

How to design a gamified class

1. Define the Central Theme: The chosen theme should be relevant to the course content and engage students in a narrative related to the topics covered in the class.

Example: The chosen theme is "Software Development Mission for a Critical System". Students take on the role of software engineers on a development team tasked with building a critical system for a global company.

2. Develop the Narrative: The teacher should write the gamified lesson, following the phases established by the process (Section \ref{processo GDS}) to create a learning experience according to the indicated structure

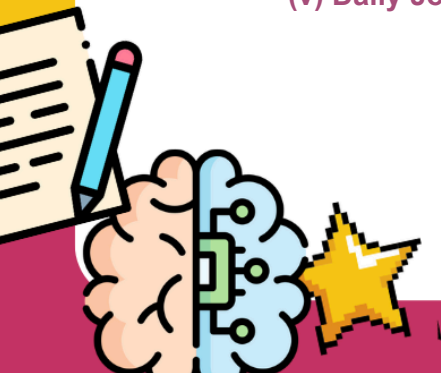
(i) Introduction: In the Software Engineering class, students take on the role of engineers from a fictitious company called TechCorps. The story begins with TechCorps being hired to develop a crisis management system. The game's rules are explained, such as the scoring system based on the quality of project deliverables, team collaboration, and time limits for

(ii) Routine: Students, now part of the TechCorps development team, participate in their first activities as software engineers. They are assigned to different departments within the team, such as requirements gathering, architecture modeling, and module implementation. Collaborative bonds between team members begin to form, and their motivations for the project's success are outlined.

(iii) Event: In the TechCorps narrative, the "villain" appears in the form of a critical problem with the client - constant changes in requirements and increasingly tight deadlines. Students must deal with revisions and redesigns of the software project, which challenges their adaptation and time management skills. They also face technical obstacles, such as unexpected bugs and difficulties in system integration.

(iv) Call to Adventure: Students, now in the middle of the system development journey, are summoned for specific missions to refine the project scope, improve the proposed architecture, and perform unit and integration tests. Progress is monitored by a phase map, where each completed mission unlocks the next stage of development. With each success, students collect points and badges, such as "Architecture Specialist" or "Tester Master".

(v) Daily Journey (three missions):



- Mission 1 - "Requirements Gathering". Students receive an initial set of vague and incomplete requirements from the client. They must use elicitation techniques to clarify the requirements, document them appropriately, and define acceptance criteria.
- Mission 2 - "Architectural Design". Based on the requirements gathered, students must design the system architecture using UML (Unified Modeling Language), ensuring that the solution meets the specified functional and non-functional needs.
- Mission 3 - "Implementation and Testing". The student team is divided into subgroups, each responsible for implementing a system module and performing unit and integration tests.

(vi) Final Challenge: This occurs when the team must deliver the complete system to the client in a final presentation. However, a critical problem arises: a bug in the system during final testing threatens to compromise the delivery. Students must work as a team to identify and resolve the problem before the deadline, applying the testing and debugging concepts they have learned.

(vii) Finalization: With the TechCorps system successfully delivered, students are awarded distinctions such as "Master of Software Development" and "Project Management Specialist". The narrative ends with the students returning to "normal life", reflecting on the lessons learned.

This structure provides a practical example of how a software engineering teacher can develop each phase of the process, guiding students through a narrative journey while they learn and apply concepts from the discipline.

After creating the story, the teacher can create slides and printed materials and use items such as costumes, decorations, or any element that helps to make the class even more fun.

