Weiterbildender Masterstudiengang

Softwaretechnik und -qualität

tech technologische universität

tech technologische universität

Weiterbildender Masterstudiengang
Softwaretechnik und -qualität

- Modalität: **online**
- Dauer: **2 Jahre**
- Qualifizierung: **TECH Technologische Universität**
- Zeitplan: **in Ihrem eigenen Tempo**
- Prüfungen: **online**

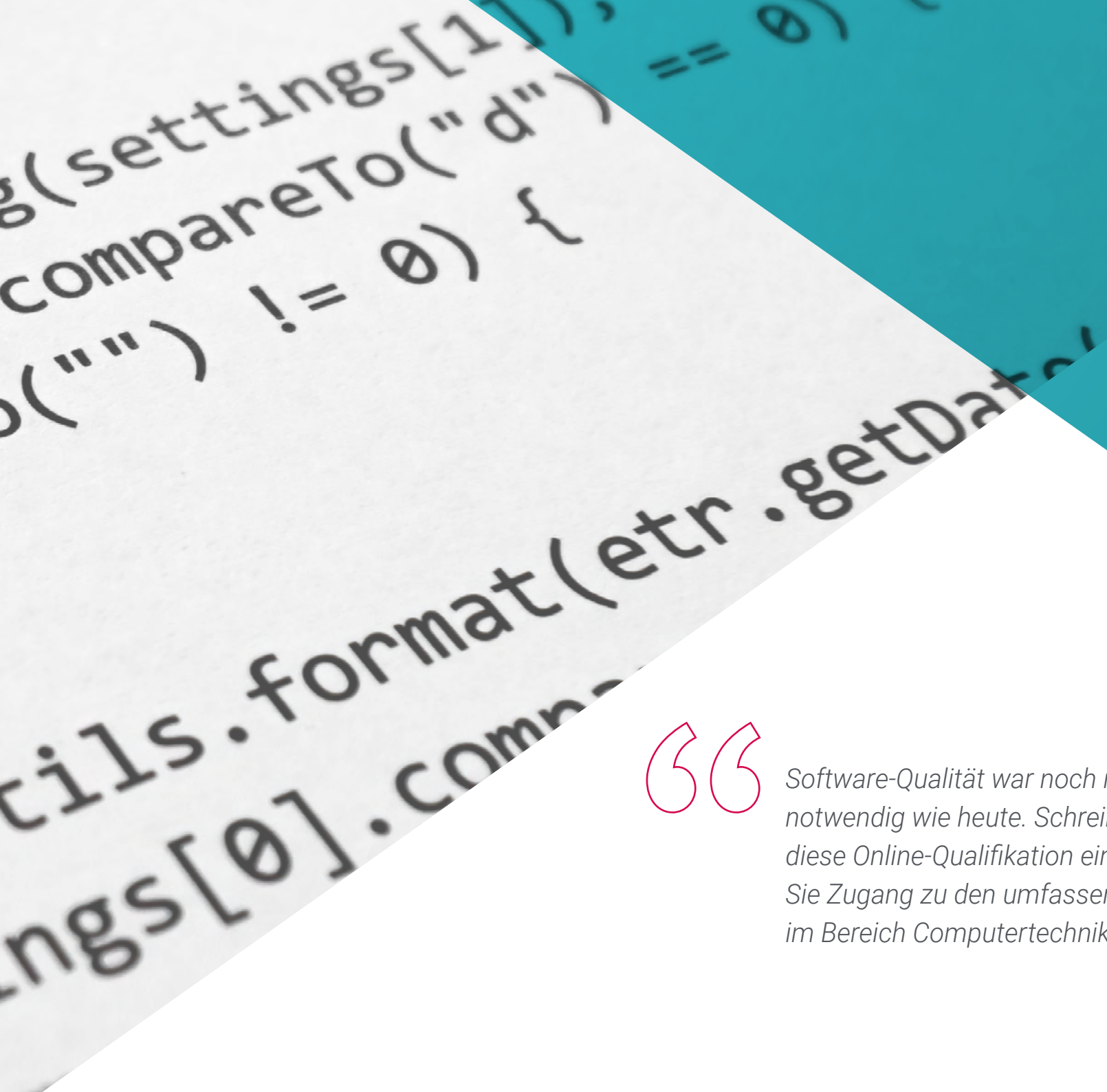
Internetzugang: www.techtitute.com/de/informatik/weiterbildender-masterstudiengang/weiterbildender-masterstudiengang-softwaretechnik-qualitat

Index

01

Präsentation

Die Entwicklung der Technologie und die Fortschritte bei den Computersystemen haben in der Branche eine große Nachfrage nach Fachleuten geschaffen, die das Software Engineering perfekt beherrschen, angefangen bei den ausgefeiltesten und genauesten Tools für die Entwicklung und Implementierung bis hin zu den Sicherheitsprotokollen, die einen unantastbaren Zugriff auf ihre Daten garantieren. Aus diesem Grund und mit dem Ziel, Fachleuten die Möglichkeit zu bieten, sich mit den aktuellsten Informationen über die in diesem Bereich angewandten Techniken auf den neuesten Stand zu bringen, hat TECH diese multidisziplinäre und 100%ige Online-Weiterbildung entwickelt. Es handelt sich um ein von Experten entwickeltes Programm, das in einer einzigen Fortbildung 3.000 Stunden der besten Inhalte über Computersysteme und Softwarequalität vereint und dem Studenten helfen wird, seine IT-Kenntnisse sofort und gezielt zu perfektionieren.



Software-Qualität war noch nie so notwendig wie heute. Schreiben Sie sich in diese Online-Qualifikation ein und erhalten Sie Zugang zu den umfassendsten Inhalten im Bereich Computertechnik"

Die Computertechnik hat in den letzten Jahren aufgrund der Entwicklung der Technologie und der digitalen Tools exponentiell zugenommen, insbesondere in Bezug auf alles, was mit dem Internet und seiner Benutzerfreundlichkeit zu tun hat. Deshalb ist die Entwicklung von Software für verschiedene Funktionen an der Tagesordnung, und der Katalog der Programme wird ständig erweitert. Diese Quantität ist jedoch nicht immer gleichbedeutend mit Qualität, und oft finden sich Anwendungen, die ihre Aufgabe nicht erfüllen, Fehler melden oder die Sicherheit von Unternehmen ernsthaft gefährden. Aus diesem Grund steigt die Nachfrage nach Computeringenieuren, die sich auf diesen Bereich spezialisiert haben.

Aus diesem Grund hat TECH beschlossen, diesen Weiterbildenden Masterstudiengang in Softwaretechnik und -qualität zu gestalten. Es handelt sich dabei um ein multidisziplinäres Programm, das von Experten auf diesem Gebiet entworfen wurde und so konzipiert ist, dass die Studenten darin alle notwendigen Werkzeuge finden, um ihr Wissen auf umfassende Weise und auf der Grundlage der neuesten Entwicklungen in diesem Sektor zu aktualisieren. Es handelt sich um eine Fortbildung, die Theorie und Praxis in 20 Modulen verbindet, in denen das Software-Engineering und die Qualität von IT-Systemprojekten vertieft werden.

Während der 24 Monate, in denen dieses 100%ige Online-Programm angeboten wird, hat der Ingenieur Zugang zum besten Studienplan, der es ihm ermöglicht, seine Fähigkeiten in der Normalisierung von Datenbanken und in der Entkopplung zwischen den Komponenten eines Systems zu verbessern sowie sein Wissen über skalierbare Architekturen, Qualitätsmetriken und kollaborative Arbeit zu erweitern.

Darüber hinaus hat er Zugriff auf ein modernes, hochmodernes virtuelles Klassenzimmer, in dem er alle Hilfsmittel findet, die es ihm ermöglichen, das Beste aus dieser Qualifikation herauszuholen, darunter Hunderte von Stunden an zusätzlichem Material in verschiedenen Formaten. Alle diese Inhalte können auf jedes Gerät mit einer Internetverbindung heruntergeladen werden, so dass sie jederzeit verfügbar sind, wenn sie gebraucht werden.

Dieser **Weiterbildender Masterstudiengang in Softwaretechnik und -qualität** enthält das vollständigste und aktuellste Programm auf dem Markt. Die hervorstechendsten Merkmale sind:

- Die Entwicklung von Fallstudien, die von Experten aus dem Bereich Ingenieurwissenschaften vorgestellt werden
- Der anschauliche, schematische und äußerst praxisnahe Inhalt soll wissenschaftliche und praktische Informationen zu den für die berufliche Praxis wesentlichen Disziplinen vermitteln
- Praktische Übungen, anhand derer der Selbstbewertungsprozess zur Verbesserung des Lernens verwendet werden kann
- Ein besonderer Schwerpunkt liegt auf innovativen Methoden bei der Konzeption und Gestaltung von Software
- Theoretische Vorträge, Fragen an den Experten, Diskussionsforen zu kontroversen Themen und individuelle Reflexionsarbeit
- Die Verfügbarkeit des Zugriffs auf die Inhalte von jedem festen oder tragbaren Gerät mit Internetanschluss

Sie werden Zugang zu Übungen im HTML-Format und deren Lösungen haben, so dass Sie Ihr Wissen und die während der Programmierung entwickelte Theorie in die Praxis umsetzen können"

Dank des DevOps-Moduls verfügen Sie über das umfangreichste und umfassendste Wissen, um den Lebenszyklus der Softwaretechnik zu beschleunigen und eine qualitativ hochwertige kontinuierliche Entwicklung zu gewährleisten"

Mit dieser Qualifikation sind Sie in der Lage, Ihr eigenes Softwareentwicklungsprojekt zu erstellen und die anspruchsvollsten und innovativsten Belastungs- und Dauertests anzuwenden, um dessen Qualität zu überprüfen.

Tauchen Sie ein in das Test Driven Development und gewinnen Sie einen umfassenden und spezialisierten Überblick über testgetriebenes Softwaredesign und -entwicklung.

Das Dozententeam besteht aus Fachleuten aus dem Bereich des Ingenieurwesens, die ihre Erfahrungen in dieses Programm einbringen, sowie aus anerkannten Spezialisten von führenden Unternehmen und renommierten Universitäten.

Die multimedialen Inhalte, die mit den neuesten Bildungstechnologien entwickelt wurden, ermöglichen den Fachleuten ein situiertes und kontextbezogenes Lernen, d.h. eine simulierte Umgebung, die ein immersives Studium ermöglicht, das auf die Fortbildung in realen Situationen ausgerichtet ist.

Das Konzept dieses Studiengangs konzentriert sich auf problemorientiertes Lernen, bei dem der Student versuchen muss, die verschiedenen Situationen der beruflichen Praxis zu lösen, die im Laufe des akademischen Kurses auftreten. Dabei wird die Fachkraft durch ein innovatives interaktives Videosystem unterstützt, das von anerkannten Experten entwickelt wurde.

02

Ziele

Die Computertechnik ist ein Sektor, der sich ständig verändert. Aus diesem Grund hat TECH diesen Studiengang entwickelt, nicht nur mit dem Ziel, dem Spezialisten ein breites und aktuelles Wissen über seinen Beruf zu vermitteln, sondern auch, um ihm ein detailliertes Wissen über die Tools an die Hand zu geben, die es ihm ermöglichen werden, auch nach Erwerb des weiterbildenden Masterstudiengangs auf dem neuesten Stand zu bleiben. Darüber hinaus wird das beste theoretische, praktische und audiovisuelle Material zur Verfügung gestellt, um dieses Programm zu einer dynamischen und äußerst förderlichen akademischen Erfahrung zu machen.

Wenn es Ihr Ziel ist, ein Experte in Softwaretechnik und -qualität zu werden, bietet Ihnen dieser weiterbildende Masterstudiengang alles, was Sie brauchen, um Ihre beruflichen Erwartungen mit absoluter Erfolgsgarantie zu übertreffen"

Allgemeine Ziele

- Entwicklung von Kriterien, Aufgaben und fortgeschrittenen Methoden, um die Bedeutung qualitätsorientierter Arbeit zu verstehen
- Analyse der wichtigsten Faktoren für die Qualität eines Softwareprojekts
- Entwicklung der relevanten regulatorischen Aspekte
- Implementierung von DevOps und Systemprozessen zur Qualitätssicherung
- Reduzierung der technischen Schuld von Projekten mit einem Qualitätsansatz anstelle eines Ansatzes, der auf Wirtschaftlichkeit und kurzen Fristen basiert
- Vermittlung von Fachwissen zur Messung und Quantifizierung der Qualität eines Softwareprojekts
- Die wirtschaftlichen Vorschläge von Projekten auf der Grundlage von Qualität verteidigen
- Erwerb neuer Kenntnisse in den Bereichen Software Engineering und Computersysteme
- Erwerb neuer Fähigkeiten in Bezug auf neue Technologien und neueste Entwicklungen im Bereich Software
- Verarbeitung der Daten, die im Rahmen der Tätigkeiten im Bereich Softwaretechnik und Computersysteme erzeugt werden

Spezifische Ziele

Modul 1. Software-Qualität. TRL-Entwicklungsstufen

- Entwicklung der Elemente, die die Softwarequalität ausmachen, in klarer und präziser Form
- Anwendung von Modellen und Standards in Bezug auf System-, Produkt- und Prozesssoftware
- Vertiefung der angewandten ISO-Qualitätsnormen sowohl im Allgemeinen als auch in spezifischen Bereichen
- Anwendung der Regeln je nach Umfeld (lokal, national und international)
- Prüfung der Technologie-Reifegrade und deren Anpassung an die verschiedenen Teile des Softwareprojekts, die zu bearbeiten sind
- Erwerb der Abstraktionsfähigkeit, um ein oder mehrere Kriterien von Elementen und Ebenen der Softwarequalität anzuwenden
- Unterscheidung der Anwendungsfälle der Standards und Reifegrade in einem simulierten Realfallprojekt

Modul 2. Software-Projektentwicklung. Funktionelle und technische Dokumentation

- Bestimmung des Einflusses des Projektmanagements auf die Qualität
- Entwicklung der verschiedenen Phasen eines Projekts
- Unterscheidung der Qualitätskonzepte für funktionale und technische Dokumentation
- Analyse der Phase der Anforderungserfassung, der Analysephase, des Teammanagements und der Konstruktionsphase
- Festlegung der verschiedenen Methoden des Software-Projektmanagements
- Kriterien erstellen, um zu entscheiden, welche Methode je nach Art des Projekts am besten geeignet ist

Modul 3. Software *Testing*. Testautomatisierung

- Die Unterschiede zwischen Produktqualität, Prozessqualität und Nutzungsqualität ermitteln
- Die ISO/IEC 15504-Norm kennen
- Die Details von CMMI ermitteln
- Erlernen der wichtigsten Aspekte der kontinuierlichen Integration, der Repositories und ihrer Auswirkungen auf ein Softwareentwicklungsteam
- Feststellung der Relevanz der Einbeziehung von Repositories bei Softwareprojekten Erfahren, wie man sie mit TFS erstellt
- Analyse der verschiedenen Arten von grundlegenden Tests, wie Last-, Einzel-, *Stress*- und Dauertests
- Erfassung der Bedeutung der Skalierbarkeit von Software bei der Konzeption und Entwicklung von Informationssystemen

Modul 4. Methoden des Software-Projektmanagements. *Waterfall*-Methodologie versus agile Methodologie

- Bestimmung, woraus die *Waterfall*-Methodologie besteht
- Vertiefung in die *SCRUM*-Methodologie
- Ermittlung der Unterschiede zwischen *Waterfall* und *SCRUM*
- Festlegung der Unterschiede zwischen der *Waterfall*- und der *SCRUM*-Methodologie und wie der Kunde sie sieht
- Untersuchung des Panel Kanban
- Bestimmung eines *Waterfall*- und *SCRUM*-Ansatzes für ein und dasselbe Projekt
- Ein Hybridprojekt einrichten

Modul 5. TDD (*Test Driven Developement*). Testgetriebener Softwareentwurf

- Kennenlernen der praktischen Anwendung von TDD und seiner Möglichkeiten für das Testen eines Softwareprojekts in der Zukunft
- Vervollständigung der vorgeschlagenen realen Simulationsfälle, um dieses TDD-Konzept kontinuierlich zu erlernen
- In den Simulationsfällen analysieren, inwieweit die Tests vom konstruktiven Standpunkt aus erfolgreich sein oder fehlschlagen können
- Bestimmung der Alternativen zu TDD und Erstellung einer vergleichenden Analyse zwischen ihnen

Modul 6. DevOps. Software-Qualitätsmanagement

- Analyse der Unzulänglichkeiten eines traditionellen Prozesses
- Bewertung möglicher Lösungen und Auswahl der am besten geeigneten Lösung
- Geschäftsanforderungen und ihre Auswirkungen auf die Implementierung verstehen
- Die Kosten der durchzuführenden Verbesserungen abschätzen
- Entwicklung eines entwicklungsfähigen Software-Lebenszyklus, angepasst an die tatsächlichen Bedürfnisse
- Mögliche Fehler vorhersehen und bereits im Entwurfsprozess vermeiden
- Die Verwendung der verschiedenen Implementierungsmodelle rechtfertigen

Modul 7. DevOps und kontinuierliche Integration. Fortgeschrittene praktische Lösungen in der Softwareentwicklung

- Die Phasen des Softwareentwicklungs- und -auslieferungszyklus identifizieren, die an bestimmte Fälle angepasst sind
- Entwurf eines Softwareentwicklungsprozesses mit kontinuierlicher Integration
- Entwicklung und Implementierung von kontinuierlicher Integration und Bereitstellung auf der Grundlage eines vorherigen Entwurfs
- Automatische Qualitätskontrollpunkte für jede Softwarelieferung einrichten
- Aufrechterhaltung eines automatisierten und robusten Softwareentwicklungsprozesses
- Anpassung zukünftiger Anforderungen an den Prozess der kontinuierlichen Integration und Bereitstellung
- Analyse und Vorhersage von Sicherheitsschwachstellen während und nach der Auslieferung der Software

Modul 8. Datenbank-Design (DB). Standardisierung und Leistung. Software-Qualität

- Bewertung der Verwendung des Entity-Relationship-Modells für den vorläufigen Entwurf einer Datenbank
- Anwendung einer Entity, eines Attributs, eines Keys, usw. für beste Datenintegrität
- Bewertung der Abhängigkeiten, Formen und Regeln der Standardisierung von Datenbanken
- Spezialisierung auf den Betrieb eines OLAP-Data-Warehouse-Systems, Entwicklung und Verwendung von Fakten und Dimensionstabellen
- Bestimmung der wichtigsten Faktoren für die Datenbankleistung
- Durchführung von vorgeschlagenen realen Simulationsfällen zum kontinuierlichen Lernen von Datenbankdesign, Normalisierung und Leistung
- In der Simulation die Optionen festlegen, die bei der Erstellung der Datenbank vom konstruktiven Standpunkt aus zu lösen sind

Modul 9. Entwurf skalierbarer Architekturen. Architektur im Lebenszyklus der Software

- Entwicklung des Konzepts der Softwarearchitektur und ihrer Merkmale
- Bestimmung der verschiedenen Arten von Skalierbarkeit in der Softwarearchitektur
- Analyse der verschiedenen Stufen, die bei der Web-Skalierbarkeit auftreten können
- Erwerb von Fachwissen über das Konzept, die Phasen und Modelle des Software-Lebenszyklus
- Bestimmung der Auswirkungen einer Architektur auf den Software-Lebenszyklus, mit ihren Vorteilen, Einschränkungen und unterstützenden Tools

- Vervollständigung der vorgeschlagenen realen Simulationsfälle, um die Architektur und den Software-Lebenszyklus kontinuierlich zu erlernen
- Bewertung in Simulationsfällen, inwieweit das Design der Architektur durchführbar oder überflüssig sein könnte

Modul 10. ISO/IEC 9126 Qualitätskriterien. Metriken zur Software-Qualität

- Entwicklung des Konzepts der Qualitätskriterien und der relevanten Aspekte
- Prüfung der Norm ISO/IEC 9126, Hauptaspekte und Indikatoren
- Analyse der verschiedenen Metriken für ein Softwareprojekt, um die vereinbarten Bewertungen zu erfüllen
- Untersuchung der internen und externen Attribute, die bei der Qualität eines Softwareprojekts zu berücksichtigen sind
- Unterscheidung der Metriken nach der Art der Programmierung (strukturiert, objektorientiert, schichtweise usw.)
- Abschluss von realen Simulationen, als kontinuierliches Lernen der Qualitätsmessung
- In den Simulationsfällen sehen, inwieweit es machbar oder unnötig ist, d.h. vom konstruktiven Standpunkt der Autoren aus gesehen

Modul 11. Methodik, Entwicklung und Qualität im Software Engineering

- Kenntnis der Grundlagen des Software Engineering sowie des Regelwerks oder der ethischen Grundsätze und der beruflichen Verantwortung während und nach der Entwicklung
- Verstehen des Softwareentwicklungsprozesses unter Berücksichtigung der verschiedenen Programmiermodelle und des objektorientierten Programmierparadigmas
- Die verschiedenen Arten der Anwendungsmodellierung und Entwurfsmuster in der Unified Modelling Language (UML) verstehen
- Aneignung der erforderlichen Kenntnisse für die korrekte Anwendung agiler Methodologien in der Softwareentwicklung, einschließlich SCRUM
- Kenntnis der Lean-Entwicklungsmethodik zur Unterscheidung der Aktivitäten, die keinen Mehrwert im Prozess bringen, um eine höhere Softwarequalität zu erreichen

Modul 12. Software-Projektmanagement

- Kenntnis der grundlegenden Konzepte des Projektmanagements und des Lebenszyklus des Projektmanagements
- Die verschiedenen Phasen des Projektmanagements wie Initiierung, Planung, *Stakeholder*-Management und Scoping zu verstehen
- Die Entwicklung des Zeitplans für Zeitmanagement, Budgetentwicklung und Risikobewältigung
- Verständnis der Funktionsweise des Qualitätsmanagements in Projekten, einschließlich Planung, Sicherung, Kontrolle, statistischer Konzepte und verfügbarer Instrumente
- Verstehen, wie die Prozesse der Projektbeschaffung, -durchführung, -überwachung, -kontrolle und -abschluss funktionieren
- Aneignung der wesentlichen Kenntnisse im Zusammenhang mit der beruflichen Verantwortung, die sich aus dem Projektmanagement ergibt

Modul 13. Plattformen für die Software-Entwicklung

- Die verschiedenen Softwareentwicklungsplattformen verstehen
- Erwerb der notwendigen Kenntnisse für die Entwicklung von Anwendungen und grafischen Oberflächen in den Sprachen Java und .NET
- Erlernen der notwendigen Techniken zur Fehlersuche und zum Testen der durchgeführten Entwicklungen
- Erlernen der Entwicklungsumgebungen für mobile Anwendungen in Android und der Prozesse des Debuggens und der Veröffentlichung
- Die Entwicklung von Cloud-basierten Anwendungen verstehen und die richtigen Verfahren für deren Implementierung festlegen
- Beherrschung der grundlegenden Konzepte, Dienste und Tools der Google Clouds-Plattform

Modul 14. Web-Client-Computing

- Den Prozess der Erstellung von Webinhalten mit Hilfe der Auszeichnungssprache HTML verstehen
- Die Verfahren und Techniken zur Verbesserung des Erscheinungsbildes eines in HTML geschriebenen Dokuments verstehen
- Die Entwicklung der JavaScript-Sprache kennen
- Erwerb der notwendigen Kenntnisse für die Entwicklung von Anwendungen auf der Web-Client-Seite
- Entwicklung von Anwendungen mit komplexen Strukturen unter Verwendung der verschiedenen Verfahren, Funktionen und Objekte, aus denen JavaScript besteht
- Lernen, wie man die DOM-Programmierschnittstelle für HTML- und XML-Dokumente verwendet, um sowohl deren Stilstruktur als auch deren Inhalt zu ändern
- Verständnis für die Verwendung von ereignisgesteuerten Abläufen und *Listeners* sowie die Verwendung moderner *Toolkits* und Ausrichtungssysteme
- Verstehen des Konzepts der Web-Usability, seiner Vorteile, Prinzipien, Methoden und Techniken, um eine Website für den Benutzer benutzbar zu machen
- Erwerb von Kenntnissen über die Barrierefreiheit im Internet, ihre Bedeutung für die heutigen digitalen Plattformen, Methoden, Normen und Standards sowie die Festlegung von Konformitätsmaßstäben

Modul 15. Web-Server-Computing

- Verstehen der grundlegenden, mittleren und fortgeschrittenen Konzepte der Sprache PHP für die Implementierung von serverseitigen Anwendungen
- Aneignung der erforderlichen Kenntnisse über Datenmodellierung, Beziehungen, Schlüssel und Normalisierungen
- Verstehen des Aufbaus des logischen Datenmodells, der Spezifikation von Tabellen, Spalten, Schlüsseln und Abhängigkeiten sowie der notwendigen Kenntnisse für die physische Handhabung von Daten, Dateitypen, Zugriffsmodi und deren Organisation
- Erlernen der Integration von in PHP entwickelten Anwendungen mit MariaDB- und MySQL-Datenbanken
- Beherrschung des Prozesses der Interaktion mit dem Kunden, durch den Einsatz von: Formularen, Cookies und Sitzungsmanagement
- Verstehen der Model-View-Controller-View (MVC)-Softwarearchitektur, die die Daten, die Benutzeroberfläche und die Steuerlogik einer Anwendung in drei verschiedene Komponenten aufteilt
- Erwerb von Fähigkeiten zur Nutzung von Webservices unter Verwendung von XML, SOA und REST

Modul 16. Sicherheitsmanagement

- Kenntnisse über den Prozess der Informationssicherheit, seine Auswirkungen auf Vertraulichkeit, Integrität, Verfügbarkeit und wirtschaftliche Kosten
- Erlernen der Anwendung bewährter Sicherheitspraktiken bei der Verwaltung von Informationstechnologiediensten
- Aneignung von Kenntnissen für die ordnungsgemäße Zertifizierung von Sicherheitsprozessen
- Authentifizierungsmechanismen und -methoden für die Zugangskontrolle sowie den Prozess der Zugangsprüfung verstehen
- Verständnis von Sicherheitsmanagementprogrammen, Risikomanagement und der Gestaltung von Sicherheitsstrategien
- Erlernen von Geschäftskontinuitätsplänen, ihren Phasen und ihrem Wartungsprozess
- Kenntnis der Verfahren für den korrekten Schutz des Unternehmens durch DMZ-Netze, den Einsatz von Systemen zur Erkennung von Eindringlingen und andere Methoden

Modul 17. Software-Sicherheit

- Verstehen der Probleme im Zusammenhang mit der Softwaresicherheit, ihrer Schwachstellen und deren Klassifizierung
- Kenntnis der Entwurfsgrundsätze, Methoden und Standards der Softwaresicherheit
- Die Anwendung von Sicherheit in den verschiedenen Phasen des Software-Lebenszyklus verstehen
- Aneignung der erforderlichen Kenntnisse für sichere Kodierungs- und Validierungstechniken während des gesamten Lebenszyklus
- Aneignung der Methoden und Verfahren zur Gewährleistung der Sicherheit bei der Entwicklung und Bereitstellung von Cloud-Diensten
- Verstehen der Grundlagen der Kryptologie und der verschiedenen derzeit verfügbaren Verschlüsselungstechniken

Modul 18. Verwaltung des Webservers

- Verstehen des Konzepts, der Funktionsweise, der Architektur, der Ressourcen und des Inhalts eines Webservers
- Verstehen der Funktionsweise, Struktur und Handhabung des HTTP-Protokolls
- Aneignung des Konzepts der verteilten Architekturen auf mehreren Servern
- Beherrschung der Funktionsweise eines Anwendungsservers und eines Proxyservers
- Analyse der verschiedenen Webserver, die heute auf dem Markt sind
- Verständnis des Prozesses der Nutzungsstatistik und des Lastausgleichs auf Webservern
- Erwerb der notwendigen Kenntnisse für die Installation, Administration, Konfiguration und Sicherheit des Microsoft Internet Information Services (IIS) Webservers sowie des freien Apache Webservers

Modul 19. Sicherheitsaudit

- Aneignung der Kenntnisse, die für die korrekte Durchführung des internen IT-Prüfungs- und Kontrollprozesses erforderlich sind
- Verständnis der für die Sicherheitsüberprüfung von Systemen und Netzen durchzuführenden Verfahren
- Verstehen der verschiedenen Hilfsmittel, Methoden und anschließenden Analysen während der Sicherheitsprüfung von Internet und mobilen Geräten
- Erlernen der Eigenschaften und Einflussfaktoren, die Geschäftsrisiken bedingen und die richtige Umsetzung eines angemessenen Risikomanagements bestimmen
- Erlernen von Maßnahmen zur Risikominderung sowie von Methoden zur Umsetzung eines Informationssicherheitsmanagementsystems und der zu verwendenden Vorschriften und Normen
- Verstehen der Verfahren zur Durchführung eines Sicherheitsaudits, seiner Nachvollziehbarkeit und der Präsentation der Ergebnisse

Modul 20. Sicherheit bei Online-Anwendungen

- Aneignung der erforderlichen Kenntnisse zur Bewertung und Erkennung von Schwachstellen in Online-Anwendungen
- Verständnis der Sicherheitsrichtlinien und -standards, die auf Online-Anwendungen anzuwenden sind
- Erlernen der Verfahren, die bei der Entwicklung von Web-Applikationen und ihrer anschließenden Validierung durch Analyse und Sicherheitstests anzuwenden sind
- Erlernen der Sicherheitsmaßnahmen für die Bereitstellung und Produktion von Web-Applikationen
- Verstehen der Konzepte, Funktionen und Technologien, die bei der Sicherheit von Webdiensten anzuwenden sind, sowie der Sicherheitstests und Schutzmaßnahmen
- Aneignung der Verfahren für ethisches *Hacking*, Malware-Analyse und forensische Analyse
- Kenntnis der Maßnahmen zur Abschwächung und Eindämmung von Zwischenfällen bei Webdiensten
- Anwendung von bewährten Verfahren für die Entwicklung und Implementierung von Online-Anwendungen

03

Kompetenzen

Das Design, die Planung, das Management und die Entwicklung von Software-Tools perfekt zu beherrschen, ist eine sehr komplexe Aufgabe, unter anderem aufgrund der Vielzahl der beteiligten Prozesse. Dieser weiterbildende Masterstudiengang wird dem Studenten jedoch alle Informationen vermitteln, die er benötigt, um seine Fähigkeiten bei der Kontrolle von Instrumenten in diesem Bereich zu perfektionieren. So kann er seine Aufgaben erfolgreich erledigen und seine Projekte mit den vielversprechendsten und hochwertigsten Ergebnissen im Bereich IT-Engineering abschließen.

Die Spezialisierung in diesem Bereich ermöglicht es Ihnen, spezifische Führungsqualitäten zu entwickeln, um Softwaremanagementprojekte zu leiten, eine Fähigkeit, die von Computertechnik-Unternehmen sehr geschätzt wird"

Allgemeine Kompetenzen

- Reduzierung der technischen Schulden von Projekten mit einem Qualitätsansatz anstelle eines Ansatzes, der auf Wirtschaftlichkeit und kurzen Fristen basiert
- Messung und Quantifizierung der Qualität eines Softwareprojekts
- Richtiges Durchführen *Test-Driven Development* (TDD), um die Qualitätsstandards der Software zu erhöhen
- Qualitätsorientierte Projektbudgetierung begründen
- Entwicklung von Qualitätsstandards, Modellen und Normen
- Prüfung verschiedener Bewertungen des Technologiereifegrads
- Reduzierung des Risikos und Gewährleistung der Pflege und Kontrolle nachfolgender Versionen
- Die Phasen beherrschen, in die ein Projekt aufgeteilt ist
- Entwurf, Verwaltung und Durchführung von Projekten in den Bereichen Software Engineering und Computersysteme

Sie werden die wichtigsten Datenbanken im Detail kennen lernen und Zugang zu Simulationen realer Projekte haben, die in Unternehmen verschiedener Branchen angewandt werden"

Spezifische Kompetenzen

- Bewertung eines Softwaresystems nach dem Grad des Fortschritts im Projektprozess
- Richtige und strategische Herangehensweise an diese Fragen der Zuverlässigkeit, Metriken und Sicherheit in Softwareprojekten
- Auf den Prozess der Entscheidungen über die im Projekt zu verwendende Methodik eingehen
- Beherrschung der wesentlichen normativen Aspekte für die Erstellung von Software
- Automatisches *Testing* entwickeln
- Aufbau einer angemessenen Kommunikation mit dem Kunden, um zu verstehen, wie er/sie das Projekt gemäß der angewandten Methodik wahrnimmt
- Ausarbeitung der Liste der Testanforderungen
- Durchführung von Abstraktion, Aufteilung in einheitlichere Tests und Eliminierung dessen, was für die gute Durchführung der Tests des durchzuführenden Softwareprojekts nicht relevant ist
- Aktualisierung der Liste der Testanforderungen auf angemessene und korrekte Weise
- Anpassung der DevOps-Kultur an die Geschäftsanforderungen
- Entwicklung der neuesten Praktiken und Tools für die kontinuierliche Integration und Bereitstellung
- Umstrukturierung und Bewältigung der Datenverwaltung und -koordination
- Die verschiedenen Arten der Anwendungsmodellierung und Entwurfsmuster in der Unified Modelling Language (UML) verstehen
- Verständnis der Funktionsweise des Qualitätsmanagements in Projekten, einschließlich Planung, Sicherung, Kontrolle, statistischer Konzepte und verfügbarer Instrumente
- Einsatz der erforderlichen Kenntnisse für die Entwicklung von Anwendungen und grafischen Oberflächen in den Sprachen Java und .NET
- Die Verfahren und Techniken zur Verbesserung des Erscheinungsbildes eines in HTML geschriebenen Dokuments verstehen
- Beherrschung des Prozesses der Interaktion mit dem Kunden, durch den Einsatz von Formularen, *Cookies* und Sitzungsmanagement
- Verständnis der Authentifizierungsmechanismen und -methoden für die Zugangskontrolle sowie des Prozesses der Zugangsprüfung
- Verstehen der Anwendung von Sicherheit in den verschiedenen Phasen des Software-Lebenszyklus
- Verstehen des Konzepts, der Funktionsweise, der Architektur, der Ressourcen und des Inhalts eines Webservers
- Verstehen der verschiedenen Hilfsmittel, Methoden und anschließenden Analysen während der Sicherheitsprüfung von Internet und mobilen Geräten
- Verständnis der Sicherheitsrichtlinien und -standards, die auf Online-Anwendungen anzuwenden sind

04

Kursleitung

Die Leitung und das Dozententeam dieses Weiterbildenden Masterstudiengangs in Softwaretechnik und -qualität werden von einem Team von Ingenieuren mit langjähriger Erfahrung im Management und in der Entwicklung von technischen und Spezialprojekten durchgeführt. Der professionelle Hintergrund verleiht dieser Qualifikation ein Plus an Qualität, das sich in einer besseren Kontextualisierung der Inhalte durch den Studenten sowie in der Implementierung realer und simulierter praktischer Fälle in die akademische Erfahrung widerspiegeln wird, aber immer mit dem Ziel, ein 100%iges Online-Programm anzubieten, das dynamisch, avantgardistisch und auf der unmittelbaren Realität des Sektors basiert.

Das Team von Ingenieuren, das für die Lehre dieses weiterbildenden Masterstudiengangs verantwortlich ist, wird Ihnen zur Verfügung stehen, um Sie anzuleiten und Ihnen zu helfen, alle Fragen oder Zweifel zu klären, die Sie über den Studienplan oder den Beruf haben"

Internationaler Gastdirektor

Darren Pulsipher ist ein sehr erfahrener **Softwarearchitekt**, ein Innovator mit einer hervorragenden internationalen Erfolgsbilanz in der **Software- und Firmwareentwicklung**. Er verfügt über hoch entwickelte **Kommunikations**-, **Projektmanagement**- und **Geschäftsfähigkeiten**, die es ihm ermöglicht haben, große globale Initiativen zu leiten.

Im Laufe seiner Karriere hatte er auch leitende Positionen mit großer Verantwortung inne, wie z. B. die des **Chefarchitekten für Lösungen für den öffentlichen Sektor** bei der **Intel Corporation**, wo er **moderne Geschäfte, Prozesse und Technologien** für Kunden, Partner und Benutzer im **öffentlichen Sektor** vorantrieb. Darüber hinaus gründete er **Yoly Inc.**, wo er auch als **CEO** fungierte, und arbeitete an der Entwicklung eines **Tools zur Aggregation und Diagnose sozialer Netzwerke** auf der Grundlage von ***Software as a Service*** (**SaaS**), das ***Big Data*** und **Web 2.0**-Technologien nutzt.

Darüber hinaus war er in anderen Unternehmen tätig, unter anderem als **leitender Ingenieur** bei **Dell Technologies**, wo er die **Abteilung *Big Data* in der Cloud** leitete und Teams in den **USA** und **China** führte, um große Projekte zu verwalten und Geschäftsbereiche für eine erfolgreiche Integration umzustrukturieren. Er war auch ***Chief Information Officer*** bei **XanGo**, wo er Projekte wie ***Helpdesk*-Support**, **Produktionssupport** und **Lösungsentwicklung** leitete.

Zu den vielen Spezialgebieten, in denen er Experte ist, gehören ***Edge-to-Cloud*-Technologie**, **Cybersicherheit**, **generative künstliche Intelligenz**, **Softwareentwicklung**, **Netzwerktechnologie**, **Cloud-native Entwicklung** und das **Container-Ökosystem**. Sein Wissen gibt er über den **wöchentlichen Podcast** und die **Newsletter *„Embracing Digital Transformation"*** weiter, die er produziert und präsentiert hat und die Organisationen dabei helfen, die **digitale Transformation** durch den Einsatz von **Menschen, Prozessen** und **Technologie** erfolgreich zu meistern.

Hr. Pulsipher, Darren

- Chefarchitekt für Lösungen für den öffentlichen Sektor bei Intel, Kalifornien, USA
- Moderator und Produzent von „Embracing Digital Transformation», Kalifornien
- Gründer und CEO von Yoly Inc., Arkansas
- Leitender Ingenieur bei Dell Technologies, Arkansas
- Chief Information Officer bei XanGo, Utah
- Leitender Architekt bei Cadence Design Systems, Kalifornien
- Leitender Projektprozessmanager bei Lucent Technologies, Kalifornien
- Software-Ingenieur bei Cemax-Icon, Kalifornien
- Software-Ingenieur bei ISG Technologies, Kanada
- MBA in Technologiemanagement von der Universität von Phoenix
- Hochschulabschluss in Informatik und Elektrotechnik von der Brigham Young University

Dank TECH werden Sie mit den besten Fachleuten der Welt lernen können"

Leitung

Hr. Molina Molina, Jerónimo

- IA Engineer & Software Architect, NASSAT - Internet Satélite en Movimiento
- Senior Berater Hexa Ingenieure Einführung in die künstliche Intelligenz (ML und CV)
- Experte für auf künstlicher Intelligenz basierende Lösungen in den Bereichen *Computer Vision,* ML/DL und NLP
- Persönliches Forschungsprojekt in Möglichkeiten der Anwendung von *Transformers* und *Reinforcement Learning*
- Universitätsexperte für Unternehmensgründung und -entwicklung, Bancaixa- FUNDEUN Alicante
- Computer-Ingenieur, Universität von Alicante
- Masterstudiengang in Künstliche Intelligenz, Katholische Universität von Avila
- MBA-Executive, Europäisches Forum Business Campus

Professoren

Hr. Martínez Calvo, Francisco Javier

- Architekt - Organischer und funktioneller Analytiker
- Technischer Berater - IT
- Entwicklung und Unterstützung des European Medical Project, FNMT, PPG und PCL Integration in HexaIngenieros
- Ausbilder Visual Studio, SqlServer, CCNA (Cisco-Router und -Switch), PHP und .NET-Webprogrammierung in mehreren Zentren (Salesianos, Maforem, Dreamsoft)
- Technischer Wirtschaftsingenieur mit Spezialisierung auf Elektrizität und Industrieelektronik
- Masterstudiengang Cibernos in .NET, MCAD
- Masterstudiengang Eidos in fortgeschrittener Programmierung, Expertenniveau
- Masterstudiengang WEB Zertifizierungen für Dreamweaver, Fireworks, Flash und ActionScript, Version MX

Hr. Tenrero Morán, Marcos

- DevOps Ingenieur- Allot Communications
- Application Lifecycle Management & DevOps- Meta4 Spain, Cegid
- QA Automation Engineer- Meta4 Spain, Cegid
- Hochschulabschluss in Computertechnik an der Universität Rey Juan Carlos
- Professionelle Anwendungsentwicklung für Android, Universität Galileo, Guatemala
- Entwicklung von Cloud-Diensten (nodeJs, JavaScript, HTML5), UPM
- Kontinuierliche Integration mit Jenkins- Meta4, Cegid
- Webentwicklung mit Angular-CLI (4), Ionic und nodeJS, Meta4, Universität Rey Juan Carlos

Dr. Acebes Tamargo, Patricia

- Operations-Abteilung, Arbeit mit Elasticsearch und Kivana, Sirt
- Forschung in der Linie Human Factor und AI Anwendungen, CTIC-Technologiezentrum
- Forschung in der Linie Geschäftseinheit, CTIC-Technologiezentrum
- Abteilung für digitale Gesundheit und aktives Altern, CTIC-Technologiezentrum
- Abteilung Data Science, CTIC-Technologiezentrum
- Doktorandin in Computertechnik
- Hochschulabschluss in Wirtschaftswissenschaften, Universität von Oviedo
- Masterstudiengang in Datenanalyse UCJC
- Masterstudiengang in Künstlicher Intelligenz (KI), UNED
- Studiengang in Mathematik und IKT-Technik, UNED
- Masterstudiengang in Blockchain, Smart Contracts und Kryptowährungen, Universität von Alcalá
- Aufbaustudium in Blockchain Engineering, EADA
- Masterstudiengang in Wirtschaftswissenschaften, Instrumente der Wirtschaftsanalyse
- Masterstudiengang in Steuern, Fachverband der Ökonomen

Fr. Rodríguez Míguez, Cándida

- CoFounder und City Leader des KI-Netzwerks von Galicien (Spain AI-Verband)
- Akademischer Streamer auf YouTube
- SISAP-Projekt für SERGAS, Web-Funktionalität Autoimpfung COVID Online-Termine, INDRA Producción S.L.,
- OSAL Aixiña Remote Zusammenarbeit mit TFM
- Dozentin Einführungssitzung über künstliche Intelligenz, WordPress Galicia
- Informatik-Ingenieurin mit Spezialisierung auf Software, ESEI Ourense, Universität von Vigo
- Masterstudiengang in Computertechnik, Spezialisierung auf die Entwicklung großer Softwaresysteme. ESEI Ourense, Universität von Vigo
- Höherer Studiengang in Handelsmanagement und Marketing, Privates Berufsbildungszentrum Novacaixagalicia Ourense
- Höherer Studiengang in Computersystemverwaltung, Privates Berufsbildungszentrum Novacaixagalicia Ourense

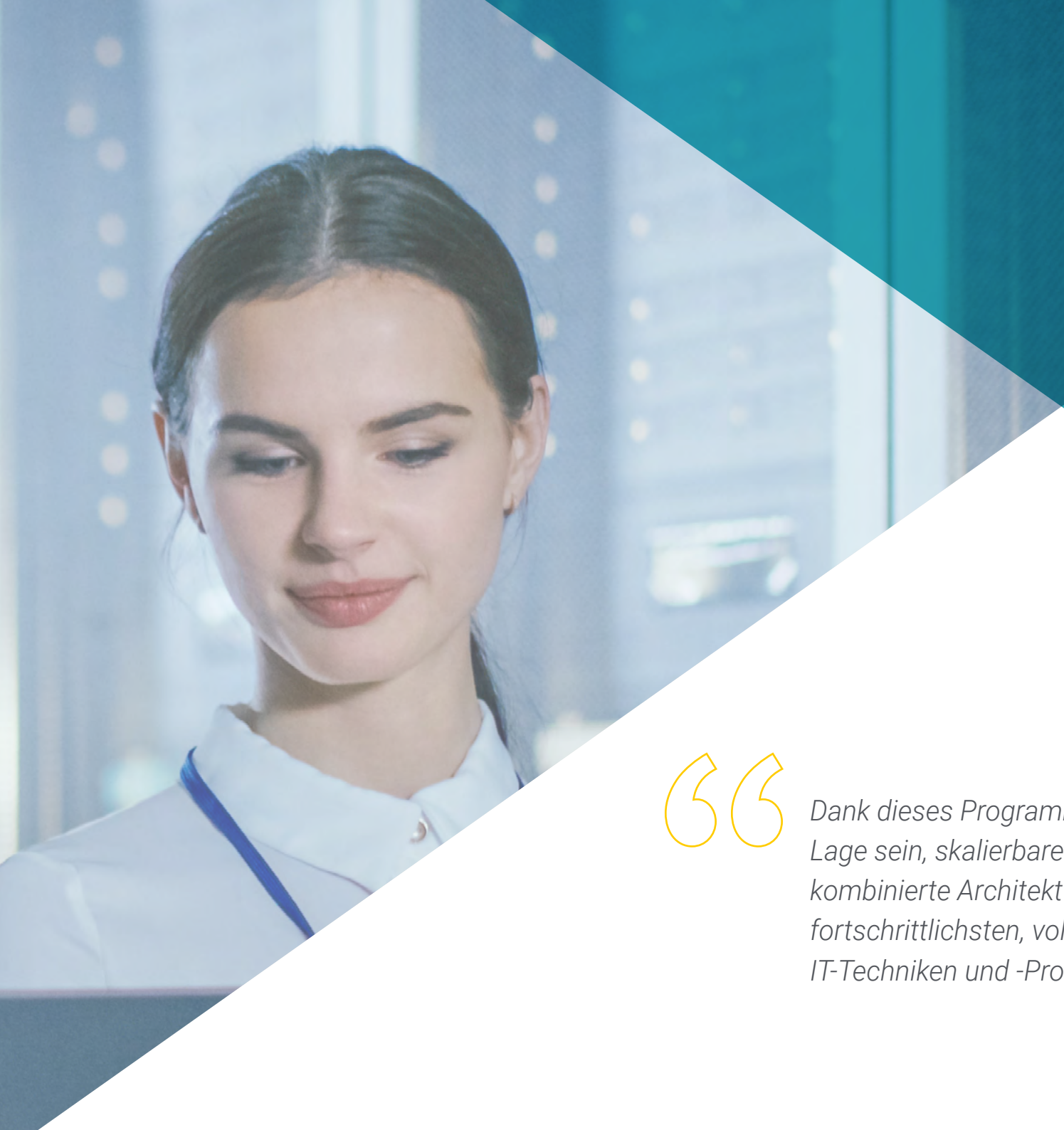
Hr. Pi Morell, Oriol

- Product Owner von Hosting und E-Mail, CDMON
- Funktionsanalytiker und Software-Ingenieur in verschiedenen Organisationen wie Fihoca, Atmira, CapGemini
- Dozent für verschiedene Kurse wie BPM in CapGemini, ORACLE Forms CapGemini, Business Processes Atmira
- Hochschulabschluss in technischem Ingenieurwesen in Computer Management von der Autonomen Universität Madrid
- Masterstudiengang in Künstlicher Intelligenz
- Masterstudiengang in Business Management und Verwaltung, MBA
- Masterstudiengang in Information Systems Management, Lehrerfahrung
- Aufbaustudium in Design Pattern, Offene Universität von Katalonien

05

Struktur und Inhalt

Bei der Entwicklung dieser Qualifikation hat TECH die Struktur des Inhalts auf drei grundlegende Säulen gestützt: die aktuellsten und vollständigsten Informationen im Bereich des IT-Engineering, spezialisiert auf den Bereich Software, die Empfehlungen des Dozententeams und die innovative Lehrmethodik des *Relearning.* Im Rahmen der Verpflichtung, ein auf die akademischen Bedürfnisse jedes Studenten zugeschnittenes und personalisiertes Programm anzubieten, nicht nur bei der Gestaltung der Studienpläne, sondern auch bei der Vertiefung des Studiums, finden die Studenten im virtuellen Klassenzimmer Hunderte von Stunden zusätzliches Material, mit dem sie die Aspekte vertiefen können, die sie für ihre berufliche Praxis für besonders relevant halten.



Dank dieses Programms werden Sie in der Lage sein, skalierbare vertikale, horizontale und kombinierte Architekturen zu entwerfen, die auf den fortschrittlichsten, vollständigsten und aktuellsten IT-Techniken und -Protokollen basieren"

Modul 1. Software-Qualität. Technologie-Reifegrad-Entwicklungsstufen

1.1. Elemente, die die Softwarequalität beeinflussen (I). Die technische Schuld
- 1.1.1. Die technische Schuld. Ursachen und Folgen
- 1.1.2. Software-Qualität. Allgemeine Grundsätze
- 1.1.3. Software mit und ohne Qualitätsprinzipien
  - 1.1.3.1. Konsequenzen
  - 1.1.3.2. Notwendigkeit der Anwendung von Qualitätsprinzipien bei Software
- 1.1.4. Software-Qualität. Typologie
- 1.1.5. Qualitätssoftware. Besondere Features

1.2. Elemente, die die Softwarequalität beeinflussen (II). Zugehörige Kosten
- 1.2.1. Software-Qualität. Beeinflussende Elemente
- 1.2.2. Software-Qualität. Missverständnisse
- 1.2.3. Software-Qualität. Zugehörige Kosten

1.3. Software-Qualitätsmodelle (I). Wissensmanagement
- 1.3.1. Allgemeine Qualitätsmodelle
  - 1.3.1.1. Total Quality Management
  - 1.3.1.2. Europäisches Modell für Business Excellence (EFQM)
  - 1.3.1.3. Six-Sigma-Modell
- 1.3.2. Modelle für das Wissensmanagement
  - 1.3.2.1. Dyba Modell
  - 1.3.2.2. Modell Seks
- 1.3.3. Erlebnisfabrik und QIP-Paradigma
- 1.3.4. Modelle mit hoher Nutzungsqualität (25010)

1.4. Software-Qualitätsmodelle (III). Qualität bei Daten, Prozessen und SEI-Modellen
- 1.4.1. Modell der Datenqualität
- 1.4.2. Software zur Prozessmodellierung
- 1.4.3. *Software&Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. SEI-Modelle
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. IISO-Software-Qualitätsnormen (I). Analyse der Standards
- 1.5.1. ISO 9000-Normen
  - 1.5.1.1. ISO 9000-Normen
  - 1.5.1.2. ISO-Familie von Qualitätsstandards (9000)
- 1.5.2. Andere ISO-Normen zum Thema Qualität
- 1.5.3. Normen zur Qualitätsmodellierung (ISO 2501)
- 1.5.4. Normen zur Qualitätsmessung (ISO 2502n)

1.6. ISO-Software-Qualitätsstandards (II). Anforderungen und Bewertung
- 1.6.1. Normen für Qualitätsanforderungen (2503n)
- 1.6.2. Normen zur Qualitätsbewertung (2504n)
- 1.6.3. ISO/IEC 24744:2007

1.7. Technologie-Reifegrad-Entwicklungsstufen (I). Stufen 1 bis 4
- 1.7.1. Technologie-Reifegrad-Stufen
- 1.7.2. Stufe 1: Grundlegende Prinzipien
- 1.7.3. Stufe 2: Konzept und/oder Anwendung
- 1.7.4. Stufe 3: kritische analytische Funktion
- 1.7.5. Stufe 4: Komponentenvalidierung in einer Laborumgebung

1.8. Technologie-Reifegrad-Entwicklungsstufen (II). Stufen 5 bis 9
- 1.8.1. Stufe 5: Komponentenvalidierung in der entsprechenden Umgebung
- 1.8.2. Stufe 6: System/Subsystem-Modell
- 1.8.3. Stufe 7: Demonstration in realer Umgebung
- 1.8.4. Stufe 8: Vollständiges und zertifiziertes System
- 1.8.5. Stufe 9: Erfolg in realer Umgebung

1.9. Technologie-Reifegrad-Entwicklungsstufen. Verwendungen
- 1.9.1. Beispiel für ein Unternehmen mit Laborumgebung
- 1.9.2. Beispiel für ein FuEuI-Unternehmen
- 1.9.3. Beispiel für ein industrielles FuEuI-Unternehmen
- 1.9.4. Beispiel für ein Joint-Venture-Unternehmen zwischen Labor und Technik

1.10. Software-Qualität. Wichtige Details
- 1.10.1. Methodische Details
- 1.10.2. Technische Details
- 1.10.3. Details zum Software-Projektmanagement
  - 1.10.3.1. Qualität der IT-Systeme
  - 1.10.3.2. Qualität von Softwareprodukten
  - 1.10.3.3. Qualität der Softwareprozesse

Modul 2. Software-Projektentwicklung. Funktionelle und technische Dokumentation

2.1. Projektmanagement
- 2.1.1. Projektmanagement für Softwarequalität
- 2.1.2. Projektmanagement. Vorteile
- 2.1.3. Projektmanagement. Typologie

2.2. Methodik des Projektmanagements
- 2.2.1. Methodik des Projektmanagements
- 2.2.2. Projekt-Methoden. Typologie
- 2.2.3. Methodik des Projektmanagements. Anwendung

2.3. Phase der Anforderungsermittlung
- 2.3.1. Identifizierung der Projektanforderungen
- 2.3.2. Verwaltung von Projekttreffen
- 2.3.3. Vorzulegende Dokumentation

2.4. Modell
- 2.4.1. Anfangsphase
- 2.4.2. Analysephase
- 2.4.3. Bauphase
- 2.4.4. Testphase
- 2.4.5. Lieferung

2.5. Zu verwendendes Datenmodell
- 2.5.1. Festlegung des neuen Datenmodells
- 2.5.2. Identifizierung des Datenmigrationsplans
- 2.5.3. Datensatz

2.6. Auswirkungen auf andere Projekte
- 2.6.1. Auswirkungen eines Projekts. Beispiele

2.7. *MUST* des Projekts
- 2.7.1. *MUST* des Projekts
- 2.7.2. Identifizierung des *MUST* des Projekts
- 2.7.3. Identifizierung der Ausführungspunkte für die Lieferung eines Projekts

2.8. Das Team für die Projektkonstruktion
- 2.8.1. Rollen, die je nach Projekt zu spielen sind
- 2.8.2. Kontakt mit der Personalabteilung für die Rekrutierung
- 2.8.3. Leistungen und Projektzeitplan

2.9. Technische Aspekte eines Softwareprojekts
- 2.9.1. Projekt Architekt. Technische Aspekte
- 2.9.2. Technische Leiter
- 2.9.3. Aufbau des Projekts Software
- 2.9.4. Bewertung der Codequalität, Sonar

2.10. Projektleistungen
- 2.10.1. Funktionsanalyse
- 2.10.2. Datenmodell
- 2.10.3. Zustandsdiagramm
- 2.10.4. Technische Dokumentation

Modul 3. *Testing* der Software. Testautomatisierung

3.1. Software-Qualitätsmodelle
- 3.1.1. Produktqualität
- 3.1.2. Prozessqualität
- 3.1.3. Qualität der Nutzung

3.2. Prozessqualität
- 3.2.1. Prozessqualität
- 3.2.2. Reifegradmodelle
- 3.2.3. ISO 15504-Norm
  - 3.2.3.1. Verwendungszwecke
  - 3.2.3.2. Kontext
  - 3.2.3.3. Etappen

3.3. ISO/IEC 15504-Norm
- 3.3.1. Prozess-Kategorien
- 3.3.2. Entwicklungsprozess. Beispiel
- 3.3.3. Profil Fragment
- 3.3.4. Etappen

3.4. CMMI (*Capability Maturity Model Integration*)
- 3.4.1. Integration des Capability Maturity Model
- 3.4.2. Modelle und Bereiche. Typologie
- 3.4.3. Prozessbereiche
- 3.4.4. Kapazitätsstufen
- 3.4.5. Prozessmanagement
- 3.4.6. Projektleitung

3.5. Verwaltung von Änderungen und Repositories
- 3.5.1. Management von Softwareänderungen
  - 3.5.1.1. Konfigurationselement. Kontinuierliche Integration
  - 3.5.1.2. Zeilen
  - 3.5.1.3. Flussdiagramme
  - 3.5.1.4. *Branches*
- 3.5.2. Repository
  - 3.5.2.1. Versionskontrolle
  - 3.5.2.2. Arbeitsteam und Nutzung des Repository
  - 3.5.2.3. Kontinuierliche Integration in das Repository

3.6. *Team Foundation Server* (TFS)
- 3.6.1. Installation und Konfiguration
- 3.6.2. Ein Team-Projekt erstellen
- 3.6.3. Hinzufügen von Inhalten zur Versionskontrolle
- 3.6.4. *TFS on Cloud*

3.7. *Testing*
- 3.7.1. Motivation für Tests
- 3.7.2. Verifizierungstests
- 3.7.3. Beta-Tests
- 3.7.4. Implementierung und Wartung

3.8. Belastungstests
- 3.8.1. *Load testing*
- 3.8.2. *LoadView*-Tests
- 3.8.3. Testen mit K6 *Cloud*
- 3.8.4. Testen mit *Loader*

3.9. Belastungs- und Dauertests für Module
- 3.9.1. Motivation für Modultests
- 3.9.2. *Unit Testing* Tools
- 3.9.3. Motivation für Belastungstests
- 3.9.4. Testen mit *StressTesting*
- 3.9.5. Motivation für Stresstests
- 3.9.6. Testen mit *LoadRunner*

3.10. Skalierbarkeit. Skalierbares Software-Design
- 3.10.1. Skalierbarkeit und Software-Architektur
- 3.10.2. Unabhängigkeit zwischen den Ebenen
- 3.10.3. Kopplung zwischen Schichten. Architektur-Muster

Modul 4. Methodologien des Software-Projektmanagements. *Waterfall*-Methodologie versus agile Methodologie

4.1. *Waterfall* Methodologie
- 4.1.1. *Waterfall* Methodologie
- 4.1.2. *Waterfall* Methodologie. Einfluss auf die Softwarequalität
- 4.1.3. *Waterfall* Methodologie. Beispiele

4.2. Agile Methodologie
- 4.2.1. Agile Methodologie
- 4.2.2. Agile Methodologie. Einfluss auf die Softwarequalität
- 4.2.3. Agile Methodologie. Beispiele

4.3. SCRUM-Methodologie
- 4.3.1. SCRUM-Methodologie
- 4.3.2. SCRUM Manifest
- 4.3.3. Einführung von SCRUM

4.4. Kanban-Panel
- 4.4.1. Kanban-Methode
- 4.4.2. Kanban-Panel
- 4.4.3. Kanban-Panel. Beispiel einer Anwendung

4.5. Projektmanagement in *Waterfall*
- 4.5.1. Phasen eines Projekts
- 4.5.2. Projektvision in *Waterfall*
- 4.5.3. Zu berücksichtigende Leistungen

4.6. Projektmanagement in SCRUM
- 4.6.1. Phasen eines Projekts SCRUM
- 4.6.2. Projektvision in SCRUM
- 4.6.3. Zu berücksichtigende Leistungen

4.7. Waterfall vs. SCRUM. Vergleich
- 4.7.1. Ansatz des Pilotprojekts
- 4.7.2. Projekt nach dem Prinzip *Waterfall*. Beispiel
- 4.7.3. Projekt nach dem Prinzip SCRUM. Beispiel

4.8. Kundenvision
- 4.8.1. Dokumente in Waterfall
- 4.8.2. Dokumente in SCRUM
- 4.8.3. Vergleich

4.9. Kanban Struktur
- 4.9.1. Anwenderberichte
- 4.9.2. *Backlog*
- 4.9.3. Kanban-Analyse

4.10. Hybride Projekte
- 4.10.1. Projekt Konstruktion
- 4.10.2. Projektleitung
- 4.10.3. Zu berücksichtigende Leistungen

Modul 5. TDD (*Test Driven Developement*). Testgetriebener Softwareentwurf

5.1. TDD. *Test Driven Development*
- 5.1.1. TDD. *Test Driven Development*
- 5.1.2. TDD. Einfluss von TDD auf die Qualität
- 5.1.3. Testgesteuertes Design und Entwicklung. Beispiele

5.2. TDD-Zyklus
- 5.2.1. Auswahl einer Anforderung
- 5.2.2. Testen. Typologien
  - 5.2.2.1. Modultests
  - 5.2.2.2. Integrationstests
  - 5.2.2.3. *End To End*-Tests
- 5.2.3. Prüfung des Tests. Misserfolge
- 5.2.4. Erstellung der Implementierung
- 5.2.5. Ausführung von automatisierten Tests
- 5.2.6. Eliminierung von Duplizierung
- 5.2.7. Aktualisierung der Liste der Anforderungen
- 5.2.8. Wiederholung des TDD-Zyklus
- 5.2.9. TDD-Zyklus. Theoretisches und praktisches Beispiel

5.3. TDD-Implementierungsstrategien
- 5.3.1. Mock-Implementierung
- 5.3.2. Dreieckige Implementierung
- 5.3.3. Offensichtliche Implementierung

5.4. TDD. Nutzung. Vorteile und Nachteile
- 5.4.1. Vorteile der Nutzung
- 5.4.2. Beschränkungen der Nutzung
- 5.4.3. Qualitätsbilanz in der Implementierung

5.5. TDD. Bewährte Verfahren
- 5.5.1. TDD-Regeln
- 5.5.2. Regel 1: Durchführung eines früheren Tests, falls dieser fehlschlägt, bevor in der Produktion programmiert wird
- 5.5.3. Regel 2: Nicht mehr als einen Modultest schreiben
- 5.5.4. Regel 3: Nicht mehr Code schreiben als nötig
- 5.5.5. Zu vermeidende Fehler und Anti-Patterns bei TDD

5.6. Simulation eines realen Projekts zur Anwendung von TDD (I)
- 5.6.1. Allgemeine Beschreibung des Projekts (Unternehmen A)
- 5.6.2. Implementierung von TDD
- 5.6.3. Vorgeschlagene Übungen
- 5.6.4. Übungen. *Feedback*

5.7. Simulation eines realen Projekts zur Anwendung von TDD (II)
- 5.7.1. Allgemeine Beschreibung des Projekts (Unternehmen B)
- 5.7.2. Implementierung von TDD
- 5.7.3. Vorgeschlagene Übungen
- 5.7.4. Übungen. *Feedback*

5.8. Simulation eines realen Projekts zur Anwendung von TDD (III)
- 5.8.1. Allgemeine Beschreibung des Projekts (Unternehmen C)
- 5.8.2. Implementierung von TDD
- 5.8.3. Vorgeschlagene Übungen
- 5.8.4. Übungen. *Feedback*

5.9. Alternativen zu 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. Theoretischer Vergleich

5.10. TDD TCR, BDD und ATDD. Praktischer Vergleich
- 5.10.1. Problemstellung
- 5.10.2. Lösen mit TCR
- 5.10.3. Lösen mit BDD
- 5.10.4. Lösen mit ATDD

Modul 6. DevOps. Software-Qualitätsmanagement

6.1. DevOps. Software-Qualitätsmanagement
- 6.1.1. DevOps
- 6.1.2. DevOps und Softwarequalität
- 6.1.3. DevOps. Vorteile der DevOps-Kultur

6.2. DevOps. Beziehung zu Agile
- 6.2.1. Beschleunigte Lieferung
- 6.2.2. Qualität
- 6.2.3. Kostensenkung

6.3. Implementierung von DevOps
- 6.3.1. Identifizierung des Problems
- 6.3.2. Implementierung in einem Unternehmen
- 6.3.3. Metriken zur Implementierung

6.4. Software-Lieferzyklus
- 6.4.1. Design-Methoden
- 6.4.2. Abkommen
- 6.4.3. Roadmap

6.5. Entwicklung von fehlerfreiem Code
- 6.5.1. Wartbarer Code
- 6.5.2. Entwicklungsmuster
- 6.5.3. Code-*Testing*
- 6.5.4. Software-Entwicklung auf Code-Ebene. Bewährte Verfahren

6.6. Automatisierung
- 6.6.1. Automatisierung. Arten von Tests
- 6.6.2. Kosten für Automatisierung und Wartung
- 6.6.3. Automatisierung. Fehler abmildern

6.7. Einsätze
- 6.7.1. Zielbewertung
- 6.7.2. Entwurf eines automatischen und angepassten Prozesses
- 6.7.3. Feedback und Reaktionsfähigkeit

6.8. Management von Zwischenfällen
- 6.8.1. Bereitschaft für Zwischenfälle
- 6.8.2. Analyse und Lösung von Vorfällen
- 6.8.3. Wie lassen sich künftige Fehler vermeiden?

6.9. Automatisierung des Einsatzes
- 6.9.1. Vorbereitungen für automatisierte Einsätze
- 6.9.2. Automatische Bewertung des Prozesszustands
- 6.9.3. Metriken und Rollback-Fähigkeit

6.10. Bewährte Verfahren. Entwicklung von DevOps
- 6.10.1. DevOps Leitfaden für bewährte Verfahren
- 6.10.2. DevOps. Methodologie für das Team
- 6.10.3. Nischen meiden

Modul 7. DevOps und kontinuierliche Integration. Fortgeschrittene praktische Lösungen in der Softwareentwicklung

7.1. Ablauf der Software-Lieferung
- 7.1.1. Identifizierung von Akteuren und Artefakten
- 7.1.2. Entwurf des Software-Lieferflusses
- 7.1.3. Ablauf der Softwarelieferung. Anforderungen zwischen den Etappen

7.2. Prozessautomatisierung
- 7.2.1. Kontinuierliche Integration
- 7.2.2. Kontinuierliche Bereitstellung
- 7.2.3. Konfiguration von Umgebungen und Verwaltung von Geheimnissen

7.3. Deklarative *Pipelines*
- 7.3.1. Unterschiede zwischen traditionellen, codeähnlichen und deklarativen *Pipelines*
- 7.3.2. Deklarative *Pipelines*
- 7.3.3. Deklarative *Pipelines* in Jenkins
- 7.3.4. Vergleich der Anbieter von kontinuierlicher Integration

7.4. Qualitätsprüfpunkte und erweitertes Feedback
- 7.4.1. Qualitätsprüfpunkte
- 7.4.2. Qualitätsstandards mit Qualitätsprüfpunkten. Wartung
- 7.4.3. Geschäftsanforderungen für Integrationsanfragen

7.5. Verwaltung von Artefakten
- 7.5.1. Artefakte und Lebenszyklus
- 7.5.2. Systeme zur Aufbewahrung und Verwaltung von Artefakten
- 7.5.3. Sicherheit bei der Verwaltung von Artefakten

7.6. Kontinuierliche Bereitstellung
- 7.6.1. Kontinuierliche Bereitstellung in Containern
- 7.6.2. Kontinuierliche Bereitstellung mit PaaS

7.7. Verbesserung der *Pipeline*-Laufzeit: statische Analyse und *Git Hooks*
- 7.7.1. Statische Analyse
- 7.7.2. Code-Stilregeln
- 7.7.3. *Git Hooks* und Modul*tests*
- 7.7.4. Die Auswirkungen der Infrastruktur

7.8. Container-Schwachstellen
- 7.8.1. Container-Schwachstellen
- 7.8.2. Scannen von Bildern
- 7.8.3. Regelmäßige Berichte und Warnmeldungen

Modul 8. Datenbank-Design (DB). Standardisierung und Leistung. Software-Qualität

8.1. Entwurf von Datenbanken
- 8.1.1. Datenbanken. Typologie
- 8.1.2. Derzeit verwendete Datenbanken
  - 8.1.2.1. Relational
  - 8.1.2.2. Schlüssel-Wert
  - 8.1.2.3. Graph-basiert
- 8.1.3. Die Datenqualität

8.2. Entwurf eines Entity-Relationship-Modells (I)
- 8.2.1. Entity-Relationship-Modell. Qualität und Dokumentation
- 8.2.2. Einheiten
  - 8.2.2.1. Starke Einheit
  - 8.2.2.2. Schwache Einheit
- 8.2.3. Attribute

8.2.4. Beziehungsset
8.2.4.1. 1 zu 1
8.2.4.2. 1 zu vielen
8.2.4.3. Viele zu 1
8.2.4.4. Viele zu viele
8.2.5. Schlüssel
8.2.5.1. Primärschlüssel
8.2.5.2. Fremdschlüssel
8.2.5.3. Schwacher Primärschlüssel der Einheit
8.2.6. Beschränkungen
8.2.7. Kardinalität
8.2.8. Vererbung
8.2.9. Aggregation
8.3. Entity-Relationship-Modell (II). Tools
8.3.1. Entity-Relationship-Modell. Tools
8.3.2. Entity-Relationship-Modell. Praktisches Beispiel
8.3.3. Durchführbares Entity-Relationship-Modell
8.3.3.1. Visuelles Beispiel
8.3.3.2. Beispiel in tabellarischer Darstellung
8.4. Standardisierung von Datenbanken (DB) (I). Erwägungen zur Softwarequalität
8.4.1. DB Standardisierung und Qualität
8.4.2. Abhängigkeit
8.4.2.1. Funktionsabhängigkeit
8.4.2.2. Eigenschaften der Funktionsabhängigkeit
8.4.2.3. Abgeleitete Eigenschaften
8.4.3. Schlüssel
8.5. Standardisierung von Datenbanken (DB) (II). Normalformen und Codd-Regeln
8.5.1. Normale Formen
8.5.1.1. Erste Normalform (1NF)
8.5.1.2. Zweite Normalform (2NF)
8.5.1.3. Dritte Normalform (3NF)
8.5.1.4. Boyce-Codd-Normalform (BCNF)
8.5.1.5. Vierte Normalform (4NF)
8.5.1.6. Fünfte Normalform (5NF)
8.5.2. Codd's Regeln
8.5.2.1. Regel 1: Information
8.5.2.2. Regel 2: Garantierter Zugang
8.5.2.3. Regel 3: Systematische Behandlung von Nullwerten
8.5.2.4. Regel 4: Beschreibung der Datenbank
8.5.2.5. Regel 5: Integrale Untersprache
8.5.2.6. Regel 6: Ansicht aktualisieren
8.5.2.7. Regel 7: Einfügen und Aktualisieren
8.5.2.8. Regel 8: Körperliche Unabhängigkeit
8.5.2.9. Regel 9: Logische Unabhängigkeit
8.5.2.10. Regel 10: Unabhängigkeit der Integrität
8.5.2.10.1. Integritätsregeln
8.5.2.11. Regel 11: Verteilung
8.5.2.12. Regel 12: Nicht-Subversion
8.5.3. Praktisches Beispiel
8.6. Datenlager / OLAP-System
8.6.1. Data Warehouse
8.6.2. Faktentabelle
8.6.3. Tabelle der Abmessungen
8.6.4. Erstellung des OLAP-Systems. Tools
8.7. Leistung der Datenbank (DB)
8.7.1. Index-Optimierung
8.7.2. Optimierung von Abfragen
8.7.3. Tabelle Partitionierung
8.8. Simulation des realen Projekts für DB-Design (I)
8.8.1. Allgemeine Beschreibung des Projekts (Unternehmen A)
8.8.2. Anwendung von Datenbankdesign
8.8.3. Vorgeschlagene Übungen
8.8.4. Vorgeschlagene Übungen. Feedback
8.9. Simulation des realen Projekts für DB-Design (II)
8.9.1. Allgemeine Beschreibung des Projekts (Unternehmen B)
8.9.2. Anwendung von Datenbankdesign
8.9.3. Vorgeschlagene Übungen
8.9.4. Vorgeschlagene Übungen. Feedback

8.10. Relevanz der DB-Optimierung für die Softwarequalität
- 8.10.1. Design-Optimierung
- 8.10.2. Optimierung des Abfragecodes
- 8.10.3. Optimierung von gespeichertem Prozedur-Code
- 8.10.4. Der Einfluss von *Triggers* auf die Softwarequalität. Empfehlungen für die Verwendung

Modul 9. Entwurf skalierbarer Architekturen. Architektur im Lebenszyklus der Software

9.1. Entwurf skalierbarer Architekturen (I)
- 9.1.1. Skalierbare Architekturen
- 9.1.2. Grundsätze einer skalierbaren Architektur
  - 9.1.2.1. Zuverlässig
  - 9.1.2.2. Skalierbar
  - 9.1.2.3. Wartbar
- 9.1.3. Arten der Skalierbarkeit
  - 9.1.3.1. Vertikal
  - 9.1.3.2. Horizontal
  - 9.1.3.3. Kombiniert

9.2. Architekturen DDD (*Domain-Driven Design*)
- 9.2.1. DDD-Modell. Domain-Ausrichtung
- 9.2.2. Schichten, Aufteilung der Verantwortung und Entwurfsmuster
- 9.2.3. Entkopplung als Grundlage für Qualität

9.3. Entwurf skalierbarer Architekturen (II). Vorteile, Einschränkungen und Designstrategien
- 9.3.1. Skalierbare Architektur. Vorteile
- 9.3.2. Skalierbare Architektur. Beschränkungen
- 9.3.3. Strategien für die Entwicklung skalierbarer Architekturen (Beschreibende Tabelle)

9.4. Lebenszyklus der Software (I). Etappen
- 9.4.1. Lebenszyklus der Software
  - 9.4.1.1. Planungsphase
  - 9.4.1.2. Analysephase
  - 9.4.1.3. Entwurfsphase
  - 9.4.1.4. Phase der Umsetzung
  - 9.4.1.5. Testphase
  - 9.4.1.6. Phase der Installation/Einrichtung
  - 9.4.1.7. Phase der Nutzung und Pflege

9.5. Software-Lebenszyklus-Modelle
- 9.5.1. Wasserfall-Modell
- 9.5.2. Wiederholtes Modell
- 9.5.3. Spiralförmiges Modell
- 9.5.4. Big Bang Modell

9.6. Lebenszyklus der Software (II). Automatisierung
- 9.6.1. Lebenszyklus der Softwareentwicklung. Lösungen
  - 9.6.1.1. Kontinuierliche Integration und Entwicklung (CI/CD)
  - 9.6.1.2. *Agile*Methodologien
  - 9.6.1.3. DevOps / Produktionsbetrieb
- 9.6.2. Zukünftige Trends
- 9.6.3. Praktische Beispiele

9.7. Software-Architektur im Software-Lebenszyklus
- 9.7.1. Vorteile
- 9.7.2. Beschränkungen
- 9.7.3. Tools

9.8. Simulation eines realen Projekts zum Entwurf einer Software-Architektur (I)
- 9.8.1. Allgemeine Beschreibung des Projekts (Unternehmen A)
- 9.8.2. Anwendung Softwarearchitektur-Design
- 9.8.3. Vorgeschlagene Übungen
- 9.8.4. Vorgeschlagene Übungen. Feedback

9.9. Simulation eines realen Projekts zum Entwurf einer Software-Architektur (II)
- 9.9.1. Allgemeine Beschreibung des Projekts (Unternehmen B)
- 9.9.2. Anwendung Softwarearchitektur-Design
- 9.9.3. Vorgeschlagene Übungen
- 9.9.4. Vorgeschlagene Übungen. Feedback

9.10. Simulation eines realen Projekts zum Entwurf einer Software-Architektur (III)
- 9.10.1. Allgemeine Beschreibung des Projekts (Unternehmen C)
- 9.10.2. Anwendung Softwarearchitektur-Design
- 9.10.3. Vorgeschlagene Übungen
- 9.10.4. Vorgeschlagene Übungen. Feedback

Modul 10. ISO, IEC 9126 Qualitätskriterien. Metriken zur Software-Qualität

10.1. Qualitätskriterien. ISO/IEC 9126 Norm
- 10.1.1. Qualitätskriterien
- 10.1.2. Software-Qualität. Gründe dafür. ISO/IEC 9126 Norm
- 10.1.3. Die Messung der Softwarequalität als Schlüsselindikator

10.2. Qualitätskriterien für Software. Eigenschaften
- 10.2.1. Verlässlichkeit
- 10.2.2. Funktionsweise
- 10.2.3. Effizienz
- 10.2.4. Benutzerfreundlichkeit
- 10.2.5. Instandhaltbarkeit
- 10.2.6. Tragbarkeit

10.3. ISO, IEC 9126 (I). Präsentation
- 10.3.1. Beschreibung von ISO, IEC 9126
- 10.3.2. Funktionsweise
- 10.3.3. Verlässlichkeit
- 10.3.4. Benutzerfreundlichkeit
- 10.3.5. Instandhaltbarkeit
- 10.3.6. Tragbarkeit
- 10.3.7. Qualität im Einsatz
- 10.3.8. Metriken zur Software-Qualität
- 10.3.9. Qualitätsmetriken in ISO 9126

10.4. ISO, IEC 9126 (II). McCall und Boehm Modelle
- 10.4.1. McCall Modell: Qualitätsfaktoren
- 10.4.2. Boehm Modell
- 10.4.3. Mittleres Niveau. Eigenschaften

10.5. Metriken zur Softwarequalität (I). Elemente
- 10.5.1. Messung
- 10.5.2. Metrisch
- 10.5.3. Indikator
  - 10.5.3.1. Arten von Indikatoren
- 10.5.4. Maßnahmen und Modelle
- 10.5.5. Umfang der Software Metriken
- 10.5.6. Klassifizierung von Softwaremetriken

10.6. Messung der Softwarequalität (II). Praxis der Messung
- 10.6.1. Metrische Datenerfassung
- 10.6.2. Messung der internen Produkteigenschaften
- 10.6.3. Messung von externen Produktattributen
- 10.6.4. Messung der Ressourcen
- 10.6.5. Metriken für objektorientierte Systeme

10.7. Design eines einzigen Software-Qualitätsindikators
- 10.7.1. Einzelner Indikator als Global Scorer
- 10.7.2. Entwicklung, Rechtfertigung und Anwendung von Indikatoren
- 10.7.3. Beispiel für eine Anwendung. Notwendigkeit, die Details zu kennen

10.8. Simulation eines realen Projekts zur Qualitätsmessung (I)
- 10.8.1. Allgemeine Beschreibung des Projekts (Unternehmen A)
- 10.8.2. Anwendung der Qualitätsmessung
- 10.8.3. Vorgeschlagene Übungen
- 10.8.4. Vorgeschlagene Übungen. *Feedback*

10.9. Simulation eines realen Projekts zur Qualitätsmessung (II)
- 10.9.1. Allgemeine Beschreibung des Projekts (Unternehmen B)
- 10.9.2. Anwendung der Qualitätsmessung
- 10.9.3. Vorgeschlagene Übungen
- 10.9.4. Vorgeschlagene Übungen. *Feedback*

10.10. Simulation eines realen Projekts zur Qualitätsmessung (III)
- 10.10.1. Allgemeine Beschreibung des Projekts (Unternehmen C)
- 10.10.2. Anwendung der Qualitätsmessung
- 10.10.3. Vorgeschlagene Übungen
- 10.10.4. Vorgeschlagene Übungen. *Feedback*

Modul 11. Methodik, Entwicklung und Qualität im Software Engineering

11.1. Modellgestützte Software-Entwicklung
- 11.1.1. Der Bedarf an
- 11.1.2. Modellierung von Objekten
- 11.1.3. UML
- 11.1.4. CASE-Werkzeuge

11.2. Anwendungsmodellierung und Entwurfsmuster mit UML
- 11.2.1. Fortgeschrittene Anforderungsmodellierung
- 11.2.2. Erweiterte statische Modellierung
- 11.2.3. Erweiterte dynamische Modellierung
- 11.2.4. Modellierung von Bauteilen
- 11.2.5. Einführung in Entwurfsmuster mit 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. Modellgestütztes Engineering
- 11.3.1. Einführung
- 11.3.2. Metamodellierung von Systemen
- 11.3.3. MDA
- 11.3.4. DSL
- 11.3.5. Modellverfeinerungen mit OCL
- 11.3.6. Modellumwandlungen

11.4. Ontologien im Software-Engineering
- 11.4.1. Einführung
- 11.4.2. Ontologie-Engineering
- 11.4.3. Anwendung der Ontologien im Software-Engineering

Modul 12. Software-Projektmanagement

12.1. Management von *Stakeholdern* und Umfang
- 12.1.1. Identifizierung von Interessengruppen
- 12.1.2. Entwicklung des Stakeholder-Management-Plans
- 12.1.3. Management der Einbindung von Stakeholdern
- 12.1.4. Überwachung des Engagements der Stakeholder
- 12.1.5. Das Projektziel
- 12.1.6. Umfangsmanagement und sein Plan
- 12.1.7. Erfassen von Anforderungen
- 12.1.8. Definieren Sie den Geltungsbereich
- 12.1.9. Erstellen des Projektstrukturplans
- 12.1.10. Überprüfung und Kontrolle des Umfangs

12.2. Die Entwicklung des Zeitplans
- 12.2.1. Zeitmanagement und sein Plan
- 12.2.2. Definieren der Aktivitäten
- 12.2.3. Festlegung der Reihenfolge der Aktivitäten
- 12.2.4. Schätzung der Ressourcen für die Aktivitäten
- 12.2.5. Geschätzte Dauer der Aktivitäten
- 12.2.6. Entwicklung des Zeitplans und Berechnung des kritischen Pfades
- 12.2.7. Zeitplan-Kontrolle

12.3. Budgetentwicklung und Risikobewältigung
- 12.3.1. Schätzung der Kosten
- 12.3.2. Entwicklung des Budgets und der S-Kurve
- 12.3.3. Kostenkontrolle und Earned-Value-Methode
- 12.3.4. Risikokonzepte
- 12.3.5. Wie man eine Risikoanalyse durchführt
- 12.3.6. Die Entwicklung des Reaktionsplans

12.4. Kommunikation und Personalwesen
- 12.4.1. Planung des Kommunikationsmanagements
- 12.4.2. Analyse der Kommunikationsanforderungen
- 12.4.3. Technologie der Kommunikation
- 12.4.4. Kommunikationsmodelle
- 12.4.5. Kommunikationsmethoden
- 12.4.6. Plan für das Kommunikationsmanagement
- 12.4.7. Verwaltung der Kommunikation
- 12.4.8. Verwaltung des Personalwesens
- 12.4.9. Hauptakteure und ihre Rolle in den Projekten
- 12.4.10. Arten von Organisationen
- 12.4.11. Projektorganisation
- 12.4.12. Das Projektteam

12.5. Beschaffung
- 12.5.1. Der Beschaffungsprozess
- 12.5.2. Planung
- 12.5.3. Beschaffung von Lieferanten und Einholung von Angeboten
- 12.5.4. Vergabe des Auftrags
- 12.5.5. Vertragsverwaltung
- 12.5.6. Verträge
- 12.5.7. Arten von Verträgen
- 12.5.8. Vertragsverhandlungen

12.6. Durchführung, Überwachung und Kontrolle sowie Abschluss
- 12.6.1. Prozessgruppen
- 12.6.2. Projektdurchführung
- 12.6.3. Projektüberwachung und -kontrolle
- 12.6.4. Abschluss des Projekts

12.7. Berufliche Verantwortung
- 12.7.1. Berufliche Verantwortung
- 12.7.2. Merkmale der sozialen und beruflichen Verantwortung
- 12.7.3. Ethischer Kodex für Projektleiter
- 12.7.4. Verantwortung vs. PMP®
- 12.7.5. Beispiele für Verantwortung
- 12.7.6. Vorteile der Professionalisierung

Modul 13. Plattformen für die Software-Entwicklung

13.1. Einführung in die Entwicklung von Applikationen
- 13.1.1. Desktop-Applikationen
- 13.1.2. Programmiersprache
- 13.1.3. Integrierte Entwicklungsumgebungen
- 13.1.4. Webanwendungen
- 13.1.5. Mobile Anwendungen
- 13.1.6. Cloud-Anwendungen

13.2. Anwendungsentwicklung und grafische Oberfläche in Java
- 13.2.1. Integrierte Entwicklungsumgebungen für Java
- 13.2.2. Wichtigste IDEs für Java
- 13.2.3. Einführung in die Eclipse-Entwicklungsplattform
- 13.2.4. Einführung in die NetBeans-Entwicklungsplattform
- 13.2.5. Controller-View-Modell für grafische Benutzeroberflächen
- 13.2.6. Entwerfen einer grafischen Benutzeroberfläche in Eclipse
- 13.2.7. Entwerfen einer grafischen Benutzeroberfläche in NetBeans

13.3. Fehlersuche und Testen in Java
- 13.3.1. Testen und Debuggen von Java-Programmen
- 13.3.2. Fehlersuche in Eclipse
- 13.3.3. Fehlersuche in NetBeans

13.4. Anwendungsentwicklung und grafische Oberfläche in .NET
- 13.4.1. Net Framework
- 13.4.2. Komponenten der .NET-Entwicklungsplattform
- 13.4.3. Visual Studio .NET
- 13.4.4. .NET Werkzeuge für GUI
- 13.4.5. Die grafische Benutzeroberfläche mit Windows Presentation Foundation
- 13.4.6. Debuggen und Kompilieren einer WPF-Anwendung

13.5. Programmierung für .NET-Netzwerke
- 13.5.1. Einführung in die .NET-Netzwerkprogrammierung
- 13.5.2. .NET-Anfragen und -Antworten
- 13.5.3. Verwendung von .NET-Anwendungsprotokollen
- 13.5.4. Sicherheit in der .NET-Netzwerkprogrammierung

13.6. Entwicklungsumgebungen für mobile Anwendungen
- 13.6.1. Mobile Anwendungen
- 13.6.2. Mobile Android-Anwendungen
- 13.6.3. Schritte für die Android-Entwicklung
- 13.6.4. Die Android Studio IDE

13.7. Entwicklung von Anwendungen in der Android Studio-Umgebung
- 13.7.1. Installieren und Starten von Android Studio
- 13.7.2. Ausführen einer Android-Anwendung
- 13.7.3. Entwicklung der grafischen Oberfläche in Android Studio
- 13.7.4. Starten von Aktivitäten in Android Studio

13.8. Debuggen und Veröffentlichen von Android-Anwendungen
- 13.8.1. Fehlersuche in einer Anwendung in Android Studio
- 13.8.2. Speichern von Anwendungen in Android Studio
- 13.8.3. Veröffentlichung einer Anwendung auf Google Play

13.9. Entwicklung von Anwendungen für die Cloud
- 13.9.1. *Cloud Computing*
- 13.9.2. *Cloud*-Ebenen: SaaS, PaaS, IaaS
- 13.9.3. Wichtigste Cloud-Entwicklungsplattformen
- 13.9.4. Bibliografische Referenzen

13.10. Einführung in die Google Cloud Platform
- 13.10.1. Grundlagen der Google Cloud Platform
- 13.10.2. Google Cloud Platform-Dienste
- 13.10.3. Google Cloud Platform-Werkzeuge

Modul 14. Web-Client-Computing

14.1. Einführung in HTML
- 14.1.1. Aufbau eines Dokuments
- 14.1.2. Farbe
- 14.1.3. Text
- 14.1.4. Hypertext-Links
- 14.1.5. Bilder
- 14.1.6. Listen
- 14.1.7. Tabellen
- 14.1.8. Rahmen (*frames*)
- 14.1.9. Formulare
- 14.1.10. Spezifische Elemente für mobile Technologien
- 14.1.11. Ausgediente Elemente

14.2. Web Style Sheets (CSS)
- 14.2.1. Elemente und Struktur einer Formatvorlage
  - 14.2.1.1. Erstellung von Stilvorlagen
  - 14.2.1.2. Anwendung von Stilen. Selektoren
  - 14.2.1.3. Stilvererbung und Kaskadierung
  - 14.2.1.4. Seitenformatierung mit Formatvorlagen
  - 14.2.1.5. Seitenstruktur mit Hilfe von Stilen. Das Kastenmodell
- 14.2.2. Gestaltung von Stilen für verschiedene Geräte
- 14.2.3. Arten von Formatvorlagen: statisch und dynamisch. Pseudo-Klassen
- 14.2.4. Bewährte Praktiken bei der Verwendung von Formatvorlagen

14.3. Einführung und Geschichte von JavaScript
- 14.3.1. Einführung
- 14.3.2. Geschichte von JavaScript
- 14.3.3. Entwicklungsumgebung, die wir verwenden werden

14.4. Grundbegriffe der Webprogrammierung
- 14.4.1. Grundlegende JavaScript-Syntax
- 14.4.2. Primitive Datentypen und Operatoren
- 14.4.3. Variablen und Domänen
- 14.4.4. Textstrings und *Template Literals*
- 14.4.5. Zahlen und Boolesche Werte
- 14.4.6. Vergleiche

14.5. Komplexe JavaScript-Strukturen
- 14.5.1. Vektoren oder *Arrays* und Objekte
- 14.5.2. Sets
- 14.5.3. Karten
- 14.5.4. Disjunktionen
- 14.5.5. Schleifen

14.6. Funktionen und Objekte
- 14.6.1. Funktionsdefinition und -aufruf
- 14.6.2. Argumente
- 14.6.3. Pfeil-Funktionen
- 14.6.4. Rückruf-Funktionen oder *Callback*
- 14.6.5. Funktionen höherer Ordnung
- 14.6.6. Wörtliche Objekte
- 14.6.7. Das Objekt *This*
- 14.6.8. Objekte als Namensräume: das *Math*-Objekt und das *Date*-Objekt

14.7. Das Dokumentenobjektmodell (DOM)
- 14.7.1. Was ist DOM?
- 14.7.2. Ein bisschen Geschichte
- 14.7.3. Navigieren und Abrufen von Elementen
- 14.7.4. Ein virtuelles DOM mit JSDOM
- 14.7.5. Abfrage-Selektoren oder *Query Selectors*
- 14.7.6. Navigation mittels Eigenschaften
- 14.7.7. Zuweisung von Attributen zu Elementen
- 14.7.8. Erstellen und Ändern von Knoten
- 14.7.9. Aktualisieren des Stils von DOM-Elementen

14.8. Moderne Webentwicklung
- 14.8.1. Ereignisgesteuerter Ablauf und *Listeners*
- 14.8.2. Moderne Web-*Toolkits* und Ausrichtungssysteme
- 14.8.3. Strikter JavaScript-Modus
- 14.8.4. Mehr über Funktionen
- 14.8.5. Asynchrone Funktionen und Versprechen
- 14.8.6. *Closures*
- 14.8.7. Funktionale Programmierung
- 14.8.8. JavaScript OOP

14.9. Web-Benutzbarkeit
- 14.9.1. Einführung in die Benutzerfreundlichkeit
- 14.9.2. Definition von Benutzerfreundlichkeit
- 14.9.3. Bedeutung des nutzerzentrierten Webdesigns
- 14.9.4. Unterschiede zwischen Barrierefreiheit und Benutzerfreundlichkeit
- 14.9.5. Vorteile und Probleme bei der Kombination von Barrierefreiheit und Benutzerfreundlichkeit
- 14.9.6. Vorteile und Schwierigkeiten bei der Umsetzung von nutzbaren Websites
- 14.9.7. Methoden zur Benutzerfreundlichkeit
- 14.9.8. Analyse der Benutzeranforderungen
- 14.9.9. Konzeptionelle Gestaltungsgrundsätze. Benutzerorientiertes Prototyping
- 14.9.10. Leitlinien für die Erstellung von nutzbaren Websites
  - 14.9.10.1. Jakob Nielsens Leitlinien zur Benutzerfreundlichkeit
  - 14.9.10.2. Bruce Tognazzinis Richtlinien zur Benutzerfreundlichkeit
- 14.9.11. Bewertung der Benutzbarkeit

14.10. Zugänglichkeit des Internets
- 14.10.1. Einführung
- 14.10.2. Definition von Barrierefreiheit im Internet
- 14.10.3. Arten von Behinderungen
  - 14.10.3.1. Vorübergehende oder dauerhafte Behinderungen
  - 14.10.3.2. Visuelle Beeinträchtigungen
  - 14.10.3.3. Beeinträchtigungen des Hörvermögens
  - 14.10.3.4. Motorische Behinderungen
  - 14.10.3.5. Neurologische oder kognitive Behinderungen
  - 14.10.3.6. Altersbedingte Schwierigkeiten
  - 14.10.3.7. Umweltbedingte Einschränkungen
  - 14.10.3.8. Hindernisse für den Zugang zum Internet
- 14.10.4. Technische Hilfsmittel und unterstützende Produkte zur Überwindung von Barrieren
  - 14.10.4.1. Hilfsmittel für Blinde
  - 14.10.4.2. Hilfsmittel für Menschen mit Sehschwäche
  - 14.10.4.3. Hilfsmittel für Menschen mit Farbenblindheit
  - 14.10.4.4. Hilfsmittel für Hörgeschädigte

14.10.4.5. Hilfsmittel für Menschen mit motorischen Behinderungen
14.10.4.6. Hilfsmittel für Menschen mit kognitiven und neurologischen Behinderungen
14.10.5. Vorteile und Schwierigkeiten bei der Umsetzung der Barrierefreiheit im Internet
14.10.6. Vorschriften und Normen für die Barrierefreiheit im Internet
14.10.7. Regulierungsstellen für die Barrierefreiheit im Internet
14.10.8. Vergleich von Normen und Standards
14.10.9. Leitlinien für die Einhaltung von Vorschriften und Normen
14.10.9.1. Beschreibung der wichtigsten Leitlinien (Bilder, Links, Videos, usw.)
14.10.9.2. Leitlinien für eine barrierefreie Navigation
14.10.9.2.1. Wahrnehmbarkeit
14.10.9.2.2. Operationalität
14.10.9.2.3. Nachvollziehbarkeit
14.10.9.2.4. Robustheit
14.10.10. Beschreibung des Prozesses der Webzugänglichkeitskonformität
14.10.11. Konformitätsstufen
14.10.12. Konformitätskriterien
14.10.13. Anforderungen an die Konformität
14.10.14. Methodik zur Bewertung der Zugänglichkeit von Websites

Modul 15. Webserver-Berechnungen

15.1. Einführung in die Programmierung im Server: PHP
15.1.1. Grundlagen der Programmierung im Server
15.1.2. Grundlegende PHP-Syntax
15.1.3. Generierung von HTML-Inhalten mit PHP
15.1.4. Entwicklungs- und Testumgebungen: XAMPP
15.2. PHP für Fortgeschrittene
15.2.1. PHP-Kontrollstrukturen
15.2.2. PHP-Funktionen
15.2.3. Management von *Array* in PHP
15.2.4. String-Verarbeitung in PHP
15.2.5. Objektorientierung in PHP
15.3. Datenmodelle
15.3.1. Begriff der Daten. Lebenszyklus der Daten
15.3.2. Datentypen
15.3.2.1. Grundlegend
15.3.2.2. Register
15.3.2.3. Dynamisch
15.4. Das relationale Modell
15.4.1. Beschreibung
15.4.2. Entitäten und Entitätstypen
15.4.3. Datenelemente. Attribute
15.4.4. Beziehungen: Typen, Untertypen, Kardinalität
15.4.5. Schlüssel. Schlüsselarten
15.4.6. Normalisierung. Normale Formen
15.5. Aufbau des logischen Datenmodells
15.5.1. Spezifikation der Tabelle
15.5.2. Definition von Spalten
15.5.3. Wichtige Spezifikation
15.5.4. Umwandlung in Normalformen. Abhängigkeit
15.6. Das physische Datenmodell. Dateien
15.6.1. Beschreibung der Datendateien
15.6.2. Datentypen
15.6.3. Zugriffsmodi
15.6.4. Organisation von Dateien
15.7. Zugriff auf Datenbanken von PHP aus
15.7.1. Einführung in MariaDB
15.7.2. Arbeiten mit einer MariaDB-Datenbank: die SQL-Sprache
15.7.3. Zugriff auf die MariaDB-Datenbank von PHP aus
15.7.4. Einführung in MySql
15.7.5. Arbeiten mit einer MySql-Datenbank: die SQL-Sprache
15.7.6. Zugriff auf die MySql-Datenbank von PHP aus
15.8. Interaktion mit dem Client über PHP
15.8.1. PHP-Formulare
15.8.2. Cookies
15.8.3. Handhabung von Sitzungen

15.9. Architektur von Webanwendungen
- 15.9.1. Das Model-View-Controller-Muster
- 15.9.2. Controller
- 15.9.3. Model
- 15.9.4. Ansicht

15.10. Einführung in Webdienste
- 15.10.1. Einführung in XML
- 15.10.2. Service-orientierte Architekturen (SOA): Webdienste
- 15.10.3. Erstellung von SOAP- und REST-Webdiensten
- 15.10.4. Das SOAP-Protokoll
- 15.10.5. Das REST-Protokoll

Modul 16. Sicherheitsmanagement

16.1. Informationssicherheit
- 16.1.1. Einführung
- 16.1.2. Die Sicherheit von Informationen setzt Vertraulichkeit, Integrität und Verfügbarkeit voraus
- 16.1.3. Sicherheit ist eine wirtschaftliche Frage
- 16.1.4. Sicherheit ist ein Prozess
- 16.1.5. Die Klassifizierung von Informationen
- 16.1.6. Informationssicherheit ist Risikomanagement
- 16.1.7. Sicherheit ist mit Sicherheitskontrollen verbunden
- 16.1.8. Sicherheit ist sowohl physisch als auch logisch
- 16.1.9. Sicherheit betrifft Menschen

16.2. Die Fachkraft für Informationssicherheit
- 16.2.1. Einführung
- 16.2.2. Informationssicherheit als Beruf
- 16.2.3. ISC2-Zertifizierungen
- 16.2.4. Die Norm ISO 27001
- 16.2.5. Bewährte Sicherheitspraktiken im IT-Service-Management
- 16.2.6. Reifegradmodelle für die Informationssicherheit
- 16.2.7. Andere Zertifizierungen, Standards und professionelle Ressourcen

16.3. Zugangskontrolle
- 16.3.1. Einführung
- 16.3.2. Anforderungen an die Zugangskontrolle
- 16.3.3. Authentifizierungsmechanismen
- 16.3.4. Genehmigungsverfahren
- 16.3.5. Zugang zu Buchhaltung und Rechnungsprüfung
- 16.3.6. Triple-A-Technologien

16.4. Programme, Verfahren und Richtlinien zur Informationssicherheit
- 16.4.1. Einführung
- 16.4.2. Programme für das Sicherheitsmanagement
- 16.4.3. Risikomanagement
- 16.4.4. Gestaltung der Sicherheitspolitik

16.5. Pläne zur Aufrechterhaltung des Geschäftsbetriebs
- 16.5.1. Einführung in BCPs
- 16.5.2. Phase I und II
- 16.5.3. Phase III und IV
- 16.5.4. Aufrechterhaltung der BCP

16.6. Verfahren für den korrekten Schutz des Unternehmens
- 16.6.1. DMZ-Netzwerke
- 16.6.2. Systeme zur Erkennung von Eindringlingen
- 16.6.3. Zugriffskontrolllisten
- 16.6.4. Vom Angreifer lernen: *Honeypot*

16.7. Sicherheitsarchitektur. Prävention
- 16.7.1. Überblick. Aktivitäten und Schichtenmodell
- 16.7.2. Perimeter-Verteidigung (Firewalls, WAFs, IPS, usw.)
- 16.7.3. Endpunktschutz (Geräte, Server und Dienste)

16.8. Sicherheitsarchitektur. Erkennung
- 16.8.1. Überblick über Erkennung und Überwachung
- 16.8.2. Logs, verschlüsselte Verkehrsunterbrechung, Aufzeichnung und *Siems*
- 16.8.3. Warnungen und Informationen

16.9. Sicherheitsarchitektur. Reaktion
- 16.9.1. Reaktion. Produkte, Dienstleistungen und Ressourcen
- 16.9.2. Management von Zwischenfällen
- 16.9.3. CERTS und CSIRTs

16.10. Sicherheitsarchitektur. Erholung
- 16.10.1. Ausfallsicherheit, Konzepte, Geschäftsanforderungen und Vorschriften
- 16.10.2. Widerstandsfähige IT-Lösungen
- 16.10.3. Krisenmanagement und Governance

Modul 17. Software-Sicherheit

17.1. Software-Sicherheitsprobleme
- 17.1.1. Einführung in das Problem der Softwaresicherheit
- 17.1.2. Schwachstellen und ihre Klassifizierung
- 17.1.3. Sichere Software-Eigenschaften
- 17.1.4. Referenzen

17.2. Grundsätze des Software-Sicherheitsdesigns
- 17.2.1. Einführung
- 17.2.2. Grundsätze des Software-Sicherheitsdesigns
- 17.2.3. Arten von S-SDLC
- 17.2.4. Software-Sicherheit in den S-SDLC-Phasen
- 17.2.5. Methodologien und Normen
- 17.2.6. Referenzen

17.3. Sicherheit im Software-Lebenszyklus in der Anforderungs- und Entwurfsphase
- 17.3.1. Einführung
- 17.3.2. Angriffsmodellierung
- 17.3.3. Missbrauchsfälle
- 17.3.4. Entwicklung von Sicherheitsanforderungen
- 17.3.5. Risikoanalyse. Architektonisch
- 17.3.6. Entwurfsmuster
- 17.3.7. Referenzen

17.4. Sicherheit im Software-Lebenszyklus in den Phasen Kodierung, Test und Betrieb
- 17.4.1. Einführung
- 17.4.2. Risikobasierte Sicherheitsprüfungen
- 17.4.3. Code-Überprüfung
- 17.4.4. Penetrationstests
- 17.4.5. Sicherheitsmaßnahmen
- 17.4.6. Externe Überprüfung
- 17.4.7. Referenzen

17.5. Sichere Kodierungsanwendungen I
- 17.5.1. Einführung
- 17.5.2. Sichere Kodierungspraktiken
- 17.5.3. Eingabeverarbeitung und Validierung
- 17.5.4. Speicherüberlauf
- 17.5.5. Referenzen

17.6. Sichere Kodierungsanwendungen II
- 17.6.1. Einführung
- 17.6.2. *Integers overflows*, Abschneidefehler und Probleme mit Typkonvertierungen zwischen ganzen Zahlen
- 17.6.3. Fehler und Ausnahmen
- 17.6.4. Datenschutz und Vertraulichkeit
- 17.6.5. Privilegierte Programme
- 17.6.6. Referenzen

17.7. Sicherheit in der Entwicklung und in der Cloud
- 17.7.1. Entwicklungssicherheit; Methodik und Praxis
- 17.7.2. PaaS, IaaS,CaaS y SaaS Modelle
- 17.7.3. Sicherheit in der Cloud und für Cloud-Dienste

17.8. Verschlüsselung
- 17.8.1. Grundlagen der Kryptologie
- 17.8.2. Symmetrische und asymmetrische Verschlüsselung
- 17.8.3. Verschlüsselung im Ruhezustand und bei der Übermittlung

17.9. Orchestrierung und Automatisierung der Sicherheit (SOAR)
- 17.9.1. Komplexität der manuellen Verarbeitung: Notwendigkeit der Automatisierung von Aufgaben
- 17.9.2. Produkte und Dienstleistungen
- 17.9.3. SOAR-Architektur

17.10. Sicherheit der Telearbeit
- 17.10.1. Bedarf und Szenarien
- 17.10.2. Produkte und Dienstleistungen
- 17.10.3. Sicherheit der Telearbeit

Modul 18. Verwaltung des Webservers

18.1. Einführung in Webserver
- 18.1.1. Was ist ein Webserver?
- 18.1.2. Architektur und Funktionsweise eines Webservers
- 18.1.3. Ressourcen und Inhalte auf einem Webserver
- 18.1.4. Anwendungsserver
- 18.1.5. Proxy-Server
- 18.1.6. Die wichtigsten Webserver auf dem Markt
- 18.1.7. Statistiken zur Webserver-Nutzung
- 18.1.8. Sicherheit des Webservers
- 18.1.9. Lastausgleich bei Webservern
- 18.1.10. Referenzen

18.2. Umgang mit dem HTTP-Protokoll
- 18.2.1. Betrieb und Struktur
- 18.2.2. Beschreibung der Abfragemethoden oder *Request Methods*
- 18.2.3. Status-Codes
- 18.2.4. Kopfzeilen
- 18.2.5. Codierung des Inhalts. Code-Seiten

  Durchführung von HTTP-Anfragen im Internet mit Hilfe eines Proxys, *Livehttpheaders* oder einer ähnlichen Methode, Analyse des verwendeten Protokolls

18.3. Beschreibung von verteilten Architekturen auf mehreren Servern
- 18.3.1. 3-Schichten-Modell
- 18.3.2. Fehlertoleranz
- 18.3.3. Lastverteilung
- 18.3.4. Sitzungsstatus-Speicher.
- 18.3.5. Cache-Speicher

18.4. Internet-Informationsdienste (IIS)
- 18.4.1. Was ist IIS?
- 18.4.2. Geschichte und Entwicklung des IIS
- 18.4.3. Die wichtigsten Vorteile und Funktionen von IIS7 und darüber hinaus
- 18.4.4. IIS7 und neuere Architektur

18.5. IIS-Installation, -Verwaltung und -Konfiguration
- 18.5.1. Präambel
- 18.5.2. Installation der Internet-Informationsdienste (IIS)
- 18.5.3. IIS-Verwaltungstools
- 18.5.4. Erstellen, Konfigurieren und Verwalten von Websites
- 18.5.5. Installieren und Verwalten von IIS-Erweiterungen

18.6. Erweiterte Sicherheit im IIS
- 18.6.1. Präambel
- 18.6.2. IIS-Authentifizierung, Autorisierung und Zugriffskontrolle
- 18.6.3. Konfigurieren einer sicheren Website auf IIS mit SSL
- 18.6.4. In IIS 18.x implementierte Sicherheitsrichtlinien

18.7. Einführung in Apache
- 18.7.1. Was ist Apache?
- 18.7.2. Die wichtigsten Vorteile von Apache
- 18.7.3. Hauptmerkmale von Apache
- 18.7.4. Architektur

18.8. Apache-Installation und -Konfiguration
- 18.8.1. Apache-Erstinstallation
- 18.8.2. Apache-Konfiguration

18.9. Installieren und Konfigurieren der verschiedenen Apache-Module
- 18.9.1. Installation von Apache-Modulen
- 18.9.2. Arten von Modulen
- 18.9.3. Sichere Apache-Konfiguration

18.10. Erweiterte Sicherheit
- 18.10.1. Authentifizierung, Autorisierung und Zugangskontrolle
- 18.10.2. Authentifizierungsmethoden
- 18.10.3. Sichere Apache-Konfiguration mit SSL

Modul 19. Sicherheitsaudit

19.1. Einführung in Informationssysteme und deren Prüfung
- 19.1.1. Einführung in Informationssysteme und die Rolle der IT-Prüfung
- 19.1.2. Definitionen von IT-Audit und interner IT-Kontrolle
- 19.1.3. Funktionen und Ziele der IT-Prüfung
- 19.1.4. Unterschiede zwischen interner Kontrolle und IT-Audit

19.2. Interne Kontrollen von Informationssystemen
- 19.2.1. Funktionsschema eines Datenverarbeitungszentrums
- 19.2.2. Klassifizierung der Kontrollen von Informationssystemen
- 19.2.3. Die Goldene Regel

19.3. Der Prozess und die Phasen der Prüfung von Informationssystemen
- 19.3.1. Risikobewertung (RRA) und andere IT-Prüfungsmethoden
- 19.3.2. Durchführung einer Prüfung der Informationssysteme. Prüfungsphasen
- 19.3.3. Grundlegende Fähigkeiten des Wirtschaftsprüfers für Informationssysteme

19.4. Technische Sicherheitsüberprüfung von Systemen und Netzen
- 19.4.1. Technische Sicherheitsaudits. Penetrationstests. Vorläufige Konzepte
- 19.4.2. Audits der Systemsicherheit. Hilfsmittel
- 19.4.3. Audits der Netzwerksicherheit. Hilfsmittel

19.5. Technische Prüfung der Sicherheit von Internet und mobilen Geräten
- 19.5.1. Internet-Sicherheitsaudit. Hilfsmittel
- 19.5.2. Prüfung der Sicherheit von mobilen Geräten. Hilfsmittel
- 19.5.3. Anhang 1. Aufbau des Kurzberichts und des technischen Berichts
- 19.5.4. Anhang 2. Inventar der Werkzeuge
- 19.5.5. Anhang 3. Methoden

19.6. Managementsystem für die Informationssicherheit
- 19.6.1. IS-Sicherheit: Eigenschaften und Einflussfaktoren
- 19.6.2. Unternehmensrisiken und Risikomanagement: Implementierung von Kontrollen
- 19.6.3. Informationssicherheitsmanagementsystem (ISMS): Konzept und kritische Erfolgsfaktoren
- 19.6.4. ISMS - PDCA-Modell
- 19.6.5. ISMS ISO-IEC 27001: organisatorischer Kontext
- 19.6.6. Organisatorischer Kontext
- 19.6.7. Führungsrolle
- 19.6.8. Planung
- 19.6.9. Support
- 19.6.10. Operation
- 19.6.11. Leistungsbewertung
- 19.6.12. Verbesserung
- 19.6.13. Anhang zu ISO 27001/ISO-IEC 27002: Zielsetzungen und Kontrollen
- 19.6.14. ISMS-Audit

19.7. Durchführung des Audits
- 19.7.1. Verfahren
- 19.7.2. Techniken

19.8. Rückverfolgbarkeit
- 19.8.1. Methoden
- 19.8.2. Analyse

19.9. Gewahrsam
- 19.9.1. Techniken
- 19.9.2. Ergebnisse

19.10. Berichterstattung und Präsentation von Beweisen
- 19.10.1. Arten von Berichten
- 19.10.2. Analyse der Daten
- 19.10.3. Vorlage von Beweismitteln

Modul 20. Sicherheit bei Online-Anwendungen

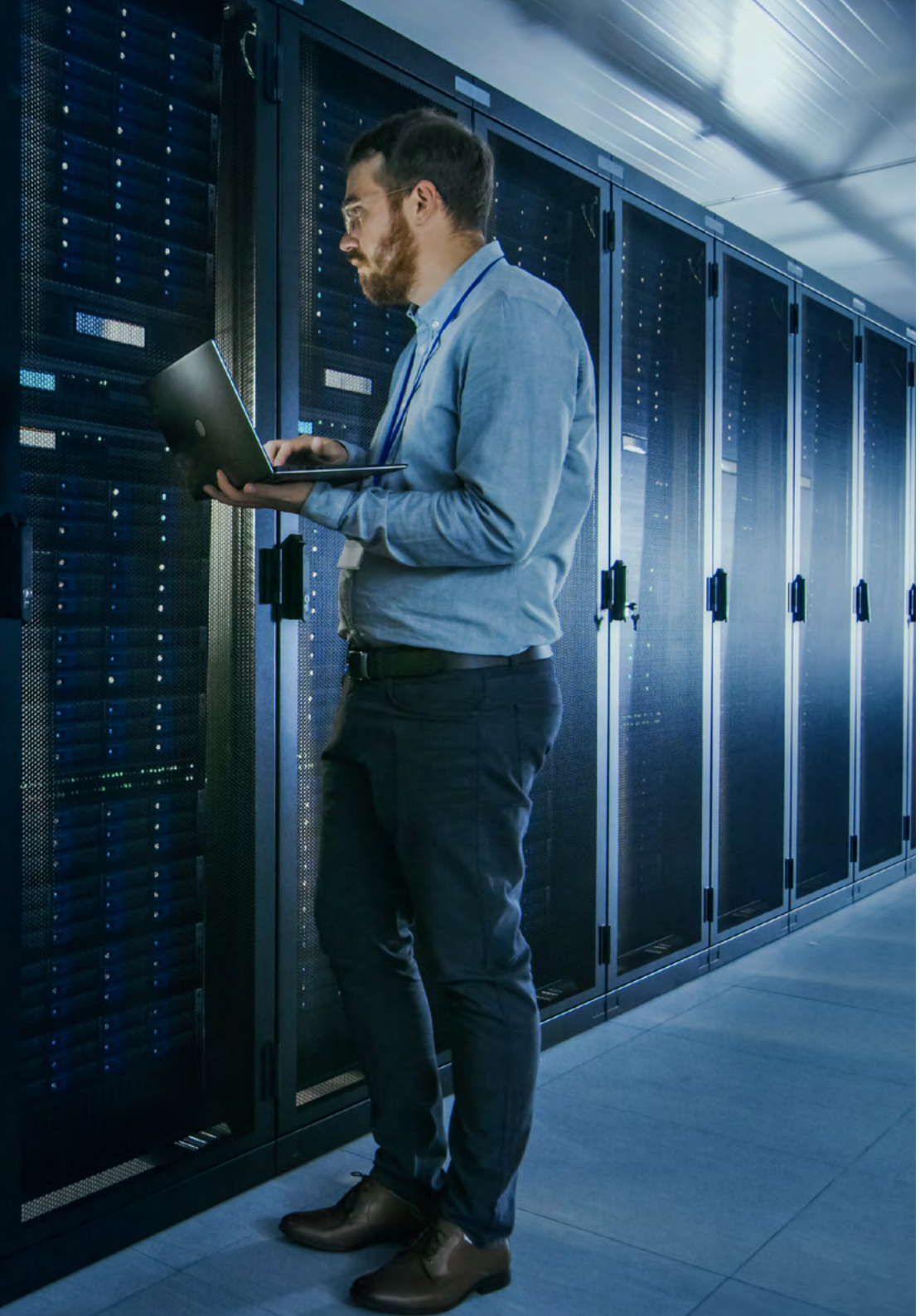
20.1. Schwachstellen und Sicherheitsprobleme in Online-Anwendungen
- 20.1.1. Einführung in die Sicherheit von Online-Anwendungen
- 20.1.2. Sicherheitsschwachstellen beim Entwurf von Webanwendungen
- 20.1.3. Sicherheitsschwachstellen Implementierung von Webanwendungen
- 20.1.4. Sicherheitsschwachstellen bei der Bereitstellung von Webanwendungen
- 20.1.5. Offizielle Listen von Sicherheitslücken

20.2. Richtlinien und Standards für die Sicherheit von Online-Anwendungen
- 20.2.1. Säulen der Sicherheit von Online-Anwendungen
- 20.2.2. Sicherheitspolitik
- 20.2.3. Managementsystem für die Informationssicherheit
- 20.2.4. Sicherer Lebenszyklus der Software Entwicklung
- 20.2.5. Standards für die Anwendungssicherheit

20.3. Sicherheit beim Entwurf von Webanwendungen
- 20.3.1. Einführung in die Sicherheit von Webanwendungen
- 20.3.2. Sicherheit beim Entwurf von Webanwendungen

20.4. Prüfung der Online-Sicherheit von Webanwendungen
- 20.4.1. Analyse und Prüfung der Sicherheit von Webanwendungen
- 20.4.2. Sicherheit bei der Bereitstellung und Produktion von Webanwendungen

20.5. Sicherheit von Webdiensten
- 20.5.1. Einführung in die Sicherheit von Webdiensten
- 20.5.2. Sicherheitsfunktionen und -technologien für Webdienste



20.6. Prüfung der Online-Sicherheit und des Schutzes von Webdiensten
- 20.6.1. Bewertung der Sicherheit von Webdiensten
- 20.6.2. Online-Schutz. *Firewalls* und *Gateways* XML

20.7. Ethisches*Hacking* Malware und *Forensik*
- 20.7.1. Ethisches *Hacking*
- 20.7.2. Malware-Analyse
- 20.7.3. Forensische Analyse

20.8. Bewährte Verfahren zur Gewährleistung der Anwendungssicherheit
- 20.8.1. Handbuch für bewährte Praktiken bei der Entwicklung von Online-Anwendungen
- 20.8.2. Handbuch für bewährte Praktiken bei der Umsetzung von Online-Anwendungen

20.9. Häufige Fehler, die die Anwendungssicherheit untergraben
- 20.9.1. Häufige Entwicklungsfehler
- 20.9.2. Häufige Fehler beim Hosting
- 20.9.3. Häufige Fehler in der Produktion

Mit dem Erwerb des weiterbildenden Masterstudiengangs machen Sie nicht nur einen entscheidenden Schritt zur Erweiterung Ihrer Kenntnisse in der technischen Informatik mit Schwerpunkt Software, sondern auch in Richtung einer erfolgreichen beruflichen Laufbahn"

06

Methodik

Dieses Fortbildungsprogramm bietet eine andere Art des Lernens. Unsere Methodik wird durch eine zyklische Lernmethode entwickelt: ***das Relearning***.

Dieses Lehrsystem wird z. B. an den renommiertesten medizinischen Fakultäten der Welt angewandt und wird von wichtigen Publikationen wie dem ***New England Journal of Medicine*** als eines der effektivsten angesehen.

Entdecken Sie Relearning, ein System, das das herkömmliche lineare Lernen aufgibt und Sie durch zyklische Lehrsysteme führt: eine Art des Lernens, die sich als äußerst effektiv erwiesen hat, insbesondere in Fächern, die Auswendiglernen erfordern"

Fallstudie zur Kontextualisierung aller Inhalte

Unser Programm bietet eine revolutionäre Methode zur Entwicklung von Fähigkeiten und Kenntnissen. Unser Ziel ist es, Kompetenzen in einem sich wandelnden, wettbewerbsorientierten und sehr anspruchsvollen Umfeld zu stärken.

Mit TECH werden Sie eine Art des Lernens erleben, die die Grundlagen der traditionellen Universitäten in der ganzen Welt verschiebt"

Sie werden Zugang zu einem Lernsystem haben, das auf Wiederholung basiert, mit natürlichem und progressivem Unterricht während des gesamten Lehrplans.

Die Studenten lernen durch gemeinschaftliche Aktivitäten und reale Fälle die Lösung komplexer Situationen in realen Geschäftsumgebungen.

Eine innovative und andersartige Lernmethode

Dieses TECH-Programm ist ein von Grund auf neu entwickeltes, intensives Lehrprogramm, das die anspruchsvollsten Herausforderungen und Entscheidungen in diesem Bereich sowohl auf nationaler als auch auf internationaler Ebene vorsieht. Dank dieser Methodik wird das persönliche und berufliche Wachstum gefördert und ein entscheidender Schritt in Richtung Erfolg gemacht. Die Fallmethode, die Technik, die diesem Inhalt zugrunde liegt, gewährleistet, dass die aktuellste wirtschaftliche, soziale und berufliche Realität berücksichtigt wird.

> *Unser Programm bereitet Sie darauf vor, sich neuen Herausforderungen in einem unsicheren Umfeld zu stellen und in Ihrer Karriere erfolgreich zu sein"*

Die Fallmethode ist das am weitesten verbreitete Lernsystem an den besten Informatikschulen der Welt, seit es sie gibt. Die Fallmethode wurde 1912 entwickelt, damit die Jurastudenten das Recht nicht nur anhand theoretischer Inhalte erlernen, sondern ihnen reale, komplexe Situationen vorlegen, damit sie fundierte Entscheidungen treffen und Werturteile darüber fällen können, wie diese zu lösen sind. Sie wurde 1924 als Standardlehrmethode in Harvard eingeführt.

Was sollte eine Fachkraft in einer bestimmten Situation tun? Mit dieser Frage konfrontieren wir Sie in der Fallmethode, einer handlungsorientierten Lernmethode. Während des gesamten Kurses werden die Studierenden mit mehreren realen Fällen konfrontiert. Sie müssen Ihr gesamtes Wissen integrieren, recherchieren, argumentieren und Ihre Ideen und Entscheidungen verteidigen.

Relearning Methodik

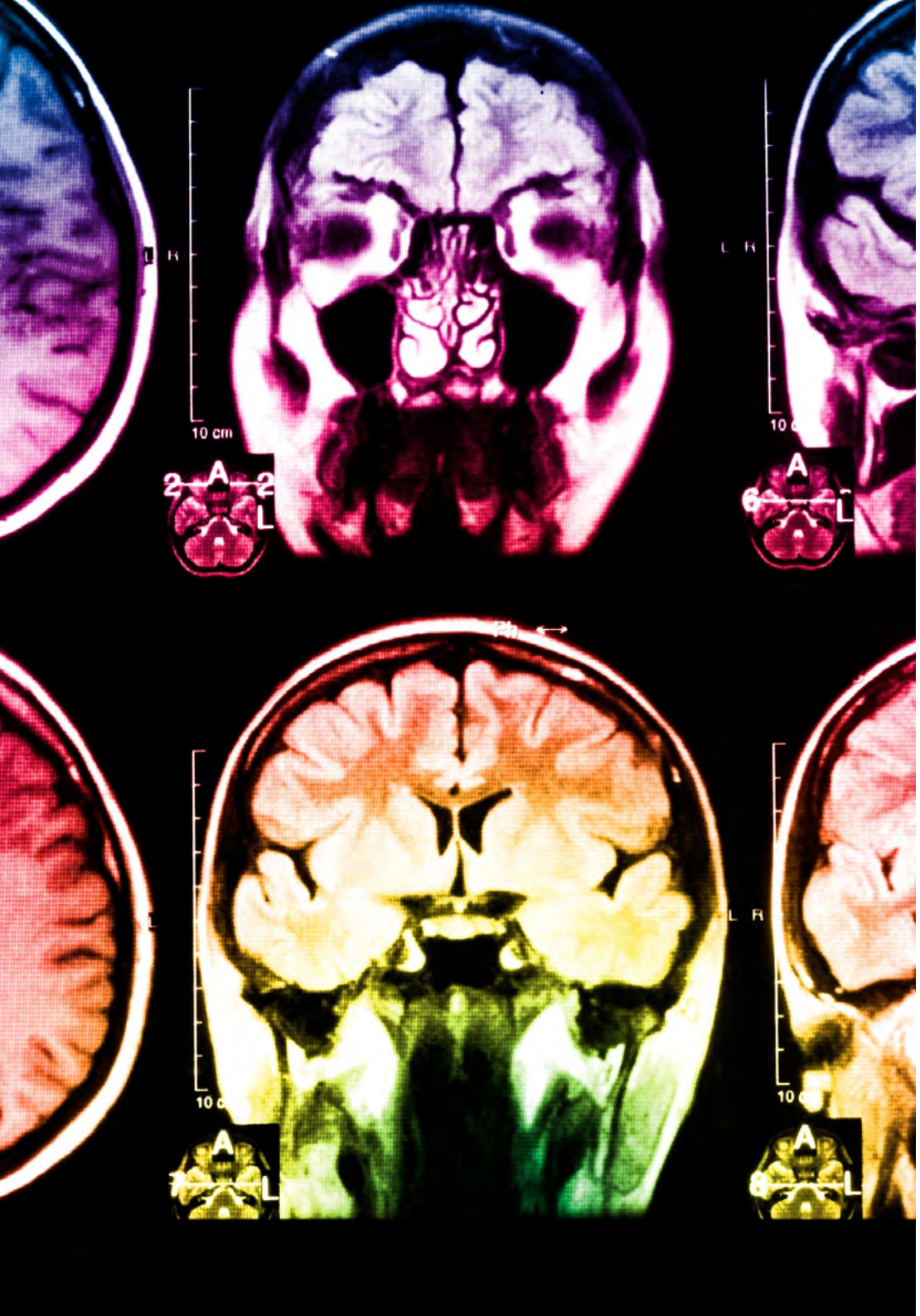
TECH kombiniert die Methodik der Fallstudien effektiv mit einem 100%igen Online-Lernsystem, das auf Wiederholung basiert und in jeder Lektion verschiedene didaktische Elemente kombiniert.

Wir ergänzen die Fallstudie mit der besten 100%igen Online-Lehrmethode: Relearning.

Im Jahr 2019 erzielten wir die besten Lernergebnisse aller spanischsprachigen Online-Universitäten der Welt.

Bei TECH lernen Sie mit einer hochmodernen Methodik, die darauf ausgerichtet ist, die Führungskräfte der Zukunft auszubilden. Diese Methode, die an der Spitze der weltweiten Pädagogik steht, wird Relearning genannt.

Unsere Universität ist die einzige in der spanischsprachigen Welt, die für die Anwendung dieser erfolgreichen Methode zugelassen ist. Im Jahr 2019 ist es uns gelungen, die Gesamtzufriedenheit unserer Studenten (Qualität der Lehre, Qualität der Materialien, Kursstruktur, Ziele...) in Bezug auf die Indikatoren der besten Online-Universität in Spanisch zu verbessern.



In unserem Programm ist das Lernen kein linearer Prozess, sondern erfolgt in einer Spirale (lernen, verlernen, vergessen und neu lernen). Daher wird jedes dieser Elemente konzentrisch kombiniert. Mit dieser Methode wurden mehr als 650.000 Hochschulabsolventen mit beispiellosem Erfolg in so unterschiedlichen Bereichen wie Biochemie, Genetik, Chirurgie, internationales Recht, Managementfähigkeiten, Sportwissenschaft, Philosophie, Recht, Ingenieurwesen, Journalismus, Geschichte, Finanzmärkte und -Instrumente ausgebildet. Dies alles in einem sehr anspruchsvollen Umfeld mit einer Studentenschaft mit hohem sozioökonomischem Profil und einem Durchschnittsalter von 43,5 Jahren.

Das Relearning ermöglicht es Ihnen, mit weniger Aufwand und mehr Leistung zu lernen, sich mehr auf Ihr Fachgebiet einzulassen, einen kritischen Geist zu entwickeln, Argumente zu verteidigen und Meinungen zu kontrastieren: eine direkte Gleichung zum Erfolg.

Nach den neuesten wissenschaftlichen Erkenntnissen der Neurowissenschaften wissen wir nicht nur, wie wir Informationen, Ideen, Bilder und Erinnerungen organisieren, sondern auch, dass der Ort und der Kontext, in dem wir etwas gelernt haben, von grundlegender Bedeutung dafür sind, dass wir uns daran erinnern und es im Hippocampus speichern können, um es in unserem Langzeitgedächtnis zu behalten.

Auf diese Weise sind die verschiedenen Elemente unseres Programms im Rahmen des so genannten neurokognitiven kontextabhängigen E-Learnings mit dem Kontext verbunden, in dem der Teilnehmer seine berufliche Praxis entwickelt.

Dieses Programm bietet die besten Lehrmaterialien, die sorgfältig für Fachleute aufbereitet sind:

Studienmaterial

Alle didaktischen Inhalte werden von den Fachleuten, die den Kurs unterrichten werden, speziell für den Kurs erstellt, so dass die didaktische Entwicklung wirklich spezifisch und konkret ist.

Diese Inhalte werden dann auf das audiovisuelle Format angewendet, um die TECH-Online-Arbeitsmethode zu schaffen. Und das alles mit den neuesten Techniken, die dem Studenten qualitativ hochwertige Stücke aus jedem einzelnen Material zur Verfügung stellen.

Meisterklassen

Die Nützlichkeit der Expertenbeobachtung ist wissenschaftlich belegt.

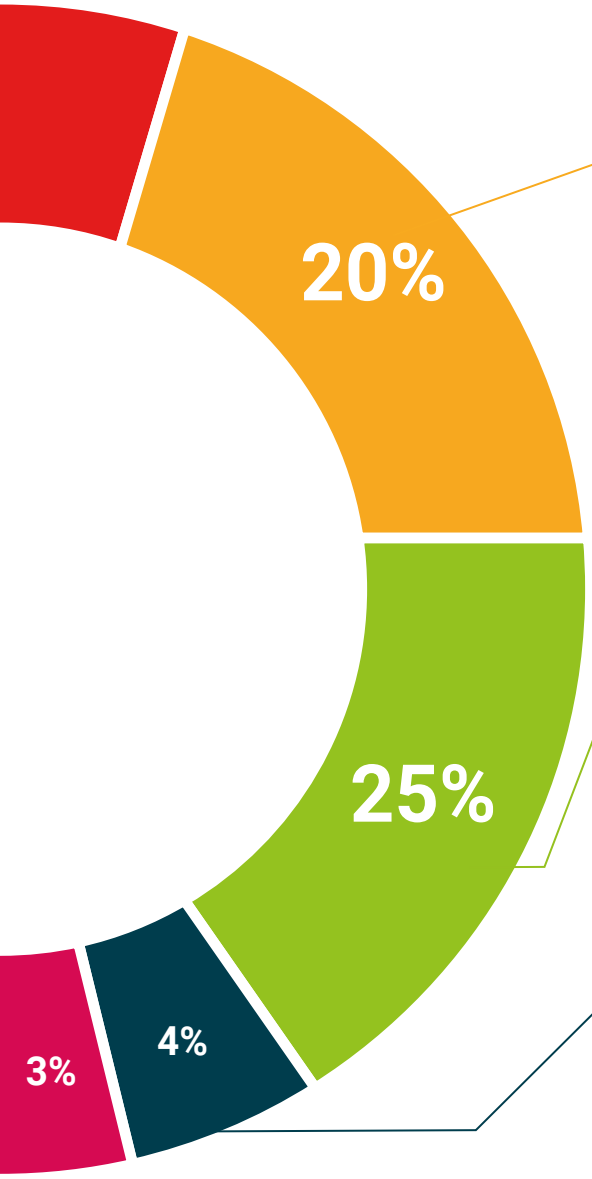
Das sogenannte Learning from an Expert baut Wissen und Gedächtnis auf und schafft Vertrauen für zukünftige schwierige Entscheidungen.

Fertigkeiten und Kompetenzen Praktiken

Sie werden Aktivitäten durchführen, um spezifische Kompetenzen und Fertigkeiten in jedem Fachbereich zu entwickeln. Praktiken und Dynamiken zum Erwerb und zur Entwicklung der Fähigkeiten und Fertigkeiten, die ein Spezialist im Rahmen der Globalisierung, in der wir leben, entwickeln muss.

Weitere Lektüren

Aktuelle Artikel, Konsensdokumente und internationale Leitfäden, u.a. In der virtuellen Bibliothek von TECH haben die Studenten Zugang zu allem, was sie für ihre Ausbildung benötigen.



Fallstudien

Sie werden eine Auswahl der besten Fallstudien vervollständigen, die speziell für diese Qualifizierung ausgewählt wurden. Die Fälle werden von den besten Spezialisten der internationalen Szene präsentiert, analysiert und betreut.

Interaktive Zusammenfassungen

Das TECH-Team präsentiert die Inhalte auf attraktive und dynamische Weise in multimedialen Pillen, die Audios, Videos, Bilder, Diagramme und konzeptionelle Karten enthalten, um das Wissen zu vertiefen.

Dieses einzigartige Bildungssystem für die Präsentation multimedialer Inhalte wurde von Microsoft als "europäische Erfolgsgeschichte" ausgezeichnet.

Prüfung und Nachprüfung

Die Kenntnisse der Studenten werden während des gesamten Programms regelmäßig durch Bewertungs- und Selbsteinschätzungsaktivitäten und -übungen beurteilt und neu bewertet, so dass die Studenten überprüfen können, wie sie ihre Ziele erreichen.

07

Qualifizierung

Der Weiterbildender Masterstudiengang in Softwaretechnik und -garantiert neben der strengsten und aktuellsten Ausbildung auch den Zugang zu einem von der TECH Technologischen Universität ausgestellten Diplom.

Schließen Sie dieses Programm erfolgreich ab und erhalten Sie Ihren Universitätsabschluss ohne lästige Reisen oder Formalitäten”

Dieser **Weiterbildender Masterstudiengang in Softwaretechnik und -qualität** enthält das vollständigste und aktuellste Programm auf dem Markt.

Sobald der Student die Prüfungen bestanden hat, erhält er/sie per Post* mit Empfangsbestätigung das entsprechende Diplom, ausgestellt von der **TECH Technologischen Universität.**

Das von **TECH Technologische Universität** ausgestellte Diplom drückt die erworbene Qualifikation aus und entspricht den Anforderungen, die in der Regel von Stellenbörsen, Auswahlprüfungen und Berufsbildungsausschüssen verlangt werden.

Titel: **Weiterbildender Masterstudiengang in Softwaretechnik und -qualität**

Anzahl der offiziellen Arbeitsstunden: **3.000 Std.**

tech technologische universität

Verleiht dieses

DIPLOM

an

Herr/Frau _____, mit Ausweis-Nr. _____

Für den erfolgreichen Abschluss und die Akkreditierung des Programms

WEITERBILDENDER MASTERSTUDIENGANG

in

Softwaretechnik und -qualität

Es handelt sich um einen von dieser Universität verliehenen Abschluss, mit einer Dauer von 3.000 Stunden, mit Anfangsdatum tt/mm/jjjj und Enddatum tt/mm/jjjj.

TECH ist eine private Hochschuleinrichtung, die seit dem 28. Juni 2018 vom Ministerium für öffentliche Bildung anerkannt ist.

Zum 17. Juni 2020

Tere Guevara Navarro
Rektorin

einzigartiger Code TECH: AFWOR23S techtitute.com/titol

Weiterbildender Masterstudiengang in Softwaretechnik und -qualität

Allgemeiner Aufbau des Lehrplans

Kurs	Modul	Stunden	Kategorie
1°	Software-Qualität. Technologie-Reifegrad-Entwicklungsstufen	150	OB
1°	Software-Projektentwicklung. Funktionelle und technische Dokumentation	150	OB
1°	Testing der Software. Testautomatisierung	150	OB
1°	Methodologien des Software-	150	OB
1°	Projektmanagements. Waterfall-Methodologie versus agile Methodologie	150	OB
1°	TDD (Test Driven Developement). Testgetriebener Softwareentwurf	150	OB
1°	DevOps. Software-Qualitätsmanagement	150	OB
1°	DevOps und kontinuierliche Integration. Fortgeschrittene praktische Lösungen in der Softwareentwicklung	150	OB
1°	Datenbank-Design (DB). Standardisierung und Leistung. Software-Qualität	150	OB
1°	Entwurf skalierbarer Architekturen. Architektur im Lebenszyklus der Software	150	OB
1°	ISO, IEC 9126 Qualitätskriterien. Metriken zur Software-Qualität	150	OB
1°	Methodik, Entwicklung und Qualität im Software Engineering	150	OB
1°	Software-Projektmanagement	150	OB
1°	Plattformen für die Software-Entwicklung	150	OB
1°	Web-Client-Computing	150	OB
1°	Webserver-Berechnungen	150	OB
1°	Sicherheitsmanagement	150	OB
1°	Software-Sicherheit	150	OB
1°	Verwaltung des Webservers	150	OB
1°	Sicherheitsaudit	150	OB
1°	Sicherheit bei Online-Anwendungen	150	OB

Tere Guevara Navarro
Rektorin

tech technologische universität

*Haager Apostille. Für den Fall, dass der Student die Haager Apostille für sein Papierdiplom beantragt, wird TECH EDUCATION die notwendigen Vorkehrungen treffen, um diese gegen eine zusätzliche Gebühr zu beschaffen.

tech technologische universität

Weiterbildender Masterstudiengang

Softwaretechnik und -qualität

- Modalität: **online**
- Dauer: **2 Jahre**
- Qualifizierung: **TECH Technologische Universität**
- Aufwand: **16 Std./Woche**
- Zeitplan: **in Ihrem eigenen Tempo**
- Prüfungen: **online**

Weiterbildender Masterstudiengang
Softwaretechnik und -qualität

tech technologische universität