Il progetto applicando Scrum. Esempio
4.8. Visione del cliente
  4.8.1. Documenti in *Waterfall*
  4.8.2. Documenti in Scrum
  4.8.3. Confronto
4.9. Struttura di Kanban
  4.9.1. Storie degli utenti
  4.9.2. *Backlog*
  4.9.3. Analisi Kanban
4.10. Progetti ibridi
  4.10.1. Costruzione del progetto
  4.10.2. Gestione dei progetti
  4.10.3. Aspetti da considerare

Modulo 5. TDD (*Test Driven Development*). Progettazione del Software guidata dai test

5.1. TDD. *Test Driven Development*
  5.1.1. TDD. *Test Driven Development*
  5.1.2. TDD. Influenza del TDD sulla qualità
  5.1.3. Progettazione e sviluppo basati sui test. Esempi
5.2. Ciclo TDD
  5.2.1. Scelta di un requisito
  5.2.2. Esecuzione di test. Tipologie
    5.2.2.1. Test unitari
    5.2.2.2. Test di integrazione
    5.2.2.3. Test *end-to-end*
  5.2.3. Verifica del test. Errori
  5.2.4. Creazione dell'Implementazione
  5.2.5. Esecuzione di test automatizzati
  5.2.6. Eliminazione dei doppioni
  5.2.7. Aggiornamento dell'elenco dei requisiti
  5.2.8. Ripetizione del ciclo TDD
  5.2.9. Ciclo TDD. Esempio teorico e pratico
5.3. Strategie di implementazione del TDD
  5.3.1. Implementazione di prova
  5.3.2. Implementazione triangolare
  5.3.3. Implementazione ovvia
5.4. TDD. Uso. Vantaggi e svantaggi
  5.4.1. Vantaggi di uso
  5.4.2. Limitazioni d'uso
  5.4.3. Equilibrio qualitativo nell'implementazione
5.5. TDD. Buone pratiche
  5.5.1. Regole TDD
  5.5.2. Regola 1: prima di codificare in produzione, eseguire un test precedente che fallisce
  5.5.3. Regola 2: non scrivere più di un test unitario
  5.5.4. Regola 3: non scrivere più codice del necessario
  5.5.5. Errori e anti-pattern da evitare in TDD

5.6. Simulazione di un progetto reale per l'utilizzo di TDD (I)
- 5.6.1. Panoramica del progetto (Azienda A)
- 5.6.2. Implementazione della TDD
- 5.6.3. Esercizi proposti
- 5.6.4. Esercizi. *Feedback*

5.7. Simulazione di un progetto reale per l'utilizzo di TDD (II)
- 5.7.1. Descrizione generale del progetto (Azienda B)
- 5.7.2. Implementazione della TDD
- 5.7.3. Esercizi proposti
- 5.7.4. Esercizi. *Feedback*

5.8. Simulazione di un progetto reale per l'utilizzo di TDD (III)
- 5.8.1. Descrizione generale del progetto (Azienda C)
- 5.8.2. Implementazione della TDD
- 5.8.3. Esercizi proposti
- 5.8.4. Esercizi. *Feedback*

5.9. Alternative al TTD *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. Confronto teorico

5.10. TDD TCR, BDD e ATDD. Confronto pratico
- 5.10.1. Definizione del problema
- 5.10.2. Risoluzione con il TCR
- 5.10.3. Risoluzione con il BDD
- 5.10.4. Risoluzione con il ATDD

Modulo 6. DevOps. Gestione della Qualità del Software

6.1. DevOps. Gestione della qualità del software
- 6.1.1. DevOps
- 6.1.2. DevOps e qualità del software
- 6.1.3. DevOps. Benefici della cultura DevOps

6.2. DevOps. Rapporto con Agile
- 6.2.1. Consegna accelerata
- 6.2.2. Qualità
- 6.2.3. Riduzione dei costi

6.3. Implementazione di DevOps
- 6.3.1. Identificazione di problemi
- 6.3.2. Implementazione in un'azienda
- 6.3.3. Metriche di implementazione

6.4. Ciclo di consegna del software
- 6.4.1. Metodi di progettazione
- 6.4.2. Convenzioni
- 6.4.3. Tabella di marcia

6.5. Sviluppo di codice privo di errori
- 6.5.1. Codice mantenibile
- 6.5.2. Modelli di sviluppo
- 6.5.3. *Testing* del codice
- 6.5.4. Sviluppo di software a livello di codice. Buone pratiche

6.6. Automatizzazione
- 6.6.1. Automatizzazione. Tipi di test
- 6.6.2. Costo dell'automazione e della manutenzione
- 6.6.3. Automatizzazione. Attenuare gli errori

6.7. Distribuzione
- 6.7.1. Valutazione dell'obiettivo
- 6.7.2. Progettazione di un processo automatico e adattato
- 6.7.3. Feedback e capacità di risposta

6.8. Gestione degli incidenti
- 6.8.1. Preparazione agli incidenti
- 6.8.2. Analisi e risoluzione degli incidenti
- 6.8.3. Come evitare errori futuri

6.9. Automazione della distribuzione
- 6.9.1. Preparazione per le distribuzioni automatiche
- 6.9.2. Valutazione dello stato di salute del processo automatico
- 6.9.3. Metriche e capacità di rollback

6.10. Buone pratiche. Evoluzione di DevOps
- 6.10.1. Guida alle migliori pratiche applicando DevOps
- 6.10.2. DevOps. Metodologia per il team
- 6.10.3. Evitare le nicchie

Modulo 7. DevOps e Integrazione Continua. Soluzioni Pratiche Avanzate nello Sviluppo di Software

7.1. Flusso della consegna del software
- 7.1.1. Identificazione di attori e artefatti
- 7.1.2. Progettazione del flusso di consegna del software
- 7.1.3. Flusso di consegna del software. Requisiti tra varie fasi

7.2. Automazione dei processi
- 7.2.1. Integrazione continua
- 7.2.2. Distribuzione continua
- 7.2.3. Configurazione degli ambienti e gestione dei segreti

7.3. Pipeline dichiarative
- 7.3.1. Differenze tra pipeline tradizionali, simili al codice e dichiarative
- 7.3.2. Pipeline dichiarative
- 7.3.3. Pipeline dichiarative in Jenkins
- 7.3.4. Confronto tra i fornitori di integrazione continua

7.4. Gateway di qualità e feedback arricchito
- 7.4.1. Gateway di qualità
- 7.4.2. Standard di qualità con gateway di qualità. Mantenimento
- 7.4.3. Requisiti aziendali sulle richieste di integrazione

7.5. Gestione degli artefatti
- 7.5.1. Artefatti e ciclo di vita
- 7.5.2. Sistemi di conservazione e gestione degli artefatti
- 7.5.3. Sicurezza nella gestione degli artefatti

7.6. Distribuzione continua
- 7.6.1. Distribuzione continua come contenitore
- 7.6.2. Distribuzione continua con PaaS
- 7.6.3. Distribuzione continua di applicazioni mobili

7.7. Migliorare il runtime della pipeline: analisi statica e *Git Hooks*
- 7.7.1. Analisi statica
- 7.7.2. Regole di stile del codice
- 7.7.3. *Git Hooks* e *test* unitari
- 7.7.4. L'impatto dell'infrastruttura

7.8. Vulnerabilità dei contenitori
- 7.8.1. Vulnerabilità dei contenitori
- 7.8.2. Scansione di immagini
- 7.8.3. Rapporti e avvisi periodici

Modulo 8. Progettazione di Database (DB). Normalizzazione e Rendimento. Qualità del software

8.1. Progettazione di database
- 8.1.1. Database: Tipologia
- 8.1.2. Database attualmente utilizzati
  - 8.1.2.1. Relazionali
  - 8.1.2.2. Chiave-Valore
  - 8.1.2.3. Basati sulla rete
- 8.1.3. Qualità del dato

8.2. Progettazione del modello entità-relazione (I)
- 8.2.1. Modello entità-relazione. Qualità e documentazione
- 8.2.2. Entità
  - 8.2.2.1. Entità forte
  - 8.2.2.2. Entità debole
- 8.2.3. Attributi
- 8.2.4. Insieme di relazioni
  - 8.2.4.1. 1 a 1
  - 8.2.4.2. 1 a molti
  - 8.2.4.3. Molti a 1
  - 8.2.4.4. Molti a molti
- 8.2.5. Chiavi
  - 8.2.5.1. Chiave primaria
  - 8.2.5.2. Chiave esterna
  - 8.2.5.3. Chiave primaria dell'entità debole
- 8.2.6. Restrizioni
- 8.2.7. Cardinalità
- 8.2.8. Ereditarietà
- 8.2.9. Aggregazione

8.3. Modello entità-relazione (II). Strumenti
- 8.3.1. Modello entità-relazione. Strumenti
- 8.3.2. Modello entità-relazione. Esempio pratico
- 8.3.3. Modello entità-relazione fattibile
  - 8.3.3.1. Campione visivo
  - 8.3.3.2. Campione in rappresentazione tabellare

8.4. Standardizzazione dei database (DB) (I). Considerazioni sulla qualità del software
- 8.4.1. Standardizzazione e qualità del DB
- 8.4.2. Dipendenze
  - 8.4.2.1. Dipendenza funzionale
  - 8.4.2.2. Proprietà della dipendenza funzionale
  - 8.4.2.3. Proprietà desunte
- 8.4.3. Chiavi

8.5. Standardizzazione dei database (DB) (II). Forme normali e regole di Codd
- 8.5.1. Forme normali
  - 8.5.1.1. Prima forma normale (1FN)
  - 8.5.1.2. Seconda forma normale (2FN)
  - 8.5.1.3. Terza forma normale (3FN)
  - 8.5.1.4. Forma normale di Boyce-Codd (BCNF)
  - 8.5.1.5. Quarta forma normale (4FN)
  - 8.5.1.6. Quinta forma normale (5FN)

8.5.2. Le regole di Codd
8.5.2.1. Regola 1: Informazione
8.5.2.2. Regola 2: accesso garantito
8.5.2.3. Regola 3: Trattamento sistematico dei valori nulli
8.5.2.4. Regola 4: descrizione del database
8.5.2.5. Regola 5: Sottolinguaggio integrale
8.5.2.6. Regola 6: aggiornamento della vista
8.5.2.7. Regola 7: inserimento e aggiornamento
8.5.2.8. Regola 8: indipendenza fisica
8.5.2.9. Regola 9: indipendenza logica
8.5.2.10. Regola 10: Indipendenza dall'integrità
8.5.2.10.1. Regole di integrità
8.5.2.11. Regola 11: Distribuzione
8.5.2.12. Regola 12: Non sovversione
8.5.3. Esempio pratico
8.6. Memorizzazione di dati/sistema OLAP
8.6.1. Memorizzazione di dati
8.6.2. Tabella dei fatti
8.6.3. Tabella delle dimensioni
8.6.4. Creazione del sistema OLAP. Strumenti
8.7. Prestazioni del database (DB)
8.7.1. Ottimizzazione dell'indice
8.7.2. Ottimizzazione delle query
8.7.3. Partizionamento delle tabelle
8.8. Simulazione di un progetto reale per il disegno di DB (I)
8.8.1. Panoramica del progetto (Azienda A)
8.8.2. Applicazioni della progettazione di database
8.8.3. Esercizi proposti
8.8.4. Esercizi proposti. *Feedback*
8.9. Simulazione di un progetto reale per il disegno di DB (II)
8.9.1. Descrizione generale del progetto (Azienda B)
8.9.2. Applicazioni della progettazione di database
8.9.3. Esercizi proposti
8.9.4. Esercizi proposti. *Feedback*
8.10. Importanza dell'ottimizzazione dei DB nella qualità del software
8.10.1. Ottimizzazione del design
8.10.2. Ottimizzazione del codice delle query
8.10.3. Ottimizzazione del codice delle procedure memorizzate
8.10.4. Influenza dei *trigger* sulla qualità del software Raccomandazioni per l'uso

Modulo 9. Progettazione di Architetture Scalabili. L'Architettura nel Ciclo di Vita del Software

9.1. Progettazione di Architetture Scalabili (I)
9.1.1. Architetture scalabili
9.1.2. Principi di un'architettura scalabile
9.1.2.1. Affidabile
9.1.2.2. Scalabile
9.1.2.3. Manutenibile
9.1.3. Tipi di scalabilità
9.1.3.1. Verticale
9.1.3.2. Orizzontale
9.1.3.3. Combinato
9.2. Architetture DDD (*Domain-Driven Design*)
9.2.1. Il modello DDD. Orientamento al dominio
9.2.2. Livelli, distribuzione delle responsabilità e modelli di progettazione
9.2.3. Il disaccoppiamento come base per la qualità
9.3. Progettazione di architetture scalabili (II). Vantaggi, limiti e strategie di progettazione
9.3.1. Architettura scalabile. Benefici
9.3.2. Architettura scalabile. Limitazioni
9.3.3. Strategie per lo sviluppo di architetture scalabili (Tabella descrittiva)

9.4. Ciclo di vita del software (I). Tappe
- 9.4.1. Ciclo di vita del software
  - 9.4.1.1. Fasi di pianificazione
  - 9.4.1.2. Fase di analisi
  - 9.4.1.3. Fase di progettazione
  - 9.4.1.4. Fase di implementazione
  - 9.4.1.5. Fase di test
  - 9.4.1.6. Fase di installazione/dispiegamento
  - 9.4.1.7. Fase di utilizzo e manutenzione

9.5. Modelli di ciclo di vita del software
- 9.5.1. Modello a cascata
- 9.5.2. Modello ripetitivo
- 9.5.3. Modello a spirale
- 9.5.4. Modello *Big Bang*

9.6. Ciclo di vita del software (II). Automatizzazione
- 9.6.1. Cicli di vita dello sviluppo del Software. Soluzioni
  - 9.6.1.1. Integrazione continua e sviluppo continuo (CI/CD)
  - 9.6.1.2. Metodologia Agile
  - 9.6.1.3. DevOps/operazioni di produzione
- 9.6.2. Tendenze future
- 9.6.3. Esempi pratici

9.7. Architettura del software nel ciclo di vita del software
- 9.7.1. Benefici
- 9.7.2. Limitazioni
- 9.7.3. Strumenti

9.8. Simulazione di un progetto reale per il disegno dell'architettura del software (I)
- 9.8.1. Panoramica del progetto (Azienda A)
- 9.8.2. Applicazioni della progettazione dell'architettura del software
- 9.8.3. Esercizi proposti
- 9.8.4. Esercizi proposti. *Feedback*

9.9. Simulazione di un progetto reale per il disegno dell'architettura del software (II)
- 9.9.1. Descrizione generale del progetto (Azienda B)
- 9.9.2. Applicazioni della progettazione dell'architettura del software
- 9.9.3. Esercizi proposti
- 9.9.4. Esercizi proposti. *Feedback*

9.10. Simulazione di un progetto reale per il disegno dell'architettura del software (III)
- 9.10.1. Descrizione generale del progetto (Azienda C)
- 9.10.2. Applicazioni della progettazione dell'architettura del software
- 9.10.3. Esercizi proposti
- 9.10.4. Esercizi proposti. *Feedback*

Modulo 10. Criteri di qualità ISO, IEC 9126. Metriche della Qualità del Software

10.1. Criteri di qualità. Normativa ISO, IEC 9126
- 10.1.1. Criteri di qualità
- 10.1.2. Qualità del software. Giustificazione. Normativa ISO, IEC 9126
- 10.1.3. Misurare la qualità del software come indicatore chiave

10.2. Criteri di qualità del software Caratteristiche
- 10.2.1. Affidabilità
- 10.2.2. Funzionalità
- 10.2.3. Efficienza
- 10.2.4. Usabilità
- 10.2.5. Mantenimento
- 10.2.6. Portabilità
- 10.2.7. Sicurezza

10.3. Normativa ISO, IEC 9126 (I). Presentazione
- 10.3.1. Descrizione della Normativa ISO, IEC 9126
- 10.3.2. Funzionalità
- 10.3.3. Affidabilità
- 10.3.4. Usabilità
- 10.3.5. Mantenimento

10.3.6. Portabilità
10.3.7. Qualità in uso
10.3.8. Metriche della qualità del software
10.3.9. Metriche di qualità in ISO 9126
10.4. Normativa ISO, IEC 9126 (II). Modelli McCall e Boehm
10.4.1. Modello McCall: fattori di qualità
10.4.2. Modello Boehm
10.4.3. Livello intermedio. Caratteristiche
10.5. Metriche di qualità del software (I). Elementi
10.5.1. Misura
10.5.2. Metriche
10.5.3. Indicatore
10.5.3.1. Tipi di indicatori
10.5.4. Misure e modelli
10.5.5. Ambito di applicazione delle metriche del software
10.5.6. Classificazione delle metriche del software
10.6. Misurazione della qualità del software (II). Pratica di misurazione
10.6.1. Raccolta dati metrici
10.6.2. Misurazione degli attributi interni del prodotto
10.6.3. Misurazione degli attributi esterni del prodotto
10.6.4. Misurazione delle risorse
10.6.5. Metriche per sistemi orientati agli oggetti
10.7. Progettazione di un unico indicatore di qualità del software
10.7.1. Singolo indicatore come qualificatore globale
10.7.2. Sviluppo, giustificazione e applicazione degli indicatori
10.7.3. Esempi di applicazione. Necessità di conoscere i dettagli
10.8. Simulazione di un progetto reale per la misurazione della qualità (I)
10.8.1. Panoramica del progetto (Azienda A)
10.8.2. Aplicazione della misurazione della qualità
10.8.3. Esercizi proposti
10.8.4. Esercizi proposti. *Feedback*
10.9. Simulazione di un progetto reale per la misurazione della qualità (II)
10.9.1. Descrizione generale del progetto (Azienda B)
10.9.2. Aplicazione della misurazione della qualità
10.9.3. Esercizi proposti
10.9.4. Esercizi proposti. *Feedback*
10.10. Simulazione di un progetto reale per la misurazione della qualità (III)
10.10.1. Descrizione generale del progetto (Azienda C)
10.10.2. Aplicazione della misurazione della qualità
10.10.3. Esercizi proposti
10.10.4. Esercizi proposti. *Feedback*

Avrai accesso a contenuti unici e specializzati. Selezionati da docenti esperti per un titolo di studio che farà emergere il tuo profilo professionale"

06

Metodologia

Questo programma ti offre un modo differente di imparare. La nostra metodologia si sviluppa in una modalità di apprendimento ciclico: ***il Relearning***.

Questo sistema di insegnamento viene applicato nelle più prestigiose facoltà di medicina del mondo ed è considerato uno dei più efficaci da importanti pubblicazioni come il ***New England Journal of Medicine.***

Scopri il Relearning, un sistema che abbandona l'apprendimento lineare convenzionale, per guidarti attraverso dei sistemi di insegnamento ciclici: una modalità di apprendimento che ha dimostrato la sua enorme efficacia, soprattutto nelle materie che richiedono la memorizzazione"

Caso di Studio per contestualizzare tutti i contenuti

Il nostro programma offre un metodo rivoluzionario per sviluppare le abilità e le conoscenze. Il nostro obiettivo è quello di rafforzare le competenze in un contesto mutevole, competitivo e altamente esigente.

Con TECH potrai sperimentare un modo di imparare che sta scuotendo le fondamenta delle università tradizionali in tutto il mondo"

Avrai accesso a un sistema di apprendimento basato sulla ripetizione, con un insegnamento naturale e progressivo durante tutto il programma.

Imparerai, attraverso attività collaborative e casi reali, la risoluzione di situazioni complesse in ambienti aziendali reali.

Un metodo di apprendimento innovativo e differente

Questo programma di TECH consiste in un insegnamento intensivo, creato ex novo, che propone le sfide e le decisioni più impegnative in questo campo, sia a livello nazionale che internazionale. Grazie a questa metodologia, la crescita personale e professionale viene potenziata, effettuando un passo decisivo verso il successo. Il metodo casistico, la tecnica che sta alla base di questi contenuti, garantisce il rispetto della realtà economica, sociale e professionale più attuali.

> *Il nostro programma ti prepara ad affrontare nuove sfide in ambienti incerti e a raggiungere il successo nella tua carriera"*

Il Metodo Casistico è stato il sistema di apprendimento più usato nelle migliori Scuole di Informatica del mondo da quando esistono. Sviluppato nel 1912 affinché gli studenti di Diritto non imparassero la legge solo sulla base del contenuto teorico, il metodo casistico consisteva nel presentare loro situazioni reali e complesse per prendere decisioni informate e giudizi di valore su come risolverle. Nel 1924 fu stabilito come metodo di insegnamento standard ad Harvard.

Cosa dovrebbe fare un professionista per affrontare una determinata situazione? Questa è la domanda con cui ti confrontiamo nel metodo dei casi, un metodo di apprendimento orientato all'azione. Durante il corso, gli studenti si confronteranno con diversi casi di vita reale. Dovranno integrare tutte le loro conoscenze, effettuare ricerche, argomentare e difendere le proprie idee e decisioni.

Metodologia Relearning

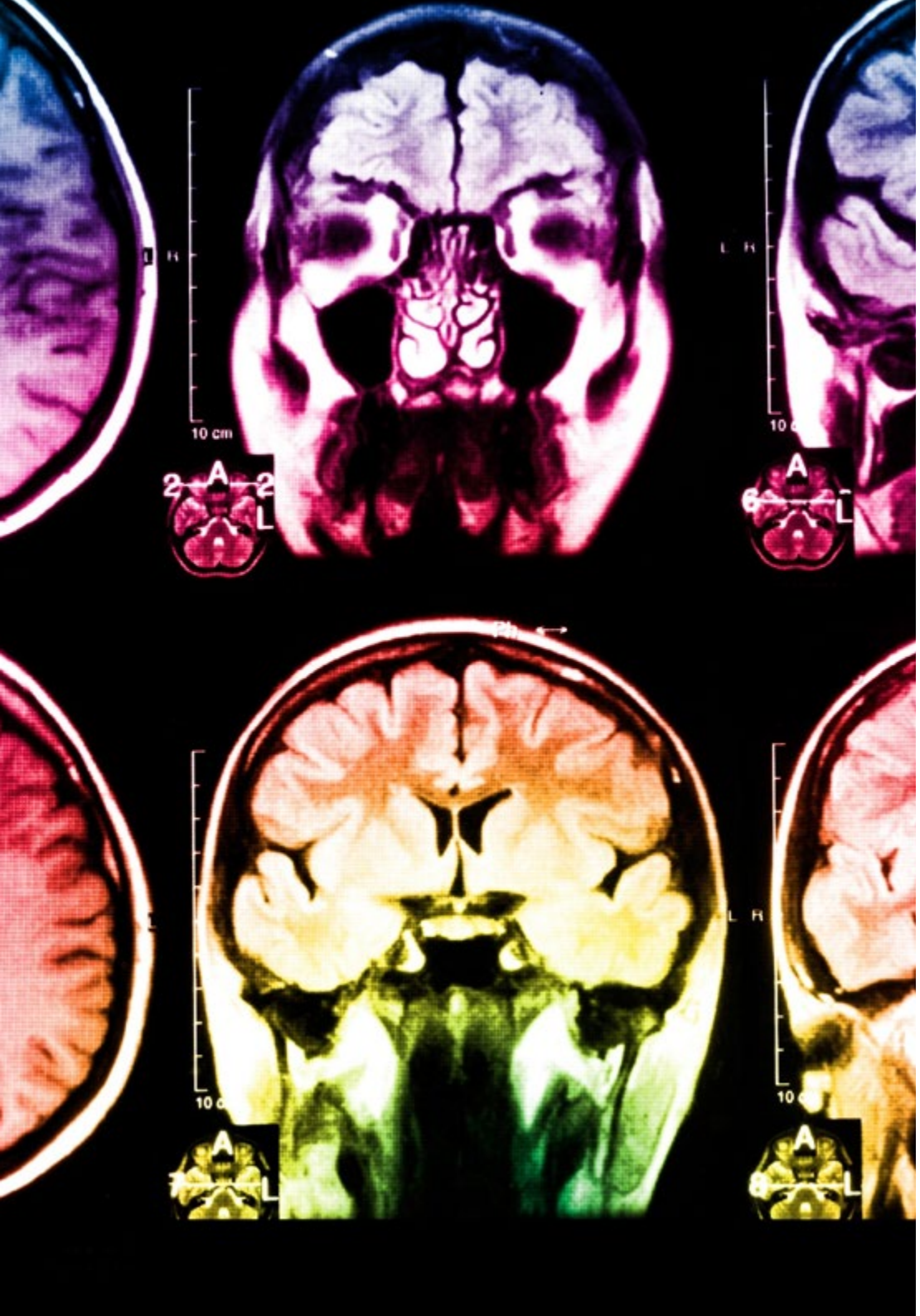
TECH coniuga efficacemente la metodologia del Caso di Studio con un sistema di apprendimento 100% online basato sulla ripetizione, che combina diversi elementi didattici in ogni lezione.

Potenziamo il Caso di Studio con il miglior metodo di insegnamento 100% online: il Relearning.

Nel 2019 abbiamo ottenuto i migliori risultati di apprendimento di tutte le università online del mondo.

In TECH imparerai con una metodologia all'avanguardia progettata per formare i manager del futuro. Questo metodo, all'avanguardia della pedagogia mondiale, si chiama Relearning.

La nostra università è l'unica autorizzata a utilizzare questo metodo di successo. Nel 2019, siamo riusciti a migliorare il livello di soddisfazione generale dei nostri studenti (qualità dell'insegnamento, qualità dei materiali, struttura del corso, obiettivi...) rispetto agli indicatori della migliore università online.



Nel nostro programma, l'apprendimento non è un processo lineare, ma avviene in una spirale (impariamo, disimpariamo, dimentichiamo e re-impariamo). Pertanto, combiniamo ciascuno di questi elementi in modo concentrico. Questa metodologia ha formato più di 650.000 laureati con un successo senza precedenti in campi diversi come la biochimica, la genetica, la chirurgia, il diritto internazionale, le competenze manageriali, le scienze sportive, la filosofia, il diritto, l'ingegneria, il giornalismo, la storia, i mercati e gli strumenti finanziari. Tutto questo in un ambiente molto esigente, con un corpo di studenti universitari con un alto profilo socio-economico e un'età media di 43,5 anni.

Il Relearning ti permetterà di apprendere con meno sforzo e più performance, impegnandoti maggiormente nella tua specializzazione, sviluppando uno spirito critico, difendendo gli argomenti e contrastando le opinioni: un'equazione diretta al successo.

Dalle ultime evidenze scientifiche nel campo delle neuroscienze, non solo sappiamo come organizzare le informazioni, le idee, le immagini e i ricordi, ma sappiamo che il luogo e il contesto in cui abbiamo imparato qualcosa è fondamentale per la nostra capacità di ricordarlo e immagazzinarlo nell'ippocampo, per conservarlo nella nostra memoria a lungo termine.

In questo modo, e in quello che si chiama Neurocognitive Context-dependent E-learning, i diversi elementi del nostro programma sono collegati al contesto in cui il partecipante sviluppa la sua pratica professionale.

Questo programma offre i migliori materiali didattici, preparati appositamente per i professionisti:

Materiale di studio

Tutti i contenuti didattici sono creati appositamente per il corso dagli specialisti che lo impartiranno, per fare in modo che lo sviluppo didattico sia davvero specifico e concreto.

Questi contenuti sono poi applicati al formato audiovisivo che supporterà la modalità di lavoro online di TECH. Tutto questo, con le ultime tecniche che offrono componenti di alta qualità in ognuno dei materiali che vengono messi a disposizione dello studente.

Master class

Esistono evidenze scientifiche sull'utilità dell'osservazione di esperti terzi.

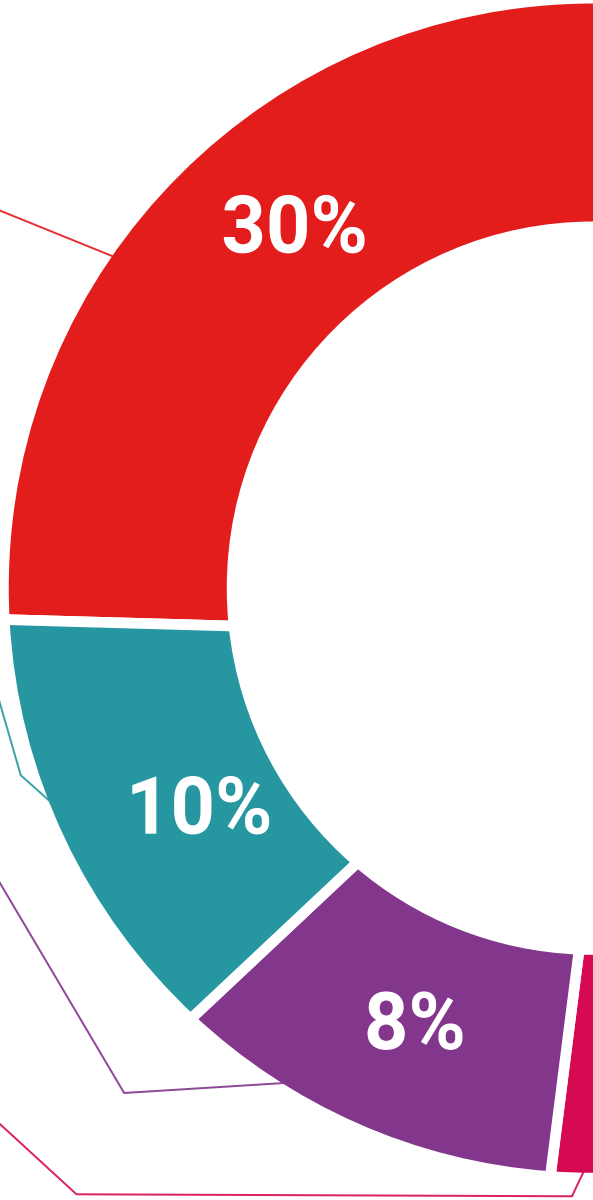
Imparare da un esperto rafforza la conoscenza e la memoria, costruisce la fiducia nelle nostre future decisioni difficili.

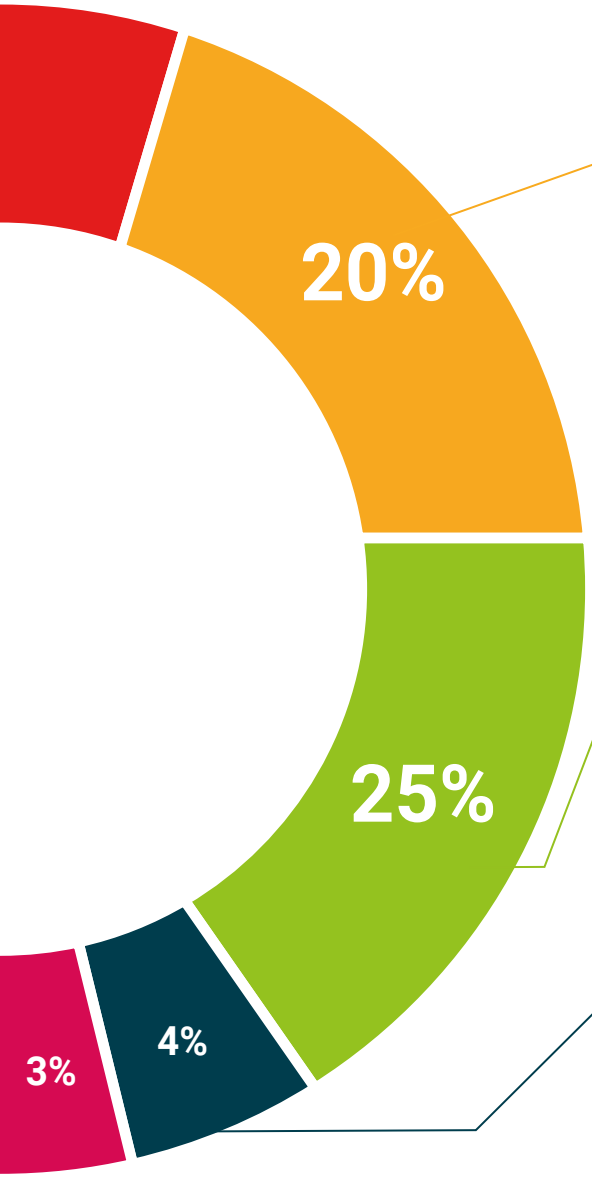
Pratiche di competenze e competenze

Svolgerai attività per sviluppare competenze e capacità specifiche in ogni area tematica. Pratiche e dinamiche per acquisire e sviluppare le competenze e le abilità che uno specialista deve sviluppare nel quadro della globalizzazione in cui viviamo.

Letture complementari

Articoli recenti, documenti di consenso e linee guida internazionali, tra gli altri. Nella biblioteca virtuale di TECH potrai accedere a tutto il materiale necessario per completare la tua specializzazione.





Casi di Studio

Completerai una selezione dei migliori casi di studio scelti appositamente per questo corso. Casi presentati, analizzati e monitorati dai migliori specialisti del panorama internazionale.

Riepiloghi interattivi

Il team di TECH presenta i contenuti in modo accattivante e dinamico in pillole multimediali che includono audio, video, immagini, diagrammi e mappe concettuali per consolidare la conoscenza.

Questo esclusivo sistema di specializzazione per la presentazione di contenuti multimediali è stato premiato da Microsoft come "Caso di successo in Europa".

Testing & Retesting

Valutiamo e rivalutiamo periodicamente le tue conoscenze durante tutto il programma con attività ed esercizi di valutazione e autovalutazione, affinché tu possa verificare come raggiungi progressivamente i tuoi obiettivi.

07

Titolo

Il Master Privato in Qualità del Software ti garantisce, oltre alla preparazione più rigorosa e aggiornata, l'accesso a una qualifica di Master Privato rilasciata da TECH Università Tecnologica.

Porta a termine questo programma e ricevi la tua qualifica universitaria senza spostamenti o fastidiose formalità"

Questo **Master Privato in Qualità del Software** possiede il programma più completo e aggiornato del mercato.

Dopo aver superato la valutazione, lo studente riceverà mediante lettera certificata* con ricevuta di ritorno, la sua corrispondente qualifica di **Master Privato** rilasciata da **TECH Università Tecnologica.**

Il titolo rilasciato da **TECH Università Tecnologica** esprime la qualifica ottenuta nel Master Privato, e riunisce tutti i requisiti comunemente richiesti da borse di lavoro, concorsi e commissioni di valutazione di carriere professionali.

Titolo: **Master Privato in Qualità del Software**

N. Ore Ufficiali: **1.500**


tech università tecnologica

Conferisce il presente

DIPLOMA

a

Dott./Dott.ssa _____, con documento d'identità n°_____

Per aver completato con esito positivo e accreditato il programma di

MASTER PRIVATO

in

Qualità del Software

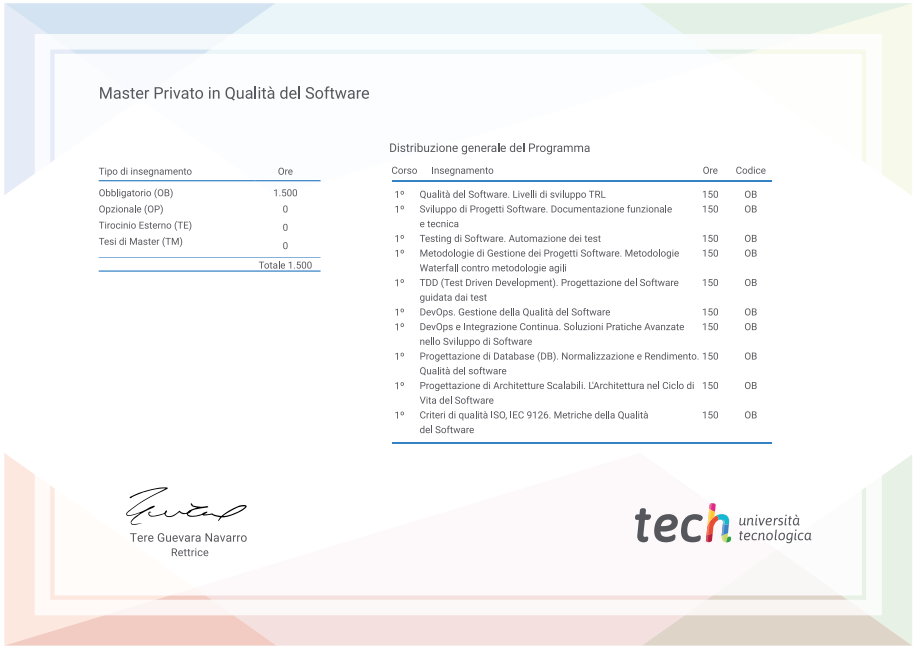
Si tratta di un titolo rilasciato da questa Università ed equivalente a 1.500 ore, con data di inizio gg/mm/aaaa e con data di fine gg/mm/aaaa.

TECH è un Istituto Privato di Istruzione Superiore riconosciuto dal Ministero della Pubblica Istruzione a partire dal 28 giugno 2018.

In data 17 Giugno 2020

Tere Guevara Navarro
Rettrice

codice unico TECH: AFWORD23S techtitute.com/titulos


Master Privato in Qualità del Software

Tipo di insegnamento	Ore
Obbligatorio (OB)	1.500
Opzionale (OP)	0
Tirocinio Esterno (TE)	0
Tesi di Master (TM)	0
	Totale 1.500

Distribuzione generale del Programma

Corso	Insegnamento	Ore	Codice
1°	Qualità del Software. Livelli di sviluppo TRL	150	OB
1°	Sviluppo di Progetti Software. Documentazione funzionale e tecnica	150	OB
1°	Testing di Software. Automazione dei test	150	OB
1°	Metodologie di Gestione dei Progetti Software. Metodologie Waterfall contro metodologie agili	150	OB
1°	TDD (Test Driven Development). Progettazione del Software guidata dai test	150	OB
1°	DevOps. Gestione della Qualità del Software	150	OB
1°	DevOps e Integrazione Continua. Soluzioni Pratiche Avanzate nello Sviluppo di Software	150	OB
1°	Progettazione di Database (DB). Normalizzazione e Rendimento. Qualità del software	150	OB
1°	Progettazione di Architetture Scalabili. L'Architettura nel Ciclo di Vita del Software	150	OB
1°	Criteri di qualità ISO, IEC 9126. Metriche della Qualità del Software	150	OB

Tere Guevara Navarro
Rettrice

tech università tecnologica

*Se lo studente dovesse richiedere che il suo diploma cartaceo sia provvisto di Apostille dell'Aia, TECH EDUCATION effettuerà le gestioni opportune per ottenerla pagando un costo aggiuntivo.

tech università tecnologica

Master Privato
Qualità del Software

- Modalità: **online**
- Durata: **12 mesi**
- Titolo: **TECH Università Tecnologica**
- Dedizione: **16 ore/settimana**
- Orario: **a scelta**
- Esami: **online**

Master Privato
Qualità del Software

tech università tecnologica