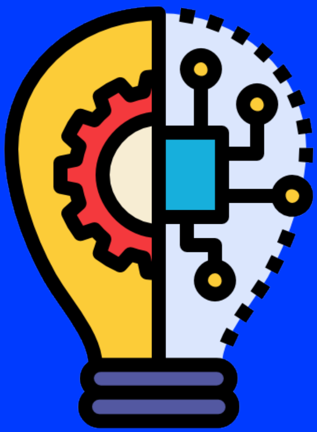
{EPITECH}

INNOVATION TRACK_

< WORKSHOP - TECHNICAL & FUNCTIONAL SPECIFICATIONS />

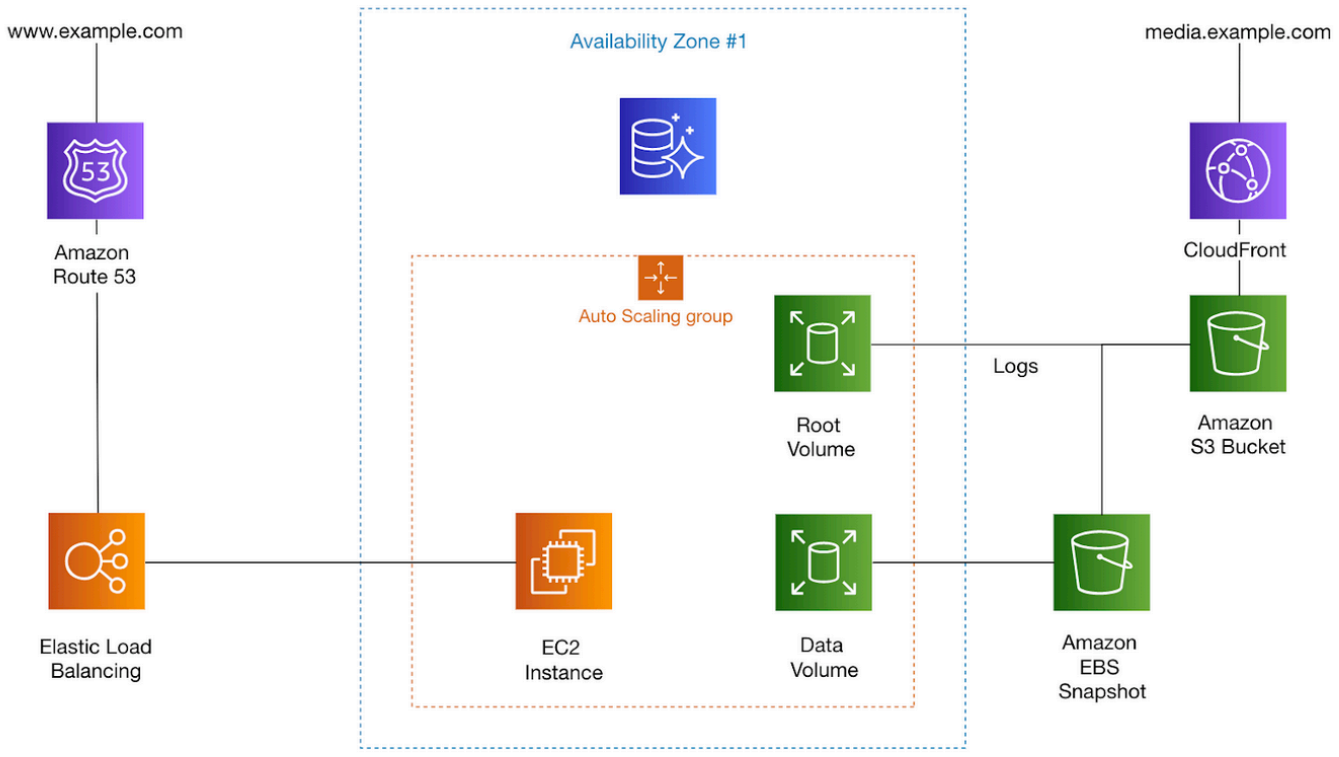


INNOVATION TRACK

Workshop: Technical & Functional Specifications

In this workshop, you will transition from the ”Idea” phase to the ”Blueprint” phase. You are no longer just imagining a product; you are defining exactly **how** it will be built.

The objective is to produce the technical and functional specifications that will guide your development phase. By the end of this workshop, you must prove that your solution is **technically viable** and **structurally coherent**.



You will work on three main pillars:

1. **System Architecture & Stack**: How parts interact and what technology drives them.
2. **Data & API Modeling**: How information flows and is stored.
3. **UI/UX Design**: How the user interacts with the system.

Part 1: System Architecture

Before writing code, you need a map. An architecture diagram is essential to understand the complexity of your system and onboarding new developers.

The C4 Model (Context, Containers, Components, Code)

We recommend using standard modeling approaches like the **C4 Model** or **UML**.

- ✓ **Context:** How your system fits into the world (Users, External Systems like Stripe or Google Maps).
- ✓ **Containers:** The high-level technical choices (Mobile App, API Server, Database, Microservices).
- ✓ **Components:** Inside the containers (Controllers, Services, Repositories).

Defense of the Tech Stack

You must choose your technologies (Language, Frameworks, Database, Cloud Provider) and **justify** them.

- ✓ *Bad justification:* ”We use React because it's trendy.”
- ✓ *Good justification:* ”We use React Native because our team has JS skills and we need to target both iOS and Android with a single codebase to meet our deadline.”

Deliverable Target: A clear architecture diagram and a one-page ”Decision Record” justifying your tech stack.

Part 2: Data & API Modeling

Software is mostly about moving data from A to B. You need to structure this information rigorously.

Database Schema

Whether you use SQL (PostgreSQL, MySQL) or NoSQL (MongoDB, Firebase), you must define your data model.

- ✓ **Entity-Relationship Diagram (ERD):** Show your users, products, orders, and how they link together.

API Definition

If you build a backend and a frontend, they need a contract to talk to each other.

- ✓ **Interface Definition:** Define your endpoints (e.g., `GET /users/:id`), the input parameters, and the output JSON format.
- ✓ *Tip:* Tools like Swagger/OpenAPI are standard for this.

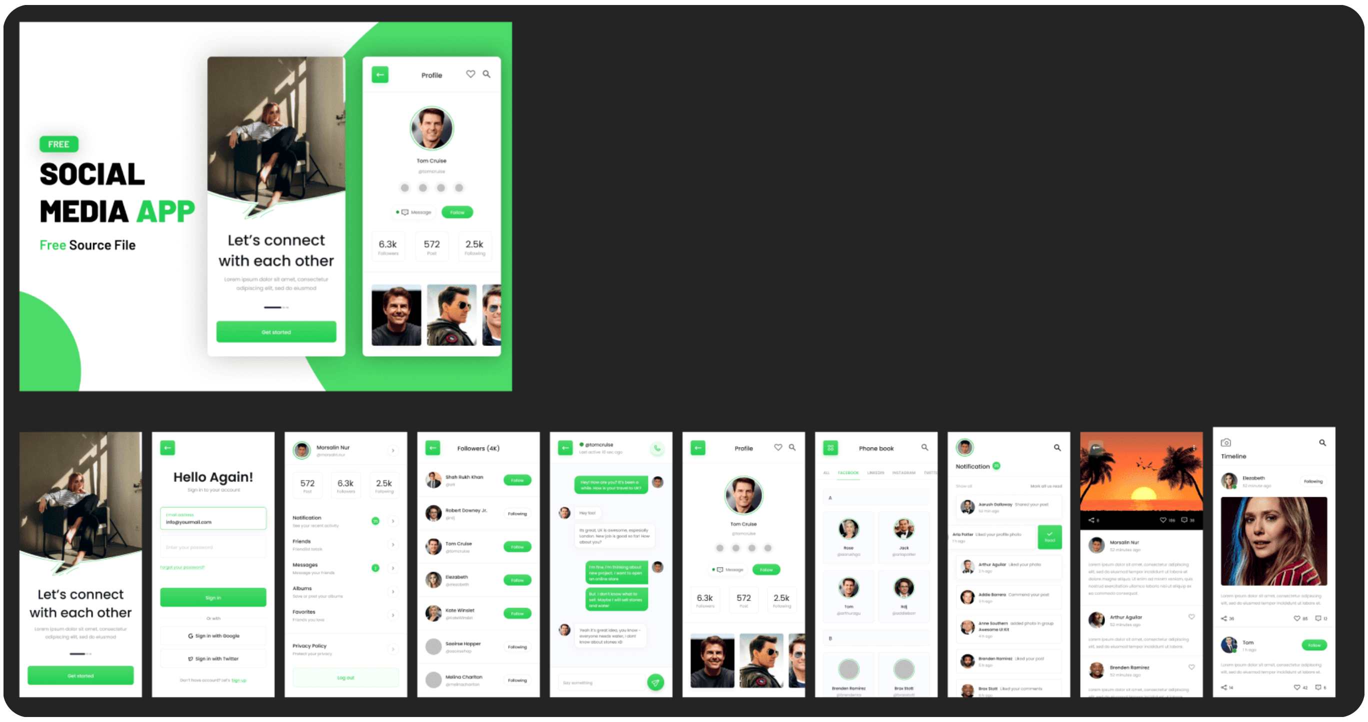
Part 3: User Interface (UI/UX) Design

For **Solution** and **Entrepreneurship** projects (and even **Tech** projects with a frontend), the interface is the product.

Wireframing & Prototyping

Using **Figma** (a leading market tool) or an equivalent, you will create an interactive mockup.

- ✓ **Wireframes:** Low-fidelity sketches to validate the user flow (navigation, layout) without getting distracted by colors.
- ✓ **High-Fidelity Mockup:** The final look and feel of your application.



Objective: Validate that the features defined in your Functional Scope (Workshop 1) are actually usable.

Specifics per Track

- ✓ **EIP Solution / Entrepreneurship:** Your mockups must be "Pixel Perfect". They are your main asset to convince stakeholders and define the frontend workload.
- ✓ **EIP Tech:** If your project is a backend framework or an engine, your "Interface" might be a CLI (Command Line Interface) or a Dashboard. You must specify the commands or the configuration files structure.

Deliverable Checklist for this Workshop:
At the end of this workshop, your slide deck should include:

1. **Architecture Diagram:** A high-level view of your system (C4 Level 2 recommended).
2. **Stack Justification:** Why did you choose these specific technologies?
3. **Data Model:** A diagram showing your main data entities and relationships.
4. **UI/UX Specification:** Key screens (Figma) or Interface definitions (CLI/API) covering the main user stories.

v 0.2

{EPITECH}