Professional Master's Degree

Software Quality

tech technological university

tech technological university

Professional Master's Degree
Software Quality

- Modality: **online**
- Duration: **12 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Website: **www.techtitute.com/in/information-technology/professional-master-degree/master-software-quality**

Index

01

Introduction

The rapid growth of the industry and the current market demands have led to a high level of technical debt in software projects. Due to the imperious need to reflect quick answers to the client's or company's requirements, without evaluating or specifying the details of the system's quality. This is where the need to take into account the scalability of the project throughout its life cycle is reflected, which requires IT expertise focused on quality from a top-down approach. This program develops the criteria, tasks and advanced methodologies to understand the relevance of a work oriented to the need to implement quality policies in Software Factories. Students will be able to study completely online for 12 months, following the methodology implemented by the world's largest online university.

Specialize in Software Quality from a technical and management perspective; graduate in 12 months, and make a difference in your professional environment"

The concept of Technical Debt currently being applied by a large number of corporations and administrations with their suppliers reflects the improvised way in which projects have been developed. Generating a new implicit cost by having to redo a project for having adopted a quick and easy solution as opposed to what should be a scalable approach in the evolution of the project.

For some years now, projects have been developed very quickly, with the aim of closing them with the client based on price and deadline criteria, instead of focusing on quality. Now, those decisions are taking their toll on many suppliers and customers.

This Professional Master's Degree will enable the IT professional to analyze the underlying criteria in Software Quality, at all levels. Criteria such as database standardization, decoupling between components of an information system, scalable architectures, metrics, documentation, both functional and technical. In addition to methodologies in the management and development of projects and other methods to ensure quality, such as collaborative work techniques, including the so-called Pair Programming, which allows knowledge to reside in the company and not in people.

The vast majority of these types of Master's Degrees are focused on a technology, a language or a tool. This program is unique in the way it makes the professional aware of the importance of software quality, reducing the technical debt of projects with a quality one instead of an approach based on economics and short deadlines; it equips the student with specialized knowledge, so that project budgeting can be justified.

To make this possible, TECH Technological University has assembled a group of experts in the area that will transmit the most up-to-date knowledge and experience. Through a modern virtual campus with theoretical and practical content, distributed in different formats. There will be 10 modules divided into various units and subunits that will make it possible to learn in 12 months, following the Relearning methodology, which facilitates memorization and learning in an agile and efficient way.

This **Professional Master's Degree in Software Quality** contains the most complete and up-to-date program on the market. The most important features include:

- The development of case studies presented by experts in software development
- The graphic, schematic, and practical contents with which they are created, provide scientific and practical information on the disciplines that are essential for professional practice
- Practical exercises where self-assessment can be used to improve learning
- Its special emphasis on innovative methodologies
- Theoretical lessons, questions for experts and individual reflection work
- Content that is accessible from any fixed or portable device with an Internet connection

The Professional Master's Degree in Software Quality analyzes the criteria underlying the subject at all levels. Broaden your expertise. Enroll now"

Develop the criteria, tasks and advanced methodologies to understand the relevance of quality-oriented work, and provide effective solutions to your company or client "

A program focused on raising awareness of the importance of software quality and the need to implement quality policies in software factories.

Learn in a practical and flexible way. Sharing your day to day life with this 100% online program exclusive to TECH Technological University.

The program's teaching staff includes professionals from the sector who contribute their work experience to this training program, as well as renowned specialists from leading societies and prestigious universities.

The multimedia content, developed with the latest educational technology, will provide the professional with situated and contextual learning, i.e., a simulated environment that will provide immersive training programmed to train in real situations.

This program is designed around Problem-Based Learning, whereby the professional must try to solve the different professional practice situations that arise during the academic year. This will be done with the help of an innovative system of interactive videos made by renowned experts.

02

Objectives

The Professional Master's Degree in Software Quality provides the student with a clear and specialized vision of the importance of quality in software development processes. As well as the most advanced tools to implement DevOps processes and quality assurance systems. In short, it will provide a broad and specialized theoretical and practical knowledge to understand the development of projects from a modern and efficient perspective.

You will be able to easily access all the contents whenever you want. From your computer or favorite device. Plus, you can download them to view them whenever you want"

General Objectives

- Develop the criteria, tasks and advanced methodologies to understand the relevance of quality-oriented work
- Analyze the key factors in the quality of a software project
- Develop the relevant regulatory aspects
- Implement DevOps and systems processes for Quality Assurance
- Reduce the technical debt of projects with a quality approach rather than an approach based on economics and short deadlines
- Provide the student with the knowledge to be able to measure and quantify the quality of a software project
- Defend the economic proposals of projects on the basis of the quality approach

Specific Objectives

Module 1. Software Quality TRL Development Levels

- Develop in a clear and concise way the elements that encompass software quality
- Apply the models and standards according to system, product and software process
- Delve into the ISO quality standards applied both in general and in specific parts of the system
- Apply the standards according to the scope of the environment
- Examine the TRL maturity levels and adapt them to the different parts of the software project to be dealt with
- Acquire capacity of abstraction to apply one or several criteria of elements and levels of software quality
- Distinguish the cases of application of the standards and maturity levels in a real case simulated project

Module 2. Software Project Development. Functional and technical documentation

- Determine the influence of project management on quality
- Develop the different phases of a project
- Differentiate the quality concepts inherent to functional and technical documentation
- Analyze the requirements gathering phase, the analysis phase, team management and the construction phase
- Establish the different software project management methodologies
- Generate criteria to decide which is the most appropriate methodology according to the type of project

Module 3. SoftwareTesting. Test automation

- Establish the differences between product quality, process quality and quality of use.
- Know the ISO/IEC 15504 standard
- Determine the details of CMMI
- Learn the keys to continuous integration, repositories and the repercussions they have on a software development team
- Establish the relevance of incorporating repositories for software projects. Learn how to create them with TFS
- Assimilate the importance of software scalability in information systems design and development

Module 4. Software Project Management Methodologies Waterfall Methodology vs Agile Methodology

- Determine what the Waterfall Methodology consists of
- Delve into the SCRUM Methodology
- Establish the differences between Waterfall and SCRUM
- Clarify the differences between Waterfall and SCRUM methodologies and how the customer sees it
- Browse the Kanban Board
- Approach a same project with WaterFall and Scrum
- Setting up a hybrid project

Module 5. TDD (Test-Driven Development). Test-Driven Software Design

- Know the practical application of TDD and its possibilities, the future testing of a software project
- Complete proposed real simulation cases, as a continuous learning of this TDD concept.
- Analyze, in the simulation cases, to what extent the tests can succeed or fail, from a constructive point of view
- Determine the alternatives to TDD, making a comparative analysis between them

Module 6. DevOps. Software Quality Management

- Analyze the shortcomings of a traditional process
- Assess the possible solutions and choose the most suitable one
- Understanding business needs and their impact on implementation
- Assess the costs of the improvements to implement
- Develop an evolvable software lifecycle, adapted to real need
- Anticipate possible errors and avoid them from the design process
- Justify the use of different implementation models

Module 7. DevOps and Continuous Integration. Advanced Practical Solutions in Software Development

- Identify the stages of the software development and delivery cycle adapted to particular cases
- Design a software delivery process using continuous integration
- Build and implement continuous integration and deployment based on your previous design
- Establish automatic quality checkpoints on each software delivery
- Maintain an automatic and robust software delivery process
- Adapt future needs to the continuous integration and deployment process
- Analyze and anticipate security vulnerabilities during and after the software delivery process

Module 8. Database (DB) Design. Standardization and performance. Software Quality

- Assess the use of the Entity-Relationship Model for the preliminary design of a database.
- Apply an entity, attribute, key, etc., for the best data integrity
- Assess the dependencies, forms and rules of database normalization
- Specialize in the operation of an OLAP data warehouse system, developing and using both fact and dimension tables
- Determine the key points for database performance
- Complete proposed real-world simulation cases as ongoing learning of database design, normalization, and performance
- Establish in the simulation cases, the options to resolve in the creation of the database from a constructive point of view

Module 9. Scalable Architecture Design Architecture in the Software Life Cycle

- Develop the concept of software architecture and its characteristics
- Determine the different types of scalability in software architecture
- Analyze the different levels that can occur in a web scalability
- Acquire specialized knowledge of the software life cycle concept, stages and models
- Determine the impact of an architecture on the software life cycle, with its advantages, limitations and support tools
- Complete proposed real simulation cases, as a continuous learning of the architecture and life cycle of the software
- Evaluate, in the simulation cases, to what extent it may be feasible or unnecessary to use the software

Module 10. ISO/IEC 9126 Quality Criteria. Software Quality Metrics

- Develop the concept of quality criteria and relevant aspects
- Examine the ISO/IEC 9126 standard, main aspects and indicators
- Analyze the different metrics for a software project to meet the agreed assessments
- Examine the internal and external attributes to be addressed in the quality of a software project
- Distinguish the metrics according to the type of programming (structured, object oriented, layered, etc.)
- Complete real simulation cases, as a continuous learning of quality measurement
- See in the simulation cases to what extent it is feasible or unnecessary, i.e. from a constructive point of view of the authors

Enhance your professional profile with this exclusive program. Obtain your degree in a practical way in 12 months with the methodology that only TECH Technological University can offer you"

03

Skills

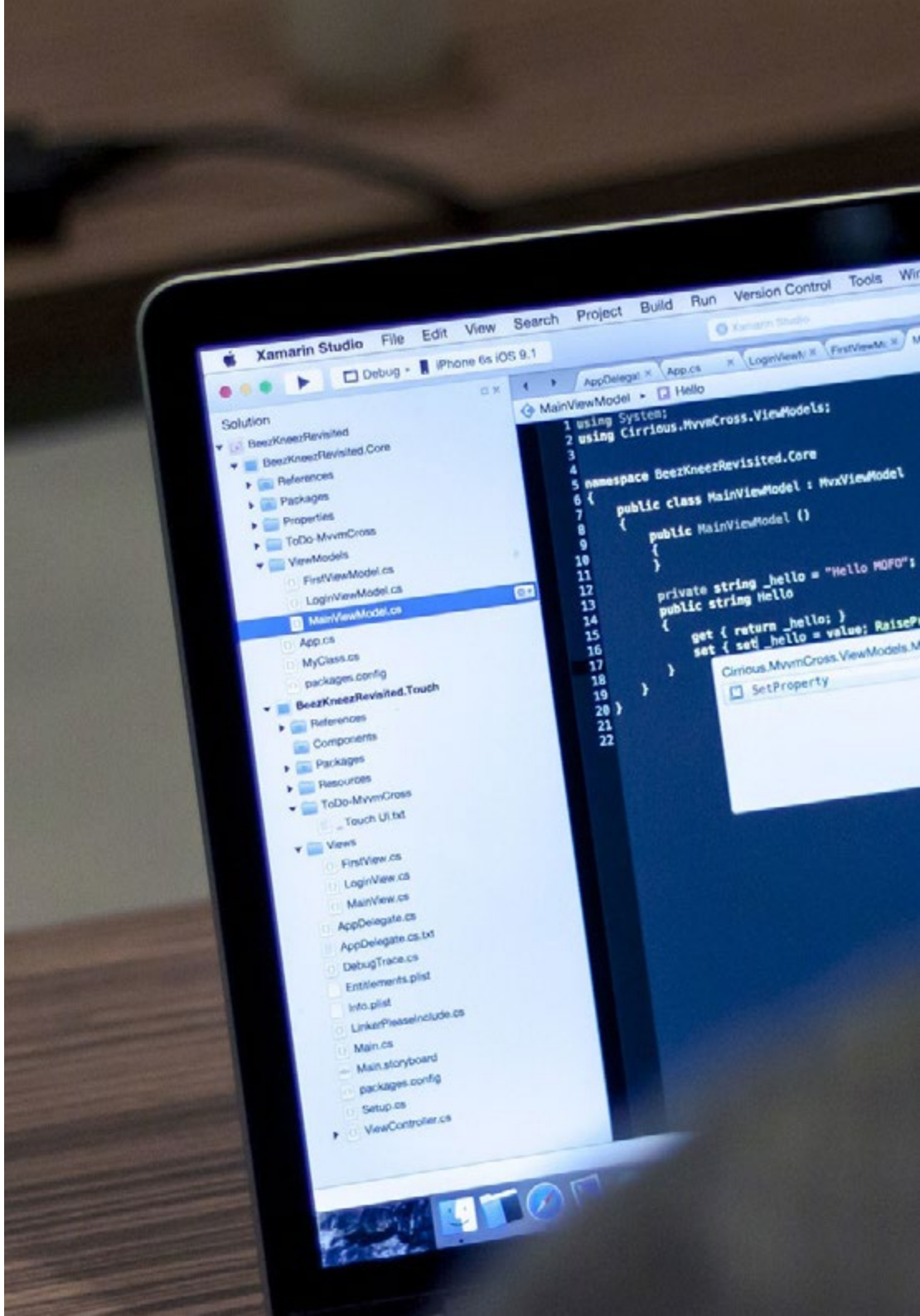
Graduates of this Professional Master's Degree in Software Quality will master the subject from both a technical and management perspective. Allowing them to develop the approach of a project, as well as the execution of the same one and to elaborate a sustainable, effective and quality architecture in the software projects that are presented. To this end, the teaching staff has contributed all their personal experience to the development of a multitude of case studies, which will serve as contextualization and evolution in the face of this "technical debt" that will no longer be present.

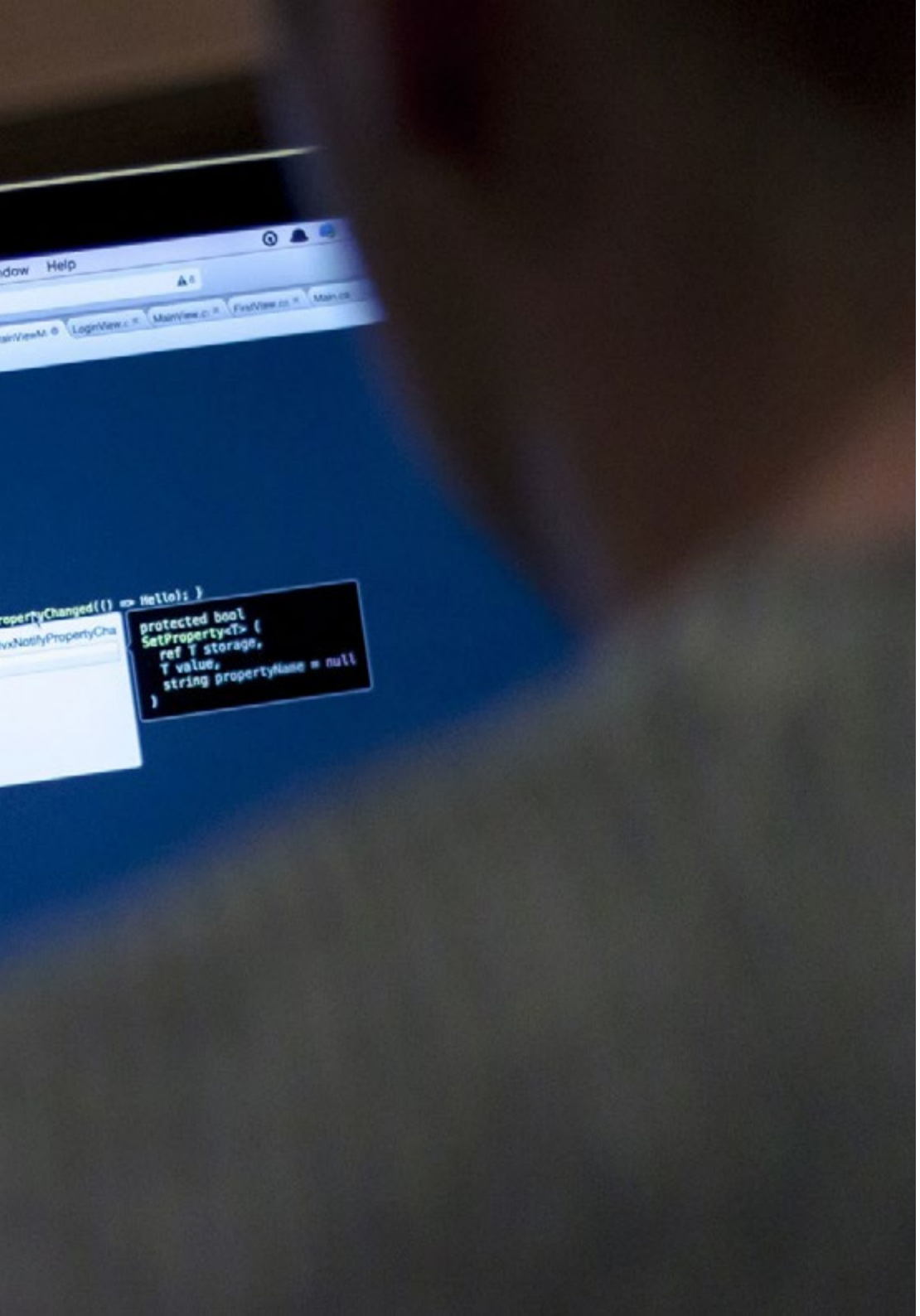
A quality-focused IT professional is a rising asset for software consulting firms and large corporations. Enroll now in this Professional Master's Degree in Software Quality"

General Skills

- Reduce the technical debt of projects with a quality approach rather than an approach based on economics and short deadlines
- Measure and quantify the quality of a software project
- Perform TDD correctly, in order to raise software quality standards
- Justify the budgeting of quality-oriented projects
- Develop quality standards, models and norms
- Examine different technology maturity assessments
- Reduce risk and ensure maintenance and control of subsequent releases
- Master the phases into which a project is broken down

Enhance your skills and discover the endless possibilities for professional growth that open up with this new experience"





Specific Skills

- Assess a software system in terms of the degree of progress in the project process
- Address these points of reliability, metrics and assurance in software projects in a correct and strategic way
- Address the process of deciding on the methodology to be used in the project
- Master the essential normative aspects for the creation of software
- Develop the Testing automatically
- Establish an adequate communication with the clients, understanding the way they perceive the project according to the applied methodology
- Elaborate the list of test requirements
- Perform abstraction, division into more unit tests and eliminate what does not apply to the good performance of the tests of the software project to be performed
- Update the list of test requirements in a measured and correct way
- Adapt DevOps culture to business needs
- Develop the latest practices and tools in continuous integration and deployment
- Refactoring and addressing data management and coordination

04

Course Management

Expert teachers with extensive experience in the field of IT solutions and software development and research, lead this Professional Master's Degree to provide the necessary tools and knowledge to the future graduate focused on quality in software development processes and the most advanced tools to implement DevOps processes and systems for quality assurance. This team of professionals will guide students at all times, in order to achieve the objectives remotely as it is a purely online program that follows the most avant-garde methodology implemented by TECH.

Specialized teachers are committed to providing you with the best content and making your learning process an agile and dynamic experience. Clarifying your doubts and accompanying you all along the way"

Management

Mr. Molina Molina, Jerónimo

- AI Engineer & Software Architect. NASSAT - Internet Satellite in Motion
- Senior Consultant at Hexa Ingenieros. Introducer of Artificial Intelligence (ML and CV).
- Expert in artificial intelligence based solutions in the fields of Computer Vision, ML/DL and NLP. Currently investigating application possibilities of Transformers and Reinforcement Learning in a personal research project.
- University Expert in Business Creation and Development. Bancaixa – FUNDEUN Alicante
- Computer Engineer. University of Alicante
- Master in Artificial Intelligence. Catholic University of Avila
- MBA-Executive. European Business Campus Forum

Professors

Mr. Pi Morell, Oriol

- Hosting and Mail Product Owner. CDMON
- Functional Analyst and Software Engineer in different organizations such as Fihoca, Atmira, CapGemini.
- Teacher of different courses such as BPM in CapGemini, ORACLE Forms CapGemini, Business Processes Atmira.
- Degree in Technical Engineering in Computer Management from the Autonomous University of Madrid
- Master's Degree in Artificial Intelligence
- Master's Degree in Business Administration MBA
- Master's Degree in Information Systems Management Teaching Experience
- Postgraduate, Postgraduate Design Patterns. Open University of Catalonia

Mr. Tenrero Morán, Marcos

- DevOps Engineer – Allot Communications
- Application Lifecycle Management & DevOps– Meta4 Spain. Cegid
- QA Automation Engineer – Meta4 Spain. Cegid
- Graduated in Computer Engineering from Rey Juan Carlos University
- Development of professional applications for Android - Galileo University (Guatemala)
- Cloud Services Development (nodeJs, JavaScript, HTML5) - UPM
- Continuous Integration with Jenkins – Meta4. Cegid
- Web Development with Angular-CLI (4), Ionic and nodeJS. Meta4 - Rey Juan Carlos University

Dr. Peralta Martín-Palomino, Arturo

- CEO and CTO at Prometeus Global Solutions
- CTO at Korporate Technologies
- CTO in AI Shephers GmbH
- Doctorate in Psychology from the University of CastillaLa
- PhD in Economics, Business and Finance from the Camilo José Cela University. Outstanding Award in her PhD
- PhD in Psychology, University of CastillaLa Mancha
- Master's Degree in Advanced Information Technologies from the University of Castilla la Mancha
- Master MBA+E (Master's Degree in Business Administration and Organisational Engineering) from the University of Castilla la Mancha.
- Associate lecturer, teaching undergraduate and master's degrees in Computer Engineering at the University of Castilla la Mancha
- Professor of the Master in Big Data and Data Science at the International University of Valencia
- Lecturer of the Master's Degree in Industry 4.0 and the Master's Degree in Industrial Design and Product Development
- Member of the SMILe Research Group of the University of Castilla la Mancha

Ms. Martínez Cerrato, Yésica

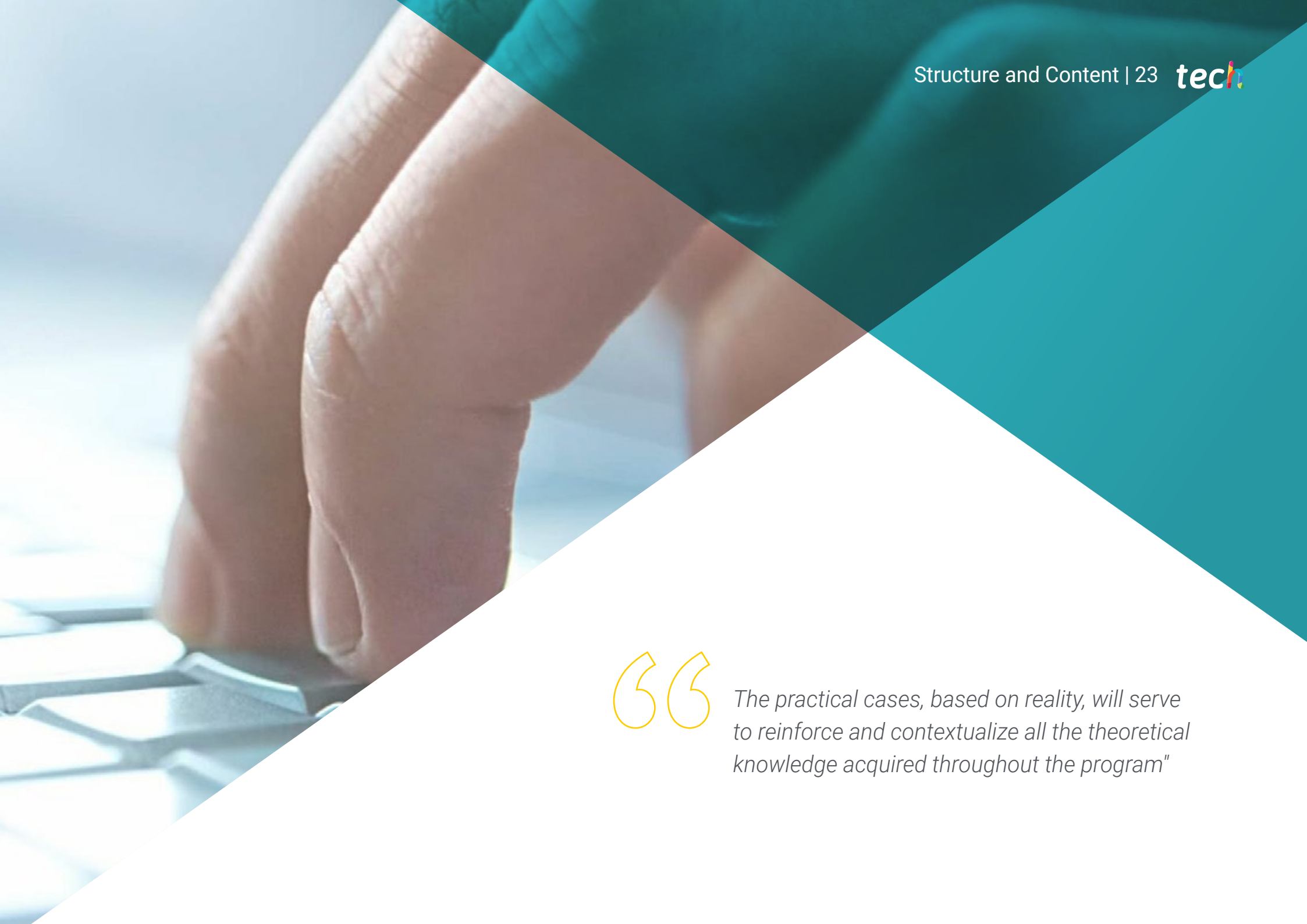
- Electronic Security Product Technician at Securitas Security Spain
- Business Intelligence Analyst at Ricopia Technologies (Alcalá de Henares) Degree in Electronic Communications Engineering at the Polytechnic School, University of Alcalá
- Responsible for training new recruits on commercial management software (CRM, ERP, INTRANET), product and procedures in Ricopia Technologies (Alcalá de Henares)
- Responsible for training new scholarship holders incorporated to the Computer Classrooms at the University of Alcalá
- Project Manager in the area of Key Accounts Integration at Correos and Telégrafos (Madrid)
- Computer Technician-Responsible for computer classrooms OTEC, University of Alcalá (Alcalá de Henares)
- Computer classes teacher at ASALUMA Association (Alcalá de Henares).
- Scholarship for Training as a Computer Technician in OTEC, University of Alcala (Alcalá de Henares)

Our teaching team will provide you with all their knowledge so that you are up to date with the latest information on the subject"

05

Structure and Content

The structure and contents of this Professional Master's Degree have been developed to cover the most important topics for the development of Quality Software. Composed of 10 teaching modules, ranging from software project development, functional and technical documentation, Test-Driven Development and different methodologies, to the implementation of advanced practical solutions with DevOps and continuous integration, all based on achieving software quality. The extensive multimedia content, rigorously selected by the expert faculty, will be of great support to alleviate the teaching load and serve as reference material for future reference.



The practical cases, based on reality, will serve to reinforce and contextualize all the theoretical knowledge acquired throughout the program"

Module 1. Software Quality TRL Development Levels

1.1. Elements that Influence Software Quality (I). Technical Debt
- 1.1 1. Technical Debt. Causes and Consequences
- 1.1.2. Software Quality General Principles
- 1.1.3. Unprincipled and Principled Quality Software
  - 1.1.3.1. Consequences
  - 1.1.3.2. Necessity of Applying Quality Principles in Software
- 1.1.4. Software Quality Typology
- 1.1.5. Quality Software. Specific Features

1.2. Elements that Influence Software Quality (II). Associated Costs
- 1.2.1. Software Quality Influencing Elements
- 1.2.2. Software Quality Misconceptions
- 1.2.3. Software Quality Associated Costs

1.3. Software Quality Models (I). Knowledge Management
- 1.3.1. General Quality Models
  - 1.3.1.1. Total Quality Management
  - 1.3.1.2. European Business Excellence Model (EFQM).
  - 1.3.1.3. Six-Sigma Model
- 1.3.2. Knowledge Management Models
  - 1.3.2.1. Dyba Model
  - 1.3.2.2. SEKS Model
- 1.3.3. Experience Factory and QIP Paradigm
- 1.3.4. Quality in Use Models (25010)

1.4. Software Quality Models (III). Quality in Data, Processes and SEI Models
- 1.4.1. Data Quality Data Model
- 1.4.2. Software Process Modeling
- 1.4.3. *Software & Systems Process Engineering Metamodel Specification* (SPEM)
- 1.4.4. SEI Models
  - 1.4.4.1. CMMI
  - 1.4.4.2. SCAMPI
  - 1.4.4.3. IDEAL

1.5. ISO Software Quality Standards (I). Analysis of the Standards
- 1.5.1. ISO 9000 Standards
  - 1.5.1.1. ISO 9000 Standards
  - 1.5.1.2. ISO Family of Quality Standards (9000)
- 1.5.2. Other ISO Standards Related to Quality
- 1.5.3. Quality Modeling Standards (ISO 2501)
- 1.5.4. Quality Measurement Standards (ISO 2502n)

1.6. ISO Software Quality Standards (II). Requirements and Assessment
- 1.6.1. Standards on Quality Requirements (2503n)
- 1.6.2. Standards on Quality Assessment (2504n)
- 1.6.3. ISO/IEC 24744:2007

1.7. TRL Development Levels (I). Levels 1 to 4
- 1.7.1. TRL Levels
- 1.7.2. Level 1: Basic Principles
- 1.7.3. Level 2: Concept and/or Application
- 1.7.4. Level 3: Critical Analytical Function
- 1.7.5. Level 4: Component Validation in Laboratory Environment 1.8.

1.8. TRL Development Levels (II). Levels 5 to 9
- 1.8.1. Level 5: Component Validation in Relevant Environment
- 1.8.2. Level 6: System/Subsystem Model
- 1.8.3. Level 7: Demonstration in Real Environment
- 1.8.4. Level 8: Complete and Certified System
- 1.8.5. Level 9: Success in Real Environment

1.9. TRL Development Levels. Uses
- 1.9.1. Example of Company with Laboratory Environment
- 1.9.2. Example of an R&D&I Company
- 1.9.3. Example of an Industrial R&D&I Company
- 1.9.4. Example of a Laboratory-Engineering Joint Venture Company

1.10. Software Quality. Key Details
- 1.10.1. Methodological Details
- 1.10.2. Technical Details
- 1.10.3. Software Project Management Details
  - 1.10.3.1. Quality of Computer Systems
  - 1.10.3.2. Software Product Quality
  - 1.10.3.3. Software Process Quality

Module 2. Software Project Development. Functional and Technical Documentation

2.1. Project Management
- 2.1.1. Project Management in Software Quality
- 2.1.2. Project Management Advantages
- 2.1.3. Project Management Typology

2.2. Methodology in Project Management
- 2.2.1. Methodology in Project Management
- 2.2.2. Project Methodologies. Typology
- 2.2.3. Methodologies in Project Management. Application

2.3. Requirements Identification Phase
- 2.3.1. Identification of Project Requirements
- 2.3.2. Management of Project Meetings
- 2.3.3. Documentation to Be Provided

2.4. Models
- 2.4.1. Initial Phase
- 2.4.2. Analysis Phase
- 2.4.3. Construction Phase
- 2.4.4. Testing Phase
- 2.4.5. Delivery

2.5. Data Model to Be Used
- 2.5.1. Determination of the New Data Model
- 2.5.2. Identification of the Data Migration Plan
- 2.5.3. Data Set

2.6. Impact on Other Projects
- 2.6.1. Impact of a Project. Examples:
- 2.6.2. Risk in the Project
- 2.6.3. Risk Management

2.7. MUST of the Project
- 2.7.1. MUST of the Project
- 2.7.2. Identification of Project MUST
- 2.7.3. Identification of the Execution Points for Project Delivery

2.8. The Project Construction Team
- 2.8.1. Roles to be Involved According to the Project
- 2.8.2. Contact with HR for Recruitment
- 2.8.3. Project Deliverables and Schedule

2.9. Technical Aspects of a Software Project
- 2.9.1. Project Architect. Technical Aspects
- 2.9.2. Technical Leaders
- 2.9.3. Construction of the Project Software
- 2.9.4. Code Quality Assessment, Sonar

2.10. Project Deliverables
- 2.10.1. Functional Analysis
- 2.10.2. Data Model
- 2.10.3. State Diagram
- 2.10.4. Technical Documentation

Module 3. Software Testing. Test Automation

3.1. Software Quality Models
- 3.1.1. Product Quality
- 3.1.2. Process Quality
- 3.1.3. Quality of Use

3.2. Process Quality
- 3.2.1. Process Quality
- 3.2.2. Maturity Models
- 3.2.3. ISO 15504 Standards
  - 3.2.3.1. Purposes
  - 3.2.3.2. Context
  - 3.2.3.3. Stages

3.3. ISO/IEC 15504 Standard
- 3.3.1. Process Categories
- 3.3.2. Development Process Example
- 3.3.3. Profile Fragment
- 3.3.4. Stages

3.4. CMMI (Capability Maturity Model Integration)
- 3.4.1. CMMI Capability Maturity Model Integration
- 3.4.2. Models and Areas. Typology
- 3.4.3. Process Areas
- 3.4.4. Capacity Levels
- 3.4.5. Process Management
- 3.4.6. Project Management

3.5. Change and Repository Management
- 3.5.1. Software Change Management
  - 3.5.1.1. Configuration Item. Continuous Integration
  - 3.5.1.2. Lines
  - 3.5.1.3. Flowcharts
  - 3.5.1.4. Branches
- 3.5.2. Repository
  - 3.5.2.1. Version Control
  - 3.5.2.2. Work Team and Use of the Repository
  - 3.5.2.3. Continuous Integration in the Repository

3.6. Team Foundation Server (TFS)
- 3.6.1. Installation and Configuration
- 3.6.2. Creation of a Team Project.
- 3.6.3. Adding Content to Source Code Control
- 3.6.4. TFS on Cloud

3.7. Testing
- 3.7.1. Motivation for Testing
- 3.7.2. Verification Testing
- 3.7.3. Beta Testing
- 3.7.4. Implementation and Maintenance

3.8. Load Testing
- 3.8.1. Load Testing
- 3.8.2. LoadView Testing
- 3.8.3. K6 Cloud Testing
- 3.8.4. Loader Testing

3.9. Unit, Stress and Endurance Tests
- 3.9.1. Reason for Unit Tests
- 3.9.2. Unit Testing Tools
- 3.9.3. Reason for Stress Tests
- 3.9.4. Testing UsingStress Testing
- 3.9.5. Reason for Endurance Tests
- 3.9.6. Tests Using LoadRunner

3.10. Scalability. Scalable Software Design
- 3.10.1. Scalability and Software Architecture
- 3.10.2. Independence Between Layers
- 3.10.3. Coupling Between Layers Architecture Patterns

Module 4. Software Project Management Methodologies Waterfall Methodology vs Agile Methodology

4.1. Waterfall Methodology
- 4.1.1. Waterfall Methodology
- 4.1.2. Waterfall Methodology Influence on Software Quality
- 4.1.3. Waterfall Methodology Examples:

4.2. Agile Methodology
- 4.2.1. Agile Methodology
- 4.2.2. Agile Methodology. Influence on Software Quality
- 4.2.3. Agile Methodology. Examples:

4.3. Scrum Methodology
- 4.3.1. Scrum Methodology
- 4.3.2. SCRUM Manifesto
- 4.3.3. SCRUM Application

4.4. Kanban Board
- 4.4.1. Kanban Method
- 4.4.2. Kanban Board
- 4.4.3. Kanban Board. Application Examples

4.5. Waterfall Project Management
- 4.5.1. Project Phases
- 4.5.2. Vision in a Waterfall Project
- 4.5.3. Deliverables to Consider

4.6. Project Management in Scrum
- 4.6.1. Phases in a Scrum Project
- 4.6.2. Vision in a Scrum Project
- 4.6.3. Deliverables to Consider

4.7. Waterfall vs. Scrum Comparison
- 4.7.1. Pilot Project Approach
- 4.7.2. Project Applying Waterfall. Example
- 4.7.3. Project Applying Scrum. Example

4.8. Customer Vision
- 4.8.1. Documents in a *Waterfall*
- 4.8.2. Documents in a Scrum
- 4.8.3. Comparison

4.9. Kanban Structure
- 4.9.1. User Stories
- 4.9.2. Backlog
- 4.9.3. Kanban Analysis

4.10. Hybrid Projects
- 4.10.1. Project Construction
- 4.10.2. Project Management
- 4.10.3. Deliverables to Consider

Module 5. TDD (Test-Driven Development). Test-Driven Software Design

5.1. TDD. Test-Driven Development
- 5.1.1. TDD. Test Driven Development
- 5.1.2. TDD. Influence of TDD on Quality
- 5.1.3. Test-Driven Design and Development. Examples

5.2. TDD Cycle
- 5.2.1. Choice of a Requirement
- 5.2.2. Performing Tests. Typology
  - 5.2.2.1. Unit Tests
  - 5.2.2.2. Integration Tests
  - 5.2.2.3. End To EndTests
- 5.2.3. Test Verification. Errors
- 5.2.4. Creation of the Implementation
- 5.2.5. Automated Test Execution
- 5.2.6. Elimination of Duplication
- 5.2.7. Requirements Lists Update
- 5.2.8. Repeating the TDD Cycle
- 5.2.9. TDD Cycle. Theoretical and Practical Example

5.3. TDD Implementation Strategies
- 5.3.1. Mock Implementation
- 5.3.2. Triangular Implementation
- 5.3.3. Obvious Implementation

5.4. TDD. Use. Advantages and Disadvantages
- 5.4.1. Advantages of Use
- 5.4.2. Limitations of Use
- 5.4.3. Quality Balance in the Implementation

5.5. TDD. Good Practices
- 5.5.1. TDD Rules
- 5.5.2. Rule 1: Have a Previous Test that Fails Before Coding in Production
- 5.5.3. Rule 2: Not to Write More than One Unit Test
- 5.5.4. Rule 3: Not to Write More Code than Necessary
- 5.5.5. Errors and Anti-Patterns to Avoid in TDD

5.6. Simulation of a Real Project to use TDD (I)
- 5.6.1. Project Overview (Company A)
- 5.6.2. Application of TDD
- 5.6.3. Proposed Exercises
- 5.6.4. Exercises Feedback

5.7. Simulation of a Real Project to use TDD (II)
- 5.7.1. Project Overview (Company B)
- 5.7.2. Application of TDD
- 5.7.3. Proposed Exercises
- 5.7.4. Exercises Feedback

5.8. Simulation of a Real Project to use TDD (III)
- 5.8.1. General Description of the Project (Company C)
- 5.8.2. Application of TDD
- 5.8.3. Proposed Exercises
- 5.8.4. Exercises Feedback

5.9. Alternatives to 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. Theoretical Comparison

5.10. TDD TCR, BDD and ATDD. Practical Comparison
- 5.10.1. Defining the Problem
- 5.10.2. Resolution with TCR
- 5.10.3. Resolution with BDD
- 5.10.4. Resolution with ATDD

Module 6. DevOps. Software Quality Management

6.1. DevOps. Software Quality Management
- 6.1.1. DevOps.
- 6.1.2. DevOps and Software Quality
- 6.1.3. DevOps. Benefits of DevOps Culture

6.2. DevOps. Relation to Agile
- 6.2.1. Accelerated Delivery
- 6.2.2. Quality
- 6.2.3. Cost Reduction

6.3. DevOps Implementation
- 6.3.1. Problem identification
- 6.3.2. Implementation in a Company
- 6.3.3. Implementation Metrics

6.4. Software Delivery Cycle
- 6.4.1. Design Methods
- 6.4.2. Agreements
- 6.4.3. Roadmap

6.5. Error-Free Code Development
- 6.5.1. Maintainable Code
- 6.5.2. Development Patterns
- 6.5.3. Code Testing
- 6.5.4. Software Development at Code Level Good Practices

6.6. Automization
- 6.6.1. Automization Types of Tests
- 6.6.2. Cost of Automation and Maintenance
- 6.6.3. Automization Mitigating Errors

6.7. Deployment
- 6.7.1. Target Assessment
- 6.7.2. Design of an Automatic and Adapted Process
- 6.7.3. Feedback and Responsiveness

6.8. Incident Management
- 6.8.1. Incident Management
- 6.8.2. Incident Analysis and Resolution
- 6.8.3. How to Avoid Future Mistakes

6.9. Deployment Automation
- 6.9.1. Preparing for Automated Deployments
- 6.9.2. Assessment of the Health of the Automated Process
- 6.9.3. Metrics and Rollback Capability

6.10. Good Practices. Evolution of DevOps
- 6.10.1. Guide of Good Practices applying DevOps
- 6.10.2. DevOps. Methodology for the Team
- 6.10.3. Avoiding Niches

Module 7. DevOps and Continuous Integration. Advanced Practical Solutions in Software Development

7.1. Software Delivery Flow
- 7.1.1. Identification of Actors and Artifacts
- 7.1.2. Software Delivery Flow Design
- 7.1.3. Software Delivery Flow. Interstage Requirements

7.2. Process Automation
- 7.2.1. Continuous Integration
- 7.2.2. Continuous Deployment
- 7.2.3. Environment Configuration and Secret Management

7.3. Declarative Pipelines
- 7.3.1. Differences Between Traditional, Code-Like and Declarative Pipelines
- 7.3.2. Declarative Pipelines
- 7.3.3. Declarative Pipelines in Jenkins
- 7.3.4. Comparison of Continuous Integration Providers

7.4. Quality Gates and Enriched Feedback
- 7.4.1. Quality Gates
- 7.4.2. Quality Standards with Quality Gates. Maintenance
- 7.4.3. Business Requirements in Integration Requests

7.5. Artifact Management
- 7.5.1. Artifacts and Life Cycle
- 7.5.2. Artifact Storage and Management Systems
- 7.5.3. Security in Artifact Management

7.6. Continuous Deployment
- 7.6.1. Continuous Deployment as Containers
- 7.6.2. Continuous Deployment with PaaS
- 7.6.3. Continuous Deployment of Mobile Applications

7.7. Improving Pipeline Runtime: Static Analysis and Git Hooks
- 7.7.1. Static Analysis
- 7.7.2. Code Style Rules
- 7.7.3. Git Hooks and Unit Tests
- 7.7.4. The Impact of Infrastructure

7.8. Vulnerabilities in Containers
- 7.8.1. Vulnerabilities in Containers
- 7.8.2. Image Scanning
- 7.8.3. Periodic Reports and Alerts

Module 8. Database (DB) Design. Standardization and performance. Software Quality

8.1. Database Design
- 8.1.1. Databases. Typology
- 8.1.2. Databases Currently Used
  - 8.1.2.1. Relationship
  - 8.1.2.2. Key-Value
  - 8.1.2.3. Based on Graphs
- 8.1.3. Data Quality

8.2. Entity-Relationship Model Design (I)
- 8.2.1. Entity-Relationship Model. Quality and Documentation
- 8.2.2. Entities
  - 8.2.2.1. Strong Entity
  - 8.2.2.2. Weak Entity
- 8.2.3. Attributes
- 8.2.4. Set of Relationships
  - 8.2.4.1. 1 to 1
  - 8.2.4.2. 1 to Many
  - 8.2.4.3. Many to 1
  - 8.2.4.4. Many to Many
- 8.2.5. Keys
  - 8.2.5.1. Primary Key
  - 8.2.5.2. Foreign Key
  - 8.2.5.3. Weak Entity Primary Key
- 8.2.6. Restrictions
- 8.2.7. Cardinality
- 8.2.8. Heritage
- 8.2.9. Aggregation

8.3. Entity-Relationship Model (II). Tools
- 8.3.1. Entity-Relationship Model. Tools
- 8.3.2. Entity-Relationship Model. Practical Example
- 8.3.3. Feasible Entity-Relationship Model
  - 8.3.3.1. Visual Sample
  - 8.3.3.2. Sample in Table Representation

8.4. Database (DB) Standardization (I). Software Quality Considerations
- 8.4.1. DB Standardization and Quality
- 8.4.2. Dependency
  - 8.4.2.1. Functional Dependence
  - 8.4.2.2. Properties of Functional Dependence
  - 8.4.2.3. Deduced Properties
- 8.4.3. Keys

8.5. Database (DB) Normalization (II). Normal Forms and Codd Rules
- 8.5.1. Normal Shapes
  - 8.5.1.1. First Normal Form (1FN)
  - 8.5.1.2. Second Normal Form (2FN)
  - 8.5.1.3. Third Normal Form (3FN)
  - 8.5.1.4. Boyce-Codd Normal Form (BCNF).
  - 8.5.1.5. Fourth Normal Form (4FN)
  - 8.5.1.6. Fifth Normal Form (5FN)
- 8.5.2. Codd's Rules
  - 8.5.2.1. Rule 1: Information
  - 8.5.2.2. Rule 2: Guaranteed Access
  - 8.5.2.3. Rule 3: Systematic Treatment of Null Values
  - 8.5.2.4. Rule 4: Description of the Database
  - 8.5.2.5. Rule 5: Integral Sub-Language
  - 8.5.2.6. Rule 6: View Update
  - 8.5.2.7. Rule 7: Insert and Update
  - 8.5.2.8. Rule 8: Physical Independence
  - 8.5.2.9. Rule 9: Logical Independence
  - 8.5.2.10. Rule 10: Integrity Independence
    - 8.5.2.10.1. Integrity Rules
  - 8.5.2.11. Rule 11: Distribution
  - 8.5.2.12. Rule 12: Non-Subversion
- 8.5.3. Practical Example

8.6. Data Warehouse/OLAP System
- 8.6.1. Data Warehouse
- 8.6.2. Fact Table
- 8.6.3. Dimension Table
- 8.6.4. Creation of the OLAP System. Tools

8.7. Database (DB) Performance
- 8.7.1. Index Optimization
- 8.7.2. Query Optimization
- 8.7.3. Table Partitioning

8.8. Simulation of Real Project for DB Design (I)
- 8.8.1. Project Overview (Company A)
- 8.8.2. Application of Database Design
- 8.8.3. Proposed Exercises
- 8.8.4. Proposed Exercises Feedback

8.9. Simulation of Real Project for BD Design (II)
- 8.9.1. Project Overview (Company B)
- 8.9.2. Application of Database Design
- 8.9.3. Proposed Exercises
- 8.9.4. Proposed Exercises Feedback

8.10. Relevance of DB Optimization to Software Quality
- 8.10.1. Design Optimization
- 8.10.2. Query Code Optimization
- 8.10.3. Stored Procedure Code Optimization
- 8.10.4. Influence of Triggers on Software Quality. Reccomendations for Use.

Module 9. Scalable Architecture Design Architecture in the Software Life Cycle

9.1. Design of Scalable Architectures (I)
- 9.1.1. Scalable Architectures
- 9.1.2. Principles of a Scalable Architecture
  - 9.1.2.1. Reliable
  - 9.1.2.2. Scalable
  - 9.1.2.3. Maintainable
- 9.1.3. Types of Scalability
  - 9.1.3.1. Vertical
  - 9.1.3.2. Horizontal
  - 9.1.3.3. Combined

9.2. Architecture DDD (Domain-Driven Design)
- 9.2.1. The DDD Model Domain Orientation
- 9.2.2. Layers, Distribution of Responsibility and Design Patterns
- 9.2.3. Decoupling as a Basis for Quality

9.3. Design of Scalable Architectures (II). Benefits, Limitations and Design Strategies
- 9.3.1. Scalable Architecture. Benefits
- 9.3.2. Scalable Architecture. Limitations
- 9.3.3. Strategies for the Development of Scalable Architectures (Descriptive Table)

9.4. Software Life Cycle (I). Stages
- 9.4.1. Software Life Cycle
  - 9.4.1.1. Planning Stage
  - 9.4.1.2. Analysis Stage
  - 9.4.1.3. Design Stage
  - 9.4.1.4. Implementation Stage
  - 9.4.1.5. Testing Stage
  - 9.4.1.6. Installation/Deployment Stage
  - 9.4.1.7. Use and Maintenance Stage

9.5. Software Life Cycle Models
- 9.5.1. Waterfall Model
- 9.5.2. Repetitive Model
- 9.5.3. Spiral Model
- 9.5.4. Big Bang Model

9.6. Software Life Cycle (II). Automation
- 9.6.1. Software Development Life Cycle. Solutions
  - 9.6.1.1. Continuous Integration and Development (CI/CD)
  - 9.6.1.2. Agile Methodologies
  - 9.6.1.3. DevOps/Production Operations
- 9.6.2. Future Trends
- 9.6.3. Practical Examples

9.7. Software Architecture in the Software Life Cycle
- 9.7.1. Benefits
- 9.7.2. Limitations
- 9.7.3. Tools

9.8. Real Project Simulation for Software Architecture Design (I)
- 9.8.1. Project Overview (Company A)
- 9.8.2. Software Architecture Design Application
- 9.8.3. Proposed Exercises
- 9.8.4. Proposed Exercises Feedback

9.9. Simulation of a Real Project for Software Architecture Design (II)
- 9.9.1. Project Overview (Company B)
- 9.9.2. Software Architecture Design Application
- 9.9.3. Proposed Exercises
- 9.9.4. Proposed Exercises Feedback

9.10. Simulation of a Real Project for Software Architecture Design (III)
- 9.10.1. General Description of the Project (Company C)
- 9.10.2. Software Architecture Design Application
- 9.10.3. Proposed Exercises
- 9.10.4. Proposed Exercises Feedback

Module 10. ISO, IEC 9126 Quality Criteria. Software Quality Metrics

10.1. Quality Criteria. ISO, IEC 9126 Standard
- 10.1.1. Quality Criteria.
- 10.1.2. Software Quality Justification. ISO, IEC 9126 Standard
- 10.1.3. Software Quality Measurement as a Key Indicator

10.2. Software Quality Criteria Features
- 10.2.1. Reliability
- 10.2.2. Functionality
- 10.2.3. Efficiency
- 10.2.4. Usability
- 10.2.5. Maintainability
- 10.2.6. Portability
- 10.2.7. Security/safety

10.3. ISO Standard, IEC 9126 (I). Introduction
- 10.3.1. Description of ISO, IEC 9126 Standard
- 10.3.2. Functionality
- 10.3.3. Reliability
- 10.3.4. Usability
- 10.3.5. Maintainability
- 10.3.6. Portability
- 10.3.7. Quality in Use
- 10.3.8. Software Quality Metrics
- 10.3.9. ISO 9126 Quality Metrics

10.4. ISO Standard, IEC 9126 (II). McCall and Boehm Models
- 10.4.1. McCall Model: Quality factors
- 10.4.2. Boehm Model
- 10.4.3. Intermediate Level. Features

10.5. Software Quality Metrics (I). Components
- 10.5.1. Measurement
- 10.5.2. Metrics
- 10.5.3. Indicator
  - 10.5.3.1. Types of Indicators
- 10.5.4. Measurements and Models
- 10.5.5. Scope of Software Metrics
- 10.5.6. Classification of Software Metrics

10.6. Software Quality Measurement (II). Measurement Practice
- 10.6.1. Metric Data Collection
- 10.6.2. Measurement of Internal Product Attributes
- 10.6.3. Measurement of External Product Attributes
- 10.6.4. Measurement of Resources
- 10.6.5. Metrics for Object-Oriented Systems

10.7. Design of a Single Software Quality Indicator
- 10.7.1. Single Indicator as a Global Qualifier
- 10.7.2. Indicator Development, Justification and Application
- 10.7.3. Example of Application. Need to Know the Detail

10.8. Simulation of Real Project for Quality Measurement (I)
- 10.8.1. Project Overview (Company A)
- 10.8.2. Application of Quality Measurement
- 10.8.3. Proposed Exercises
- 10.8.4. Proposed Exercises Feedback

10.9. Real Project Simulation for Quality Measurement (II)
- 10.9.1. Project Overview (Company B)
- 10.9.2. Application of Quality Measurement
- 10.9.3. Proposed Exercises
- 10.9.4. Proposed Exercises Feedback

10.10. Real Project Simulation for Quality Measurement (III)
- 10.10.1. General Description of the Project (Company C)
- 10.10.2. Application of Quality Measurement
- 10.10.3. Proposed Exercises
- 10.10.4. Proposed Exercises Feedback

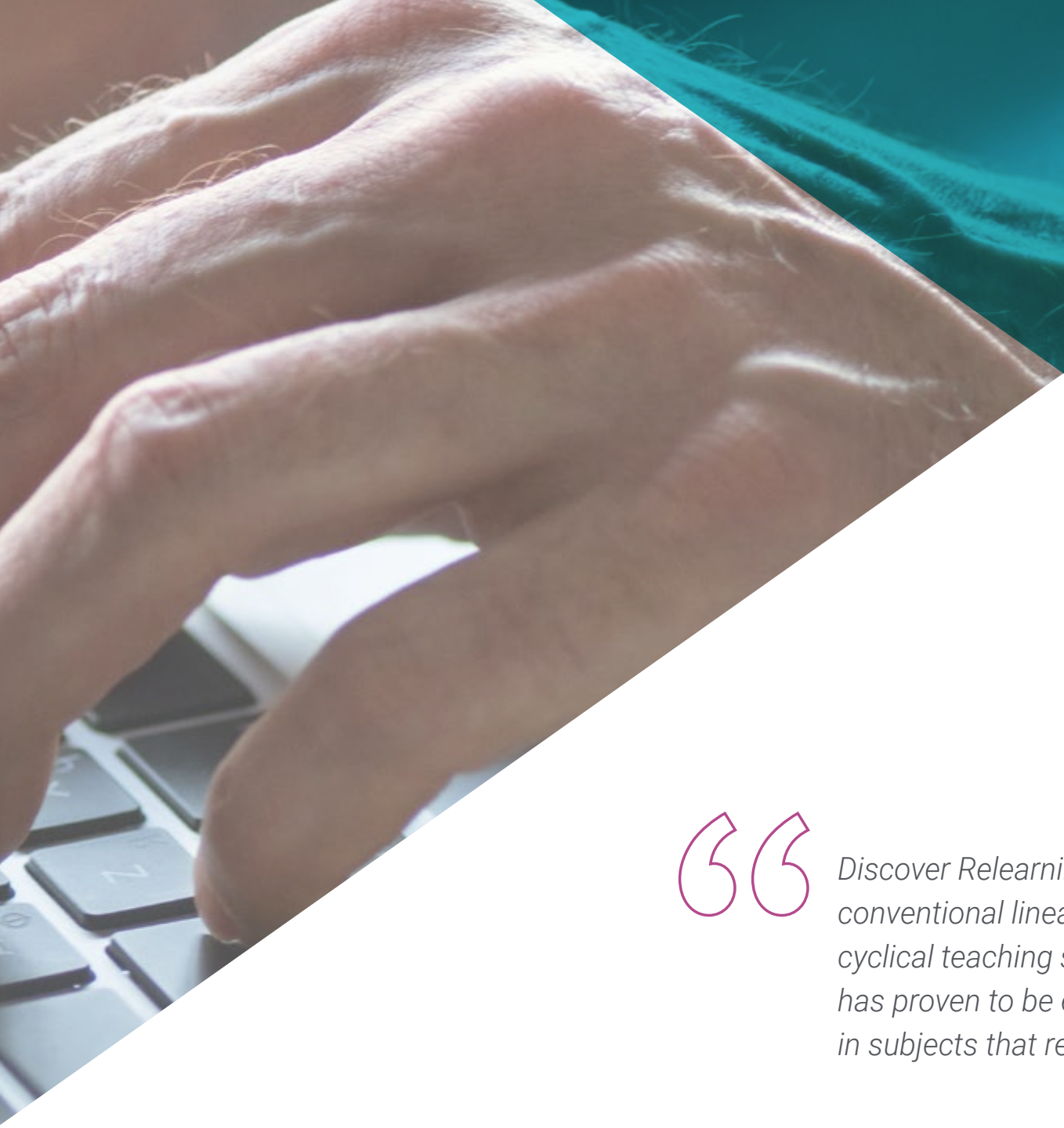
You will have access to unique and specialized content. Selected by expert teachers for a qualification that will make your professional profile transcend"

06

Methodology

This academic program offers students a different way of learning. Our methodology uses a cyclical learning approach: **Relearning.**

This teaching system is used, for example, in the most prestigious medical schools in the world, and major publications such as the **New England Journal of Medicine** have considered it to be one of the most effective.



Discover Relearning, a system that abandons conventional linear learning, to take you through cyclical teaching systems: a way of learning that has proven to be extremely effective, especially in subjects that require memorization"

Case Study to contextualize all content

Our program offers a revolutionary approach to developing skills and knowledge. Our goal is to strengthen skills in a changing, competitive, and highly demanding environment.

At TECH, you will experience a learning methodology that is shaking the foundations of traditional universities around the world"

You will have access to a learning system based on repetition, with natural and progressive teaching throughout the entire syllabus.

The student will learn to solve complex situations in real business environments through collaborative activities and real cases.

A learning method that is different and innovative

This TECH program is an intensive educational program, created from scratch, which presents the most demanding challenges and decisions in this field, both nationally and internationally. This methodology promotes personal and professional growth, representing a significant step towards success. The case method, a technique that lays the foundation for this content, ensures that the most current economic, social and professional reality is taken into account.

> *Our program prepares you to face new challenges in uncertain environments and achieve success in your career"*

The case method has been the most widely used learning system among the world's leading Information Technology schools for as long as they have existed. The case method was developed in 1912 so that law students would not only learn the law based on theoretical content. It consisted of presenting students with real-life, complex situations for them to make informed decisions and value judgments on how to resolve them. In 1924, Harvard adopted it as a standard teaching method.

What should a professional do in a given situation? This is the question that you are presented with in the case method, an action-oriented learning method. Throughout the course, students will be presented with multiple real cases. They will have to combine all their knowledge and research, and argue and defend their ideas and decisions.

Relearning Methodology

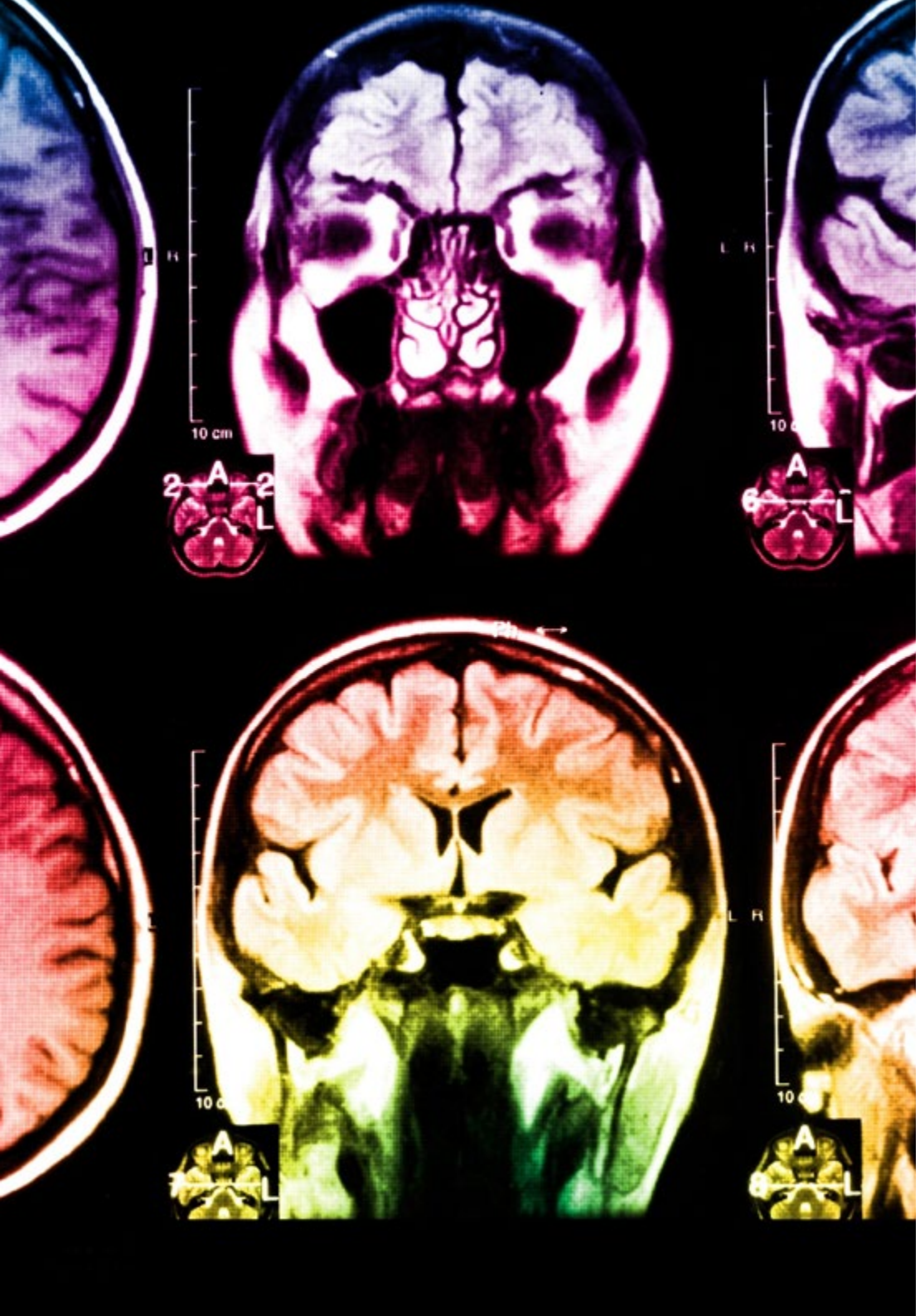
TECH effectively combines the Case Study methodology with a 100% online learning system based on repetition, which combines different teaching elements in each lesson.

We enhance the Case Study with the best 100% online teaching method: Relearning.

In 2019, we obtained the best learning results of all online universities in the world.

At TECH you will learn using a cutting-edge methodology designed to train the executives of the future. This method, at the forefront of international teaching, is called Relearning.

Our university is the only one in the world authorized to employ this successful method. In 2019, we managed to improve our students' overall satisfaction levels (teaching quality, quality of materials, course structure, objectives...) based on the best online university indicators.



In our program, learning is not a linear process, but rather a spiral (learn, unlearn, forget, and re-learn). Therefore, we combine each of these elements concentrically. This methodology has trained more than 650,000 university graduates with unprecedented success in fields as diverse as biochemistry, genetics, surgery, international law, management skills, sports science, philosophy, law, engineering, journalism, history, and financial markets and instruments. All this in a highly demanding environment, where the students have a strong socio-economic profile and an average age of 43.5 years.

Relearning will allow you to learn with less effort and better performance, involving you more in your training, developing a critical mindset, defending arguments, and contrasting opinions: a direct equation for success.

From the latest scientific evidence in the field of neuroscience, not only do we know how to organize information, ideas, images and memories, but we know that the place and context where we have learned something is fundamental for us to be able to remember it and store it in the hippocampus, to retain it in our long-term memory.

In this way, and in what is called neurocognitive context-dependent e-learning, the different elements in our program are connected to the context where the individual carries out their professional activity.

This program offers the best educational material, prepared with professionals in mind:

Study Material

All teaching material is produced by the specialists who teach the course, specifically for the course, so that the teaching content is highly specific and precise.

These contents are then applied to the audiovisual format, to create the TECH online working method. All this, with the latest techniques that offer high quality pieces in each and every one of the materials that are made available to the student.

Classes

There is scientific evidence suggesting that observing third-party experts can be useful.

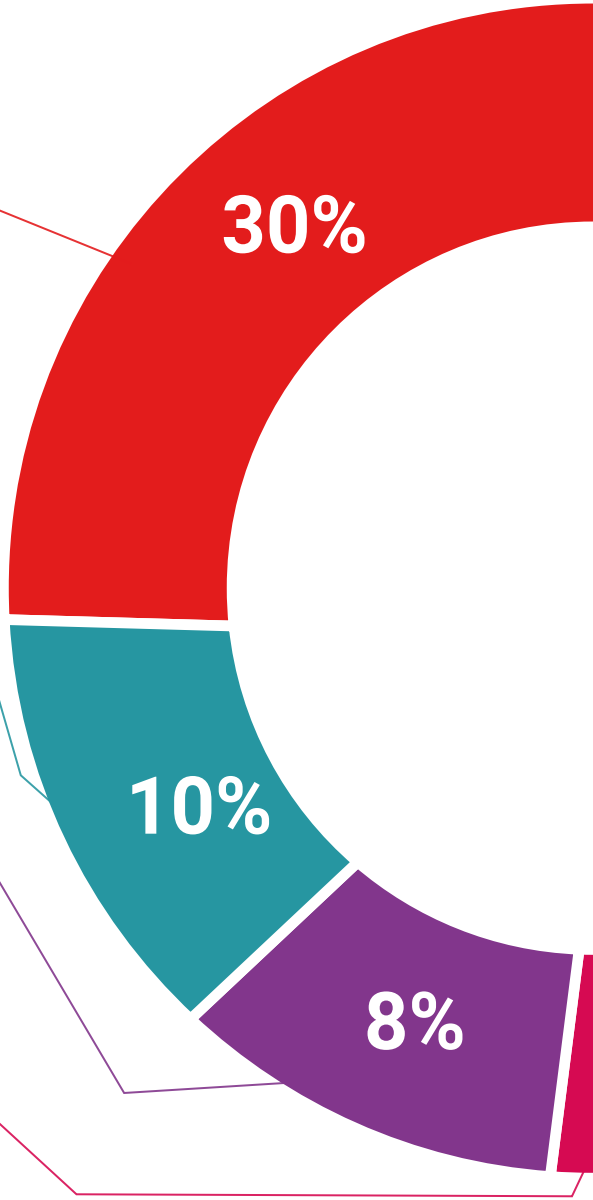
Learning from an Expert strengthens knowledge and memory, and generates confidence in future difficult decisions.

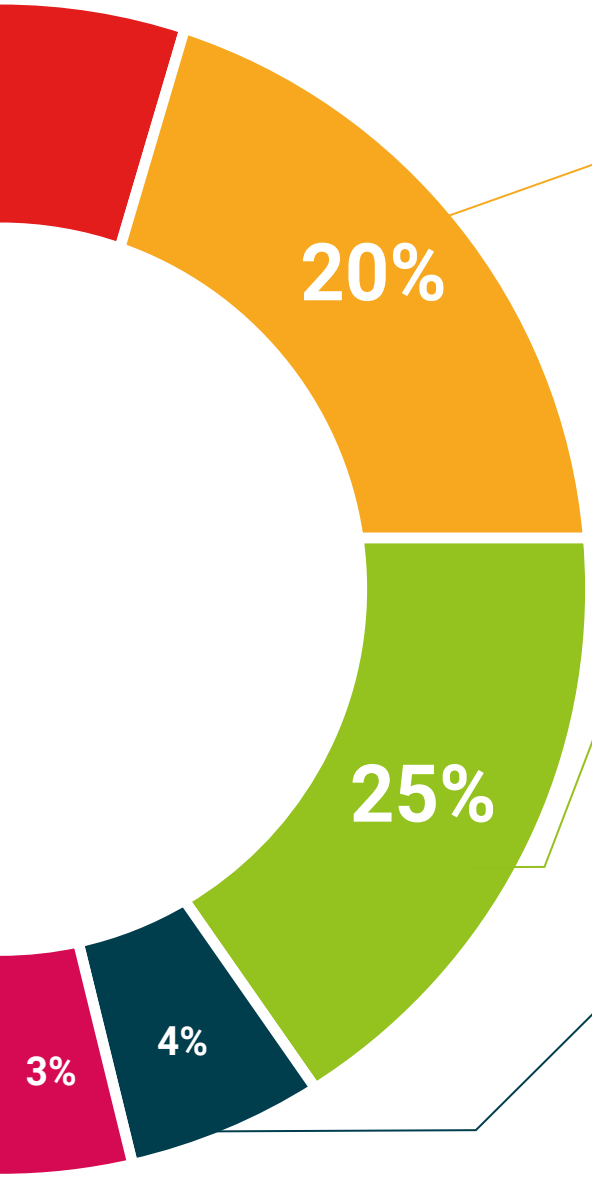
Practising Skills and Abilities

They will carry out activities to develop specific skills and abilities in each subject area. Exercises and activities to acquire and develop the skills and abilities that a specialist needs to develop in the context of the globalization that we are experiencing.

Additional Reading

Recent articles, consensus documents and international guidelines, among others. In TECH's virtual library, students will have access to everything they need to complete their course.





Case Studies

Students will complete a selection of the best case studies chosen specifically for this program. Cases that are presented, analyzed, and supervised by the best specialists in the world.

Interactive Summaries

The TECH team presents the contents attractively and dynamically in multimedia lessons that include audio, videos, images, diagrams, and concept maps in order to reinforce knowledge.

This exclusive educational system for presenting multimedia content was awarded by Microsoft as a "European Success Story".

Testing & Retesting

We periodically evaluate and re-evaluate students' knowledge throughout the program, through assessment and self-assessment activities and exercises, so that they can see how they are achieving their goals.

07

Certificate

The Professional Master's Degree in Software Quality guarantees students, in addition to the most rigorous and up-to-date education, access to a Professional Master's Degree issued by TECH Technological University.

Successfully complete this program and receive your university qualification without having to travel or fill out laborious paperwork"

This **Professional Master's Degree in Software Quality** contains the most complete and up-to-date program on the market.

After the student has passed the assessments, they will receive their corresponding **Professional Master's Degree** certificate issued by **TECH Technological University** via tracked delivery*.

The certificate issued by **TECH Technological University** will express the qualification obtained in the Professional Master's Degree, and meets the requirements commonly demanded by labor exchanges, competitive examinations, and professional career evaluation committees.

Title: **Professional Master's Degree in Software Quality**

Official Nº of Hours: **1,500 h.**


tech technological university

Awards the following

CERTIFICATE

to

Mr./Ms. _____, with identification number _____.

For having successfully passed and accredited the following program

PROFESSIONAL MASTER'S DEGREE

in

Software Quality

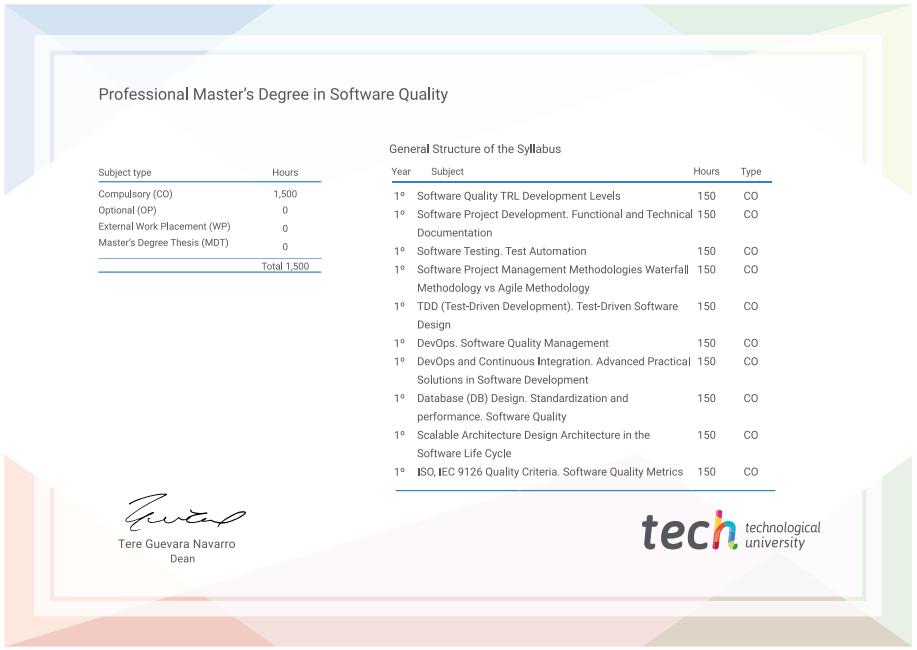
This is a qualification awarded by this University, equivalent to 1,500 hours, with a start date of dd/mm/yyyy and an end date of dd/mm/yyyy.

TECH is a Private Institution of Higher Education recognized by the Ministry of Public Education as of June 28, 2018.

June 17, 2020

Tere Guevara Navarro
Dean

This qualification must always be accompanied by the university degree issued by the competent authority to practice professionally in each country. Unique TECH Code: AFWORD23S techtitute.com/certificates


Professional Master's Degree in Software Quality

Subject type	Hours
Compulsory (CO)	1,500
Optional (OP)	0
External Work Placement (WP)	0
Master's Degree Thesis (MDT)	0
	Total 1,500

General Structure of the Syllabus

Year	Subject	Hours	Type
1º	Software Quality TRL Development Levels	150	CO
1º	Software Project Development. Functional and Technical Documentation	150	CO
1º	Software Testing. Test Automation	150	CO
1º	Software Project Management Methodologies Waterfall Methodology vs Agile Methodology	150	CO
1º	TDD (Test-Driven Development). Test-Driven Software Design	150	CO
1º	DevOps. Software Quality Management	150	CO
1º	DevOps and Continuous Integration. Advanced Practical Solutions in Software Development	150	CO
1º	Database (DB) Design. Standardization and performance. Software Quality	150	CO
1º	Scalable Architecture Design Architecture in the Software Life Cycle	150	CO
1º	ISO, IEC 9126 Quality Criteria. Software Quality Metrics	150	CO

Tere Guevara Navarro
Dean

tech technological university

*Apostille Convention. In the event that the student wishes to have their paper certificate issued with an apostille, TECH EDUCATION will make the necessary arrangements to obtain it, at an additional cost.

tech technological university

Professional Master's Degree
Software Quality

- Modality: **online**
- Duration: **12 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Professional Master's Degree

Software Quality

tech technological university