Postgraduate Diploma
Software Engineering

tech technological university

tech technological university

Postgraduate Diploma

Software Engineering

- Modality: **online**
- Duration: **6 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Website: www.techtitute.com/pk/information-technology/postgraduate-diploma/postgraduate-diploma-software-engineering

Index

01

Introduction

The professionals will be able to update their knowledge in Software Engineering with the best professionals in the sector. With this high-level education, you will learn the latest trends and developments in the field. A unique opportunity to give your career the boost it needs.

This Postgraduate Diploma will allow you to update your knowledge in Software Engineering in a practical way, 100% Online, without renouncing to the maximum academic rigor”

This program is intended for those interested in achieving a higher level of knowledge in Software Engineering. The main objective is to enable the student to apply in the real world the knowledge acquired in this Postgraduate Diploma, in a work environment that reproduces conditions that can be found in their future, in a rigorous and realistic way.

This Postgraduate Diploma will prepare students for professional practice of Computer Engineering, thanks to a transversal and versatile academic experience adapted to new technologies and innovations in this field. You will obtain wide knowledge in Software Engineering, from the hand of professionals in the sector.

The professional should take advantage of the opportunity and take this program in a 100% online format, without having to give up their obligations. Update your knowledge and get your Postgraduate Diploma Certificate to continue growing personally and professionally.

This **Postgraduate Diploma in Software Engineering** contains the most complete and up-to-date program on the market. The most important features include:

- Development of 100 simulated scenarios presented by experts in Software Engineering
- Its graphic, schematic and practical contents, with which they are conceived gather scientific and practical information on Software Engineering
- News on the latest advances in Software Engineering
- Practical exercises where self-assessment can be used to improve learning
- Interactive learning system based on the case method and its application to real practice
- All of this will be complemented by 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

Learn the latests techniques and strategies with this program and achieve success as an IT Engineer"

This program will enhance your skills and update your Software Engineering knowledge"

Take advantage of the latest educational technology to get updated in Software Engineering without leaving home.

Learn about the latest techniques in Software Engineering from experts in the field.

It includes in its teaching staff a team of professionals belonging to the field of Computer Engineering, who pour into this program their work experience, in addition to recognized specialists belonging to reference societies and prestigious universities.

The multimedia content developed with the latest educational technology will allow the professional a situated and contextual learning, that is, a simulated environment that will provide an immersive learning programmed to prepare for real situations.

This program is designed around Problem-Based Learning, whereby the professional must try to solve different professional practice situations that arise throughout the program. For this, the professional will be assisted by an innovative interactive video system created by recognized experts in Software Engineering with extensive teaching experience.

02

Objectives

The objective of this program is to provide IT professionals with the knowledge and skills necessary to carry out their activity using the most advanced protocols and techniques of the moment. Through a work approach that is fully adaptable to the student, this Postgraduate Diploma will progressively lead them to acquire skills that will propel them to a higher professional level.

Achieve the knowledge level you desire and dominate fundamental concepts in Software Engineering with this high-level program"

General Objectives

- To educate scientifically and technologically, as well as to prepare for professional practice of Computer Engineering, all this with a transversal and versatile training adapted to new technologies and innovations in this field
- To obtain a wide knowledge in computer science field, computer structure and software engineering, including mathematical, statistical and physical basis essential in engineering

Achieve professional success as a computer engineer with this intensive program, developed by professionals with extensive experience in the sector"

Specific Objectives

- To build software engineering and modeling foundations, learning the main processes and concepts
- To understand software process and different models for its development, including agile technologies
- To understand requirements engineering, its development, elaboration, negotiation and validation
- To learn modeling of requirements and different elements such as scenarios, information, analysis classes, flow, behavior and patterns
- To understand concepts and processes of software design, learning also about architecture design and about component-level and pattern-based design
- To know main standards related to software quality and project management
- To know in depth the different agile methodologies used in software engineering
- To learn how to develop using Scrum, extreme programming and reuse-based software development techniques
- To understand different patterns of system architectures and software design, as well as the architecture of cloud applications
- To learn how to test software, with methodologies such as Test-Driven Development, Acceptance Test-Driven Development, Behavior-Driven Development, BDD and Cucumber
- To deepen software development process improvement and software quality improvement using ISO/IEC standards
- To introduce DevOps concept and its main practices
- To understand the importance of requirements engineering in the software development process
- To deepen in sources of requirements and requirements elicitation techniques, as they are an essential part of the process
- To understand and apply prototyping as an essential part of the development process
- To learn how to perform requirements analysis, as well as how to properly document them
- To understand requirements validation and negotiation processes, as well as requirements modeling and management
- To acquire necessary knowledge for management of critical systems and the formal specification of requirements
- To know software engineering framework and the ISO/IEC 12207 standard
- To learn characteristics of unified software development process and planning in the context of agile software development
- To learn different styles of distributed software design and service-oriented software architectures
- To learn essential concepts in graphical user interface design
- To understand the basics of web application development
- To deepen into software testing strategies and techniques, software quality factors and different metrics used

03

Structure and Content

The contents structure has been designed by a team of Computer Engineering professionals, aware of relevance in today's education in order to deepen in this area of knowledge, with the objective of humanistically enriching the student and raising their knowledge level in Software Engineering by means of the latest educational technologies available.

This Postgraduate Diploma in Software Engineering contains the most complete and updated learning program on the market"

Module 1. Software Engineering

1.1. Introduction to Software Engineering and Modeling
- 1.1.1. The Nature of Software
- 1.1.2. The Unique Nature of Webapps
- 1.1.3. Software Engineering
- 1.1.4. The Software Process
- 1.1.5. Software Engineering Practice
- 1.1.6. Software Myths
- 1.1.7. How It All Begins
- 1.1.8. Object-Oriented Concepts
- 1.1.9. Introduction to UML

1.2. The Software Process
- 1.2.1. A General Process Model
- 1.2.2. Prescriptive Process Models
- 1.2.3. Specialized Process Models
- 1.2.4. The Unified Process
- 1.2.5. Personal and Team Process Models
- 1.2.6. What is Agility?
- 1.2.7. What is an Agile Process?
- 1.2.8. Scrum
- 1.2.9. Agile Process Toolkit

1.3. Software Engineering Guiding Principles
- 1.3.1. Principles Guiding the Process
- 1.3.2. Principles Guiding the Practice
- 1.3.3. Principles of Communication
- 1.3.4. Planning Principles
- 1.3.5. Modeling Principles
- 1.3.6. Construction Principles
- 1.3.7. Deployment Principles

1.4. Understanding the Requirements
- 1.4.1. Requirements Engineering
- 1.4.2. Establish the Basis
- 1.4.3. Inquiry of Requirements
- 1.4.4. Development of Cases Studies
- 1.4.5. Elaboration of the Requirements Model
- 1.4.6. Negotiation of Requirements
- 1.4.7. Validation of Requirements

1.5. Requirements Modeling: Scenarios, Information and Analysis Classes
- 1.5.1. Analysis of Requirements
- 1.5.2. Scenario-Based Modeling
- 1.5.3. UML Models that provide the Case Study
- 1.5.4. Data Modeling Concepts
- 1.5.5. Class-Based Modeling
- 1.5.6. Class Diagrams

1.6. Requirements Modeling: Flow, Behavior and Patterns
- 1.6.1. Requirements that Shape Strategies
- 1.6.2. Flow-Oriented Modeling
- 1.6.3. Status Diagrams
- 1.6.4. Creation of a Behavioral Model
- 1.6.5. Sequence Diagrams
- 1.6.6. Communication Diagrams
- 1.6.7. Patterns for Requirements Modeling

1.7. Design Concepts
- 1.7.1. Design in Software Engineering
- 1.7.2. The Design Process
- 1.7.3. Design Concepts
- 1.7.4. Object-Oriented Design Concepts
- 1.7.5. Model of the Design

1.8. Designing the Architecture:
- 1.8.1. Software Architecture
- 1.8.2. Architectural Genres
- 1.8.3. Architectural Styles
- 1.8.4. Architectural Design
- 1.8.5. Evolution of Alternative Designs for Architecture
- 1.8.6. Mapping the Architecture Using the Data Flow

1.9. Component-Level and Pattern-Based Design
- 1.9.1. What is a Component?
- 1.9.2. Class-Based Component Design
- 1.9.3. Realization of the Design at the Component Level
- 1.9.4. Design of Traditional Components
- 1.9.5. Component-Based Development
- 1.9.6. Design Patterns
- 1.9.7. Pattern-Based Software Design
- 1.9.8. Architectural Patterns
- 1.9.9. Design Patterns at the Component Level
- 1.9.10. User Interface Design Patterns

1.10. Software Quality and Project Management
- 1.10.1. Quality
- 1.10.2. Software Quality
- 1.10.3. The Software Quality Dilemma
- 1.10.4. Achieving Software Quality
- 1.10.5. Software Quality Assurance
- 1.10.6. The Administrative Spectrum
- 1.10.7. The Staff
- 1.10.8. The product
- 1.10.9. The Process
- 1.10.10. The Project
- 1.10.11. Principles and Practices

Module 2. Advanced Software Engineering

2.1. Introduction to Agile Methodologies
- 2.1.1. Process Models and Methodologies
- 2.1.2. Agility and Agile Processes
- 2.1.3. Agile Manifesto
- 2.1.4. Some Agile Methodologies
- 2.1.5. Agile vs. Traditional

2.2. Scrum
- 2.2.1. Scrum origins and philosophy
- 2.2.2. Scrum Values
- 2.2.3. Scrum Process Flow
- 2.2.4. Scrum Roles
- 2.2.5. Scrum Artifacts
- 2.2.6. Scrum Events
- 2.2.7. User Stories
- 2.2.8. Scrum Extensions
- 2.2.9. Agile Estimates
- 2.2.10. Scrum Scaling

2.3. Extreme Programming
- 2.3.1. Justification and Overview of XP
- 2.3.2. The XP Life Cycle
- 2.3.3. The Five Core Values
- 2.3.4. The Twelve Basic Practices in XP
- 2.3.5. Roles of Participants
- 2.3.6. XP Industrial
- 2.3.7. Critical Assessment of XP

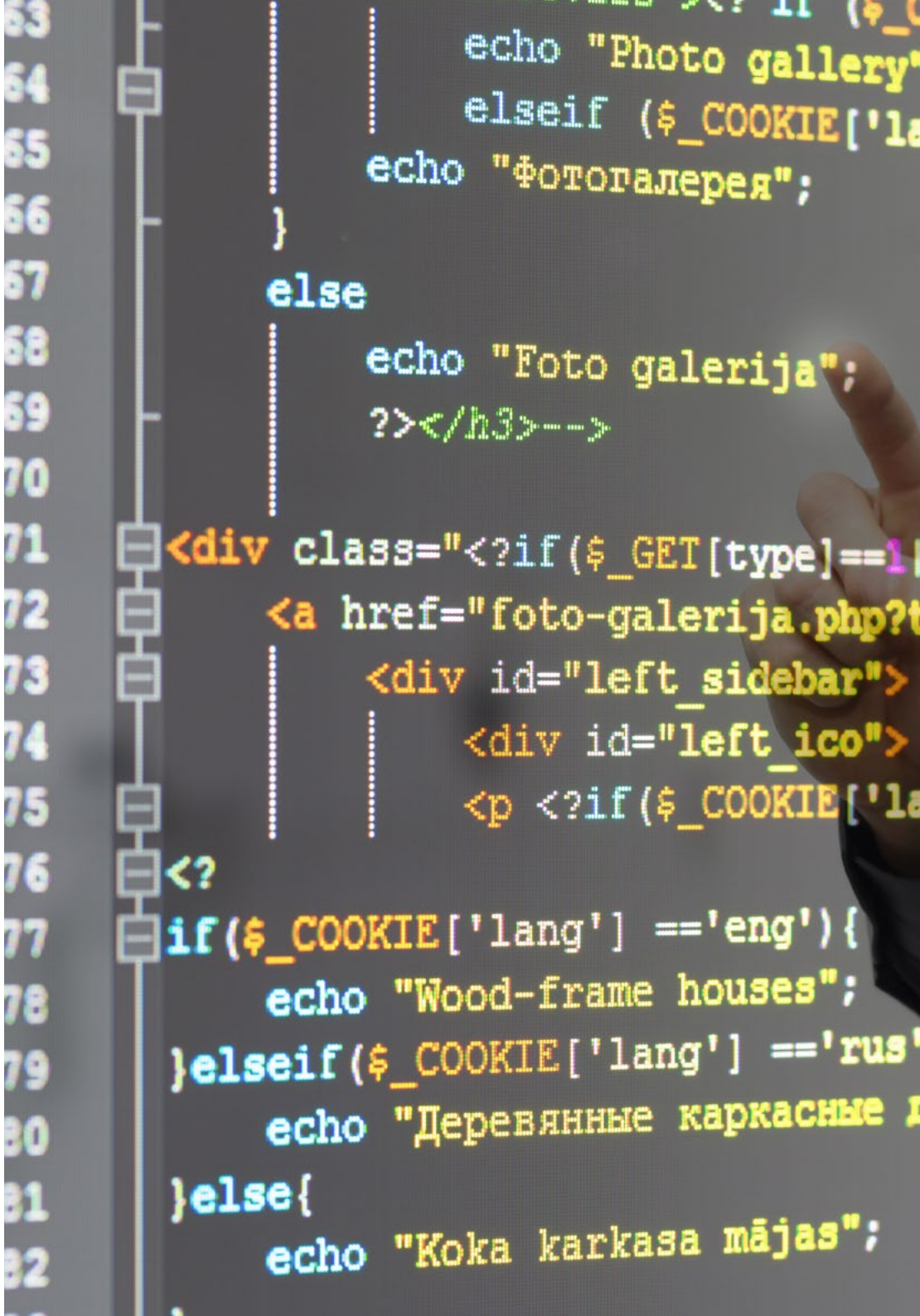
2.4. Software Development Based on Reusability
- 2.4.1. Software Reuse
- 2.4.2. Code Reuse Levels
- 2.4.3. Specific Reuse Techniques
- 2.4.4. Component-Based Development
- 2.4.5. Benefits and Problems of Reuse
- 2.4.6. Reuse Planning

2.5. System Architecture and Software Design Patterns
- 2.5.1. Architectural Design
- 2.5.2. General Architectural Patterns
- 2.5.3. Fault Tolerant Architectures
- 2.5.4. Distributed Systems Architectures
- 2.5.5. Design Patterns
- 2.5.6. Gamma Patterns
- 2.5.7. Interaction Design Patterns

2.6. Cloud Application Architecture
- 2.6.1. Cloud Computing Fundamentals
- 2.6.2. Cloud Application Quality
- 2.6.3. Architectural Styles
- 2.6.4. Design Patterns

2.7. Software Testing: TDD, ATDD and BDD
- 2.7.1. Software Verification and Validation
- 2.7.2. Software Testing
- 2.7.3. Test Driven Development (TDD)
- 2.7.4. Acceptance Test Driven Development (ATDD)
- 2.7.5. Behavior Driven Development (BDD)
- 2.7.6. BDD and Cucumber

2.8. Software Process Improvement
- 2.8.1. Software Process Improvement
- 2.8.2. The Process Improvement Approach
- 2.8.3. Maturity Models
- 2.8.4. The CMMI Model
- 2.8.5. CMMI V2.0
- 2.8.6. CMMI and Agile



2.9. The Quality of the Software Product: Square
- 2.9.1. Software Quality
- 2.9.2. Software Product Quality Models
- 2.9.3. ISO/IEC 25000 Family
- 2.9.4. ISO/IEC 25010: Quality Model and Quality Characteristics
- 2.9.5. ISO/IEC 25012: the Quality of the Data
- 2.9.6. ISO/IEC 25020 Software Quality Measurement
- 2.9.7. ISO/IEC 25022, 25023 and 25024: Software and Data Quality Metrics
- 2.9.8. ISO/IEC 25040 Software Assessment
- 2.9.9. Accreditation Process

2.10. Introduction to DevOps
- 2.10.1. DevOps Concept
- 2.10.2. Core Practices

Module 3. Requirements Engineering

3.1. Introduction to Requirements Engineering
- 3.1.1. The Importance of Requirements
- 3.1.2. Concept of Requirement
- 3.1.3. Dimensions of Requirements
- 3.1.4. Levels and Types of Requirements
- 3.1.5. Requirements Characteristics
- 3.1.6. Requirements Engineering
- 3.1.7. Requirements Engineering Process
- 3.1.8. Frameworks for Requirements Engineering
- 3.1.9. Best Practices in Requirements Engineering
- 3.1.10. The Business Analyst

3.2. Sources of Requirements
- 3.2.1. The Requirements Network
- 3.2.2. The Stakeholders
- 3.2.3. Business Requirements
- 3.2.4. Vision and Scope Document

3.3. Requirements Elicitation Techniques
- 3.3.1. Elicitation of Requirements
- 3.3.2. Problems of Requirements Elicitation
- 3.3.3. Contexts of Discovery
- 3.3.4. Interviews
- 3.3.5. Observation and "Learning
- 3.3.6. Ethnography
- 3.3.7. Workshops
- 3.3.8. Focus groups
- 3.3.9. Questionnaires
- 3.3.10. Brainstorming and Creative Techniques
- 3.3.11. Group Media
- 3.3.12. Analysis of System Interfaces
- 3.3.13. Document Analysis and "Archeology"
- 3.3.14. Case Studies and Scenarios
- 3.3.15. Prototypes
- 3.3.16. Reverse Engineering
- 3.3.17. Reuse of Requirements
- 3.3.18. Good Elicitation Practices

3.4. User Requirements
- 3.4.1. Person
- 3.4.2. Case Studies and User Stories
- 3.4.3. Scenarios
- 3.4.4. Types of Scenarios
- 3.4.5. How to Discover Scenarios

3.5. Prototyping Techniques
- 3.5.1. Prototyping
- 3.5.2. Prototypes According to their Scope
- 3.5.3. Prototypes According to their Seasonality
- 3.5.4. The Fidelity of a Prototype
- 3.5.5. User Interface Prototypes
- 3.5.6. Evaluation of Prototypes

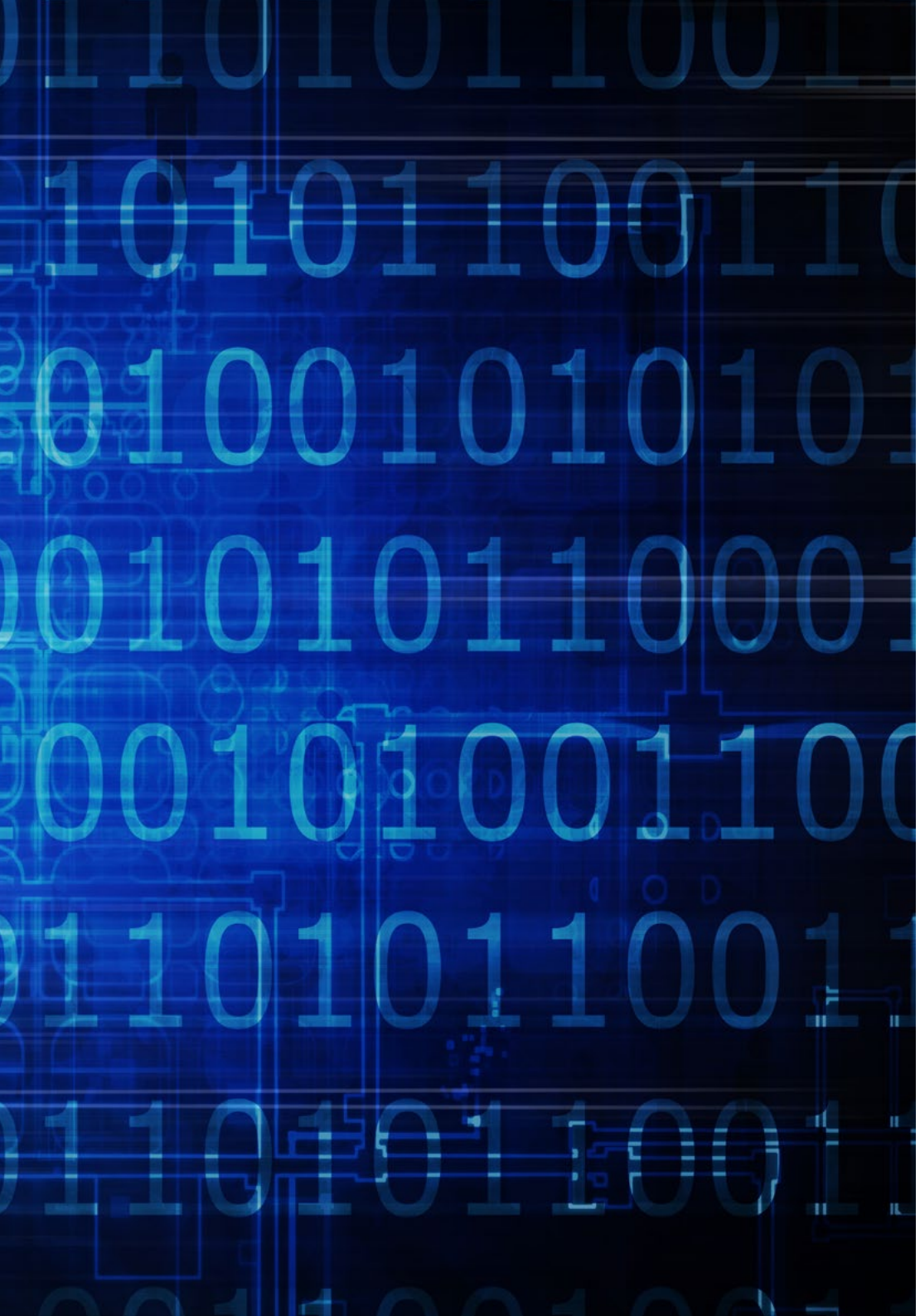
3.6. Requirements Analysis
- 3.6.1. Requirements Analysis
- 3.6.2. Requirements Analysis Best Practices
- 3.6.3. The Data Dictionary
- 3.6.4. Prioritization of Requirements

3.7. Documentation of Requirements
- 3.7.1. The Requirements Specification Document
- 3.7.2. Structure and Contents of an SRS
- 3.7.3. Natural Language Documentation
- 3.7.4. EARS: Easy Approach to Requirements Syntax
- 3.7.5. Non-Functional Requirements
- 3.7.6. Attributes and Templates in Table Form
- 3.7.7. Good Specifications Practices

3.8. Validation and Negotiation of Requirements
- 3.8.1. Validation of Requirements
- 3.8.2. Requirements Validation Techniques
- 3.8.3. Negotiation of Requirements

3.9. Modeling and Requirements Management
- 3.9.1. Requirements Modeling
- 3.9.2. The User's Perspective
- 3.9.3. The Data Perspective
- 3.9.4. The Functional or Flow-Oriented Perspective
- 3.9.5. The Behavioral Perspective
- 3.9.6. Volatility of Requirements
- 3.9.7. Requirements Management Process
- 3.9.8. Tools for Requirements Management
- 3.9.9. Best Practices in Requirements Management

3.10. Critical Systems and Formal Specification
- 3.10.1. Critical Systems
- 3.10.2. Risk-Driven Specification
- 3.10.3. Formal Specification



Module 4. Software Engineering Processes

4.1. Software Engineering Framework
- 4.1.1. Software Features
- 4.1.2. The Main Processes in Software Engineering
- 4.1.3. Software Development Process Models
- 4.1.4. Standard Reference Framework for the Software Development Process: The ISO/IEC 12207 Standard

4.2. Unified Software Development Process
- 4.2.1. The Unified Process
- 4.2.2. Dimensions of the Unified Process
- 4.2.3. Case Studies Driven Development Process
- 4.2.4. Fundamental Workflows of Unified Processes

4.3. Planning in the Context of Agile Software Development
- 4.3.1. Characteristics of Agile Software Development
- 4.3.2. Different Planning Time Horizons in Agile Development
- 4.3.3. Scrum Agile Development Framework and Planning Time Horizons
- 4.3.4. User Stories as a Planning and Estimating Unit
- 4.3.5. Common Techniques for Deriving an Estimate
- 4.3.6. Scales for Interpreting Estimates
- 4.3.7. Planning Poker
- 4.3.8. Common Scheduling Types: Delivery Scheduling and Iteration Scheduling

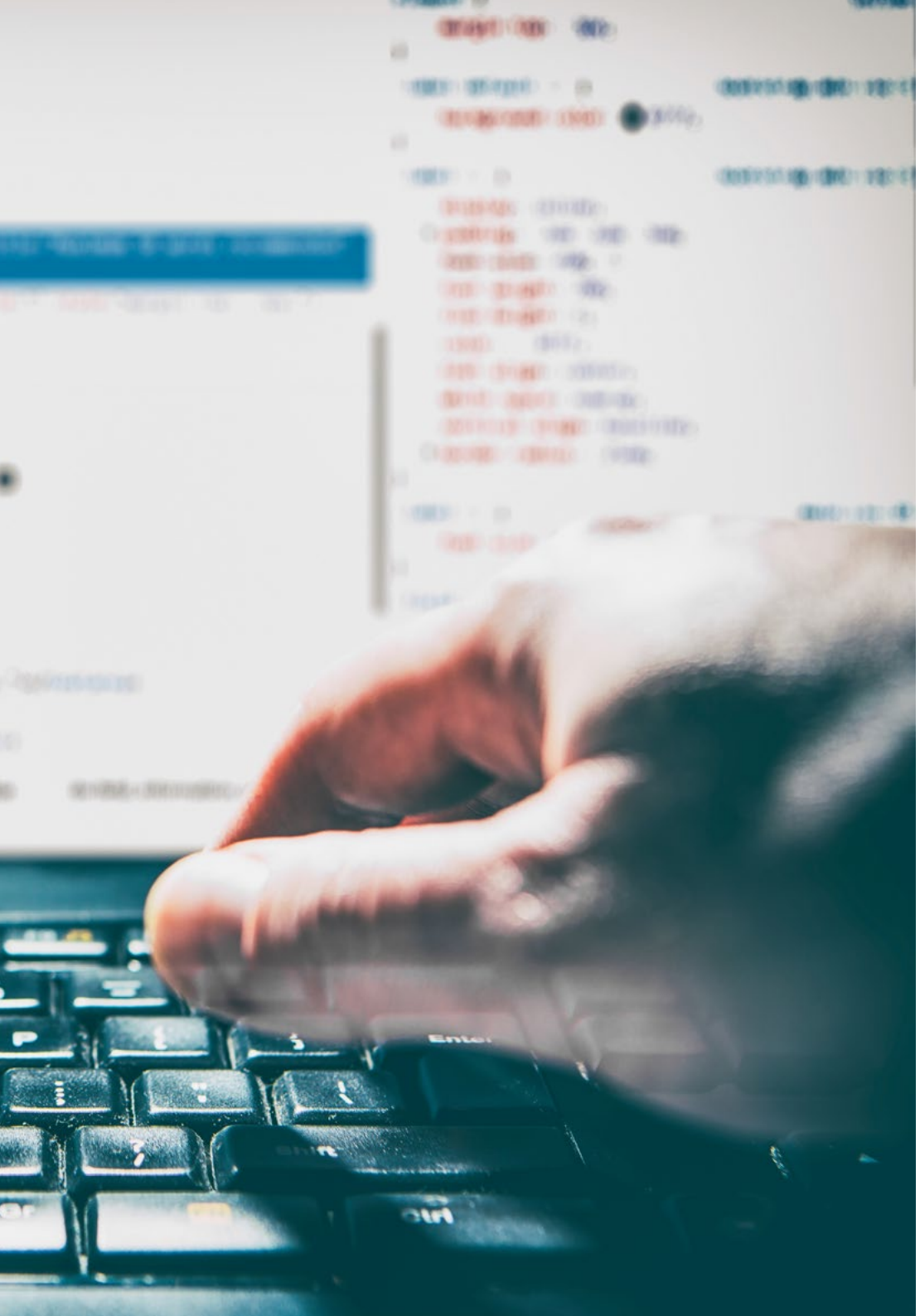
4.4. Distributed Software Design Styles and Service-Oriented Software Architectures
- 4.4.1. Communication Models in Distributed Software Systems
- 4.4.2. Middleware
- 4.4.3. Architecture Patterns for Distributed Systems
- 4.4.4. General Software Service Design Process
- 4.4.5. Design Aspects of Software Services
- 4.4.6. Composition of Services
- 4.4.7. Web Services Architecture
- 4.4.8. Infrastructure and SOA Components

4.5. Introduction to Model Driven Software Development
- 4.5.1. The Model Concept
- 4.5.2. Model-Driven Software Development
- 4.5.3. MDA Model-Driven Development Framework
- 4.5.4. Elements of a Transformation Model

4.6. Graphical User Interface Design
- 4.6.1. Principles of User Interface Design
- 4.6.2. Architectural Design Patterns for Interactive Systems: Model View Controller (MVC)
- 4.6.3. UX User Experience
- 4.6.4. User-Centered Design
- 4.6.5. Graphical User Interface Analysis and Design Process
- 4.6.6. Usability of User Interfaces
- 4.6.7. Accessibility in User Interfaces

4.7. Web Application Design
- 4.7.1. Characteristics of Web Applications
- 4.7.2. Web Application User Interface
- 4.7.3. Navigation Design
- 4.7.4. Basic Interaction Protocol for Web Applications
- 4.7.5. Architecture Styles for Web Applications

4.8. Software Testing Strategies and Techniques and Software Quality Factors
- 4.8.1. Testing Strategies
- 4.8.2. Test Case Designs
- 4.8.3. Value for Money
- 4.8.4. Quality Models
- 4.8.5. ISO/IEC 25000 Family of Standards (SQuaRE)
- 4.8.6. Product Quality Model (ISO 2501n)
- 4.8.7. Data Quality Models (ISO 2501n)
- 4.8.8. Software Quality Management



4.9. Introduction to Software Engineering Metrics
- 4.9.1. Basic Concepts: Measurements, Metrics and Indicators
- 4.9.2. Metric Types in Software Engineering
- 4.9.3. The Measurement Process
- 4.9.4. ISO 25024. External and Quality Metrics in Use
- 4.9.5. Object-Oriented Metrics

4.10. Software Maintenance and Reengineering
- 4.10.1. Maintenance Process
- 4.10.2. Standard Maintenance Process Framework. ISO/EIEC 14764
- 4.10.3. Software Reengineering Process Model
- 4.10.4. Inverse Engineering

A unique, key, and decisive educational experience to boost your professional development"

04

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

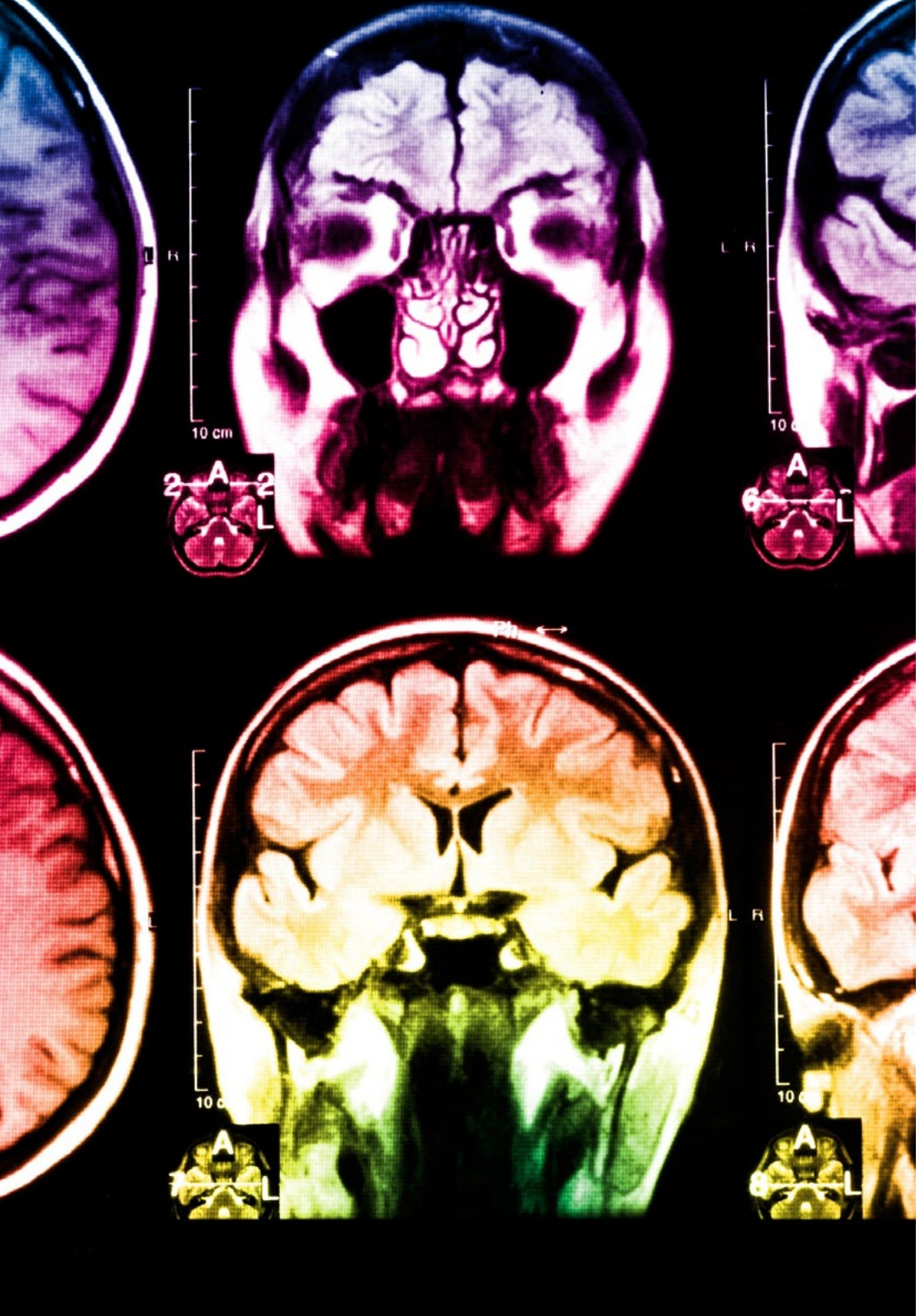
TECH effectively combines the Case Study methodology with a 100% online learning system based on repetition, which combines different teaching elements in each lesson.

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.

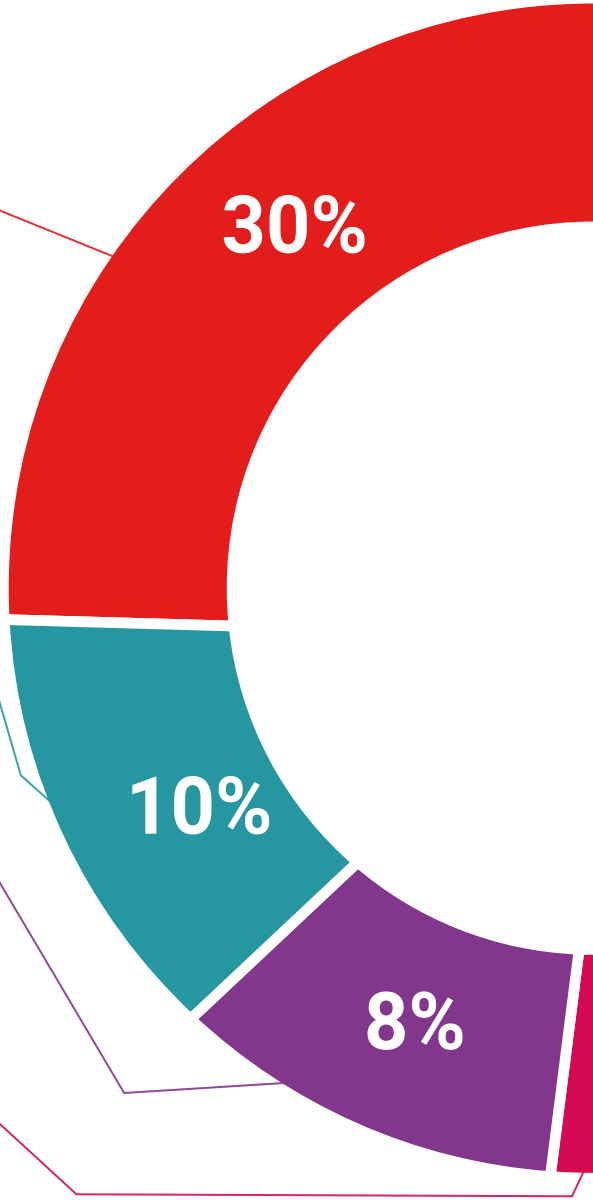
Learning from an Expert strengthens knowledge and memory, and generates confidence in future difficult decisions.

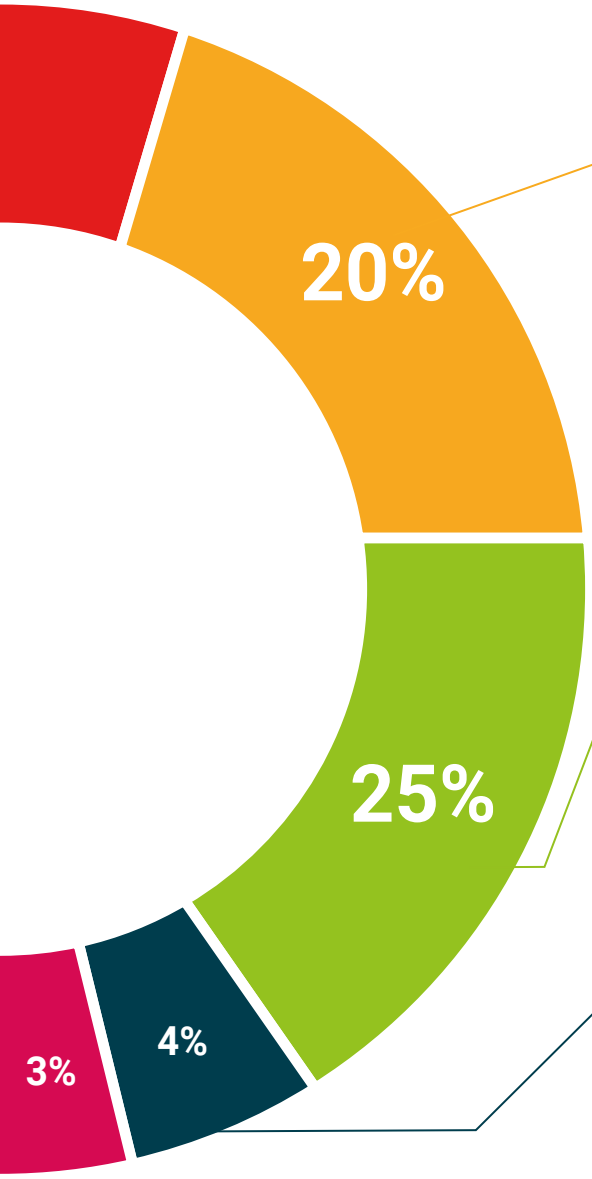
Practising Skills and Abilities

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.

05

Certificate

The Postgraduate Diploma in Software Engineering guarantees students, in addition to the most rigorous and up-to-date education, access to a Postgraduate Certificate issued by TECH Technological University.

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

This **Postgraduate Diploma in Software Engineering** contains the most complete and up-to-date program on the market.

After the student has passed the assessments, they will receive their corresponding **Postgraduate Diploma** issued by **TECH Technological University** via tracked delivery*.

The certificate issued by **TECH Technological University** will reflect the qualification obtained in the Postgraduate Diploma, and meets the requirements commonly demanded by labor exchanges, competitive examinations, and professional career evaluation committees.

Title: **Postgraduate Diploma in Software Engineering**

Official Nº of Hours: **600 h.**


tech technological university

Awards the following

CERTIFICATE

to

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

For having passed and accredited the following program

POSTGRADUATE DIPLOMA

in

Software Engineering

This is a qualification awarded by this University, equivalent to 600 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

Unique TECH Code: AFWORD23S techtitute.com/certificates

*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

Postgraduate Diploma
Software Engineering

- Modality: **online**
- Duration: **6 months**
- Certificate: **TECH Technological University**
- Dedication: **16h/week**
- Schedule: **at your own pace**
- Exams: **online**

Postgraduate Diploma
Software Engineering

tech technological university