Advanced Master's Degree

Software Engineering and Quality

tech global university

tech global university

Advanced Master's Degree
Software Engineering and Quality

- Modality: **online**
- Duration: **2 years**
- Certificate: **TECH Global University**
- Credits: **120 ECTS**
- Schedule: **at your own pace**
- Exams: **online**

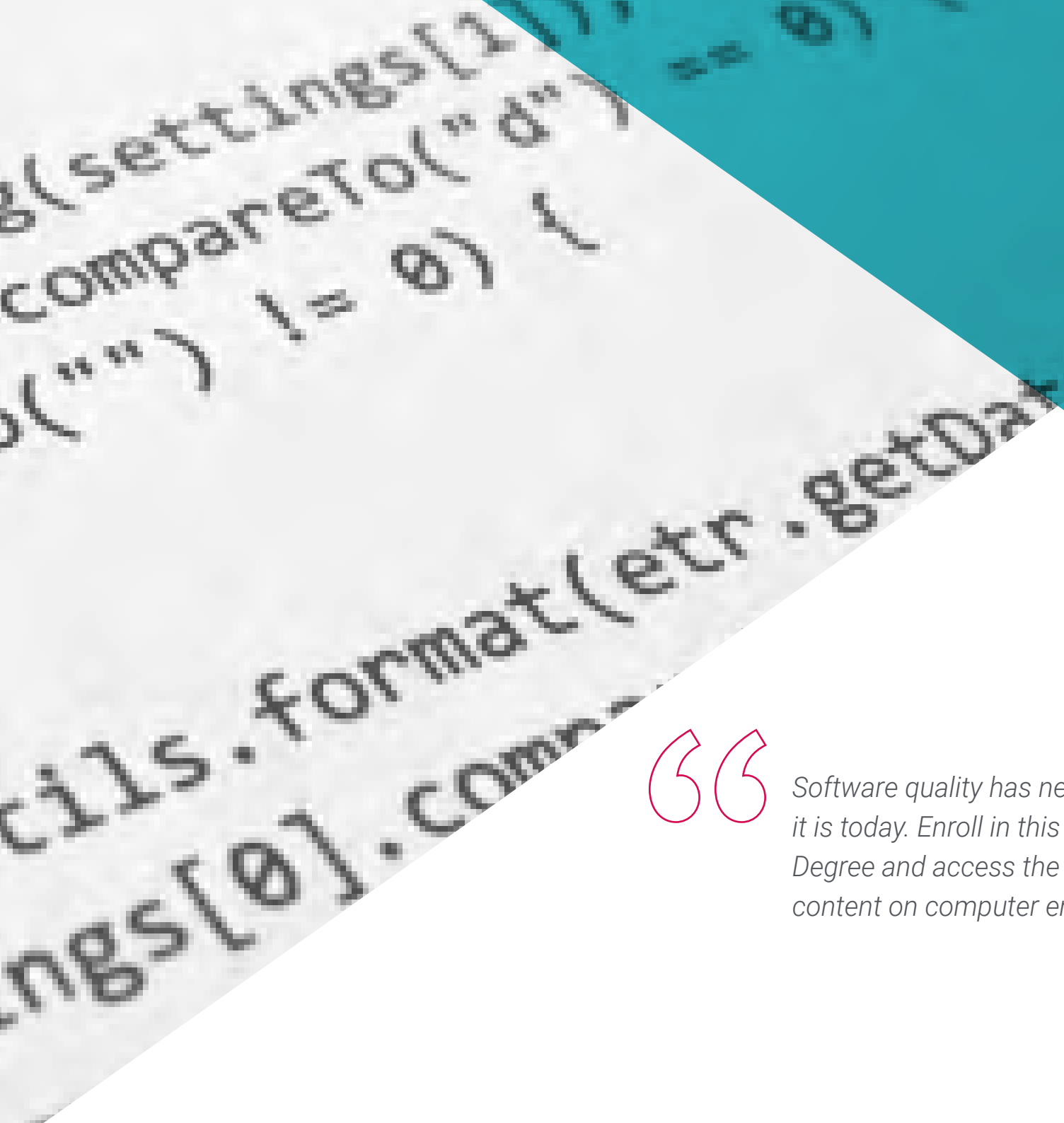
Website: www.techtitute.com/us/information-technology/advanced-master-degree/advanced-master-degree-software-engineering-quality

Index

01

Introduction

The development of technology and advances in computer systems have created a great demand from the industry for professionals who perfectly handle Software Engineering, from the most sophisticated and accurate tools for its design and implementation, to the security protocols that guarantee inviolable access to your data. For this reason, and with the aim of offering specialists the opportunity to get up to date with the most cutting-edge information on engineering applied to this area, TECH has developed this multidisciplinary and 100% online program. It is a program designed by experts that combines, in a single syllabus, 3,000 hours of the best content on computer systems and software quality, and that will help the graduate improve their computer skills in an immediate and specific way.



Software quality has never been as necessary as it is today. Enroll in this online Advanced Master's Degree and access the most comprehensive content on computer engineering"

Computer engineering has grown exponentially in recent years due to the evolution of technology and digital tools, especially in everything related to the web and its usability. That is why today the development of software for various functions is the order of the day and the catalog of programs is growing. However, this quantity is not always synonymous of quality, so that frequently there are applications that do not fulfill their purpose, that return errors or that seriously violate the security of the companies. For this reason, computer engineers specialized in this area are increasingly in demand.

That is why TECH has decided to design this Advanced Master's Degree in Software Engineering and Quality, a multidisciplinary program designed by experts in the area and designed in such a way that the graduate will find in it all the necessary tools to update their knowledge in a comprehensive manner and based on the latest developments in the sector. It is an educational program that combines theory and practice in 20 modules in which software engineering and the quality of information systems projects are studied in depth.

Throughout the 24 months in which this 100% online program is taught, the engineer will have access to the best syllabus that will allow him to improve his skills in the standardization of databases and in the decoupling between components of a system, as well as to expand his knowledge in scalable architectures, quality metrics and collaborative work.

In addition, you will have access to a modern and cutting-edge virtual classroom where you will find all the tools that will allow you to get the most out of this certificate, including hundreds of hours of additional material in different formats. All this content can be downloaded to any device with an internet connection, ensuring that you can consult it whenever you want and need it.

This **Advanced Master's Degree in Software Engineering and Quality** contains the most complete and up-to-date educational program on the market. Its most notable features are:

- Case studies presented by engineering experts
- 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
- Special emphasis on innovative methodologies in Software design and construction
- Theoretical lessons, questions to the expert, debate forums on controversial topics, and individual reflection assignments
- Content that is accessible from any fixed or portable device with an Internet connection

You will have access to HTML exercises and their answers, so you will be able to put into practice your knowledge and the theory developed throughout the programming"

Thanks to the dedicated DevOps module you will have the most comprehensive and in-depth knowledge to speed up the software development lifecycle and ensure continuous high-quality delivery"

Thanks to this certificate you will be able to start your own software development project and apply the most sophisticated and innovative unit stress and endurance tests to check its quality.

Delve into Test-Driven Development and gain a broad and specialized view of test-driven software design and development.

It includes, in its teaching staff, professionals belonging to the field of engineering who contribute their work experience to this program, as well as renowned specialists from prestigious universities and reference societies.

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 an immersive learning experience designed to prepare for real-life situations.

This program is designed around Problem-Based Learning, whereby the student must try to solve the different professional practice situations that arise throughout the program. For this purpose, the professional will be assisted by an innovative interactive video system created by renowned and experienced experts.

02

Objectives

Computer engineering is an industry that is constantly changing. This is why TECH has developed this certification, not only with the objective of providing the specialist with a broad and up-to-date knowledge of their profession, but also to provide them with a detailed knowledge of the tools that will allow them to keep up to date after the completion of this Advanced Master's Degree. In addition, the best theoretical, practical and audiovisual material will be available to make this program a dynamic and highly empowering academic experience.

If your goal is to become a specialist in Software Engineering and Quality, this Advanced Master's Degree will provide you with everything you need to exceed your professional expectations with a total guarantee of success"

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
- Acquire new knowledge in Software and Computer Systems Engineering
- Acquire new skills in terms of new technologies and the latest software developments
- Process the data generated in Software and Computer Systems Engineering activities

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 (local, national and international)
- Examine the TRL maturity levels and adapt them to the different parts of the software project to be dealt with
- Acquire the 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. Software Testing. 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
- Analyze the different types of fundamental tests, such as load, unit, *stress* and endurance tests
- 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 on
- 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 Panel
- 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 in 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. Design of Scalable Architectures. The 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 a 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

Module 11. Methodologies, Development and Quality in Software Engineering

- Know the basics of Software Engineering, as well as the set of rules or principles of ethics and professional responsibility during and after development
- Understand the software development process under the different programming models and the object-oriented programming paradigm
- Understand the different types of application modeling and design patterns in the Unified Modeling Language (UML)
- Acquire the knowledge for the correct application of agile methodologies in Software development such as SCRUM, among others
- Know the Lean development methodology to identify the activities that do not add value to the process, in order to obtain a higher quality software

Module 12. Software Project Management

- Know the fundamental concepts of project management and the project management life cycle
- Understand the different stages of project management such as initiation, planning, stakeholder management and scoping
- Learning schedule development for time management, budget development and risk response
- Understand how quality management works in projects, including planning, assurance, control, statistical concepts and available tools
- Understand the functioning of the processes of procurement, execution, monitoring, control and closure of a project
- Acquire the essential knowledge related to the professional responsibility derived from project management

Module 13. Software Development Platforms

- Understand the different software development platforms
- Acquire the necessary knowledge for the development of applications and graphical interfaces in Java and .NET languages
- Know the techniques required for the debugging and testing of the developments made
- Learn Android mobile application development environments and debugging and publishing processes
- Understand cloud-based application development and determine the correct procedures for its implementation
- Master the basic concepts, services and tools of the Google Clouds platform

Module 14. Web-Client Computing

- Understand the process of creating web content through HTML markup language
- Understand the procedures and techniques to improve the appearance of a document written in HTML
- Know the evolution of the JavaScript language
- Acquire the necessary knowledge for the development of web client-side applications
- Develop applications with complex structures, by using the different procedures, functions and objects that integrate JavaScript
- Learn how to use the DOM programming interface for HTML and XML documents to modify their structure, style and content
- Understand the use of event-based flow and Listeners, as well as the use of modern Toolkit and alignment systems
- Know the concept of web usability, its advantages, principles, methods and techniques to make a web site usable by the user
- Establish knowledge of web accessibility, its importance in current digital platforms, methodologies, norms, standards and determine compliance

Module 15. Web Server Computing

- Understand the basic, intermediate and advanced concepts of the PHP language for the implementation of server-side applications
- Acquire the necessary knowledge for data modeling, relationships, keys and normalizations
- Understand the construction of the logical data model, the specification of tables, columns, keys and dependencies, as well as the knowledge necessary for the physical handling of data, file types, access modes and file organization
- Learn how to integrate applications developed in PHP with MariaDB and MySQL databases
- Master the process of customer interactions, using forms, Cookies and session management
- Understand the Model View Controller View (MVC) Software architecture that separates an application's data, user interface, and control logic into three distinct components
- Acquire the skills for the use of web services using XML, SOA and REST

Module 16. Safety Management

- Know the information security process, its implications on confidentiality, integrity, availability and economic costs
- Learn the use of good security practices in the management of information technology services
- Acquire the knowledge for the correct certification of security processes
- Understand authentication mechanisms and methods for access control, as well as the access audit process
- Understand security management programs, risk management and security policy design
- Learn about business continuity plans, their phases and maintenance process
- Know the procedures for the correct protection of the company through DMZ networks, the use of intrusion detection systems and other methodologies

Module 17. Software Security

- Understand Software security issues, vulnerabilities and how they are classified
- Know the design principles, methodologies and standards in software security
- Understand the application of security in the different phases of the software life cycle
- Acquire the necessary knowledge for secure coding of the life cycle and its validation techniques
- Understand the methodologies and processes to guarantee security during the development and delivery of cloud services
- Understand the basics of cryptology and the different encryption techniques currently available

Module 18. Web Server Administration

- Know the concept, operation, architecture, resources and contents of a web server
- Understand the functioning, structure and HTTP protocol handling
- Understand the concept of distributed multi-server architectures
- Master the functioning of an application server and another proxy
- Analyze the different web servers that are trending in today's market
- Understand the process of usage statistics and load balancing on web servers
- Acquire the necessary knowledge for the installation, administration, configuration and security of the Microsoft Internet Information Services (IIS) web server as well as the free Apache web server

Module 19. Security Audit

- Acquire the knowledge required for the correct execution of the audit process and internal computer control
- Understand the processes to carry out for the security audit in systems and networks
- Understand the different support tools, methodologies and subsequent analysis during internet and mobile device security auditing
- Learn the properties and influencing factors that condition business risks and determine the correct implementation of appropriate risk management
- Know the risk mitigation measures, as well as the methodologies for the implementation of an Information Security Management System and the norms and standards to be used
- Understand the procedures for conducting the security audit, its traceability and presentation of results

Module 20. Online Application Security

- Acquire the knowledge required to evaluate and detect the vulnerabilities of online applications
- Understand the security policies and standards to be applied to online applications
- Know the procedures to use during the development of web applications and their subsequent evaluation through analysis and security tests
- Learn the security measures for the deployment and production of web applications
- Understand the concepts, functions and technologies to be applied in the security of web services, as well as security tests and protective measures
- Assimilate the procedures for ethical *hacking*, malware analysis and forensic analysis
- Know the mitigation and containment measures for incidents on web services
- Incorporate best practice techniques for the development and implementation of online applications

03

Skills

Handling Software design, planning, management and development tools perfectly is a very complex task, among other things, because of the number of processes involved. However, this Advanced Master's Degree will provide the graduates with all the information they need to perfect their skills in the control of the tools in this area. In this way, they will be able to successfully complete their tasks and complete their projects with the most promising and high-quality results in the computer engineering sector.

Specializing in this field will allow you to develop specific leadership skills to lead software management projects, a skill highly valued by companies dedicated to computer engineering"

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 Test-Driven Development (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
- Design, manage and implement Software engineering and information systems projects

You will be able to learn in detail about the main databases and access simulations of real projects for their design applied to companies in different sectors"

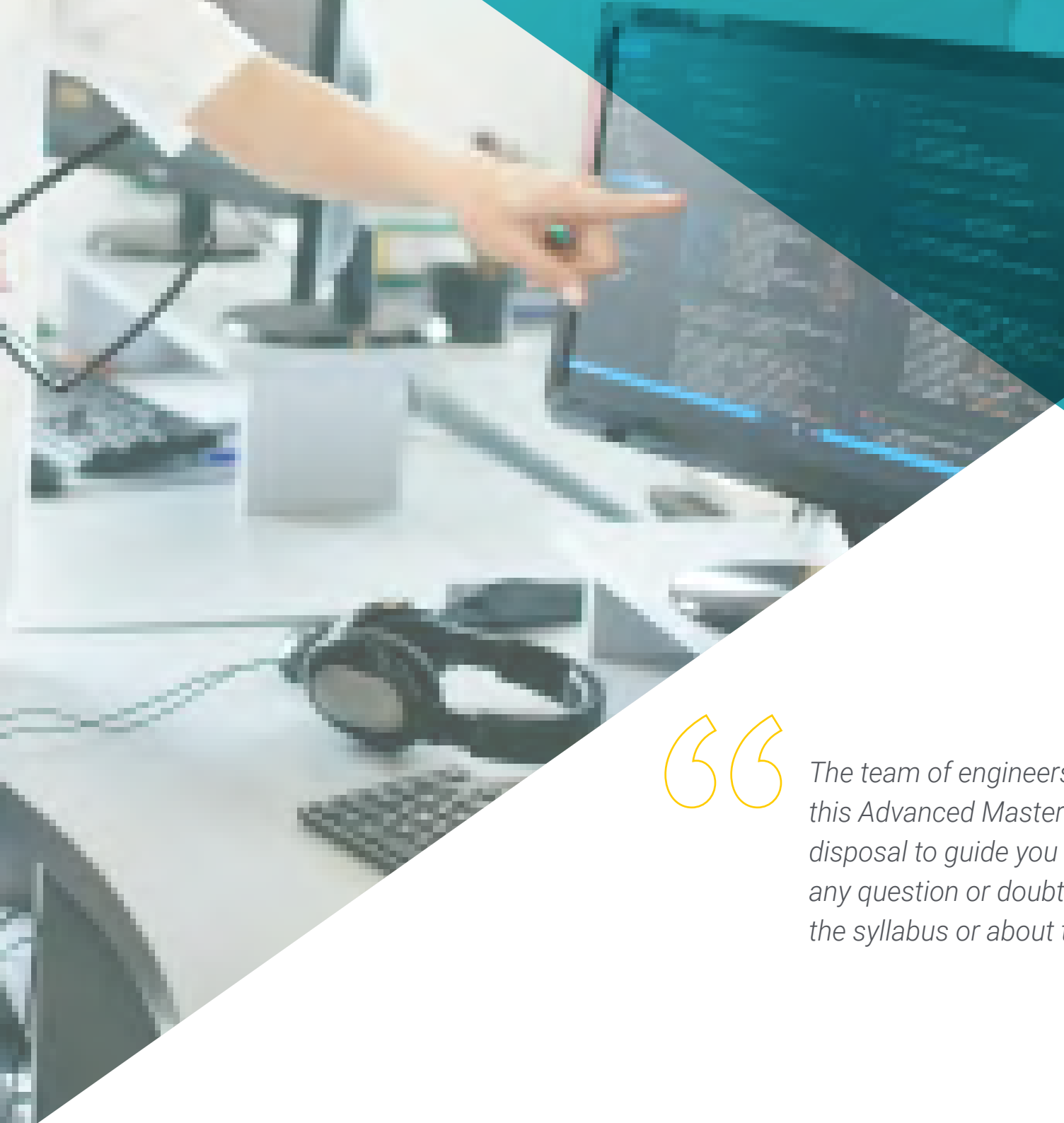
Specific Skills

- 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
- Understand the different types of application modeling and design patterns in the Unified Modeling Language (UML)
- Understand how quality management works in projects, including planning, assurance, control, statistical concepts and available tools
- Use the necessary knowledge for the development of applications and graphical interfaces in Java and .NET languages
- Understand the procedures and techniques to improve the appearance of a document written in HTML
- Master the process of customer interactions, using forms, Cookiesand session management
- Understand authentication mechanisms and methods for access control, as well as the access audit process
- Understand the application of security in the different phases of the software life cycle
- Know the concept, operation, architecture, resources and contents of a web server
- Understand the different support tools, methodologies and subsequent analysis during internet and mobile device security auditing
- Understand the security policies and standards to be applied to online applications

04

Course Management

The direction and teaching of this Advanced Master's Degree in Software Engineering and Quality is carried out by a team of engineering experts with many years of experience in the management and development of technical and specialized projects. Their professional background brings to this program a boost in quality that will be reflected in a better contextualization of the content by the graduate, as well as the implementation to the academic experience of real and simulated case studies, but always aimed at offering a 100% online, dynamic and avant-garde program based on the immediate reality of the sector.



The team of engineers in charge of teaching this Advanced Master's Degree will be at your disposal to guide you and help you to solve any question or doubt you may have about the syllabus or about the profession"t

International Guest Director

Darren Pulsipher is a highly experienced software architect, an innovator with an outstanding international track record in software and firmware development. In fact, he possesses highly developed communication, project management and business skills, which have enabled him to lead major global initiatives.

He has also held senior positions of great responsibility throughout his career, such as Chief Solution Architect for the Public Sector at Intel Corporation, where he has promoted modern business, processes and technologies for customers, partners and users in the public sector. In addition, he founded Yoly Inc. where he has also served as CEO, working to develop a social network aggregation and diagnostic tool based on Software as a Service (SaaS), using Big Data and Web 2.0 technologies.

Additionally, he has served in other companies, as Senior Director of Engineering, at Dell Technologies, where he led the Big Data in the Cloud Business Unit, leading teams in the United States and China for the management of large projects and the restructuring of business divisions for their successful integration. He has also worked as Chief Information Officer at XanGo, where he managed projects such as Help Desk support, production support and solution development.

Among the many specialties in which he is an expert, Edge to Cloud technology, cybersecurity, Generative Artificial Intelligence, software development, networking technology, cloud-native development and the container ecosystem stand out. Knowledge he has shared through the "Embracing Digital Transformation" podcast and weekly newsletter, which he produced and hosted, helping organizations successfully navigate digital transformation by leveraging people, processes and technology.

Mr. Pulsipher, Darren

- Chief Solution Architect for Public Sector at Intel, California, United States
- Presenter and Producer of "Embracing Digital Transformation", California
- Founder and CEO at Yoly Inc., Arkansas
- Senior Director of Engineering at Dell Technologies, Arkansas
- Chief Information Technology Officer, XanGo, Utah
- Senior Architect at Cadence Design Systems, California
- Senior Project Process Manager at Lucent Technologies, California
- Software Engineer at Cemax-Icon, California
- Software Engineer at ISG Technologies, Canada
- MBA in Technology Management from the University of Phoenix, Phoenix, California
- B.S. in Computer Science and Electrical Engineering from Brigham Young University

Thanks to TECH, you will be able to learn with the best professionals in the world"

Management

Mr. Molina Molina, Jerónimo

- AI Engineer & Software Architect. NASSAT - Internet Satellite in Motion
- Sr. 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. Martínez Calvo, Francisco Javier

- Architect - Organic and functional analyst
- Technical Consultant - IT
- Development and Support to European Medical Project, FNMT, PPG and PCL integration in HexaIngenieros
- Trainer Visual Studio, SqlServer, CCNA (Cisco routers and switch), PHP and .NET web programming in several centers (Salesianos, Maforem, Dreamsoft)
- Industrial Technical Engineer, specialization in Electricity, Industrial Electronics
- Master Cibernos in .NET. MCAD
- Eidos Master's Degree in Advanced Programming. Expert Level
- WEB Master Certifications Dreamweaver, Fireworks, Flash and ActionScript, MX versions

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
- Professional application development for Android-University Galileo, 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. Acebes Tamargo, Patricia

- Operations Department, working with Elasticsearch and Kivana. Sirt
- Researcher Human Factor Line and AI Aplications. CTIC Technology Center
- Business Unit Researcher. CTIC Technology Center
- Digital Health and Active Aging Department. CTIC Technology Center
- Data Science Department. CTIC Technology Center
- PhD Candidate in Computer Engineering
- Degree in Economics University of Oviedo
- Studying for a Master's Degree in Data Analysis UCJC
- Studying Master in Artificial Intelligence (ia). UNED
- Studying Mathematics and tic engineering UNED
- Master's Degree in Blockchain, Smart Contracts and Cryptocurrencies. University of Alcalá
- Postgraduate in Blockchain Engineering. EADA
- Master's Degree in Economics and Economic Analysis Tools
- Master's Degree in Taxation

Ms. Rodríguez Míguez, Cándida

- CoFounder and City Leader of Galicia AI network (Spain AI association)
- Academic Streamer on YouTube
- SISAP project for SERGAS, web functionality auto vaccination COVID Internet Appointment. INDRA Production S.L.
- OSAL Aixiña. Collaboration in remote TFM
- Teaching introductory session on Artificial Intelligence. WordPress Galicia
- Computer Engineer specialized in Software. ESEI Ourense. University of Vigo
- Master's Degree in Computer Engineering, specializing in Large Software Systems Development. ESEI Ourense. University of Vigo
- Superior Cycle in Commercial Management and Marketing. Novacaixagalicia Ourense Private Vocational Training Center
- Superior Cycle in Computer Systems Administration. Novacaixagalicia Ourense Private Vocational Training Center

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

05

Structure and Content

For the development of this program, TECH has based the structure of its content on three fundamental pillars: the most up-to-date and complete information from the IT Engineering sector specialized in the Software area, the recommendations of the teaching team, and the innovative pedagogical methodology of Relearning. In addition, in its commitment to offer a program adapted and personalized to the academic needs of each graduate, not only in the design of schedules, but also in the level of deepening, students will find in the virtual classroom hundreds of hours of additional material with which they can delve into the aspects they consider most relevant for their professional practice.

Thanks to this program, you will be able to design scalable vertical, horizontal and combined architectures, based on the most advanced, complete and up-to-date IT techniques and protocols"

Module 1. Software Quality TRL Development Levels

1.1. Elements that Influence Software Quality (I). The Technical Debt
- 1.1.1. The 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 Product 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:

2.7. 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. 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. Motivation of Unit Tests
- 3.9.2. Unit Testing Tools
- 3.9.3. Motivation for Stress Testing
- 3.9.4. Testing UsingStressTesting
- 3.9.5. Motivation for stress Resistance
- 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. Panel Kanban
- 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 Waterfall. 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 End Tests
- 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. Automation
- 6.6.1. Automation Types of Tests
- 6.6.2. Cost of Automation and Maintenance
- 6.6.3. Automation 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. Inter-Stage 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

7.7. Pipeline Runtime Improvement: 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. Relational
  - 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. Entity-Relationship Model feasible
  - 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. Database Design Application
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. Recommendations for Use

Module 9. Design of Scalable Architectures. The 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. Application of Software Architecture Design
- 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

10.3. 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. General Description of the Project (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. General Description of the Project (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

Module 11. Methodologies, Development and Quality in Software Engineering

11.1. Model-Based Software Development
- 11.1.1. The Need for
- 11.1.2. Object Modeling
- 11.1.3. UML
- 11.1.4. CASE Tools

11.2. Application Modeling and Design Patterns with UML
- 11.2.1. Advanced Requirements Modeling
- 11.2.2. Advanced Static Modeling
- 11.2.3. Advanced Dynamic Modeling
- 11.2.4. Component Modeling
- 11.2.5. Introduction to Design Patterns with 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. Model-Driven Engineering
- 11.3.1. Introduction
- 11.3.2. Metamodeling of Systems
- 11.3.3. MDA
- 11.3.4. DSL
- 11.3.5. Model Refinements with OCL
- 11.3.6. Model Transformations

11.4. Ontologies in Software Engineering
- 11.4.1. Introduction
- 11.4.2. Ontology Engineering
- 11.4.3. Application of Ontologies in Software Engineering

Module 12. Software Project Management

12.1. Stakeholders and Outreach Management
- 12.1.1. Identify Stakeholders
- 12.1.2. Develop Plan for Stakeholder Management
- 12.1.3. Manage Stakeholder Engagement
- 12.1.4. Control Stakeholder Engagement
- 12.1.5. The Objective of the Project
- 12.1.6. Scope Management and its Plan
- 12.1.7. Gathering Requirements
- 12.1.8. Define the Scope Statement
- 12.1.9. Create the WBS
- 12.1.10. Verify and Control the Scope

12.2. The Development of the Time-Schedule
- 12.2.1. Time Management and its Plan
- 12.2.2. Define Activities
- 12.2.3. Establishment of the Sequence of Activities
- 12.2.4. Estimated Resources for Activities
- 12.2.5. Estimated Duration of Activities
- 12.2.6. Development of the Time-Schedule and Calculation of the Critical Path
- 12.2.7. Schedule Control

12.3. Budget Development and Risk Response
- 12.3.1. Estimate Costs
- 12.3.2. Develop Budget and S-Curve
- 12.3.3. Cost Control and Earned Value Method
- 12.3.4. Risk Concepts
- 12.3.5. How to Perform a Risk Analysis
- 12.3.6. The Development of the Response Plan

12.4. Communication and Human Resources
- 12.4.1. Planning Communications Management
- 12.4.2. Communications Requirements Analysis
- 12.4.3. Communication Technology
- 12.4.4. Communication Models
- 12.4.5. Communication Methods
- 12.4.6. Communications Management Plan
- 12.4.7. Manage Communications
- 12.4.8. Management of Human Resources
- 12.4.9. Main Stakeholders and their Roles in the Projects
- 12.4.10. Types of Organization
- 12.4.11. Project Organization
- 12.4.12. The Work Equipment

12.5. Procurement
- 12.5.1. The Procurement Process
- 12.5.2. Plan
- 12.5.3. Search for Suppliers and Request for Quotations
- 12.5.4. Contract Allocation
- 12.5.5. Contract Administration
- 12.5.6. Contracts
- 12.5.7. Types of Contracts
- 12.5.8. Contract Negotiation

12.6. Execution, Monitoring and Control and Closure
- 12.6.1. Process Groups
- 12.6.2. Project Execution
- 12.6.3. Project Monitoring and Control
- 12.6.4. Project Closure

12.7. Professional Responsibility
- 12.7.1. Professional Responsibility
- 12.7.2. Characteristics of Social and Professional Responsibility
- 12.7.3. Project Leader Code of Ethics
- 12.7.4. Liability vs. PMP®
- 12.7.5. Examples of Liability
- 12.7.6. Benefits of Professionalization

Module 13. Software Development Platforms

13.1. Introduction to Application Development
- 13.1.1. Desktop Applications
- 13.1.2. Programming Language
- 13.1.3. Integrated Development Environments
- 13.1.4. Web Applications
- 13.1.5. Mobile Applications
- 13.1.6. Cloud Applications

13.2. Application Development and Graphical User Interface in Java
- 13.2.1. Integrated Development Environments for Java
- 13.2.2. Main IDE for Java
- 13.2.3. Introduction to the Eclipse Development Platform
- 13.2.4. Introduction to the NetBeans Development Platform
- 13.2.5. Controller View Model for Graphical User Interfaces
- 13.2.6. Design a Graphical Interface in Eclipse
- 13.2.7. Design a Graphical Interface in NetBeans

13.3. Debugging and Testing in Java
- 13.3.1. Testing and Debugging of Java programs
- 13.3.2. Debugging in Eclipse
- 13.3.3. Debugging in NetBeans

13.4. Application Development and Graphical User Interface in. NET
- 13.4.1. Net Framework
- 13.4.2. Components of the .NET Development Platform
- 13.4.3. Visual Studio .NET
- 13.4.4. .NET tools for GUI
- 13.4.5. The GUI with Windows Presentation Foundation
- 13.4.6. Debugging and Compiling a WPF Application

13.5. Programming for .NET Networks
- 13.5.1. Introduction to .NET Network Programming
- 13.5.2. Requests and Responses in .NET
- 13.5.3. Use of Application Protocols in .NET
- 13.5.4. Security in .NET Network Programming

13.6. Mobile Application Development Environments
- 13.6.1. Mobile Applications
- 13.6.2. Android Mobile Applications
- 13.6.3. Steps for Development in Android
- 13.6.4. The IDE Android Studio

13.7. Development of Applications in the Environment Android Studio
- 13.7.1. Install and Start Android Studio
- 13.7.2. Running an Android Application
- 13.7.3. Development of the Graphic Interface in Android Studio
- 13.7.4. Starting Activities in Android Studio

13.8. Debugging and Publishing of Android Applications
- 13.8.1. Debugging an Application in Android Studio
- 13.8.2. Memorizing Applications in Android Studio
- 13.8.3. Publishing an Application on Google Play

13.9. Cloud Application Development
- 13.9.1. Cloud Computing
- 13.9.2. Cloud Levels: SaaS, PaaS, IaaS
- 13.9.3. Main Development Platforms in the Cloud
- 13.9.4. Bibliographical References

13.10. Introduction to Google Cloud Platform
- 13.10.1. Basic Concepts of Google Cloud Platform
- 13.10.2. Google Cloud Platform Services
- 13.10.3. Tools in Google Cloud Platform

Module 14. Web-Client Computing

14.1. Introduction to HTML
- 14.1.1. Structure of the Document
- 14.1.2. Color
- 14.1.3. Text:
- 14.1.4. Hypertext Links
- 14.1.5. Images
- 14.1.6. Lists
- 14.1.7. Tables
- 14.1.8. Frames
- 14.1.9. Forms
- 14.1.10. Specific Elements for Mobile Technologies
- 14.1.11. Obsolete Elements

14.2. Cascading Style Sheets (CSS)
- 14.2.1. Elements and Structure of a Cascading Style Sheet
  - 14.2.1.1. Creation of Style Sheets
  - 14.2.1.2. Application of Styles Selectors
  - 14.2.1.3. Style Inheritance and Cascading
  - 14.2.1.4. Page Formatting Using Styles
  - 14.2.1.5. Page Structuring Using Styles. The Box Model
- 14.2.2. Style Design for different Devices
- 14.2.3. Types of Style Sheets: Static and Dynamic The Pseudo-Classes
- 14.2.4. Best Practices in the Use of Style Sheets

14.3. Introduction and History of JavaScript
- 14.3.1. Introduction
- 14.3.2. History of JavaScript
- 14.3.3. Development Environment to be Used

14.4. Basic Notions of Web Programming
- 14.4.1. Basic JavaScript Syntax
- 14.4.2. Primitive Data Types and Operators
- 14.4.3. Variables and Areas
- 14.4.4. Text Strings and Template Literals
- 14.4.5. Numbers and Booleans
- 14.4.6. Comparisons

14.5. Complex JavaScript Structures
- 14.5.1. Vectors or Arrays and Objects
- 14.5.2. Sets
- 14.5.3. Maps
- 14.5.4. Disjunctive
- 14.5.5. Loops

14.6. Functions and Objects
- 14.6.1. Function Definition and Invocation
- 14.6.2. Arguments
- 14.6.3. Arrow Functions
- 14.6.4. Callback Functions
- 14.6.5. Higher Order Functions
- 14.6.6. Literal Objects
- 14.6.7. The This Object
- 14.6.8. Objects as Namespaces: theMaths and Date Objects

14.7. The Document Object Model (DOM)
- 14.7.1. What is DOM?
- 14.7.2. A Bit of History
- 14.7.3. Navigation and Element Retrieval
- 14.7.4. A Virtual DOM with JSDOM
- 14.7.5. Query Selectors
- 14.7.6. Navigation using Properties
- 14.7.7. Assigning Attributes to Elements
- 14.7.8. Creation and Modification of Nodes
- 14.7.9. Updated Styling of the DOM Elements

14.8. Modern Web Development
- 14.8.1. Event-Driven Flow and Listeners
- 14.8.2. Modern Web Toolkits and Alignment Systems
- 14.8.3. Strict JavaScript Mode
- 14.8.4. More about Functions
- 14.8.5. Asynchronous Promises and Functions
- 14.8.6. Closures
- 14.8.7. Functional Programming
- 14.8.8. POO in JavaScript

14.9. Web Usability
- 14.9.1. Introduction to Usability
- 14.9.2. Definition of Usability
- 14.9.3. Importance of User-Centered Web Design
- 14.9.4. Differences Between Accessibility and Usability
- 14.9.5. Advantages and Problems in Combining Accessibility and Usability
- 14.9.6. Advantages and Difficulties in the Implementation of Usable Websites
- 14.9.7. Usability Methods
- 14.9.8. User Requirements Analysis
- 14.9.9. Conceptual Design Principles. User-Oriented Prototyping
- 14.9.10. Guidelines for the Creation of Usable Web Sites
  - 14.9.10.1. Usability Guidelines of Jakob Nielsen
  - 14.9.10.2. Usability Guidelines of Bruce Tognazzini
- 14.9.11. Usability Evaluation

14.10. Web Accessibility
- 14.10.1. Introduction
- 14.10.2. Definition of Web-Accessibility
- 14.10.3. Types of Disabilities
  - 14.10.3.1. Temporary or Permanent Disabilities
  - 14.10.3.2. Visual Impairment
  - 14.10.3.3. Hearing Impairment
  - 14.10.3.4. Motor Impairment
  - 14.10.3.5. Neurological or Cognitive Disabilities
  - 14.10.3.6. Difficulties Arising from Aging
  - 14.10.3.7. Limitations Arising from the Environment
  - 14.10.3.8. Barriers Preventing Access to the Web
- 14.10.4. Technical Aids and Support Products to Overcome Barriers
  - 14.10.4.1. Aids for the Blind
  - 14.10.4.2. Aids for Persons with Low Vision
  - 14.10.4.3. Aids for People with Color Blindness
  - 14.10.4.4. Aids for the Hearing Impaired
  - 14.10.4.5. Aids for the Motor Impaired
  - 14.10.4.6. Aids for the and Neurological Impaired
- 14.10.5. Advantages and Difficulties in the Implementation of Web Accessibility
- 14.10.6. Web Accessibility Regulations and Standards
- 14.10.7. Web Accessibility Regulatory Bodies
- 14.10.8. Comparison of Standards and Regulations
- 14.10.9. Guidelines for Compliance with Regulations and Standards
  - 14.10.9.1. Description of the Main Guidelines (Images, links, videos, etc.)
  - 14.10.9.2. Guidelines for Accessible Navigation
    - 14.10.9.2.1. Perceptibility
    - 14.10.9.2.2. Operability
    - 14.10.9.2.3. Comprehensibility
    - 14.10.9.2.4. Robustness
- 14.10.10. Description of the Web Accessibility Compliance Process
- 14.10.11. Compliance Levels
- 14.10.12. Compliance Criteria
- 14.10.13. Compliance Requirements
- 14.10.14. Web Site Accessibility Evaluation Methodology

Module 15. Web Server Computing

15.1. Introduction to Server-Side Programming: PHP
- 15.1.1. Server-Side Programming Basics
- 15.1.2. Basic PHP Syntax
- 15.1.3. HTML Content Generation with PHP
- 15.1.4. Development and Testing Environments: XAMPP

15.2. Advanced PHP
- 15.2.1. Control Structures with PHP
- 15.2.2. PHP Functions
- 15.2.3. Array Handling in PHP
- 15.2.4. String Handling with PHP
- 15.2.5. Object Orientation in PHP

15.3. Data Models
- 15.3.1. Concept of Data. Life Cycle of Data
- 15.3.2. Types of Data
  - 15.3.2.1. Basic
  - 15.3.2.2. Records
  - 15.3.2.3. Dynamics

15.4. Relational Model
- 15.4.1. Description
- 15.4.2. Entities and Types of Entities
- 15.4.3. Data Elements. Attributes
- 15.4.4. Relationships: Types, Subtypes, Cardinality
- 15.4.5. Keys Types of Keys
- 15.4.6. Normalization. Normal Shapes

15.5. Construction of the Logical Data Model
- 15.5.1. Specification of Tables
- 15.5.2. Definition of Columns
- 15.5.3. Key Specification
- 15.5.4. Conversion to Normal Shapes. Dependency

15.6. The Physical Data Model. Data Files
- 15.6.1. Description of Data Files
- 15.6.2. Types of Files
- 15.6.3. Access Modes
- 15.6.4. File Organization

15.7. Database Access from PHP
- 15.7.1. Introduction to MariaDB
- 15.7.2. Working with a MariaDB Database: the SQL Language
- 15.7.3. Accessing the MariaDB Database from PHP
- 15.7.4. Introduction to MySql
- 15.7.5. Working with a MySql Database: The SQL Language
- 15.7.6. Accessing MySql Database from PHP

15.8. Client Interaction from PHP
- 15.8.1. PHP Forms
- 15.8.2. Cookies
- 15.8.3. Session Management

15.9. Web Application Architecture
- 15.9.1. The Model-View-Controller Pattern
- 15.9.2. Controller
- 15.9.3. Models
- 15.9.4. View

15.10. Introduction to Web Services
- 15.10.1. Introduction to XML
- 15.10.2. Service-Oriented Architecture (SOA): Web Services
- 15.10.3. Creation of SOAP and REST Web Services
- 15.10.4. The SOAP Protocol
- 15.10.5. The REST Protocol

Module 16. Safety Management

16.1. Information Security
- 16.1.1. Introduction
- 16.1.2. Information Security Involves Confidentiality, Integrity and Availability
- 16.1.3. Safety is an Economic Issue
- 16.1.4. Safety is a Process
- 16.1.5. Classification of Information
- 16.1.6. Information Security Involves Risk Management
- 16.1.7. Security is Articulated with Security Controls
- 16.1.8. Security is both Physical and Logical
- 16.1.9. Safety Involves People

16.2. The Information Security Professional
- 16.2.1. Introduction
- 16.2.2. Information Security as a Profession
- 16.2.3. ISC2 Certifications
- 16.2.4. The ISO 27001 Standard
- 16.2.5. Best Security Practices in IT Service Management
- 16.2.6. Information Security Maturity Models
- 16.2.7. Other Certifications, Standards and Professional Resources

16.3. Access Control
- 16.3.1. Introduction
- 16.3.2. Access Control Requirements
- 16.3.3. Authentication Mechanisms
- 16.3.4. Authorization Methods
- 16.3.5. Access Accounting and Auditing
- 16.3.6. Triple A Technologies

16.4. Information Security Programs, Processes and Policies
- 16.4.1. Introduction
- 16.4.2. Security Management Programs
- 16.4.3. Risk Management
- 16.4.4. Design of Security Policies

16.5. Business Continuity Plans
- 16.5.1. Introduction to BCPs
- 16.5.2. Phase I and II
- 16.5.3. Phase III and IV
- 16.5.4. Maintenance of the BCP

16.6. Procedures for the Correct Protection of the Company
- 16.6.1. DMZ Networks
- 16.6.2. Intrusion Detection Systems
- 16.6.3. Access Control Lists
- 16.6.4. Learning from the Attacker: Honeypot

16.7. Security Architecture Prevention
- 16.7.1. Overview. Activities and Layer Model
- 16.7.2. Perimeter Defence (Firewalls, WAFs, WAFs, IPS, etc.)
- 16.7.3. Endpoint Defence (Equipment, Servers and Services)

16.8. Security Architecture Detection
- 16.8.1. Overview Detection and Monitoring
- 16.8.2. Logs, Encrypted Traffic Breaking, Recording and Siems
- 16.8.3. Alerts and Intelligence

16.9. Security Architecture Reaction
- 16.9.1. Reaction Products, Services and Resources
- 16.9.2. Incident Management
- 16.9.3. CERTS and CSIRTs

16.10. Security Architecture Recuperation
- 16.10.1. Resilience, Concepts, Business Requirements and Regulations
- 16.10.2. IT Resilience Solutions
- 16.10.3. Crisis Management and Governance

Module 17. Software Security

17.1. Software Security Problems
- 17.1.1. Introduction to the Problem of Software Security
- 17.1.2. Vulnerabilities and their Classification
- 17.1.3. Secure Software Properties
- 17.1.4. References

17.2. Software Safety Design Principles
- 17.2.1. Introduction
- 17.2.2. Software Safety Design Principles
- 17.2.3. Types of S-SDLC
- 17.2.4. Software Safety in S-SDLC Phases
- 17.2.5. Methodologies and Standards
- 17.2.6. References

17.3. Software Lifecycle Safety in the Requirements and Design Phases
- 17.3.1. Introduction
- 17.3.2. Attack Modeling
- 17.3.3. Cases of Abuse
- 17.3.4. Safety Requirements Engineering
- 17.3.5. Risk Analysis Architectural
- 17.3.6. Design Patterns
- 17.3.7. References

17.4. Software Lifecycle Safety in the Coding, Testing and Operation Phases
- 17.4.1. Introduction
- 17.4.2. Risk-Based Safety Testing
- 17.4.3. Code Review
- 17.4.4. Penetration Test
- 17.4.5. Security Operations
- 17.4.6. External Review
- 17.4.7. References

17.5. Secure Coding Applications I
- 17.5.1. Introduction
- 17.5.2. Secure Coding Practices
- 17.5.3. Manipulation and Validation of Inputs
- 17.5.4. Memory Overflow
- 17.5.5. References

17.6. Secure Coding Applications II
- 17.6.1. Introduction
- 17.6.2. Integers Overflows, Truncation Errors and Problems with Type Conversions between Integers
- 17.6.3. Errors and Exceptions
- 17.6.4. Privacy and Confidentiality
- 17.6.5. Privileged Programs
- 17.6.6. References

17.7. Development and Cloud Security
- 17.7.1. Safety in Development; Methodology and Practice
- 17.7.2. PaaS, IaaS, CaaS and SaaS Models
- 17.7.3. Security in the Cloud and for Cloud Services

17.8. Encryption
- 17.8.1. Fundamentals of Cryptology
- 17.8.2. Symmetric and Asymmetric Encryption
- 17.8.3. Encryption at Rest and in Transit

17.9. Security Automation and Orchestration (SOAR)
- 17.9.1. Complexity of Manual Processing: Need to Automate Tasks
- 17.9.2. Products and Services
- 17.9.3. SOAR Architecture

17.10. Telework Safety
- 17.10.1. Need and Scenarios
- 17.10.2. Products and Services
- 17.10.3. Telework Safety

Module 18. Web Server Administration

18.1. Introduction to Web Servers
- 18.1.1. What is a Web Server?
- 18.1.2. Architecture and Operation of a Web Server
- 18.1.3. Resources and Contents on a Web Server
- 18.1.4. Application Servers
- 18.1.5. Proxy Servers
- 18.1.6. Main Web Servers on the Market
- 18.1.7. Web Server Usage Statistics
- 18.1.8. Web Server Security
- 18.1.9. Load Balancing on Web Servers
- 18.1.10. References

18.2. HTTP Protocol Handling
- 18.2.1. Operation and Structure
- 18.2.2. Request Methods
- 18.2.3. Status Codes
- 18.2.4. Headers
- 18.2.5. Content Coding Code Pages

  Performing HTTP Requests on the Internet Using a Proxy, Livehttpheadersor similar Method, Analyzing the Protocol Used

18.3. Description of Distributed Multi-Server Architectures
- 18.3.1. 3-Layer Model
- 18.3.2. Fault Tolerance
- 18.3.3. Load Sharing
- 18.3.4. Session State Stores
- 18.3.5. Cache Stores

18.4. Internet Information Services (IIS)
- 18.4.1. What is IIS?
- 18.4.2. History and Evolution of IIS
- 18.4.3. Main Advantages and Features of IIS7 and Later Versions
- 18.4.4. IIS7 Architecture and Later Versions

18.5. IIS Installation, Administration and Configuration
- 18.5.1. Preamble
- 18.5.2. Internet Information Services (IIS) Installation
- 18.5.3. IIS Administration Tools
- 18.5.4. Web Site Creation, Configuration and Administration
- 18.5.5. Installation and Management of IIS Extensions

18.6. Advanced Security in IIS
- 18.6.1. Preamble
- 18.6.2. Authentication, Authorization, and Access Control in IIS
- 18.6.3. Configuring a Secure Website on IIS with SSL
- 18.6.4. Security Policies Implemented in IIS 18.x

18.7. Introduction to Apache
- 18.7.1. What is Apache?
- 18.7.2. Main Advantages of Apache
- 18.7.3. Main Features of Apache
- 18.7.4. Architecture

18.8. Apache Installation and Configuration
- 18.8.1. Initial Installation of Apache
- 18.8.2. Apache Configuration

18.9. Installation and Configuration of the Different Apache Modules
- 18.9.1. Apache Module Installation
- 18.9.2. Types of Modules
- 18.9.3. Secure Apache Configuration

18.10. Advanced Security
- 18.10.1. Authentication, Authorization and Access Control
- 18.10.2. Authentication Methods
- 18.10.3. Secure Apache Configuration with SSL

Module 19. Security Audit

19.1. Introduction to Information Systems in the Company
- 19.1.1. Introduction to Information Systems in the Company and the Role of IT Auditing
- 19.1.2. Definitions of "IT Audit" and "IT Internal Control"
- 19.1.3. Functions and Objectives of IT Auditing
- 19.1.4. Differences between Internal Control and IT Auditing

19.2. Internal Controls of Information Systems
- 19.2.1. Functional Flowchart of a Data Processing Center
- 19.2.2. Classification of Information Systems Controls
- 19.2.3. The Golden Rule

19.3. The Process and Phases of the Information Systems Audit
- 19.3.1. Risk Assessment and Other IT Auditing Methodologies
- 19.3.2. Execution of an Information Systems Audit. Phases of the Audit
- 19.3.3. Fundamental Skills of the Auditor of an IT System

19.4. Technical Audit of Security in Systems and Networks
- 19.4.1. Technical Security Audits. Intrusion Test. Previous Concepts
- 19.4.2. Security Audits in Systems. Support Tools
- 19.4.3. Security Audits in Networks. Support Tools

19.5. Technical Audit of Security on the Internet and in Mobile Devices
- 19.5.1. Internet Security Audit. Support Tools
- 19.5.2. Mobile Devices Security Audit. Support Tools
- 19.5.3. Annex 1. Structure of an Executive Report and Technical Report
- 19.5.4. Annex 2. Tools Inventory
- 19.5.5. Annex 3. Methods

19.6. Information Security Management System
- 19.6.1. Security of IS: Properties and Influential Factors
- 19.6.2. Business Risks and Risk Management: Implementing Controls
- 19.6.3. Information Security Management System (ISMS): Concept and Critical Success Factors
- 19.6.4. ISMS-PDCA Model
- 19.6.5. ISMS ISO-IEC 27001: Organizational Context
- 19.6.6. Context of the Organization
- 19.6.7. Leadership
- 19.6.8. Plan
- 19.6.9. Support
- 19.6.10. Operation
- 19.6.11. Performance Evaluation
- 19.6.12. Improvement
- 19.6.13. Annex to ISO 27001/ISO-IEC 27002: Objectives and Controls
- 19.6.14. ISMS Audit

19.7. Carrying Out the Audit
- 19.7.1. Procedures
- 19.7.2. Techniques

19.8. Traceability
- 19.8.1. Methods
- 19.8.2. Analysis

19.9. Copyright
- 19.9.1. Techniques
- 19.9.2. Results

19.10. Reports and Presenting Proof
- 19.10.1. Types of Reports
- 19.10.2. Analysis of Data
- 19.10.3. Presenting Proof

Module 20. Online Application Security

20.1. Vulnerabilities and Security Issues in Online Applications
- 20.1.1. Introduction to Online Application Security
- 20.1.2. Security Vulnerabilities in the Design of Web Applications
- 20.1.3. Security Vulnerabilities in the Implementation of Web Applications
- 20.1.4. Security Vulnerabilities in the Deployment of Web Applications
- 20.1.5. Official Lists of Security Vulnerabilities

20.2. Policies and Standards for Online Application Security
- 20.2.1. Pillars for the Security of Online Applications
- 20.2.2. Security Policy
- 20.2.3. Information Security Management System
- 20.2.4. Secure Software Development Life Cycle
- 20.2.5. Standards for Application Security

20.3. Security in the Design of Web Applications
- 20.3.1. Introduction to Web Application Security
- 20.3.2. Security in the Design of Web Applications

20.4. Testing the Online Safety and Security of Web Applications
- 20.4.1. Web Application Security Testing and Analysis
- 20.4.2. Web Application Deployment and Production Security

20.5. Web Services Security
- 20.5.1. Introduction to Web Services Security
- 20.5.2. Web Services Security Functions and Technologies

20.6. Testing the Online Safety and Security of Web Services
- 20.6.1. Evaluation of Web Services Security
- 20.6.2. Online Protection. Firewalls and Gateways XML

20.7. Ethical Hacking, malware and Forensic
- 20.7.1. Ethical Hacking
- 20.7.2. Malware Analysis
- 20.7.3. Forensic Analysis

20.8. Best Practices to Ensure Application Security
- 20.8.1. Handbook of Best Practices in the Development of Online Applications
- 20.8.2. Handbook of Good Practices in the Implementation of Online Applications

20.9. Common Errors that Undermine Application Security
- 20.9.1. Common Errors in Development
- 20.9.2. Common Errors in Hosting
- 20.9.3. Common Production Errors

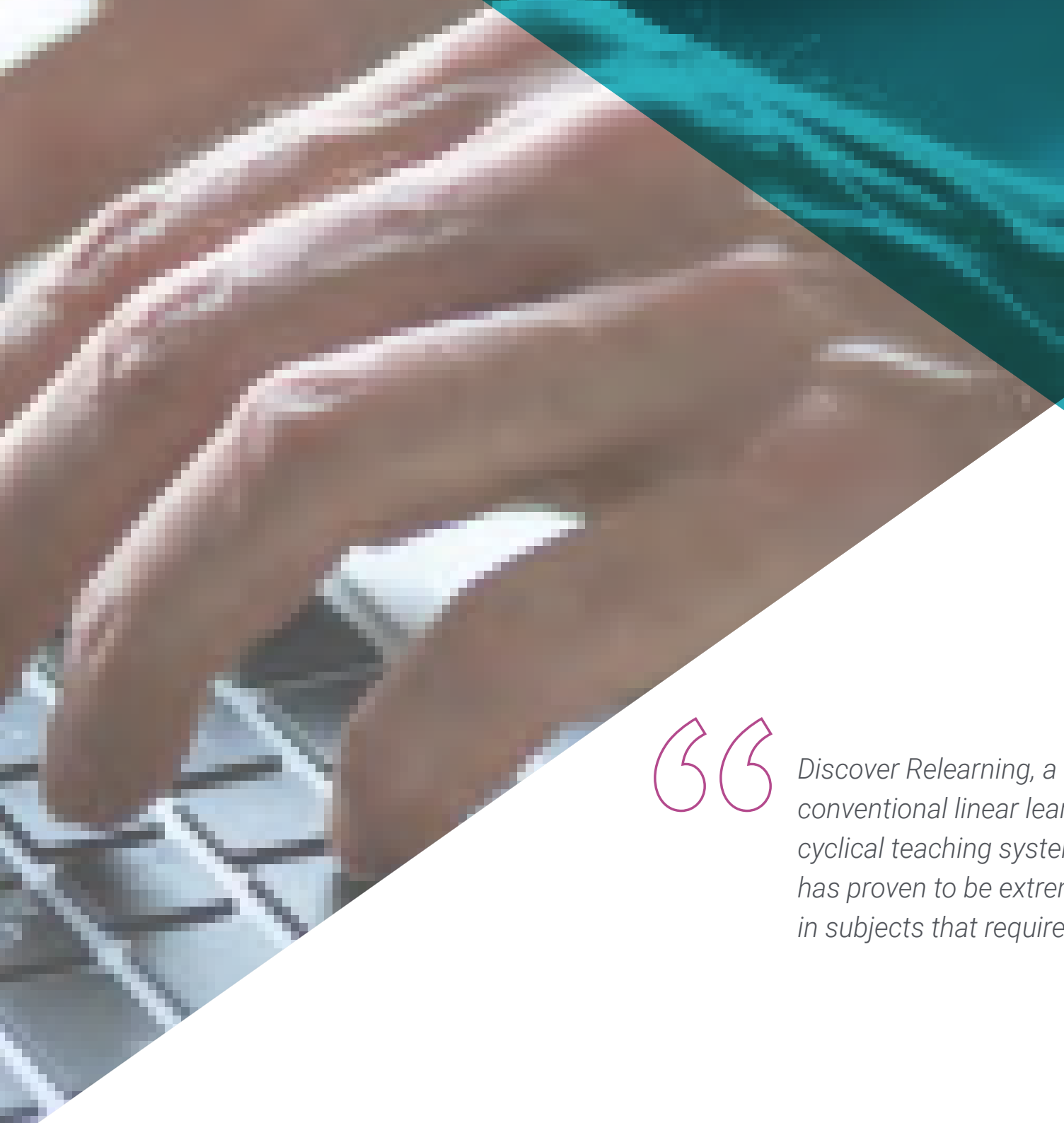
By accessing this Advanced Master's Degree, you will not only be taking a decisive step to broaden your knowledge of computer engineering specialized in software, but also towards a prosperous and successful professional projection"

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

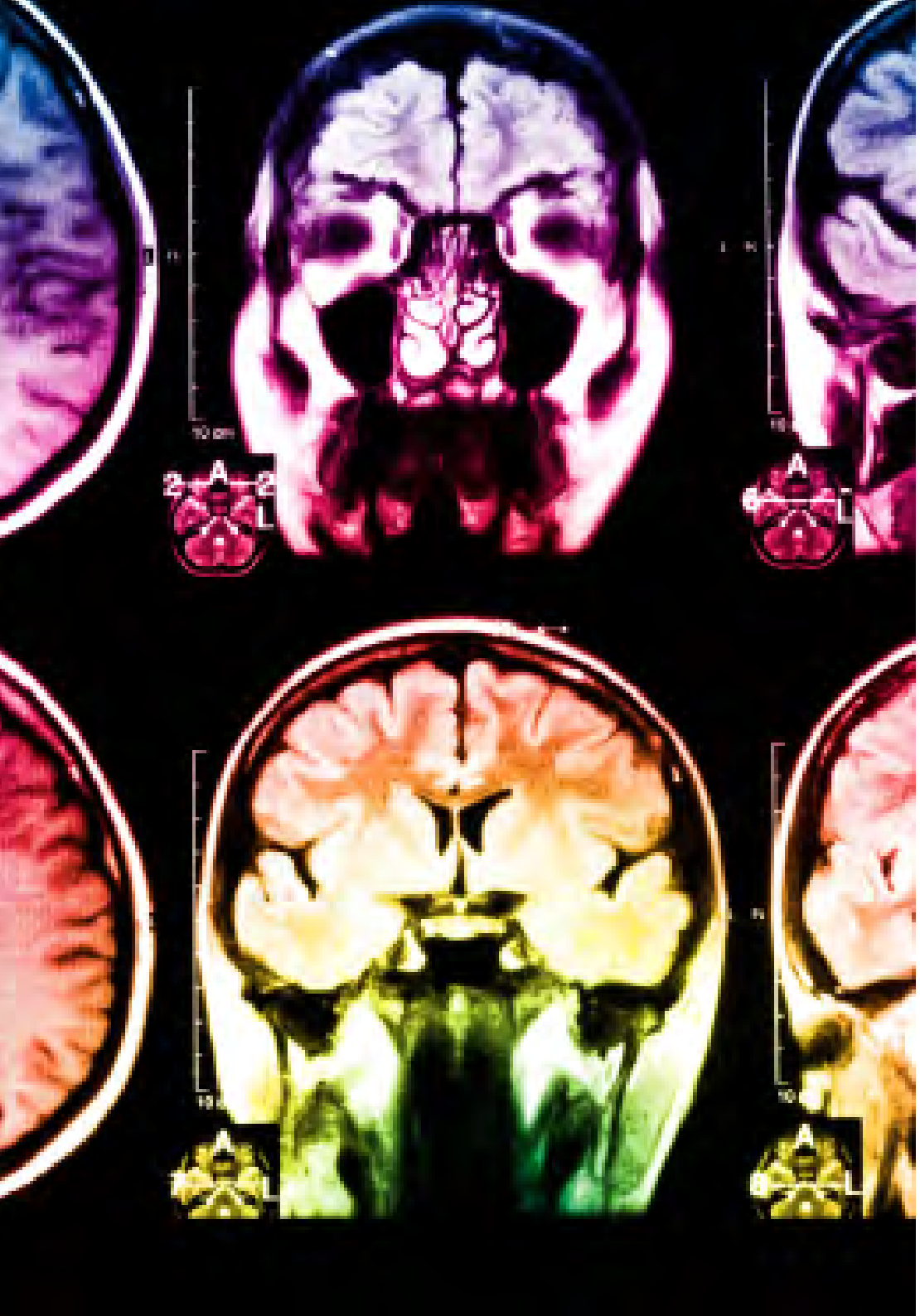
TECH effectively combines the Case Study methodology with a 100% online learning system based on repetition, which combines different teaching elements in each lesson.

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.

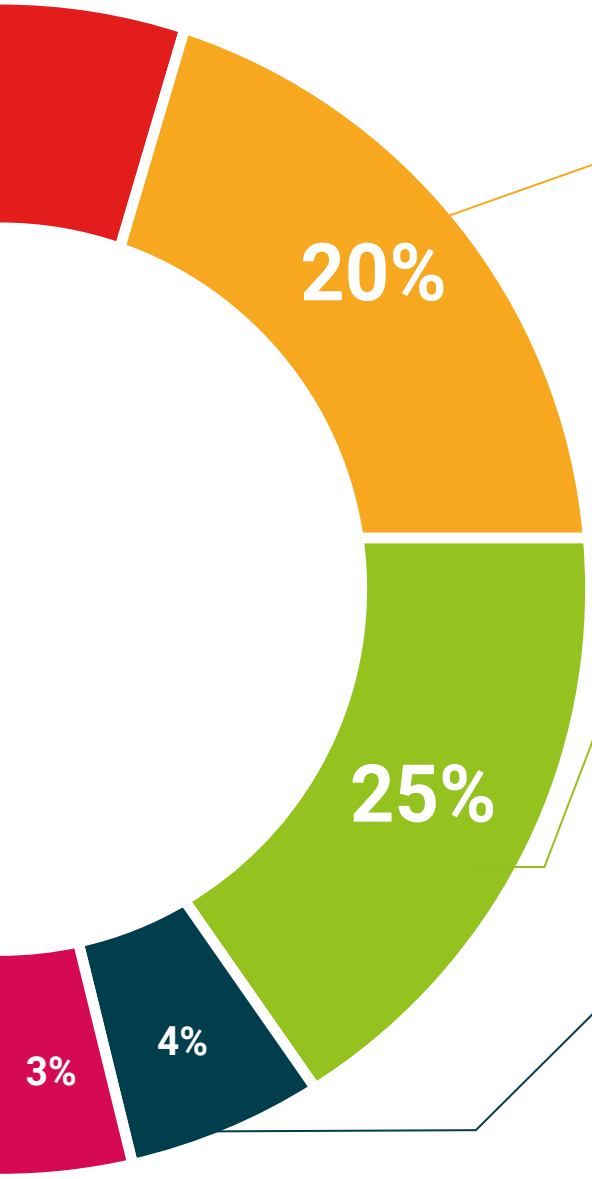
Learning from an Expert strengthens knowledge and memory, and generates confidence in future difficult decisions.

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 Advanced Master's Degree in Software Engineering and Quality guarantees students, in addition to the most rigorous and up-to-date education, access to an Advanced Master's Degree issued by TECH Global University.

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

This program will allow you to obtain your **Advanced Master's Degree diploma in Software Engineering and Quality** endorsed by **TECH Global University**, the world's largest online university.

TECH Global University is an official European University publicly recognized by the Government of Andorra (***official bulletin***). Andorra is part of the European Higher Education Area (EHEA) since 2003. The EHEA is an initiative promoted by the European Union that aims to organize the international training framework and harmonize the higher education systems of the member countries of this space. The project promotes common values, the implementation of collaborative tools and strengthening its quality assurance mechanisms to enhance collaboration and mobility among students, researchers and academics.

This **TECH Global University** title is a European program of continuing education and professional updating that guarantees the acquisition of competencies in its area of knowledge, providing a high curricular value to the student who completes the program.

Title: **Advanced Master's Degree in Software Engineering and Quality**

Modality: **online**

Duration: **2 years**

Accreditation: **120 ECTS**


tech global university

Mr./Ms. _____, with identification document _____
has successfully passed and obtained the title of:

Advanced Master's Degree in Software Engineering and Quality

This is a program of 3,000 hours of duration equivalent to 120 ECTS, with a start date of dd/mm/yyyy and an end date of dd/mm/yyyy.

TECH Global University is a university officially recognized by the Government of Andorra on the 31st of January of 2024, which belongs to the European Higher Education Area (EHEA).

In Andorra la Vella, on the 28th of February of 2024

Dr. Pedro Navarro Illana
Rector

Unique TECH Code: AFWORD23S techtitute.com/certificates


Advanced Master's Degree in Software Engineering and Quality

General Structure of the Syllabus

Year	Subject	ECTS	Type
1º	Software Quality TRL Development Levels	6	CO
1º	Software Project Development. Functional and technical documentation	6	CO
1º	Software Testing. Test Automation	6	CO
1º	Software Project Management Methodologies. Watertall Methodology vs Agile Methodology	6	CO
1º	TDD (Iest-Driven Development). Iest-Driven Software Design	6	CO
1º	DevOps. Software Quality Management	6	CO
1º	DevOps and Continuous Integration. Advanced Practical Solutions in Software Development	6	CO
1º	Database (DB) Design. Standardization and performance. Software Quality	6	CO
1º	Design of Scalable Architectures. The Architecture in the Software life Crcle	6	CO

Year	Subject	ECTS	Type
2º	ISO, IEC 9126 Quality Criteria. Software Quality Metrics	6	CO
2º	Methodologies, Development and Quality in Software Engineering	6	CO
2º	Software Project Management	6	CO
2º	Software Development Platforms	6	CO
2º	Web-Client Computing	6	CO
2º	Web Server Computing	6	CO
2º	Safety Management	6	CO
2º	Software Security	6	CO
2º	Web Server Administration	6	CO
2º	Security Audit	6	CO
2º	Online Application Security	6	CO

Dr. Pedro Navarro Illana
Rector

tech global university

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

tech global university

Advanced Master's Degree
Software Engineering and Quality

- Modality: **online**
- Duration: **2 years**
- Certificate: **TECH Global University**
- Credits: **120 ECTS**
- Schedule: **at your own pace**
- Exams: **online**

Advanced Master's Degree
Software Engineering and Quality

tech global university