FACULDADE ENGENHARIA
Departamento de Informática

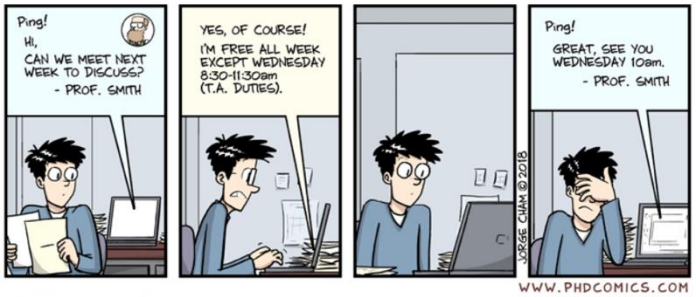
# Engenharia de Software
## (14341, 16230, 15386)

### Course Outline 2025/26

# Lecturers

- Nuno Pombo, PhD
  - ngpombo@di.ubi.pt
  - Personal webpage:
  https://www.di.ubi.pt/~ngpombo/
  - Course webpage :
  https://www.di.ubi.pt/~ngpombo/units/se.html
  - Office Hours:
  Mondays 10:30 AM–12:30 PM, or by appointment. Lecturer's office (Room 4.10).

- Carlos Cunha, PhD
  - carlos.cunha@di.ubi.pt




**footnote: Thanks to Laura from Queens U. for this comic idea!

# Challenges

- A software engineering project is fundamentally a **team endeavor**; being a solitary programming expert is not enough. Competent software engineers must demonstrate proficiency in **communication** and **collaboration**.

- While programming can often be an individual activity, software engineering is inherently **collaborative**. It is deeply intertwined with professionalism, **teamwork**, and effective **communication**.

# Challenges

- A software engineering project generally has an **expected lifespan**; it needs to work correctly for months, years, or even decades.
- **Features** may be added or removed to meet evolving product requirements.
- The engineering team itself is likely to **change**. The technological context will evolve as our computing platforms advance, programming languages **change**, dependencies are updated, etc.
- This exposure to time and change is **unprecedented** compared to a typical programming project: it is not enough to build something that works—it must work and keep working over time.

# Challenges

- The integration of **AI tools** in the development lifecycle (e.g., for code generation, test generation, refactoring, and documentation) introduces new dependencies and learning curves. Developers must now critically evaluate AI-generated outputs and understand their limitations.
- The use of autonomous or semi-autonomous AI agents (e.g., DevOps bots, copilots, or orchestrators) requires new forms of **coordination, governance, and accountability**. Engineers must ensure these agents align with the project's goals and do not propagate technical debt or security risks.
- Teams must learn to **collaborate effectively with AI assistants**, rebalancing roles and responsibilities. Engineers must refine their critical thinking to question, validate, or override AI decisions.
- Developers and software engineers are expected to acquire **new competences** related to prompt engineering, AI tool orchestration, and AI system evaluation, in addition to traditional software engineering knowledge.

# Intended learning outcomes - Knowledge

**K1**: Explain the principles, processes, and lifecycle models used in modern software engineering.

**K2**: Describe Agile development methods, including Scrum, and their application in iterative software development.

**K3**: Identify and understand key software requirements engineering concepts, including user stories, scenarios, and model-driven approaches.

**K4**: Recognize fundamental system design principles, modeling techniques (UML, SysML), and architectural decisions.

**K5**: Explain the concepts and role of quality assurance, automated testing, and continuous integration in the software lifecycle.

**K6**: Understand the impact of intelligent tools and AI in software engineering practices such as testing, refactoring, and code generation.

**K7**: Identify code smells and describe strategies and tools for automated refactoring.

**K8**: Understand the structure and function of modern engineering platforms, including DevOps, pipelines, and AI-augmented software workflows.

# Intended learning outcomes - Skills

**S1**: Gather and specify software requirements through user stories and model-driven techniques.

**S2**: Apply Agile practices to plan, track, and reflect on project development through sprints and iterations.

**S3**: Model software behavior and structure using system modeling tools (e.g., use case diagrams).

**S4**: Implement basic and intermediate-level testing (unit, integration, exploratory) including using AI-assisted tools for automated test generation.

**S5**: Analyze software for quality concerns and apply intelligent tools to detect and refactor code smells.

**S6**: Design and operate a basic CI/CD pipeline with integrated quality checks and automation tools.

**S7**: Use intelligent assistants responsibly in the context of software development activities.

**S8**: Communicate effectively within teams, participating in retrospectives, planning meetings, and demos.

# Intended learning outcomes - Competences

**C1**: The ability to design, develop, and deliver a software system through iterative sprints in a team environment using Scrum.

**C2**: Proficiency in using collaborative tools (e.g., GitHub, issue trackers, DevOps pipelines) in a software engineering workflow.

**C3**: Critical thinking and reflection in evaluating the effectiveness of tools, practices, and personal/team progress.

**C4**: Awareness of ethical and sustainable considerations in modern software engineering, including the responsible use of AI.

**C5**: Lifelong learning capabilities to adapt to evolving software practices, tools, and technologies through self-regulated learning and peer feedback.

# Syllabus

1. Introduction to Software Development and Software Products
2. Lifecycle Models: Waterfall, Iterative, Incremental
3. Software Processes and Agile Development
4. Features, Scenarios, and User Stories
5. Model-Driven Requirements Engineering
6. System Architecture
7. System Modeling
8. Software Testing and AI-Assisted Testing
9. Automated Refactoring and Code Smells with Intelligent Tools
10. Continuous Integration and Software Engineering Platforms
11. AI-Assisted Software Engineering

# Assessment Criteria

- During the **teaching-learning period:**

  **Theoretical Component** (12 points. Minimum passing grade: 6 points (50%))

  2 Written Exams:

  » [F1] 1st Exam (5 points): **31.10.2025**, 9h, Room: 6.01

  » [F2] 2nd Exam (6 points): **19.12.2025**, 9h, Room: 6.01

  » [B] Badge to be completed during a theoretical lecture, and a certification in SCRUM (1 point)

  ***IMPORTANT:*** *A student who, due to their working student status, is unable to complete the badges during class time must contact the instructor to schedule an alternative date and time for their completion.*

  **Practical Component (**8 points. Minimum passing grade: 4 points (50%))

  [P] Project to be developed throughout the semester

  **Final Grade = F1*5/20 + F2*6/20 + P*8/20 + B**

- In **exam period**:
  - [E] Written exam (12 points, minimum score 6 points)
  - [P] *The practical component is* ***mandatory*** *and carries over to the exam period; however, it may be subject to new requirements and/or extended scope as determined by the instructor.*

# Relevance of Soft skills

The dispositions identified by multiple knowledge areas in CS2023(*) as essential for Computer Science graduates include:

- **Adaptable**, as the subject matter is constantly evolving
- **Collaborative**, since most real-world applications are team efforts
- **Inventive**, to create new solutions and apply existing ones in new contexts
- **Meticulous**, to ensure accuracy and completeness of solutions
- **Persistent**, because solving computational problems is an iterative process
- **Proactive**, to anticipate issues related to usability, security, ethics, etc.
- **Responsible** in all aspects of a solution, including design, implementation, and maintenance
- **Self-directed**, as lifelong learning is necessary due to the rapid evolution of each discipline

(*) Amruth N. Kumar, Rajendra K. Raj, Sherif G. Aly, Monica D. Anderson, Brett A. Becker, Richard L. Blumenthal, Eric Eaton, Susan L. Epstein, Michael Goldweber, Pankaj Jalote, Douglas Lea, Michael Oudshoorn, Marcelo Pias, Susan Reiser, Christian Servin, Rahul Simha, Titus Winters, and Qiao Xiang. 2024. *Computer Science Curricula 2023*. Association for Computing Machinery, New York, NY, USA.

# Main bibliography


- *Engineering AI Systems: Architecture and DevOps Essentials,* Len Bass et. al., Pearson, 2025.
- *Engineering Software Products: An Introduction to Modern Software Engineering*, Ian Sommerville, Pearson, 2020.
- *Refactoring* (2nd Edition), Martin Fowler, Addison-Wesley Signature Series, 2018.
- *Software Engineering: International Version* (10th Edition), Ian Sommerville, Pearson, 2015.
- *Introdução à Engenharia de Software*, Sérgio Guerreiro, FCA, 2015.

WHY
WHERE
WHEN
WHERE
WHY
WHO
WHEN
WHAT
WHAT
WHAT
WHY
WHEN
WHO
WHERE
HOW
WHY