MODUS CREATE

MICRO FRONTENDS

Scaling Digital Presence and Operations

WORKSHOP AGENDA

- What are Micro Frontends and why businesses should care
- Creating a simple Module Federation app cluster
- Types of dependencies and dealing with conflicts
- How to communicate between federated modules
- Module Federation in hybrid apps
- How to organize teams, repositories, & code sharing
- Q&As

ENVIRONMENT SETUP

VS Code + Live Share + Codespaces

UPGRADE YOUR BUSINESS WITHOUT MAINTENANCE BREAKS

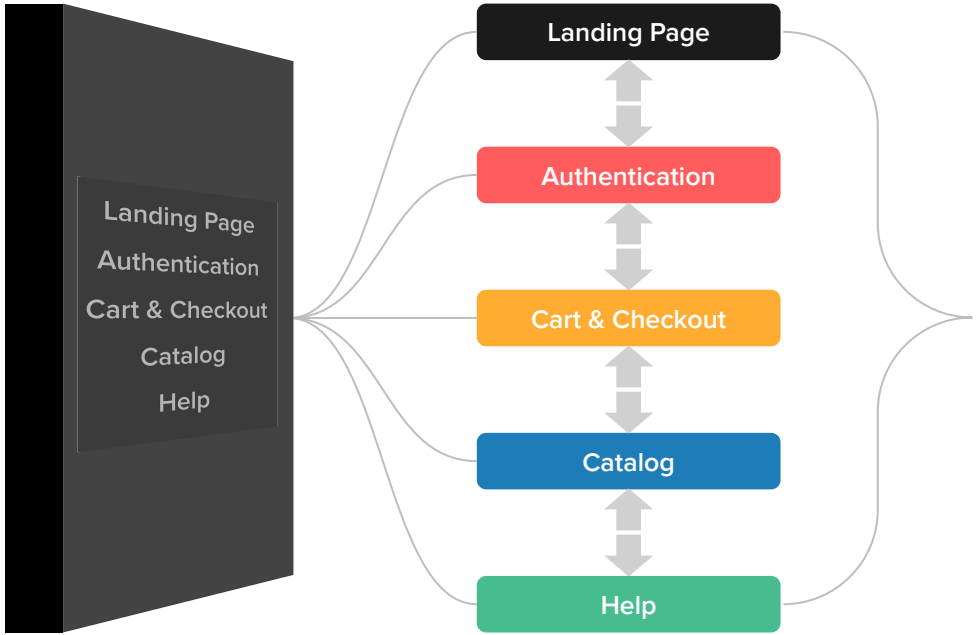
What if a race car’s engine could be fixed without shutting it down, at 200 mph?

THE BIG PICTURE IS MADE OF SMALL ONES

Democratize web development
to empowered teams

WHAT ARE MICRO FRONTENDS?

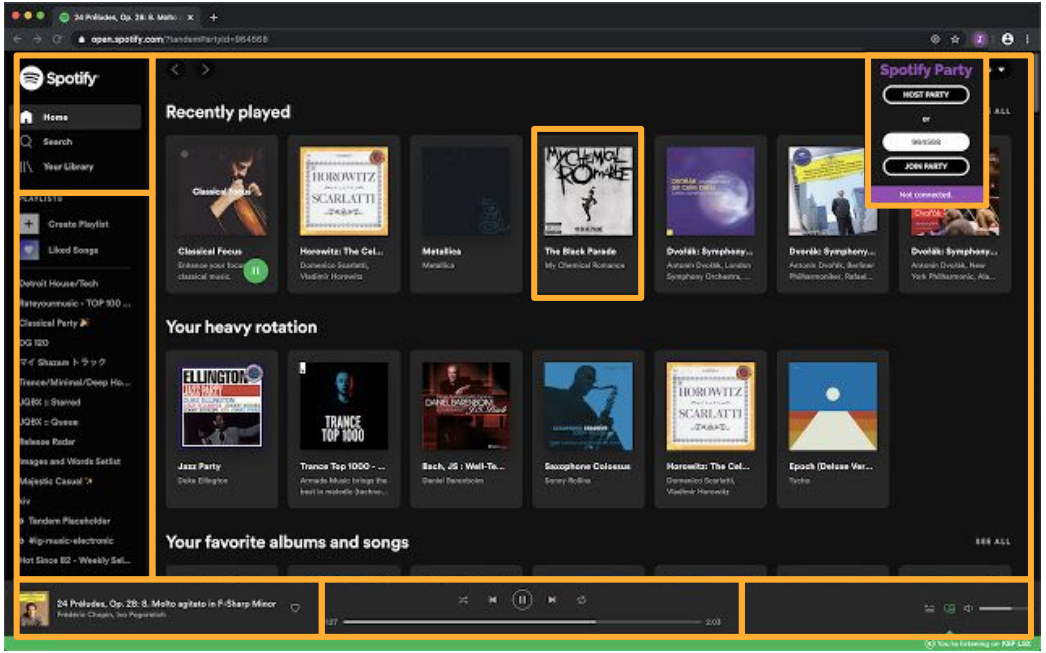
The technical pattern for decoupling a product into **business subdomains** (features) owned by **independent teams**. Features are integrated into a whole, but governed independently.



Micro frontends is an engineering architecture that promotes decentralization by decoupling a monolith into independent feature services. Features are owned by dedicated teams and automatically integrated where needed.

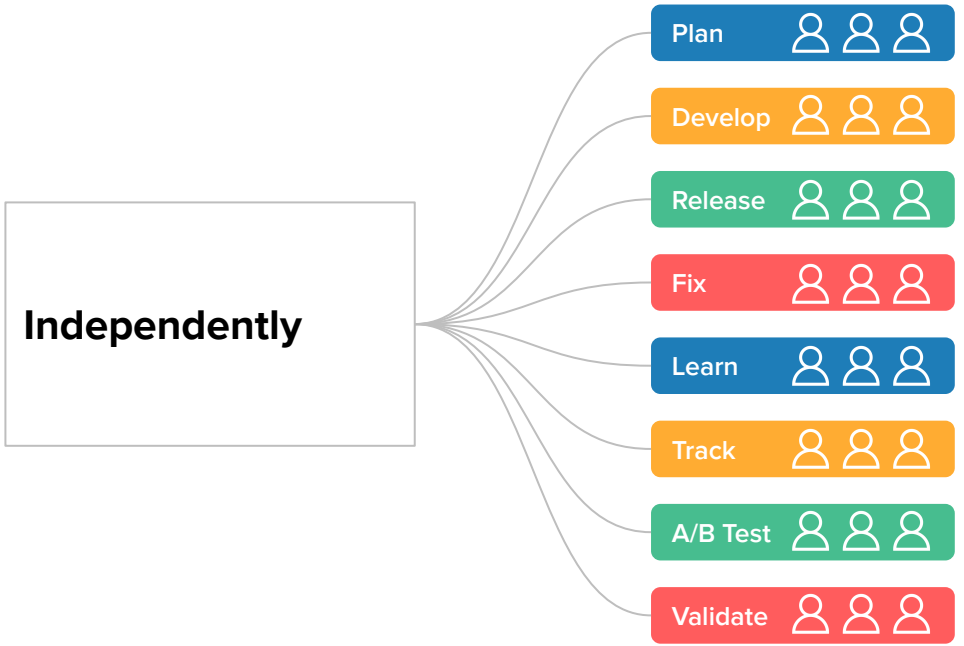
AGILE SCALING AT SPOTIFY

Spotify uses the Micro Frontend architecture to optimize how they **learn**, **improve**, and **increment**.



TRULY EMPOWER SELF-ORGANIZED TEAMS

Micro frontends democratize development by enforcing ownership of a business subdomain. Teams decide how to plan, develop, release, validate, and fix. It's no longer required to plan for maintenance of the entire system, when features can update independently **without breaking integration**.



ARCHITECTURAL PATTERNS FOR MICRO FRONTENDS

+ Frames
+ Server-side includes (SSI)
+ Edge-side includes
+ Git (submodules, monorepo)
+ NPM modules
+ Module Federation
+ Others (mostly custom build-step processes)

WHAT MAKES MODULE FEDERATION SPECIAL?

VERSUS

+ Frames
+ Server-side includes
+ Edge-side includes
+ Git
+ NPM modules
+ Others

+ Module injection at runtime
+ Automatic dependency resolver
+ Avoid duplication
+ Chooses the best version available
+ Vanilla JavaScript, framework-agnostic
+ Standards-based
+ Universal (browser, server, mobile, etc.)

EXERCISE 1

Hello, Federated World!

EXERCISE 2

Adding Dependencies

EXERCISE 3

Maintaining Dependencies

EXERCISE 4

Load Federated Modules Dynamically

COMMUNICATION PATTERNS

Sharing Data and Triggering Common Actions

EVENT BUS [OBSERVABLES]

Communication Pattern

Centralized listeners triggered with data payload

\+ Agnostic of the other apps presence
\+ Can be captured by multiple instances
\+ No intermediaries (dependencies) required

\- Lazy-loaded (async) code might not be loaded in time
\- Deferred code cannot access trigger history (repeatable state)

SERVICES

Communication Pattern

Dedicated modules that provide APIs for business logic (e.g. localization, tracking)

\+ Can manage state and history

\+ Can use federated [lazy] loading capabilities

– Creates dependencies (coupling)

– May create data corruption when services

SERVER-SIDE

Communication Pattern

Using a cloud or
a server-side logic

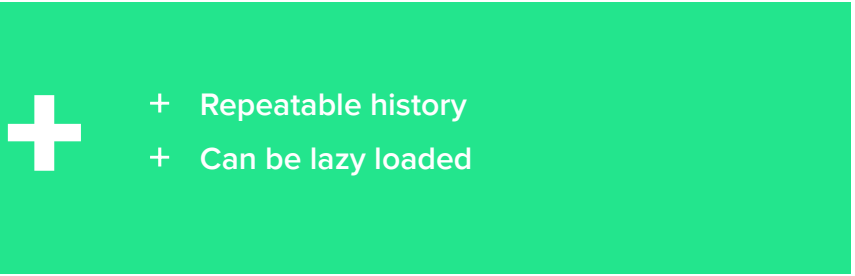
+ Reliable, up-to-date business logic
+ Features can load with pre-defined state for better performance

– Laggy (to very laggy)
– No awareness of the existing capabilities in the front-end (e.g. loaded module versions)

SHARED STATE

Communication Pattern

Redux or other state managers



ROUTING

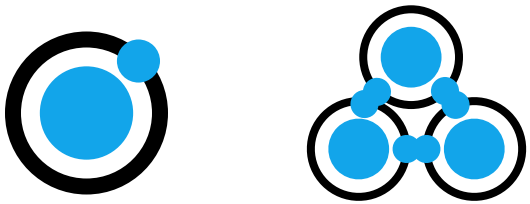
Communication Pattern

Communicate via navigation

+ Universal
+ Repeatable history
+ Transparent
+ Shareable

– Requires managing a singleton history instance
– Awkward to use with some requirements like tracking

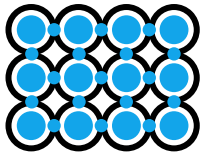
MICRO FRONTENDS VS DESIGN SYSTEM



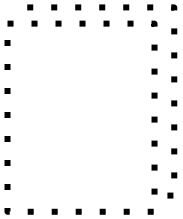
ATOMS

MOLECULES

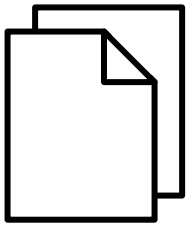
Design System UI
Component Library



ORGANISMS



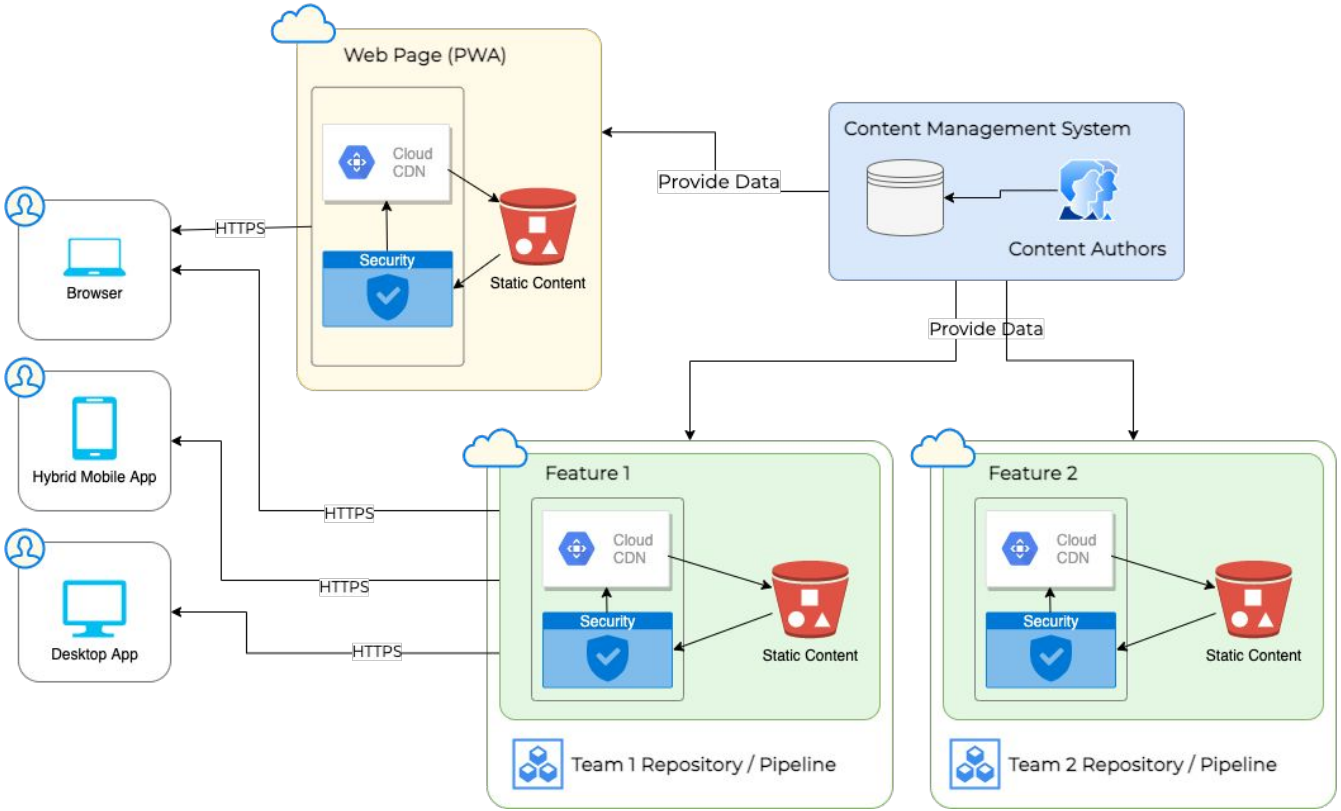
TEMPLATES



PAGES

Micro Frontends

FEATURE AS A SERVICE ARCHITECTURE



TIPS

Things you should know early to avoid headaches later

- Think globally, act locally
- Always ask: am I creating coupling
- Deploy Features in versioned subfolders
- Use separate cloud stacks for better analytics
- Wrap injected Features in Error Boundaries
  - Self-heal
- Communicate via routes only
- Always create backwards-compatible increments
- Maintain a custom shared memory object
- CORS Allow * for statics

FAQ

Questions Other People Often Ask

Q: CAN I MIX & MATCH JS FRAMEWORKS?

A: Would you do that in an SPA? From the holistic point of view, this creates more coupling, facades, and awkwardness than the value it might bring.

Q: WE HAVE A STRICT ORGANIZATION. CAN WE DEFINE GLOBAL RULES FOR ALL TEAMS?

A: Yes. But - the goal is to give teams a little bit of extra freedom to work out the best way to be efficient. There will always be some contractual agreements (e.g. the framework, routing mechanisms, design system, etc) and there will be opportunities to customize stacks (like eslint, type safety, test suites, CI setup, etc.).

Q: SHOULD I USE A MONOREPO FOR STORING ALL FEDERATED MODULES?

A: Yes, but that might not be a good idea. Micro Frontends are great at scale. As your project and teams scale, it will be more difficult to maintain such a large monorepo (think merge conflicts). Monorepos also restrict team decisions.

HOW TO CONVINCE YOUR BOSS OR CLIENT

... Because They don't Speak the Same Language

MICRO FRONTENDS PROMOTE AGILE ORGANIZATIONS

Learn quickly, **iterate** independently.

GRANULARITY OF GROWTH

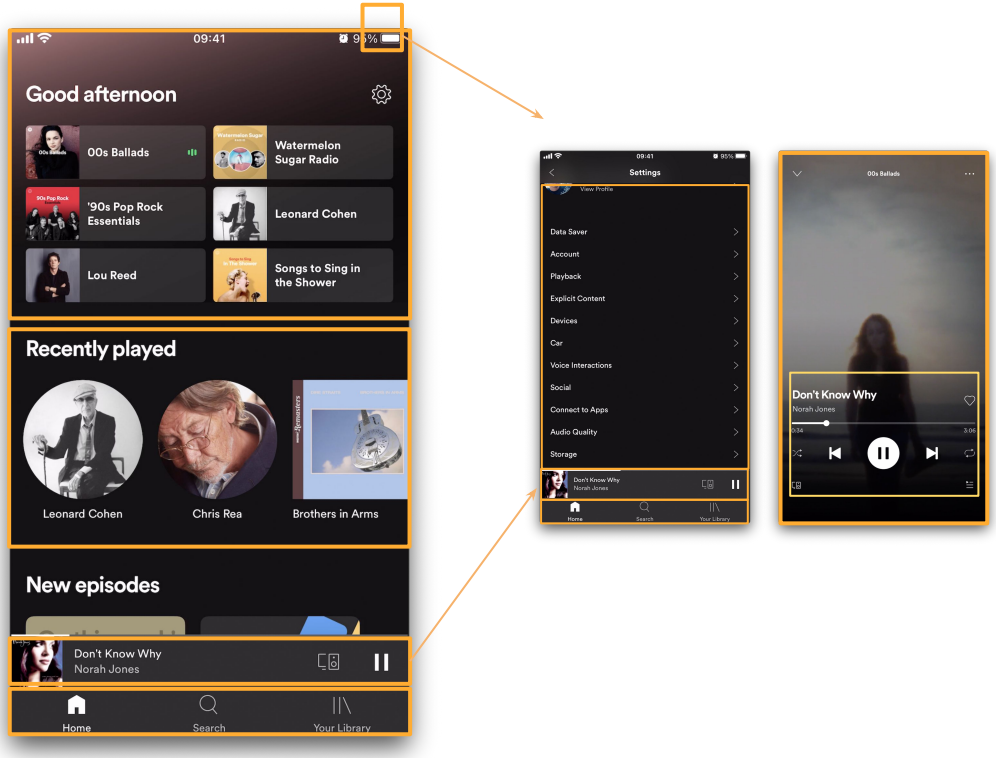
Micro Frontends can be as homogeneous or as granular as works best for your teams, business, and users.

SIMPLIFY MIGRATIONS AND FUTURE-PROOF YOUR PRODUCT

Feature-by-feature migration (aka the Strangler Pattern) is cheaper, faster, and more agile.

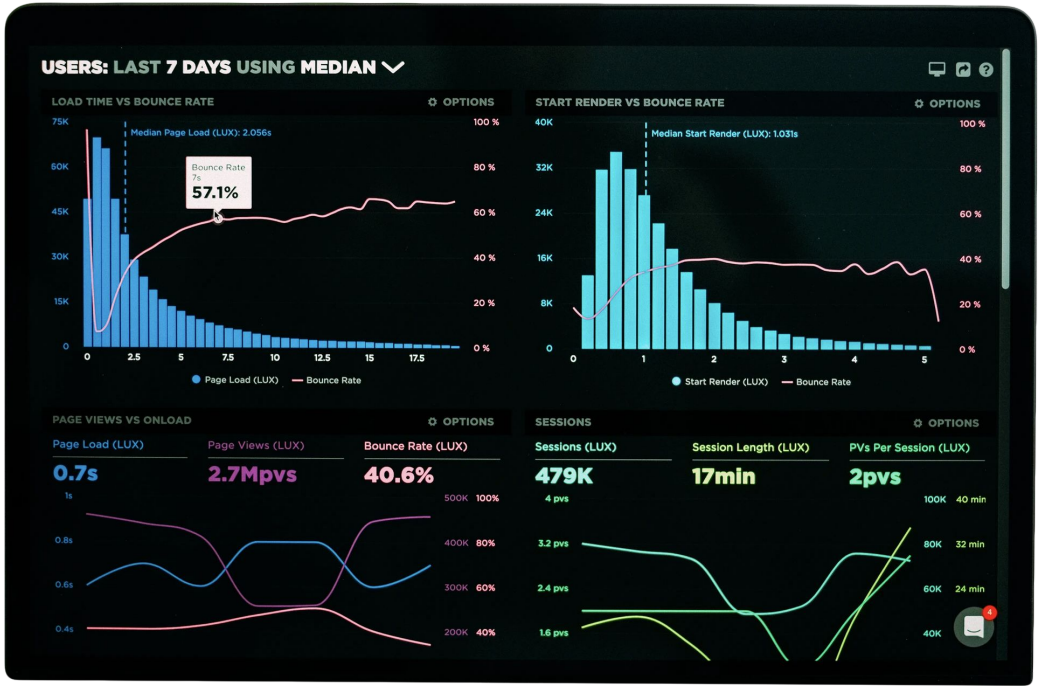
OTA MOBILE UPDATES

Spotify uses the Micro Frontend architecture to optimize how they **learn**, **improve**, and **increment**.



ARE YOUR FEATURES PROFITABLE?

Make educated decisions based on detailed cost and benefit data. Track more information with Micro Frontends.



FEATURE AS A SERVICE

Deploy features to a CDN and inject them where needed, as needed

- A small JavaScript runtime is generated at build time, allowing for integration and dependency management
- Dependencies are resolved automatically using semver to avoid duplication and to ensure consistency
- Features can be loaded dynamically (injectable, e.g. based on configuration)

WEBPACK 5 MODULE FEDERATION

Module Federation is the long anticipated industry standard and a game changer for Micro Frontend architectures.

Development Efficiency

Engineers can focus on the business logic and goals

Automatic Code Reuse

Dependencies are shared automatically, minding versioning

Continuous Improvement

The best patterns bubble up and spread across teams

Lean Validation

Better A/B testing opportunities with regression limited to fixed scope

YOU ARE NOT ALONE

World's leading organizations utilize Module Federation to scale their digital presence

Q&A

Communication

MODUS IS HIRING!

This is the slide where we say that you can join Modus and **live an amazing life**. Your star will be added to Hollywood boulevard and everybody in your village will chant your name in celebration of your stardom. Your offspring will become immortal through the heritage of your work.

THANK YOU!

Let’s make something great together.