Project of Software Development

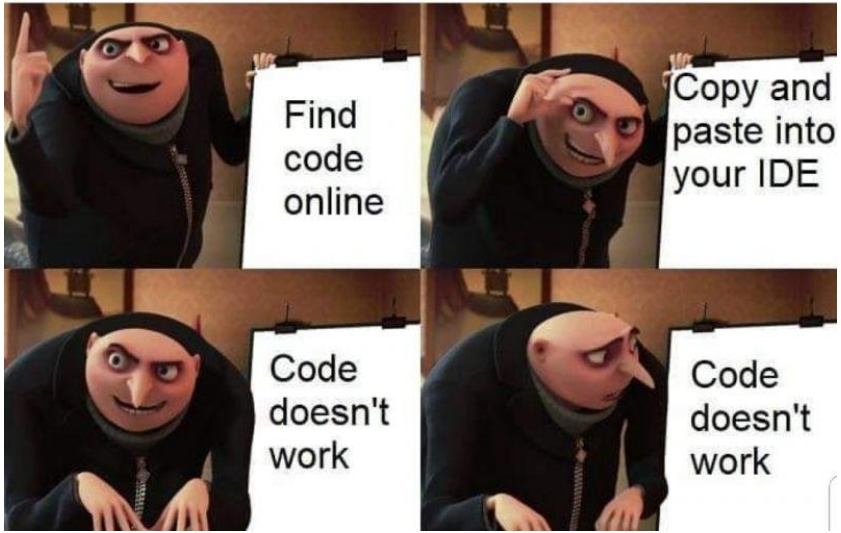
Lesson 2 - 30Jan25
Software Engineering

ISEG Lisbon School of Economics & Management Universidade de Lisboa
U LISBOA | UNIVERSIDADE DE LISBOA

Lecture Topic

Software Engineering:

- Project, Software, and Development
- Software Engineering Layers
- Software Engineering Processes
- Software Architecture
- Architecture Styles



Bibliography:

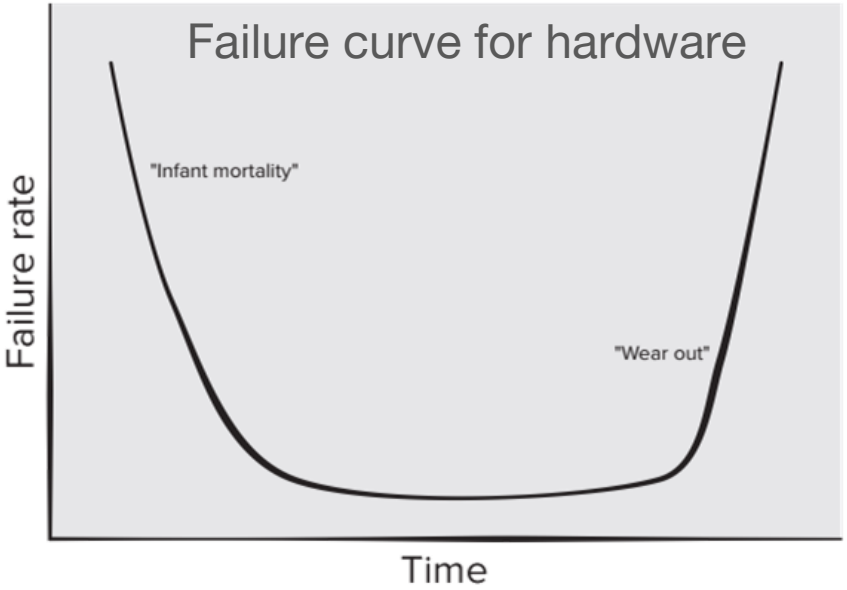
- Pressman, Roger S. "Software Engineering - A Practitioner's Approach, 9th Edition", chapters 1, 2, 10 (McGraw-Hill) 2020

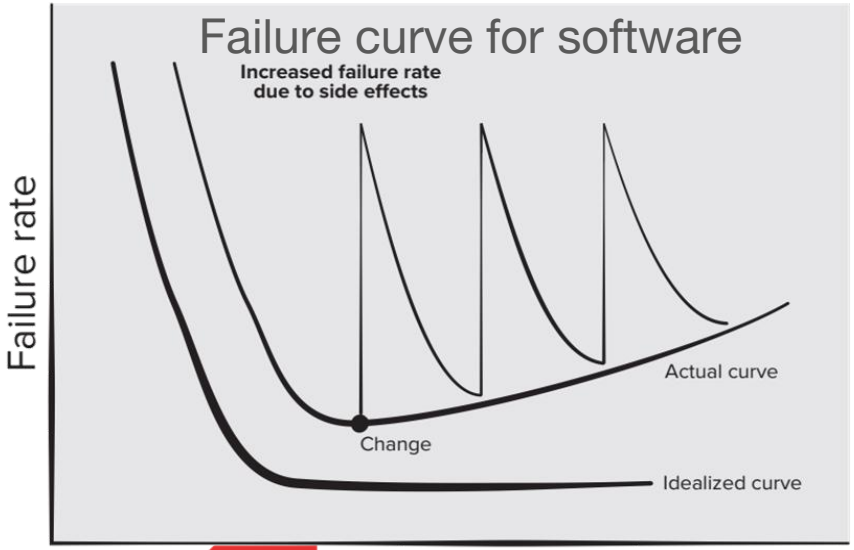
Project of Software Development

- Project? of Software Development
  - Will be detailed in the course IT Project Management: “a temporary endeavor undertaken to create a unique project service or result.” [PMBOK]
- Project of Software? Development
  - (1) instructions (computer programs) that when executed provide desired features, function, and performance
  - (2) data structures that enable the programs to adequately manipulate information
  - (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs [Pressman]

Project of Software Development

- Project of **Software** Development
  - Software fails in a different way to hardware: software doesn't "wear out"



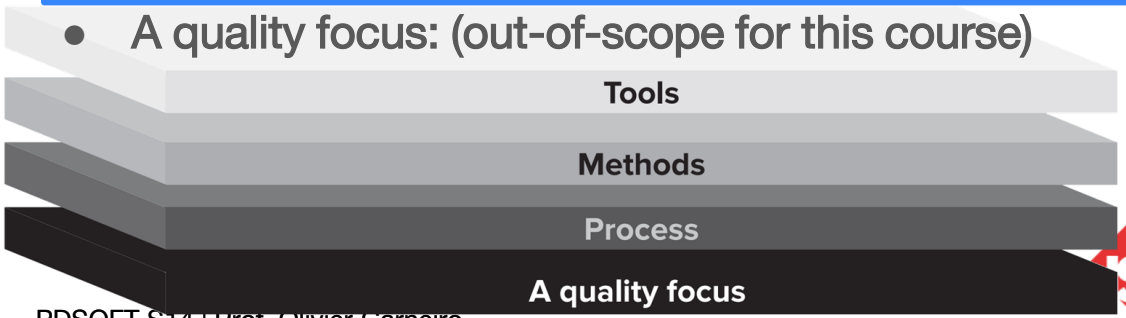


Project of Software Development

- **Project** of Software Development
  - Will be detailed in the course IT Project Management: “a temporary endeavor undertaken to create a unique project service or result.” [PMBOK]
- Project of **Software** Development
  - (1) instructions (computer programs) that when executed provide desired features, function, and performance
  - (2) data structures that enable the programs to adequately manipulate information
  - (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs [Pressman]
- Project of Software **Development?**
  - Part of **Software Engineering**: The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software [SWEBOK]

Software Engineering Layers

- **Tools:** Provide automated or semi-automated support for the process and the methods. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called computer-aided software engineering, is established
- **Methods:** Provide the technical how-to's for building software. Methods encompass a broad array of tasks that include communication, requirements analysis, design modeling, program construction, testing, and support
- **Process:** Forms the basis for management control of software projects and establishes the context in which technical methods are applied, work products (models, documents, data, reports, forms, etc.) are produced, milestones are established, quality is ensured, and change is properly managed
- A quality focus: (out-of-scope for this course)

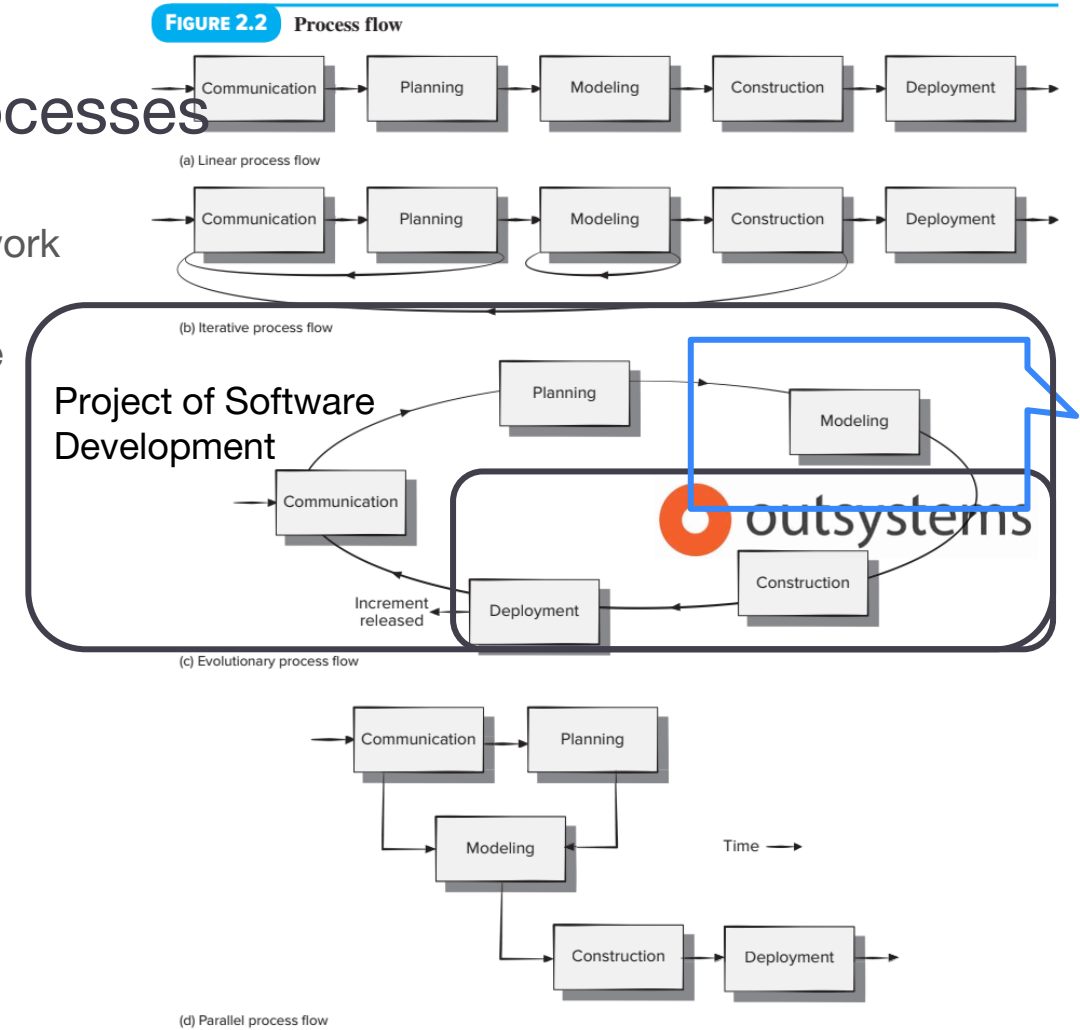


Software Engineering Processes

- Software Engineering Process: a **framework** for the activities, actions, and tasks required to build high-quality software
- A generic process **framework** for software engineering defines five framework activities: communication, planning, modeling, construction, and deployment
- Process flow: describes **how the framework** activities and the actions and tasks that occur within each framework activity **are organized** with respect to sequence and time
- Examples:

Software Engineering Processes

Process flow: describes how the framework activities and the actions and tasks that occur within each framework activity are organized with respect to sequence and time



Software Engineering Process: Modeling

Software Architecture is the set of **significant decisions** about:

- The organization of a software system
- The selection of the structural elements and their interfaces by which the system is composed
- Their behavior as specified in the collaborations among those elements
- The composition of these structural and behavioral elements into progressively larger subsystems
- The **architectural style** that guides this organization: these elements and their interfaces, their collaborations, and their composition [The Unified Modeling Language User Guide, Addison Wesley, 1999]

Architectural Styles

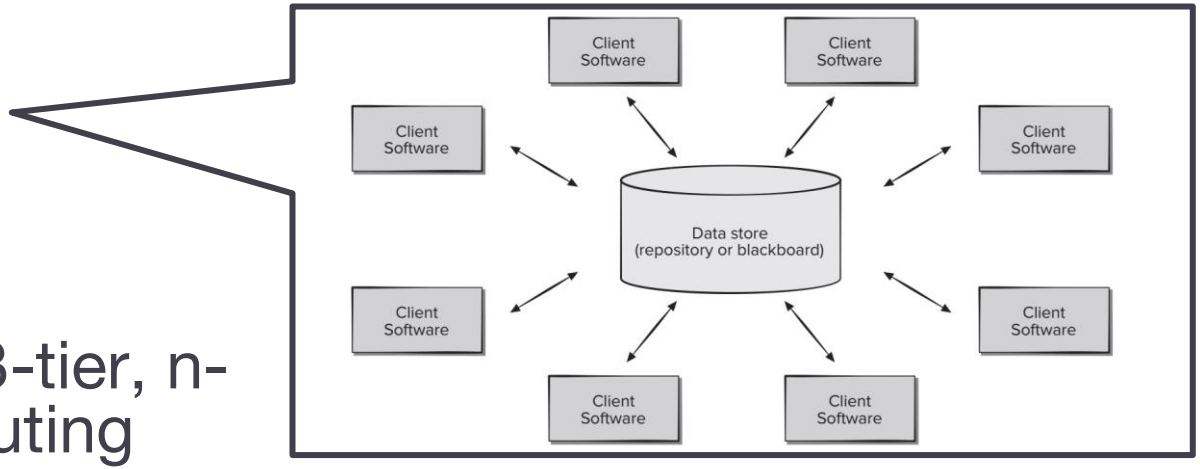
Each architectural style describes a system category that encompasses

- A set of components (e.g., a database, computational modules) that perform a function required by a system
- A set of connectors that enable “communication, coordination and cooperation” among components
- Constraints that define how components can be integrated to form the system
- Semantic models that enable a designer to understand the overall properties of a system by analyzing the known properties of its constituent parts

[Pressman, chapter 10]

Architectural Styles

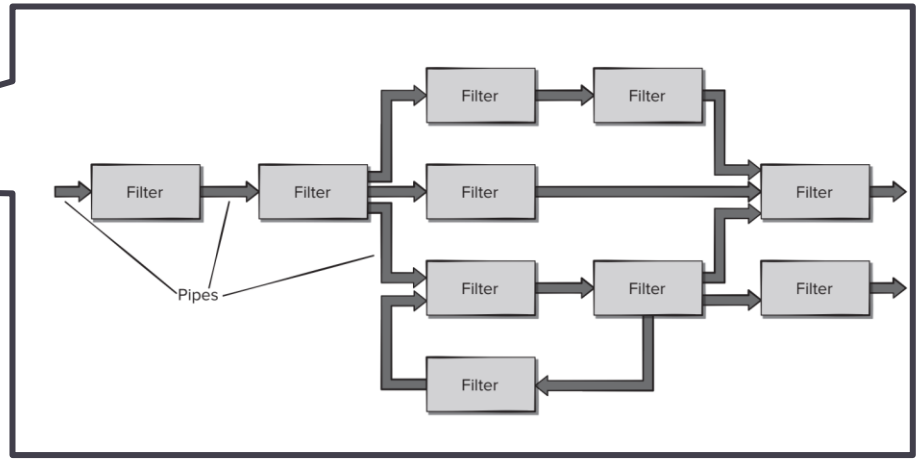
1. Data-Centered
2. Data-Flow
3. Call-and-Return
4. Object-Oriented
5. Layered: 2-tier, 3-tier, n-tier, cloud computing



Data-Centered Architectures: a data store (e.g., a file or database) resides at the center of this architecture and is accessed frequently by other components that update, add, delete, or otherwise modify data within the store. Ex: MS Access, Oracle Forms.
Remember: each architectural style describes a system category that encompasses (1) set of components (ex: database, computational modules) that perform a function required by a system, (2) set of connectors that enable “communication, coordination and cooperation” among components, (3) constraints that define how components can be integrated to form the system, and (4) semantic models

Architectural Styles

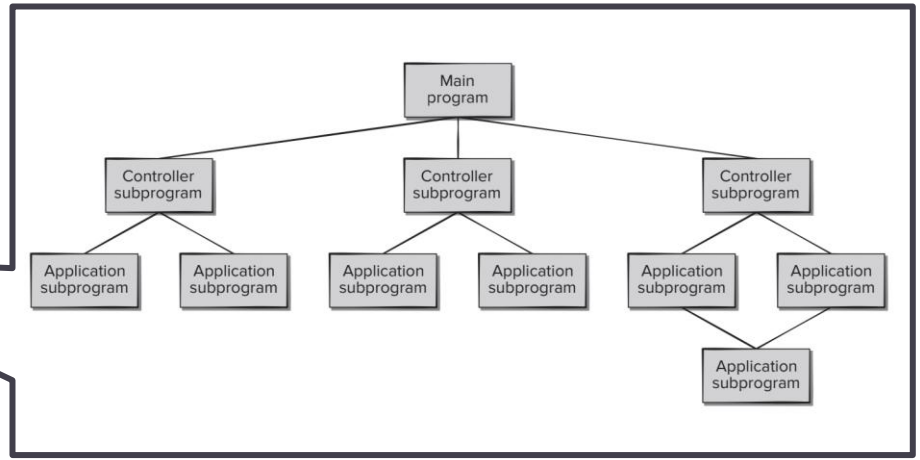
1. Data-Centered
2. Data-Flow
3. Call-and-Return
4. Object-Oriented
5. Layered: 2-tier, 3-tier, n-tier, cloud computing



Data-Flow Architectures: is applied when input data are to be transformed through a series of computational or manipulative components into output data. A pipe-and-filter pattern has a set of components, called filters, connected by pipes that transmit data from one component to the next. Ex: digital signal processing, network routing, graphics processing, telemetry
Remember: each architectural style describes a system category that encompasses (1) set of components (ex: database, computational modules) that perform a function required by a system, (2) set of connectors that enable “communication, coordination and cooperation” among components, (3) constraints that define how components can be integrated to form the system, and (4) semantic models

Architectural Styles

1. Data-Centered
2. Data-Flow
3. Call-and-Return
4. Object-Oriented
5. Layered: 2-tier, 3-tier, n-tier, cloud computing



Call-and-Return Architectures: enables a program structure that is relatively easy to modify and scale. Two substyles that exist within this category: (a) Main program/subprogram architectures; (b) Remote procedure call architectures. Ex: your PHP/Python programs
Remember: each architectural style describes a system category that encompasses (1) set of components (ex: database, computational modules) that perform a function required by a system, (2) set of connectors that enable “communication, coordination and cooperation” among components, (3) constraints that define how components can be integrated to form the system, and (4) semantic models

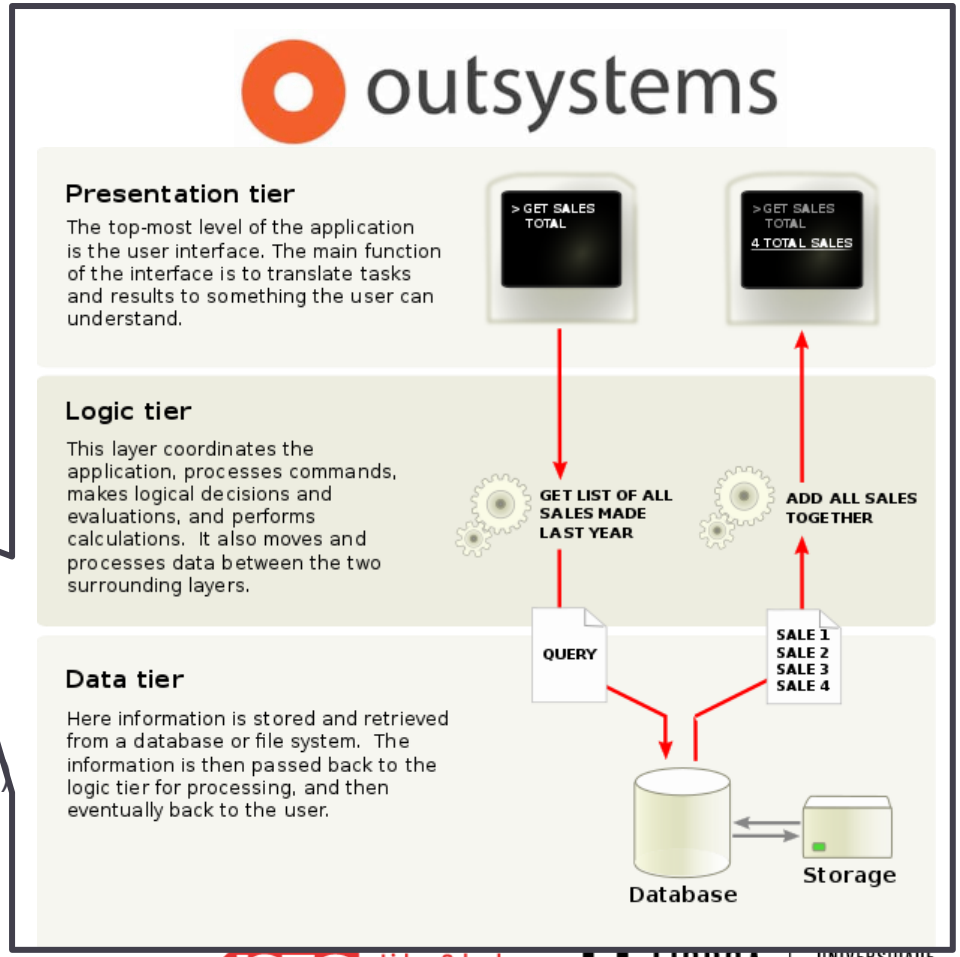
Architectural Styles

1. Data-Centered
2. Data-Flow
3. Call-and-Return
4. Object-Oriented
5. **Layered:** 2-tier, **3-tier**, n-tier, cloud computing

3-tier architecture is a client-server software architecture style in which:

- The user interface (presentation)
- The functional process logic ("business rules")
- Computer data storage and data access

are developed and maintained as independent modules, most often on separate platforms



Learning Goals

- Software Development is part of Software Engineering
- Software Engineering is layered in: tools, methods, and process
  - OutSystems is a tool to, amongst other features, develop software and deploy it.
- Software Engineering Process is a framework for the activities, actions, and tasks required to build high-quality software
- A generic process framework for software engineering defines five framework activities: communication, planning, modeling, construction, and deployment
  - Construction (Development) is the primary goal of this course
- Modeling includes defining a Software Architecture within an Architecture Style.
- Most common corporate software uses Layered Architecture Style
  - OutSystems allows the development of software using a 3-tier Architecture (Interface, Business Logic and Data)

Homework

- Create your groups with 5 members in e-Business
- The groups for e-Business will be the same for this course
  - Students not enrolled in e-Business must join another existing group
  - Who is not enrolled in e-Business?
- Register your group at Fenix, again for this course

Homework

Léo Andrade [Aprenda OutSystems]
https://www.youtube.com/watch?v=tjN2hVv8GTY&list=PLY-9oEzuBhdeqnuD66kykLg-779qOiM3K
Set subtitles in your preferred language

1. “#02 Service Studio overview” (13 minutes)
2. “#03 Primeiras páginas e navegação” (16 minutes)

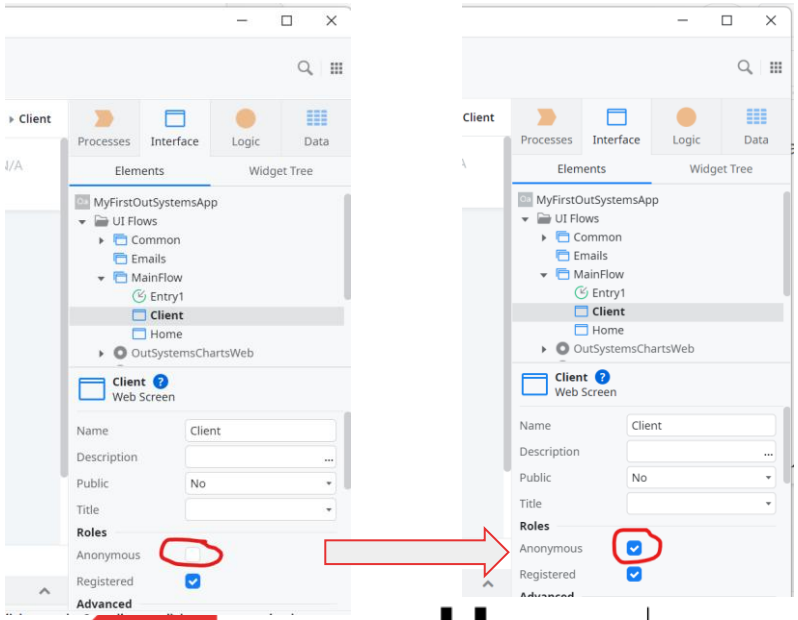
Implement the WebApp as described by Léo Andrade in video “#03 Primeiras páginas e navegação”:

The first step (at 01:00) is to name the application. To avoid name collision with your colleagues, always start the name of your apps with your student nr (eg: I58460_MyFirstApp)

Homework

After having implemented your Web App as described by Léo Andrade in video #03, do the following:

1. Add the “anonymous” role to all your Web Screens:
   a. Click on the Web Screen (widget tree on the right side)
   b. The attributes area will open. Click on “Anonymous”
   c. Repeat (a) and (b) for each Web Screen you created
   d. Publish and test
2. Submit the address of your Web Application, following the steps defined in email you received. Please confirm you received the email.



Expected total effort: 60 to 90 minutes