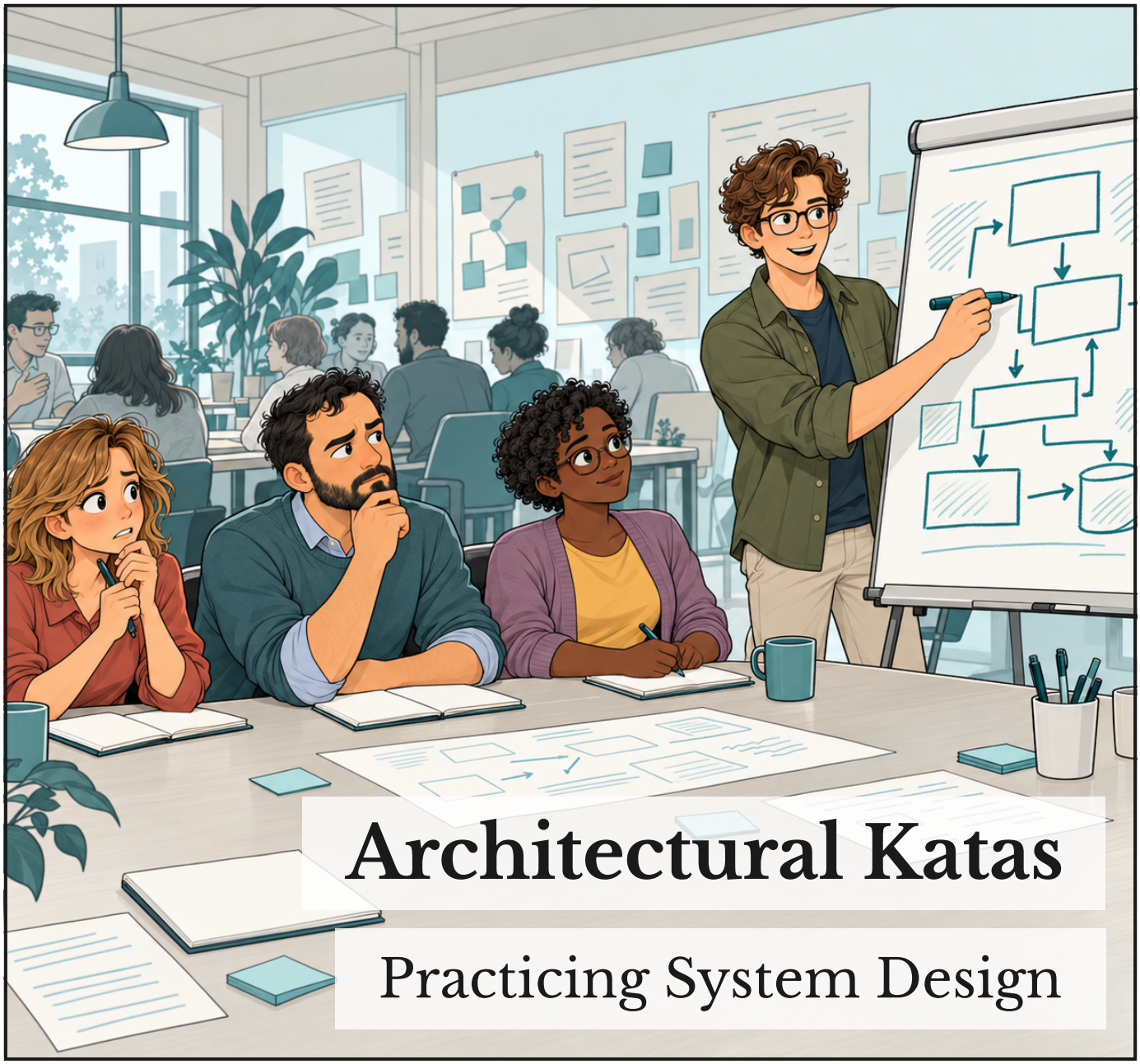


Architectural Katas

Practicing System Design

VENUE

FOSSCOMM/26

2026.fosscomm.gr

DATE

November 1, 2026

FACILITATORS

Aggelos Bellos

Staff Engineer, Epignosis

/bellangelo

/in/aggelos-bellos

Yannis Rizos

Chief Software Architect, Epignosis

/yrizos

/in/yrizos

About the Workshop

Most engineers design a system from scratch only a few times in their career. Ted Neward created Architectural Katas so they could do it far more often.

Each team gets a realistic brief, makes the design calls and defends them when the room pushes back. When another team solves the same problem differently, you start to see the assumptions you didn't know you were making.

We run katas at Epignosis to give our engineers that experience away from production. Whether you're a developer new to architecture, a tech lead who wants your decisions challenged or an architect after more reps, you'll get something out of today.

Rules and practice problems are at architecturalkatas.com, so you can run one with your own team.

The Show Must Go On!

Description

A team of organizers runs a large annual technology conference, bringing together thousands of developers, technology professionals, and industry experts.

Each year features a new lineup of speakers, talks, and activities. The organizers need a platform to manage the conference, keep attendees informed, and support interactions between speakers and participants.

Early-bird ticket sales open on December 1, 2026.

The platform must also maintain an archive of previous editions, allowing visitors to explore past schedules, speakers, and presentations.

Users

Thousands
of attendees

Hundreds
of speakers

Dozens
of organizers and event staff

Requirements

1 Conference Management

Organizers and speakers can manage talks, schedules, room assignments, and conference information.

2 Attendee Engagement

Attendees can explore the program, interact with talks, provide feedback, and receive important updates.

3 Conference Archive

Visitors can explore previous editions, including their programs, speakers, and presentations.

4 Mobile Accessibility

All capabilities must be usable on mobile devices, including during the conference.

5 Reliability

The platform must remain useful during periods of high demand and unreliable venue connectivity.

Cheat Sheet

1 Understand and Prioritize

List the actors and what they do, write down assumptions, pick the top three characteristics, and decide what must ship first and what can wait.

2 Design

Group actions into components, decide what to build and what to buy, choose a style, draw the architecture, and note each key decision on the diagram with the trade-off it accepts.

3 Share

Present your top characteristics, diagram, release plan, and one key trade-off.

Architectural Characteristics

- Availability
- Scalability
- Elasticity
- Performance
- Security
- Maintainability
- Testability
- Deployability
- Interoperability
- Cost

Architectural Styles

- Layered
- Modular monolith
- Service-based
- Microservices
- Event-driven
- Microkernel

Product Decisions

- Build or buy
- Off-the-shelf, open source, or custom
- Managed service or self-hosted
- Extend an existing system or replace it

Phasing

First release

The smallest scope that delivers value to the main actor.

Later releases

Everything else, ordered by value.

Time is a factor. The date you must ship by limits what fits in the first release.

Ground Rules

- There is no correct answer, only trade-offs.
- No technology names until the shape is clear.
- An unstated assumption is a hidden decision.

Catas to Avoid

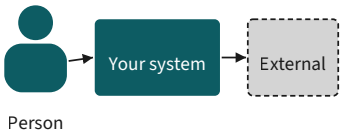
Kata + catastrophe = cata. Read about the most common kata mistakes and how to avoid them.

github.com/Bellangelo/architecture-catas

C4 Model

Simon Brown created the C4 model and maintains its official site at c4model.com. The model describes a software system as a small set of diagrams, each one zooming further in, much like the zoom levels of a map. It prescribes no shapes or colors, so the notation below is one common convention.

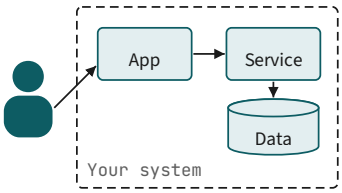
Four Levels of Zoom



1 System Context

Who uses the system, and what does it depend on?

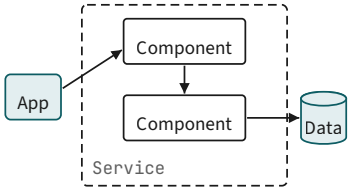
Start here, but don't spend too much time.



2 Containers

What are the main building blocks, and how do they communicate?

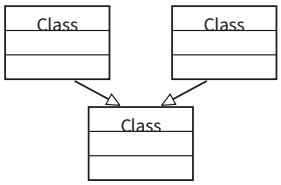
This is what you present. Start with containers and name each technology by its category, such as relational database, rather than by vendor.



3 Components

How is one container organized inside?

Also what you present. Draw it only for the container that carries your most important decision.



4 Code

How is a component implemented?

Skip it in the kata.

Notation



Person · Your system · External system · Container · Data store · Boundary · Relationship

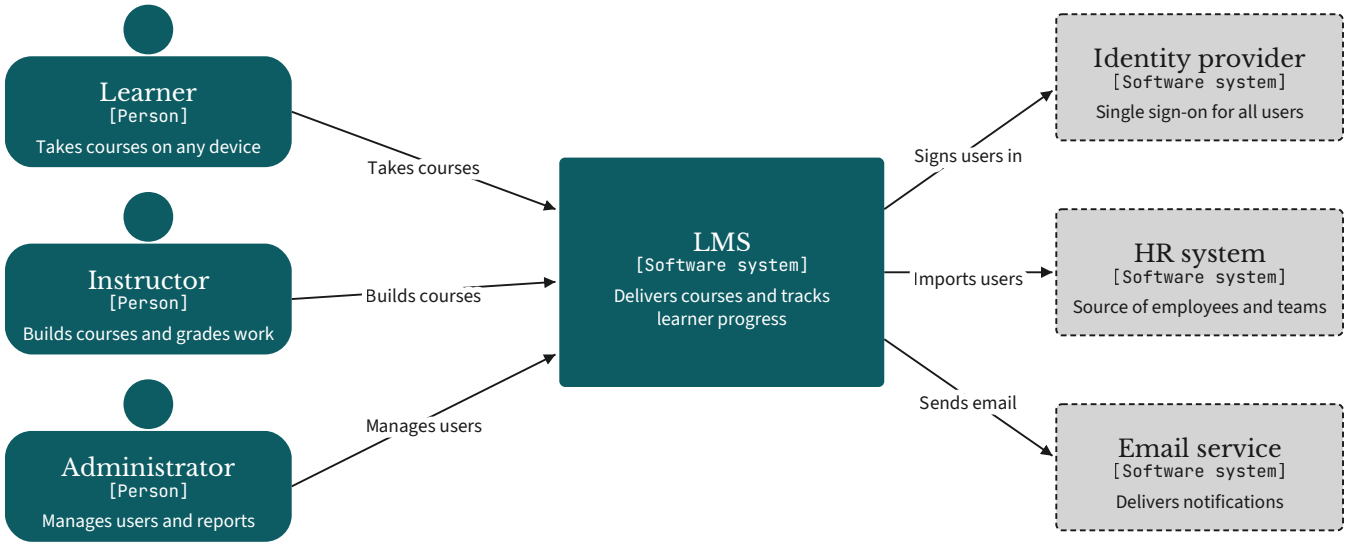
Tips

- Give each diagram a title and a key for your shapes and lines.
- Label every arrow with what it does. An unlabeled arrow hides a decision.
- Draw each arrow in one direction and make its label read the same way.
- Show build or buy on the element itself, as in `[Container: managed relational database]`.
- Mark each key decision on the diagram with its number from the decisions log.

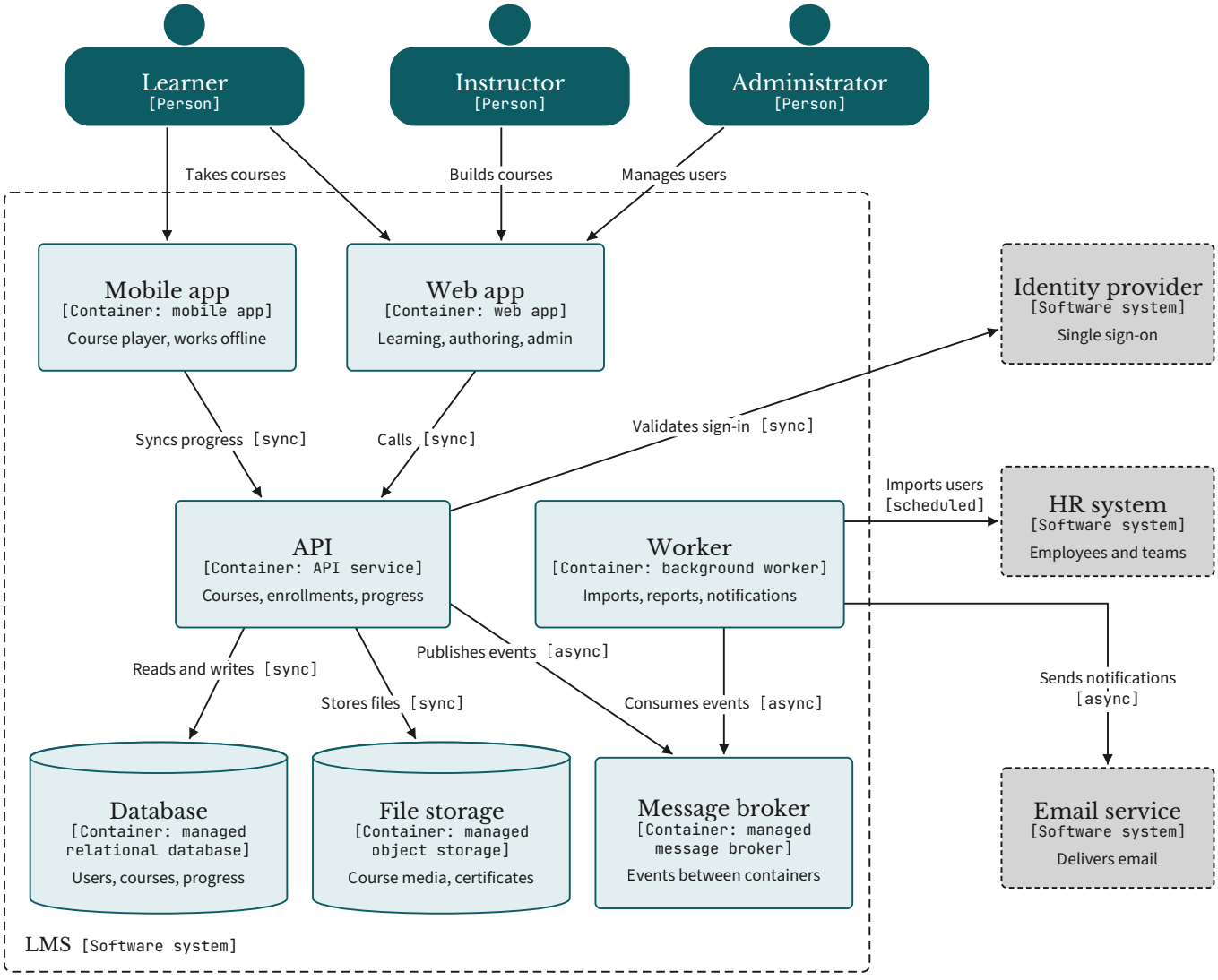
C4 Example

Here are the first three levels for a learning management system, drawn with the notation from the previous page. Each level zooms into one box from the level above it.

1 System Context



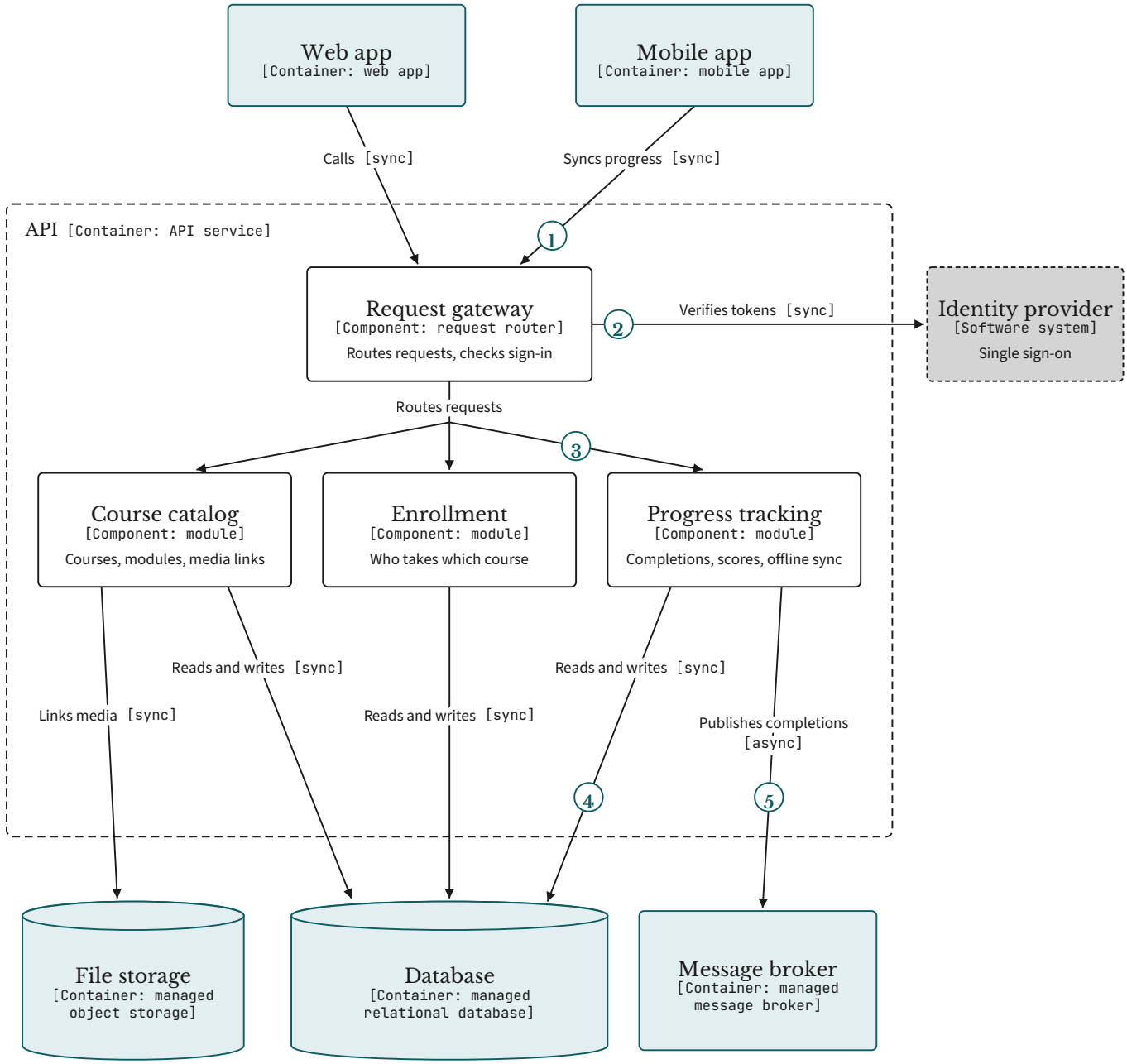
2 Containers



C4 Example

The component diagram zooms into the API container from the previous page. Its components group related responsibilities behind clear interfaces, and all of them deploy together as one service.

3 Components



Follow One Request

1. The Mobile app syncs a learner's progress to the Request gateway.
2. The gateway verifies the token with the Identity provider.
3. It routes the request to Progress tracking.
4. Progress tracking reads and writes the Database.
5. It publishes the completion to the Message broker.

Architecture Kata

TEAM _____

Top Three Characteristics

1	2	3

Users → Features

Decisions & Timeline

TEAM _____

Decisions

1	DECISION	
	ALTERNATIVE A	ALTERNATIVE B
	KEY TRADE-OFF	

2	DECISION	
	ALTERNATIVE A	ALTERNATIVE B
	KEY TRADE-OFF	

3	DECISION	
	ALTERNATIVE A	ALTERNATIVE B
	KEY TRADE-OFF	

Timeline
