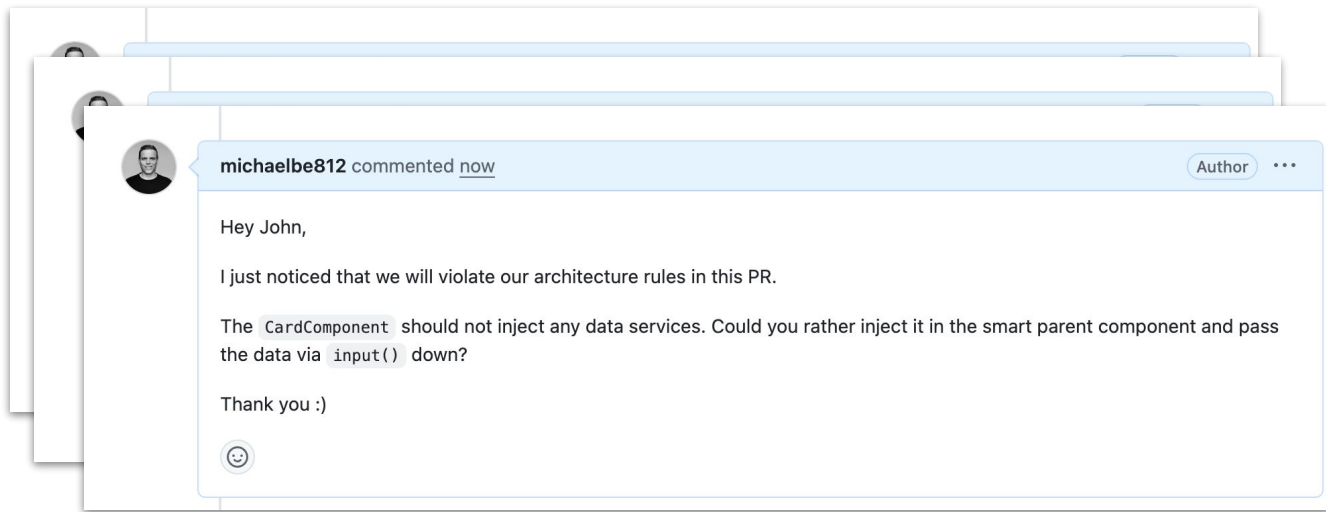


Architecture enforcement done right



Familiar PR comment?





Agenda

Agenda

- The problem
- Intro about architecture
- Architecture Validation
- Case Study / Demo
- Conclusion

Michael Berger

Freelancer | Trainer | Consultant

michael.berger@berger-engineering.io



@michaelbe812



www.berger-engineering.io



michaelbe812



<https://www.linkedin.com/in/michael-berger-angular-expert/>



Berger Software Engineering



Architecture

Architecture | What is it?

With architecture we define a **ruleset** how we want to **structure things** and how these things should **work together**.

Code organization /

Modularization

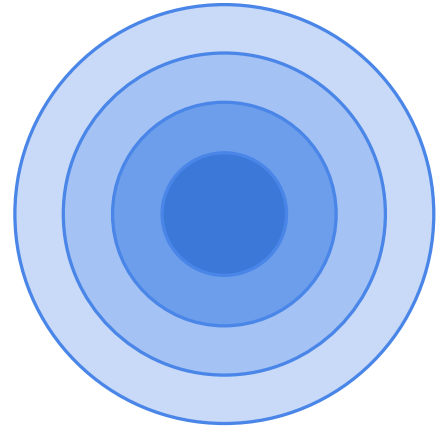
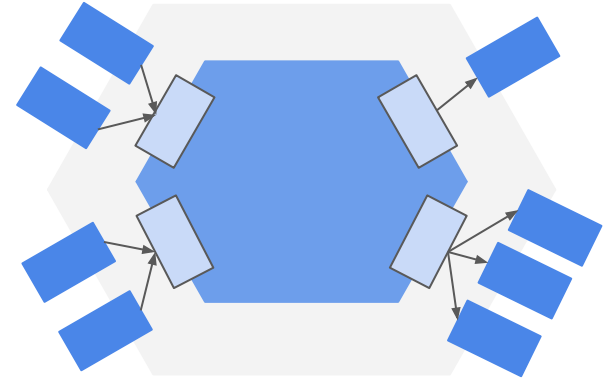
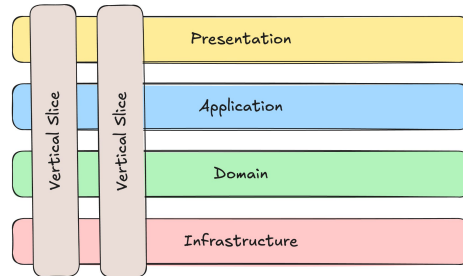
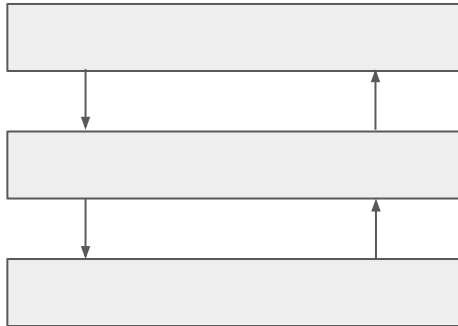
Dependencies

API's

Encapsulation

Architecture | Patterns

- Layered Architecture
- Vertical Slice Architecture / Verticals
- Clean Architecture
- Hexagonal Architecture

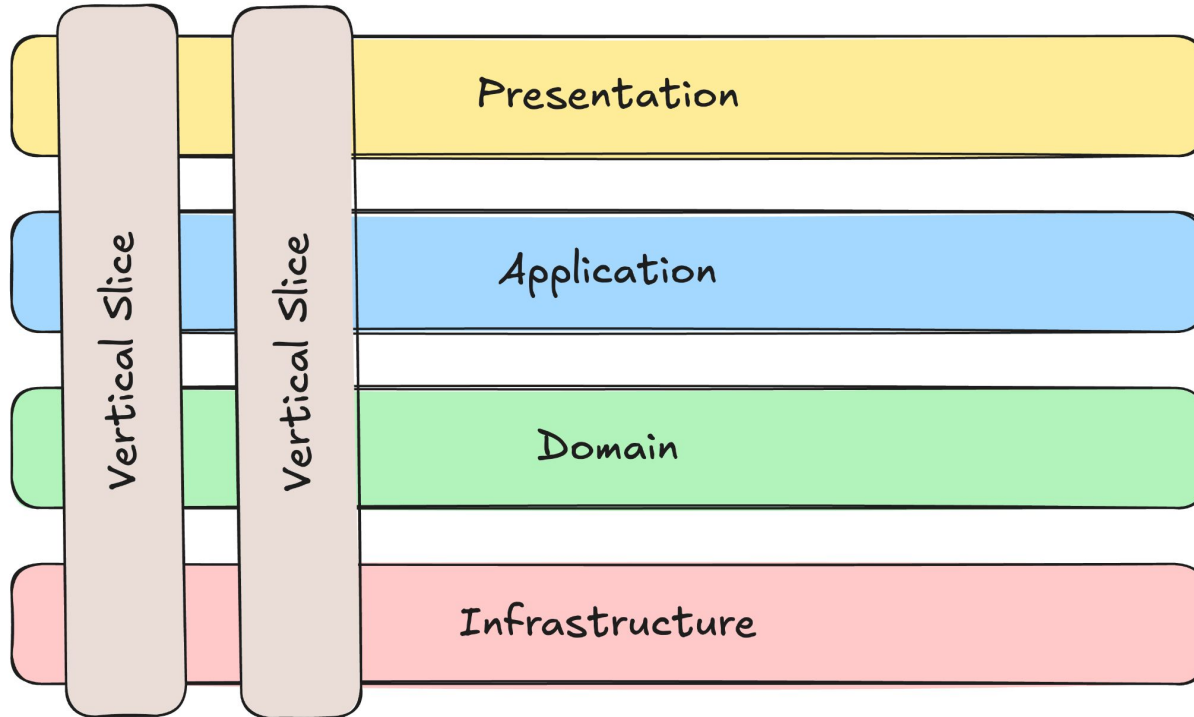


Verticals | What

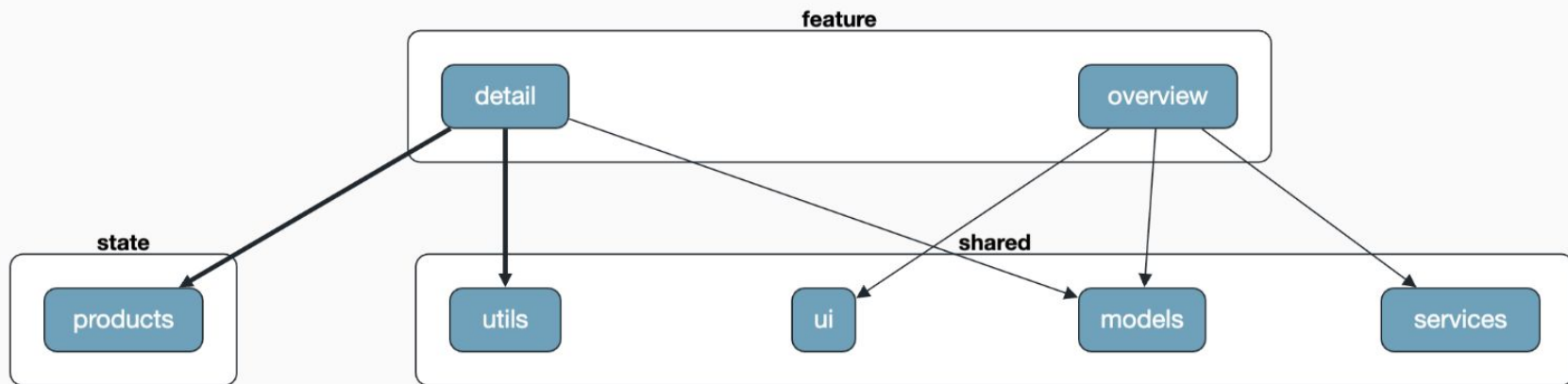
Verticals, more precisely ***Vertical Slice Architecture (VSA)***, is a pattern that aims to organize the application by vertical slices.

Each Slice encapsulates a specific functionality or feature. This means that each vertical layer contains all necessary components to provide a specific functionality.

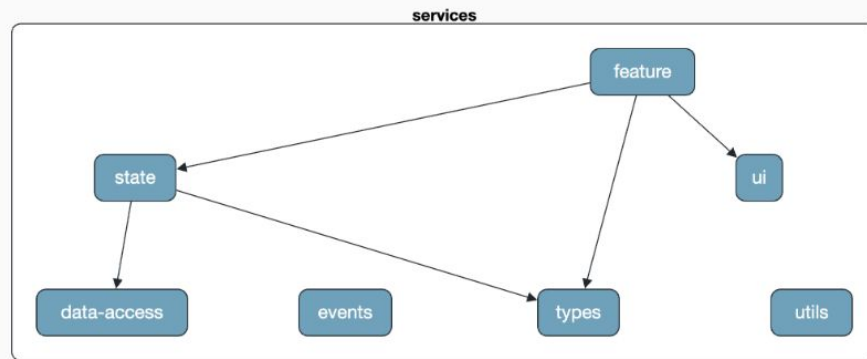
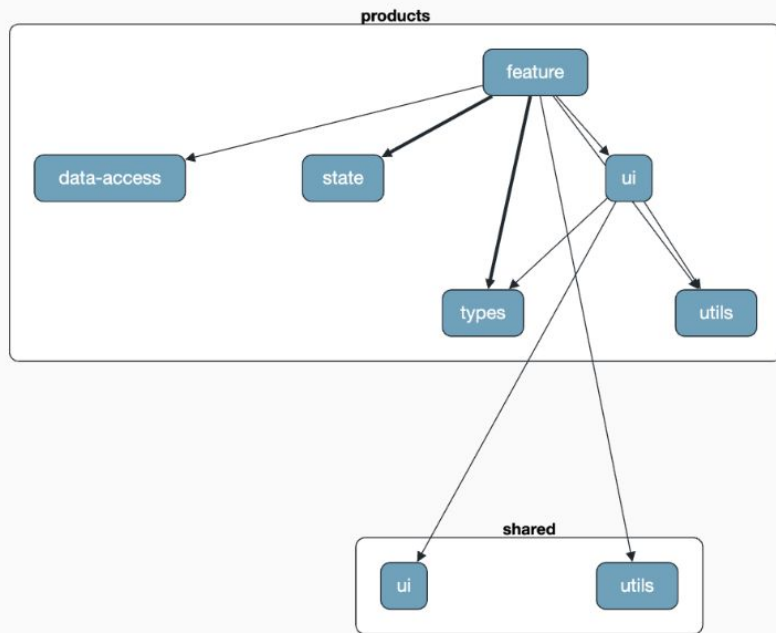
Verticals | Characteristics



Verticals | Impact visualized



Verticals | Impact visualized



Verticals | Pros and Cons

Pros

- Natural module encapsulation
- Improved Code co-location
- Better DX
- Rigid Structure
- Good natural decoupling
- Scales with complexity

Cons

- Not always DRY

Architecture Validation

Prerequisites for Architecture Validation

- Rules must be deterministic
- Therefore architecture must be predictable by patterns → e.g. always the same folder structure

Overview about the tooling landscape

- Tools for Architecture Validation which work together with ESLint:
 - Sheriff
 - nx module boundaries
 - eslint-plugin-boundaries

Tools | nx/module-boundaries



Core Principle:

- Tagging nx projects and implementing access rules based on tags
- multi-dimensional, e.g. 2 dimensions:
 - type: feature, ui, data-access, utility
 - scope: e.g. admin, shopping-cart → e.g. a sub-domain/bounded context

Tools | eslint-plugin-boundaries

Core principles

- Tagging of modules based on path structure
- Defining dependency rules and access rules between modules

eslint-plugin-boundaries

5.0.1 • Public • Published 10 months ago

Readme

Code Beta

4 Dependencies

25 Dependents

45 Versions

build no status coverage 99% quality gate passed

renovate enabled last commit november 2024 release date november 2024

downloads 8934/month license MIT

eslint-plugin-boundaries

In words of Robert C. Martin, "Software architecture is the art of drawing lines that I call boundaries. Those boundaries separate software elements from one another, and restrict those on one side from knowing about those on the other." (*"acknowledgements"*)

This plugin ensures that your architecture boundaries are respected by the elements in your project checking the folders and files structure and the dependencies between them. It is not a replacement for `eslint-plugin-import`, on the contrary, the combination of both plugins is

Install

```
> npm i eslint-plugin-boundaries
```

Repository

[github.com/javierbrea/eslint-plugin-bo...](https://github.com/javierbrea/eslint-plugin-boundaries)

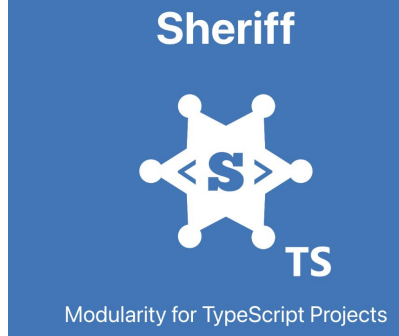
Homepage

[github.com/javierbrea/eslint-plugin-b...](https://github.com/javierbrea/eslint-plugin-boundaries)

Weekly Downloads

236,271

Tools | sheriff



Core Principles

- Enforces module boundaries for encapsulation. Works with and without barrel files ([index.ts](https://www.npmjs.com/package/index.ts))
- Very flexible dependency rule configuration to model access-rules based on tagging
- Zero external dependencies
- Can be integrated with ESLint and/or used via a standalone CLI



Credits to
Rainer Hahnekamp

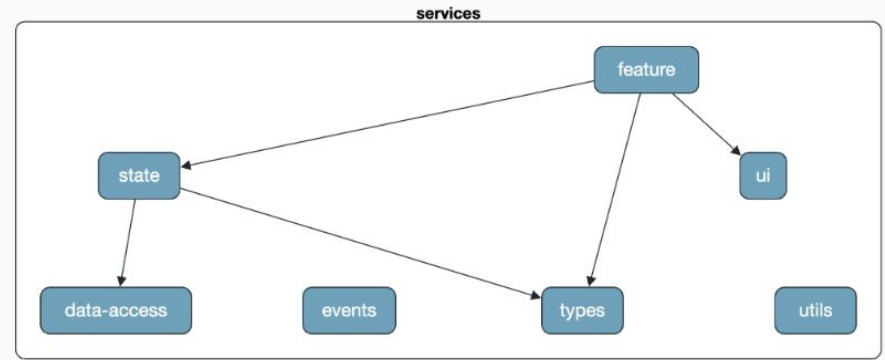
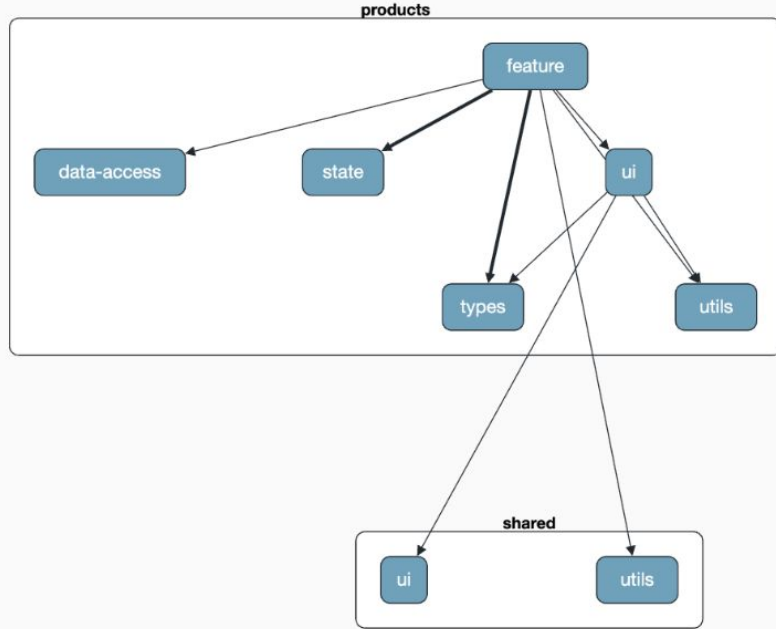
Tool Comparison

	Nx	Sheriff	eslint-plugin boundaries
Flexibility	-	↑	↑
Setup	↑	-	-
Boilerplate	↓	↑	-

↑ Good - OK ↓ Not good

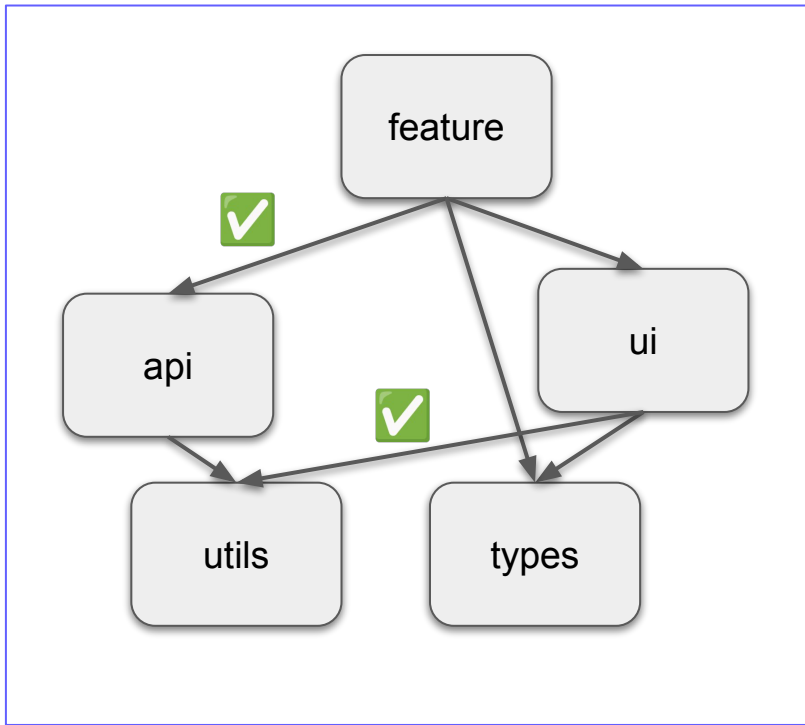
Case Study / Demo

Recap



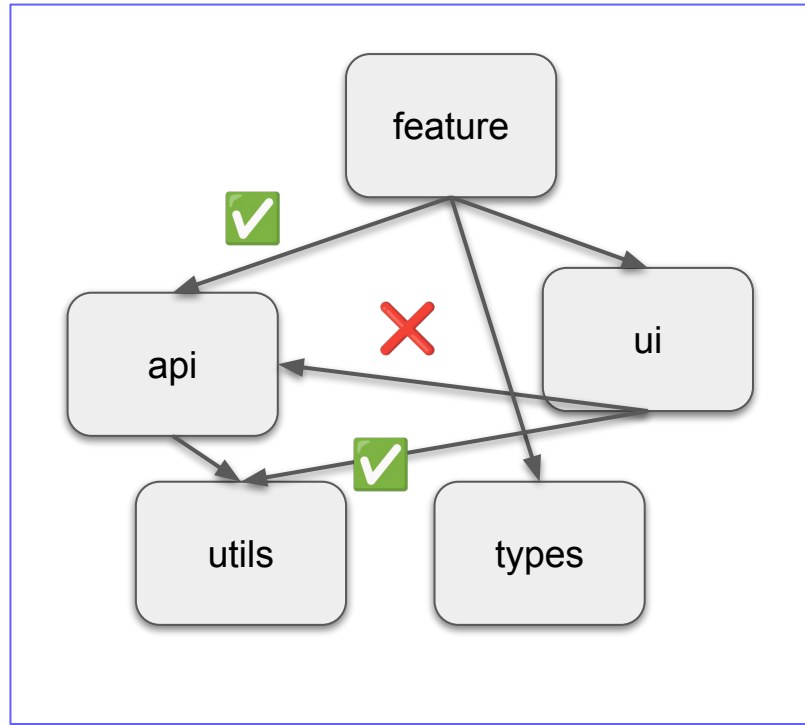
Isolated features

feature



✗

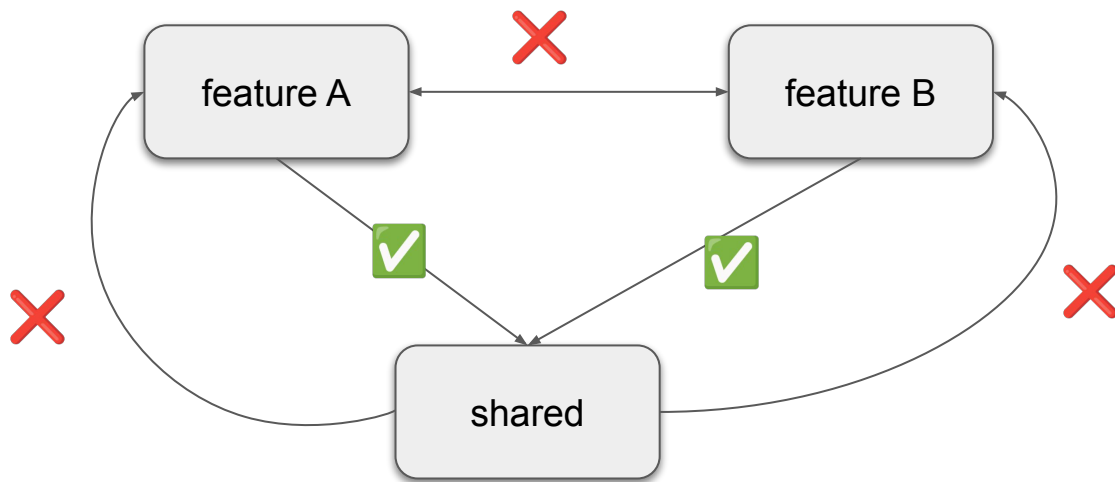
feature



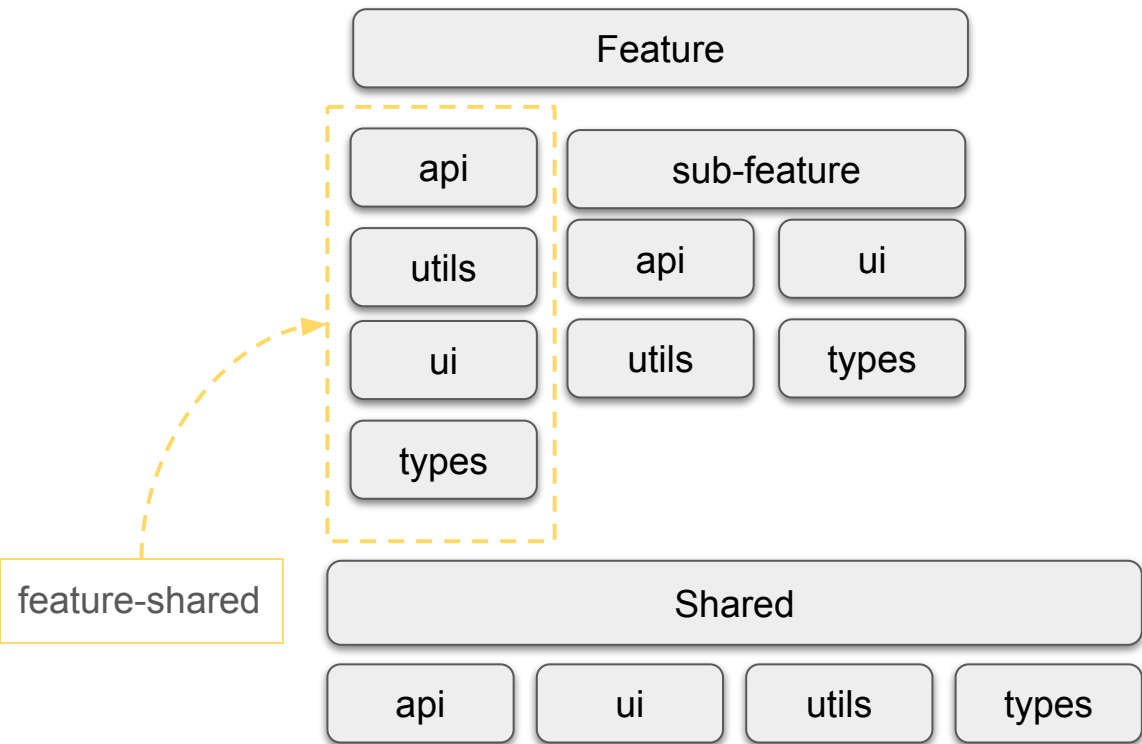
Access Rules within a feature

	feature	ui	api	utils	types
from feature	to ✓	✓	✓	✓	✓
ui	✗	✓	✗	✓	✓
api	✗	✗	✓	✓	✓
utils	✗	✗	✗	✓	✓
types	✗	✗	✗	✗	✓

Access rules between features



Complex Modularization



Demo-Time

Conclusion

Takeaways



Enforcing architecture patterns is relatively easy with the right tool

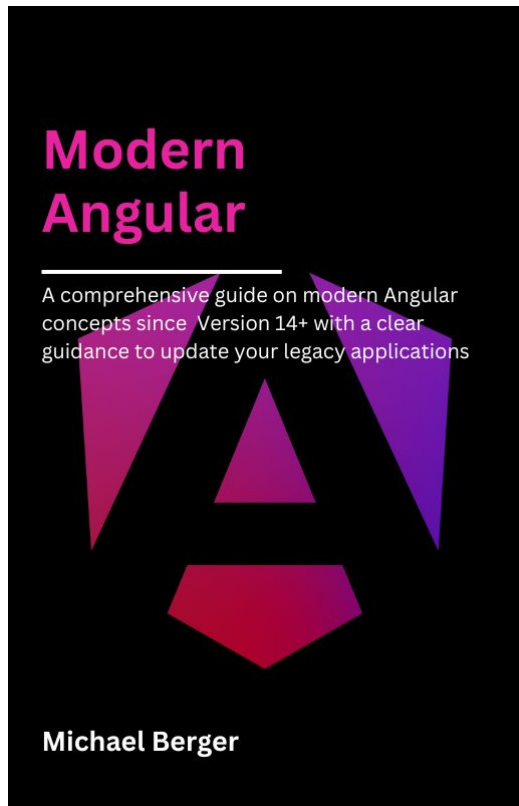


Verticals are a great fit for evolving architectures



Architecture validation is a key measure to preserve a healthy project

THANK YOU



THE guide for modern Angular

This book will cover:

- All modern Angular concepts
- A clear decision guidance for each concept when and how to update “legacy” apps
- + free [Agents.md](https://agents.md) for a modern Angular AI Agent

**Exclusive Pre-Order
discount until 15.11.2025**

<https://book.modern-angular.com/>



Guides for modern Angular Developers

Architectures in Angular that

really scale

Learn hands-on how to implement a real scalable architecture that evolves with your app-size and ensure that your team's performance does not suffer from bad architecture.

Michael Berger

RxJS and Signals united together

A comprehensive guide on when to use Angular signals and RxJs observables in modern Angular apps.

Michael Berger

NgRx SignalStore modern, lightweight state management

The NgRx SignalStore promises to be a lightweight and composable state management solution which grows with the individual needs of an application unless the heavier global store.

This guide will walk you through how to use it and scale it efficiently with a growing app.

Signal Forms

Angular forms completely reimagined.

Learn the future way of building complex forms with ease in Angular.

Meet Ressources - the modern async reactive primitive

Ressources are the new signal-based reactive primitive to handel async-operations.

In this comprehensive guide you will best practices how to use them efficiently.

<https://guides.modern-angular.com/>

scan to visit



Michael Berger

Freelancer | Trainer | Consultant

michael.berger@berger-engineering.io



call



<https://www.berger-engineering.io/appointments>



@michaelbe812



michaelbe812



www.berger-engineering.io

Slides & Demo Repo



Repo

<https://github.com/michaelbe812/talk-automated-architecture-validation>



Slides

<https://www.berger-engineering.io/talks#ng-stuttgart-2025-10-22>