Comparing microservices and monolithic architecture

Learning objectives

- **Understand** the structure of a **monolithic architecture** and how it **differs from microservices** in terms of deployment, development, and maintenance.
- Understand **case studies** where the transition from a monolithic to a microservices architecture solved specific business or technical challenges.
- Understand **modular monoliths**
- Know when to **choose** one **architecture** over the other

Monolithic architecture

- The entire application is built and deployed as one unit.
- Components are often tightly coupled, making changes or updates challenging.
- All parts of the application share the same resources and codebase.

Pros of monolithic design

- Single application to deploy
- Single database
- Simple to develop, build, test and deploy
- Simple communication between components
- Easy to scale

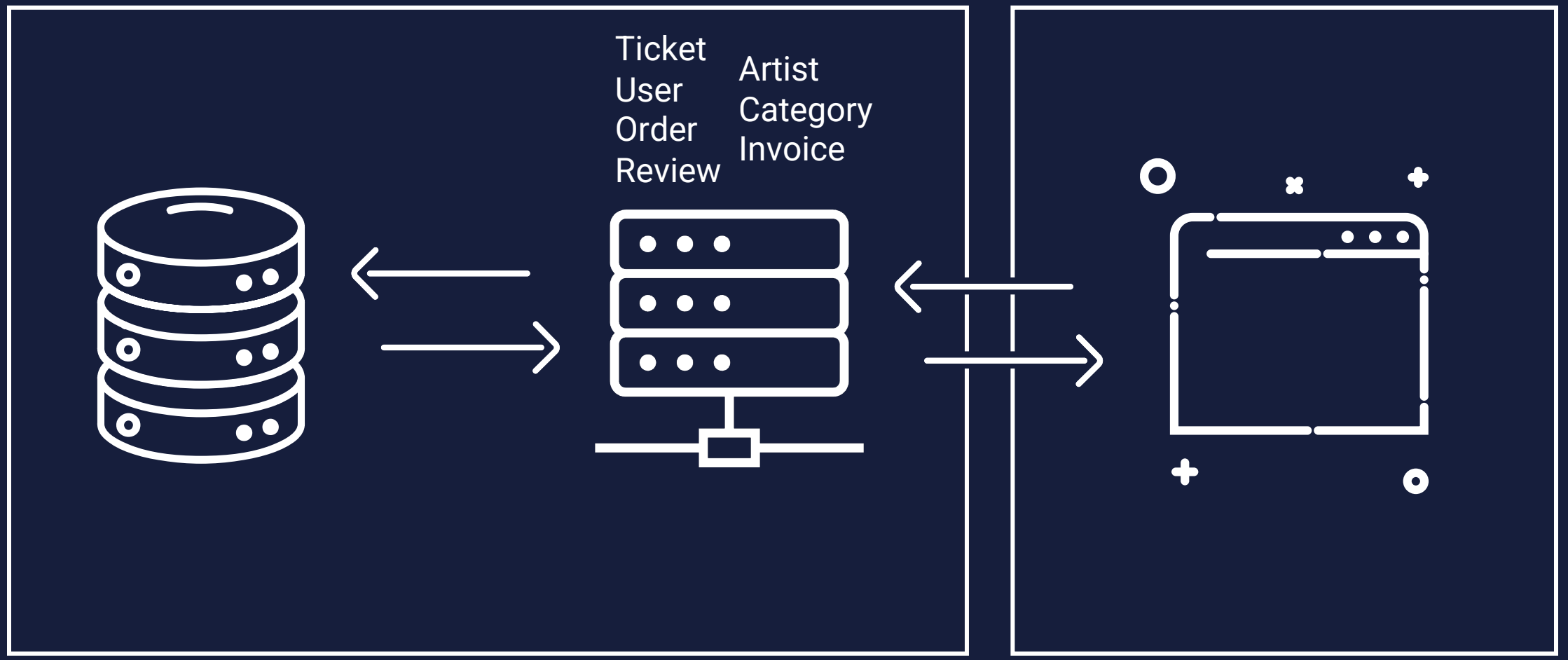
Cons of monolithic design

- Code harder to understand
- No emerging technologies
- Scale unnecessary parts
- Huge database
- Hard to add team members / learn code base

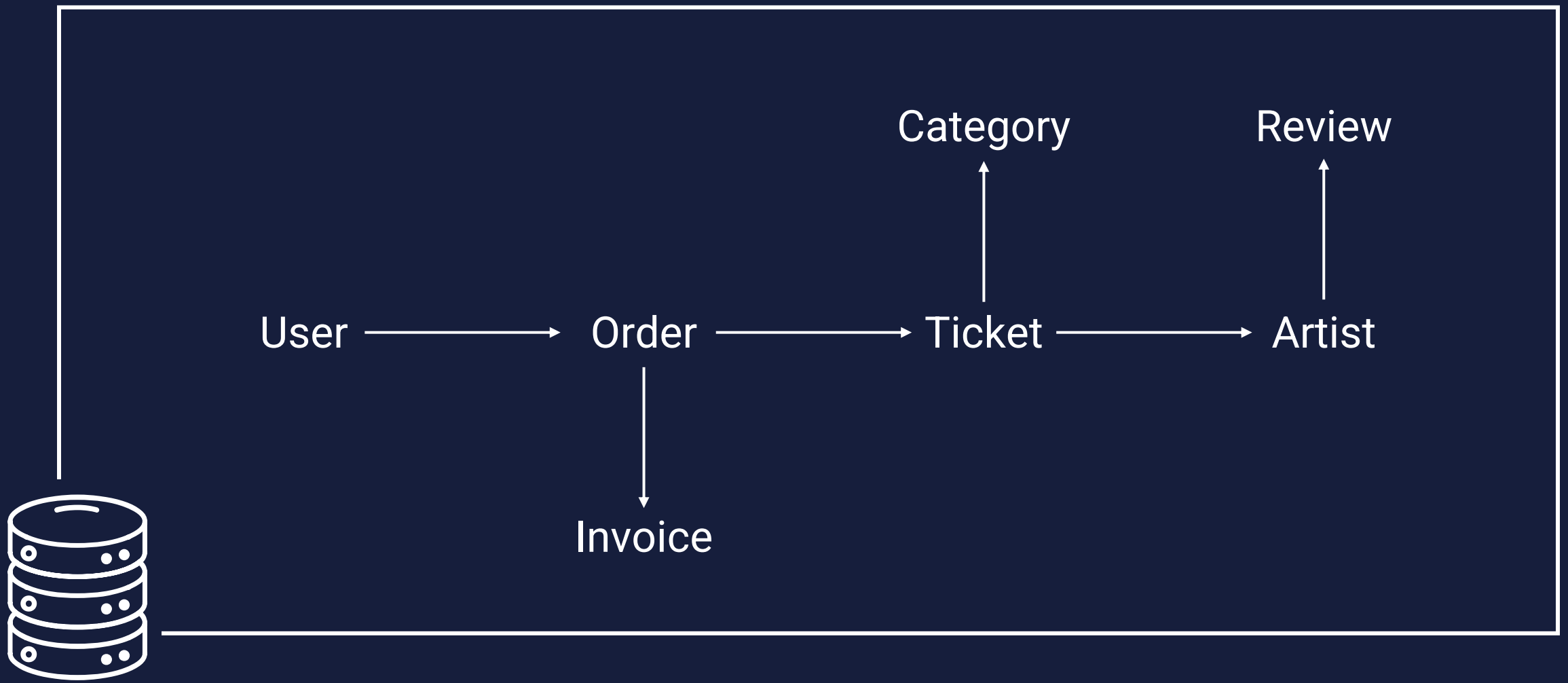
Monolithic design

- Tickets for concerts
- Single database
- Single user interface

Monolithic design



Monolithic design - backend



Exercise

Analyze and deconstruct a monolithic application

Software lifecycle

Development workflow

Multiple teams might work on the same codebase, which can lead to conflicts

Deployment process

Packaged and deployed as a single executable / archive

Maintenance

Updating one part of the application also (typically) affects other parts

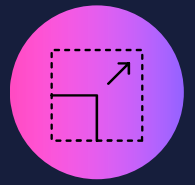
Containerization in monolithic applications

- Common but not necessary
- It helps to run in the same environment
- It’s easier to manage dependencies and deployment processes
- Containerizing a monolith doesn’t solve the scalability and maintenance issues

Microservices

Software development practices

Increase development speed



Scaling up

Not bound to a certain technology

Principles and patterns

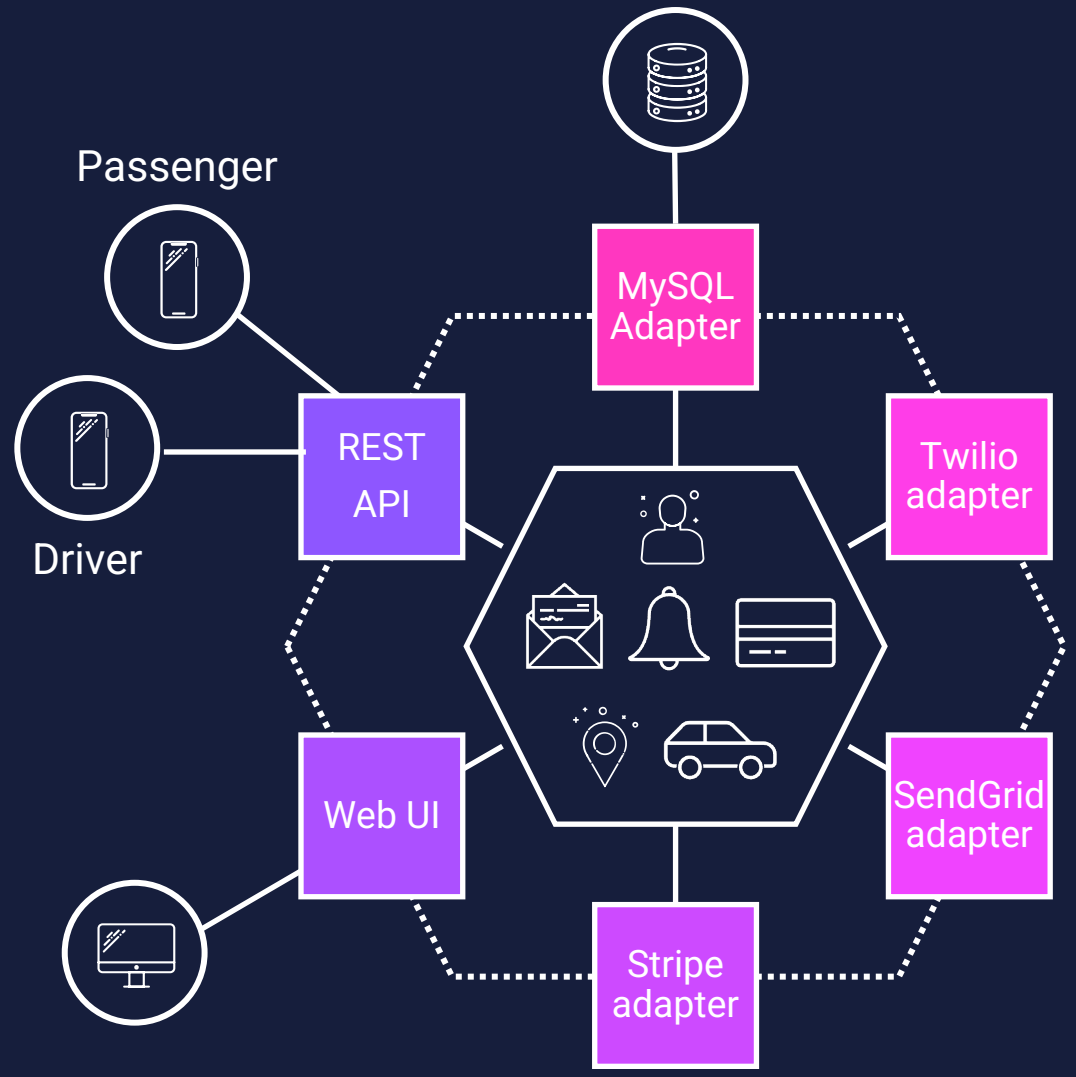
If you can't build a monolith, what makes you think microservices are the answer?

Simon Brown

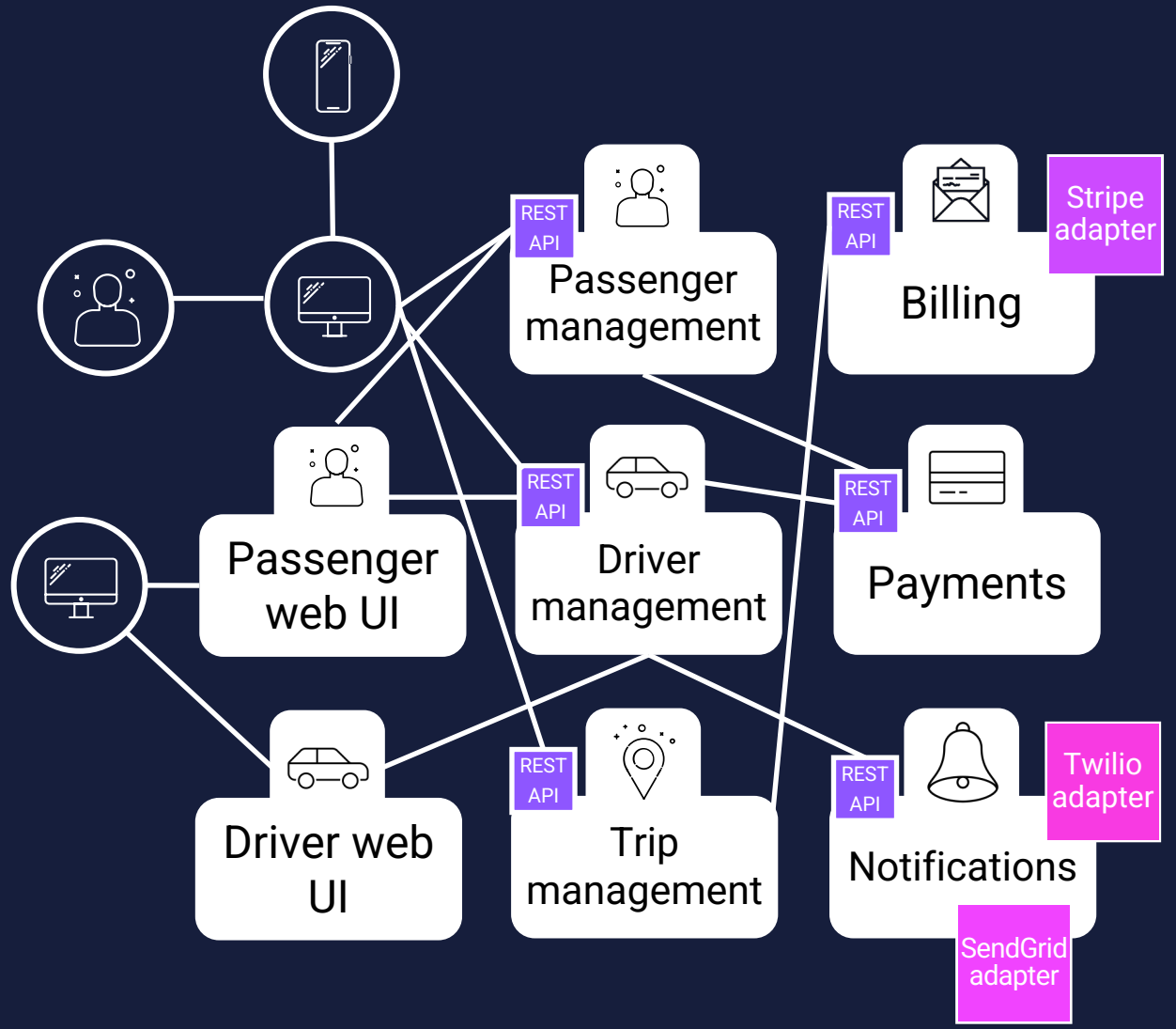
Exercise

Microservices vs monoliths

Monolith vs microservices



Monolithic architecture



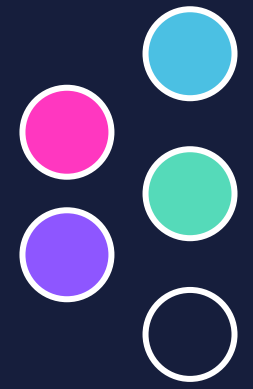
Microservices architecture

Monolith vs microservices

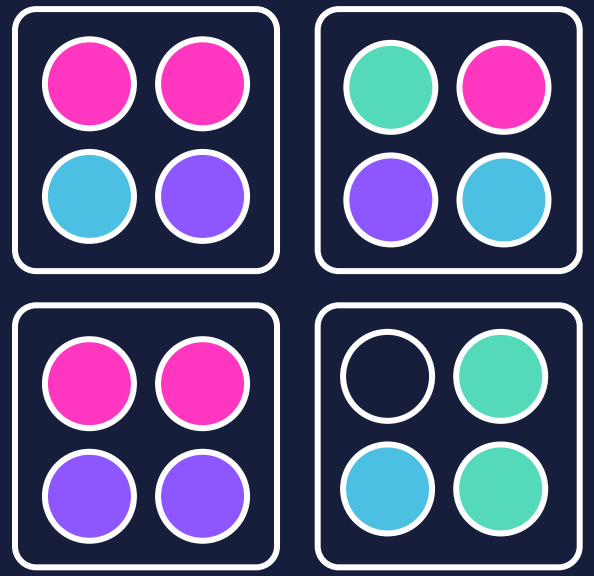
A monolith application combines all its functionality into a single process...

...and scales by replicating the entire application across multiple servers

A microservice architecture separates functionality into individual services...

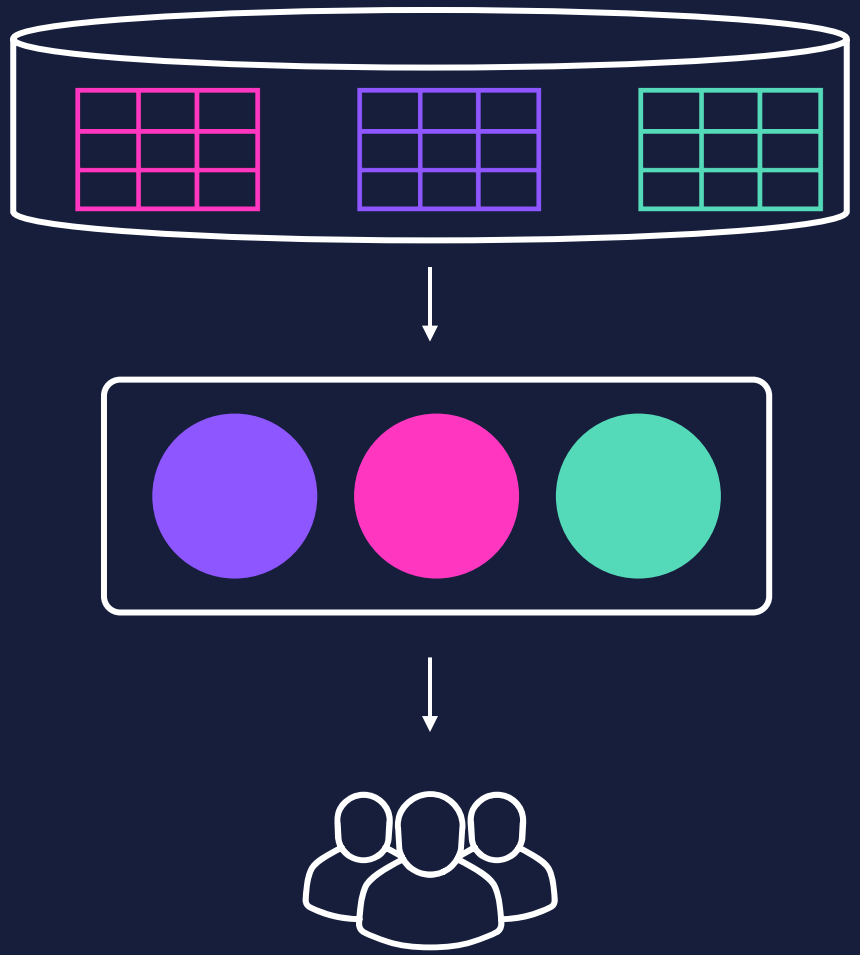


...scaling by distributing and replicating them across servers as needed

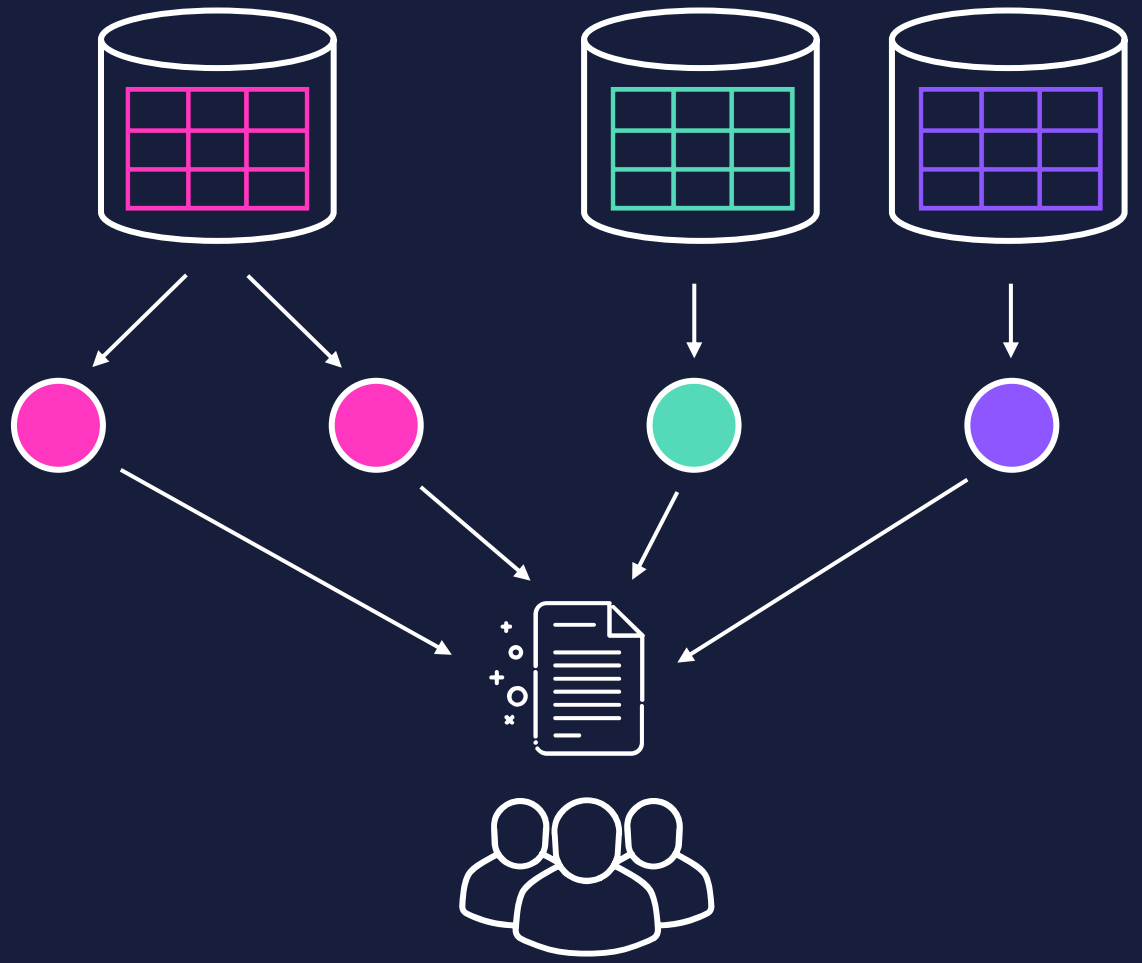


Decentralized data management

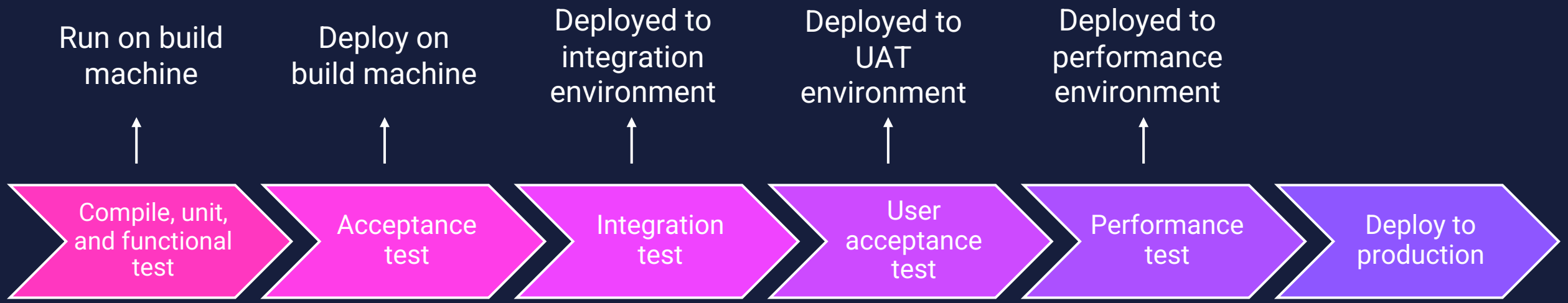
Monolith - single database



Microservices - application databases



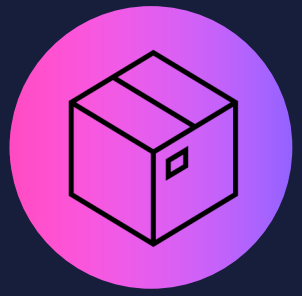
Infrastructure automation



- Scale up or down as needed
- Speed up deployment cycles
- Built for resilience and high availability

12 factor app

One **codebase** tracked in revision control, many deploys.



Explicitly declare and isolate **dependencies**.

Store **config** in the environment.

12 factor app

Treat **backing services** as attached resources.

Strictly **separate** build and run **stages**.

Execute the app as one or more stateless **processes**.

12 factor app

Export services via **port binding**.

Scale out via the process model (**concurrency**).

Maximize robustness with fast startup and graceful shutdown (**disposability**).

12 factor app

Keep development, staging, and production as similar as possible (**dev/prod parity**).

Treat **logs** as event streams.

Run admin/management tasks as **one-off processes**.

There's always a third option...

Modular monoliths

- A modular monolith is an architectural style where a monolithic application is designed with a modular structure internally.
- Clear boundaries between modules within the monolith.
- Still deployed as one unit but with better internal organization.

Pros of modular monoliths

- Easier to understand and modify individual modules
- Improved code organization and maintainability
- Easier to onboard new developers due to clear module boundaries
- Can serve as a steppingstone towards microservices if needed

Cons of modular monoliths

Cannot scale modules independently at runtime

Changes in one module still require redeploying the entire application

Use cases modular monoliths

When the application doesn’t warrant microservices

Teams with limited resources or early-stage startups

Exercise

Choosing the right design

Which of the following statements best describes a monolithic architecture?

{A} An application composed of small, independent services that communicate over well-defined APIs.

{B} An application where all components are interconnected and interdependent, deployed as a single unit.

{C} An architecture that allows each service to be developed, deployed, and scaled independently.

{D} An application that uses containers to isolate services from each other.

Which of the following statements best describes a monolithic architecture?

{A} An application composed of small, independent services that communicate over well-defined APIs.

{B} **An application where all components are interconnected and interdependent, deployed as a single unit.**

{C} An architecture that allows each service to be developed, deployed, and scaled independently.

{D} An application that uses containers to isolate services from each other.

What is a primary advantage of a microservices architecture over a traditional monolithic architecture?

{A} Simplified deployment due to a single codebase.

{B} Reduced complexity in managing distributed systems.

{C} Unified technology stack across the entire application.

{D} Easier scaling of specific components independently.

What is a primary advantage of a microservices architecture over a traditional monolithic architecture?

{A} Simplified deployment due to a single codebase.

{B} Reduced complexity in managing distributed systems.

{C} Unified technology stack across the entire application.

{D} Easier scaling of specific components independently.

Which of the following best describes a modular monolith (a.k.a. "majestic monolith")?

{A} A monolithic application with no internal modularity.

{B} A microservices application that is deployed as multiple units.

{C} A monolithic application designed with a modular internal structure but still deployed as a single unit.

{D} An application that uses containerization to deploy services independently.

Which of the following best describes a modular monolith (a.k.a. "majestic monolith")?

{A} A monolithic application with no internal modularity.

{B} A microservices application that is deployed as multiple units.

{C} **A monolithic application designed with a modular internal structure but still deployed as a single unit.**

{D} An application that uses containerization to deploy services independently.

Next up:

Building and containerizing microservices

Questions or suggestions?

maaikejvp@gmail.com

See you tomorrow!