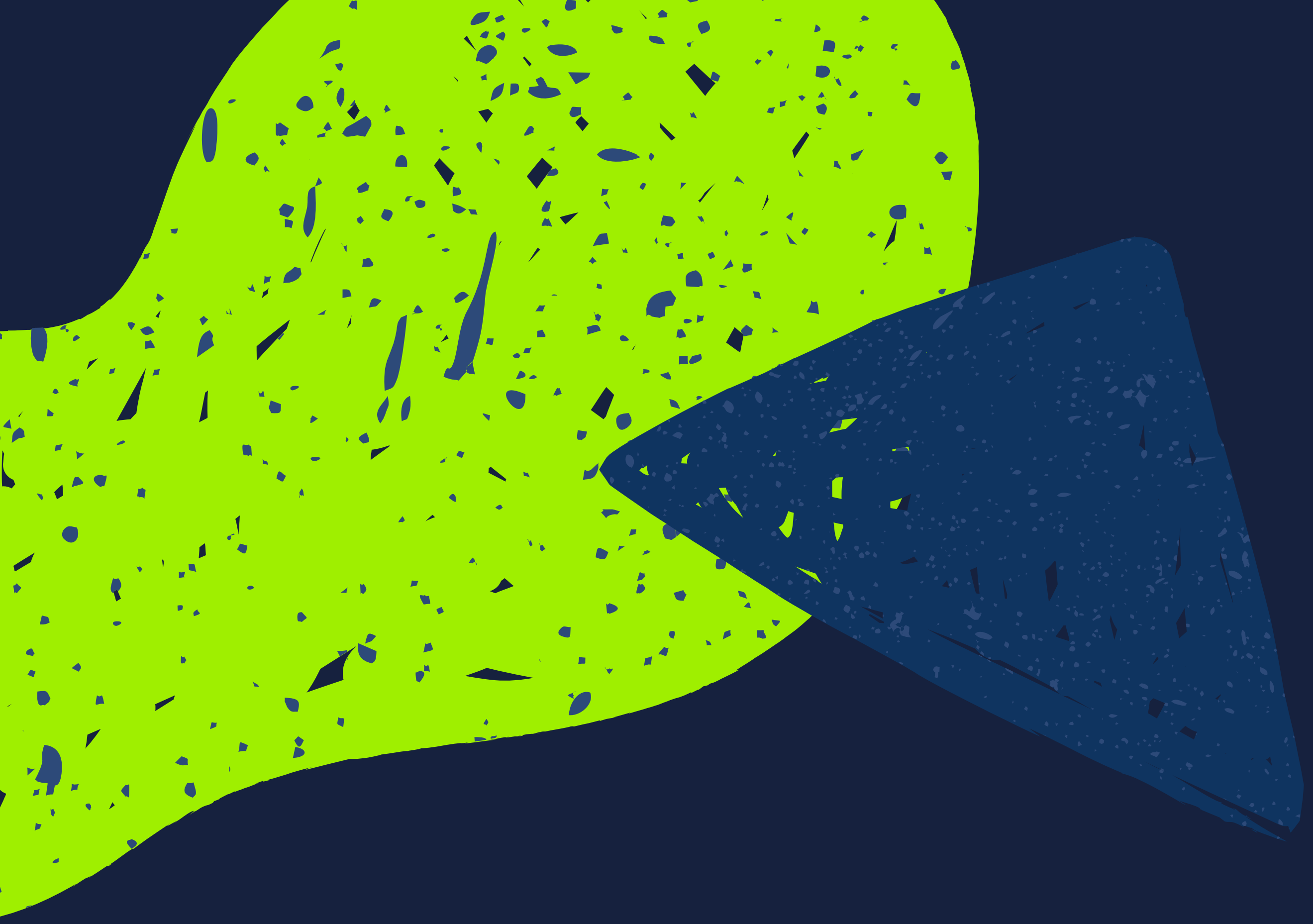
Introduction to Distributed Systems

Overview of Web Services Concepts

Ph.D Mohammed Sahraoui

Content



Understanding Distributed Systems Fundamentals

Distributed systems are crucial for modern applications, enabling **scalability, resource sharing**, and **fault tolerance** across diverse environments. They play a vital role in improving performance and reliability in today's interconnected software landscape.

Scalability

Scalability allows systems to handle increased load effectively.

Resource Sharing

Resource sharing optimizes utilization of available computing resources.

Fault Tolerance

Fault tolerance ensures system reliability despite component failures.

Understanding Distributed Systems Fundamentals

A distributed system is a collection of independent computers that appear to the user as a single coherent system. Instead of one powerful machine doing all the work, tasks are split across multiple nodes that communicate and coordinate over a network

Core Characteristics

Concurrency:
Many processes run simultaneously across nodes.

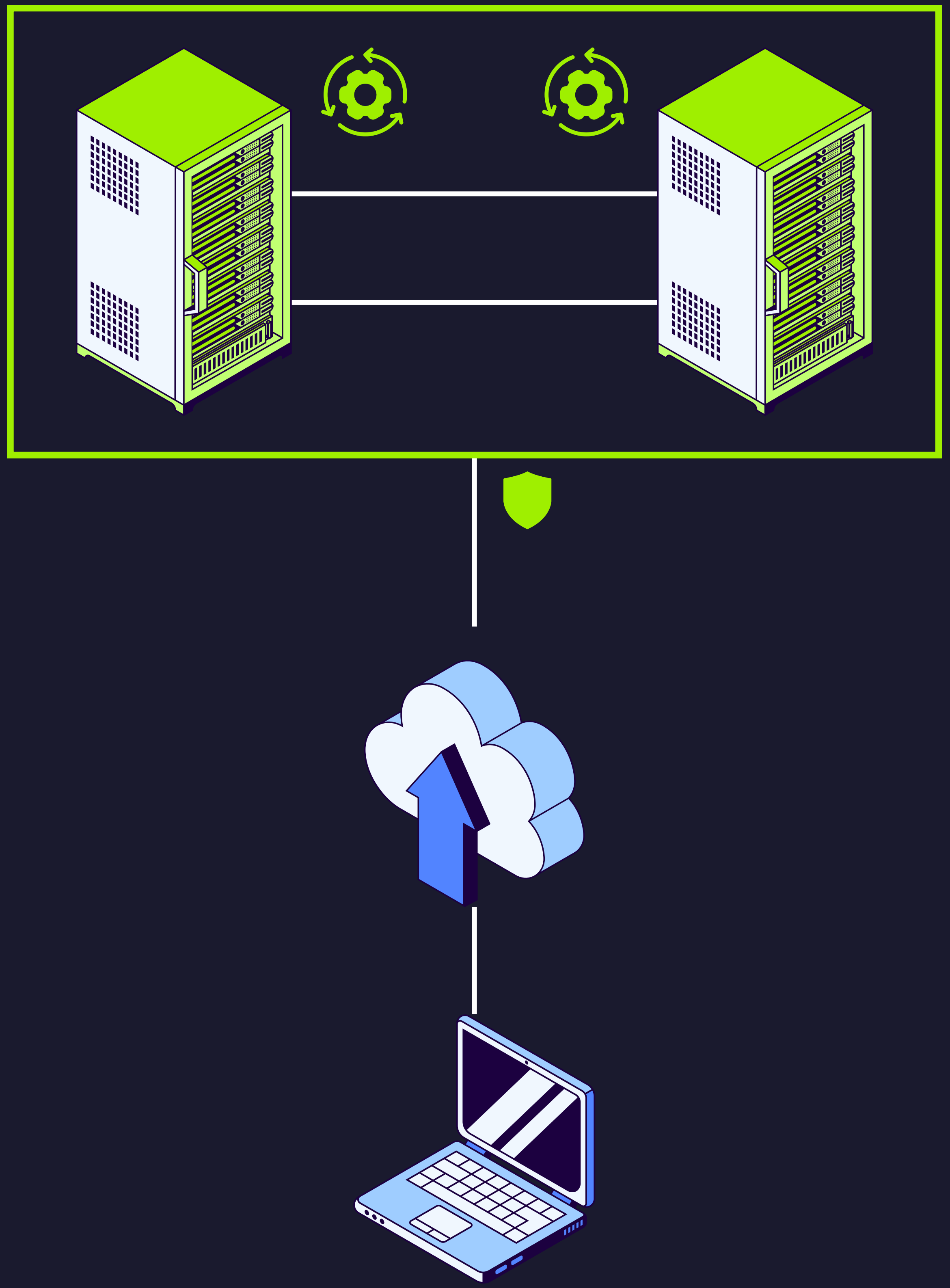
Scalability:
Systems can grow by adding more machines.

Fault Tolerance:
If one node fails, others can continue the work.

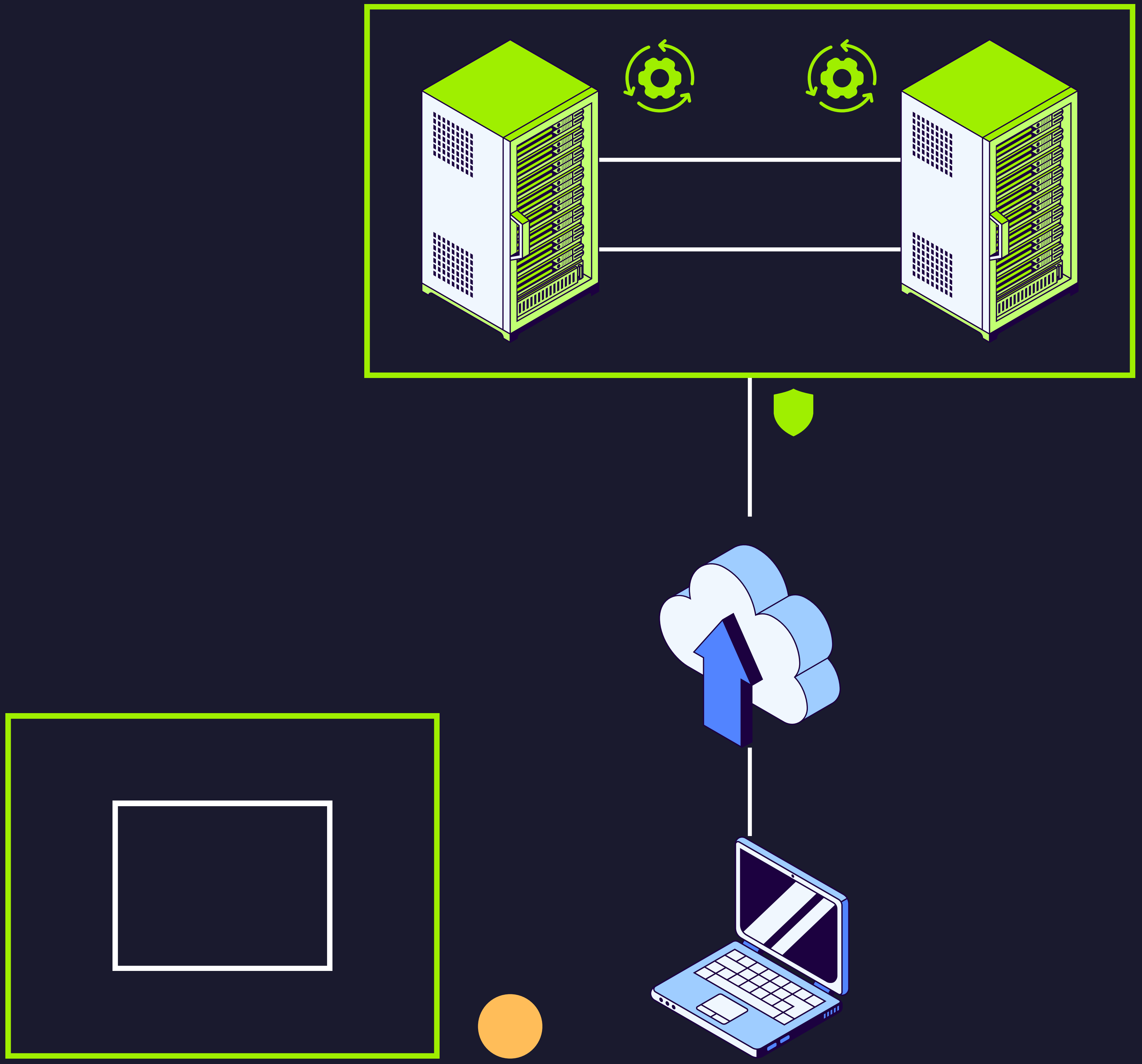
Transparency:
Users shouldn’t notice the complexity (location, replication, failures are hidden).

Heterogeneity:
Nodes may differ in hardware, OS, or network, but still cooperate.

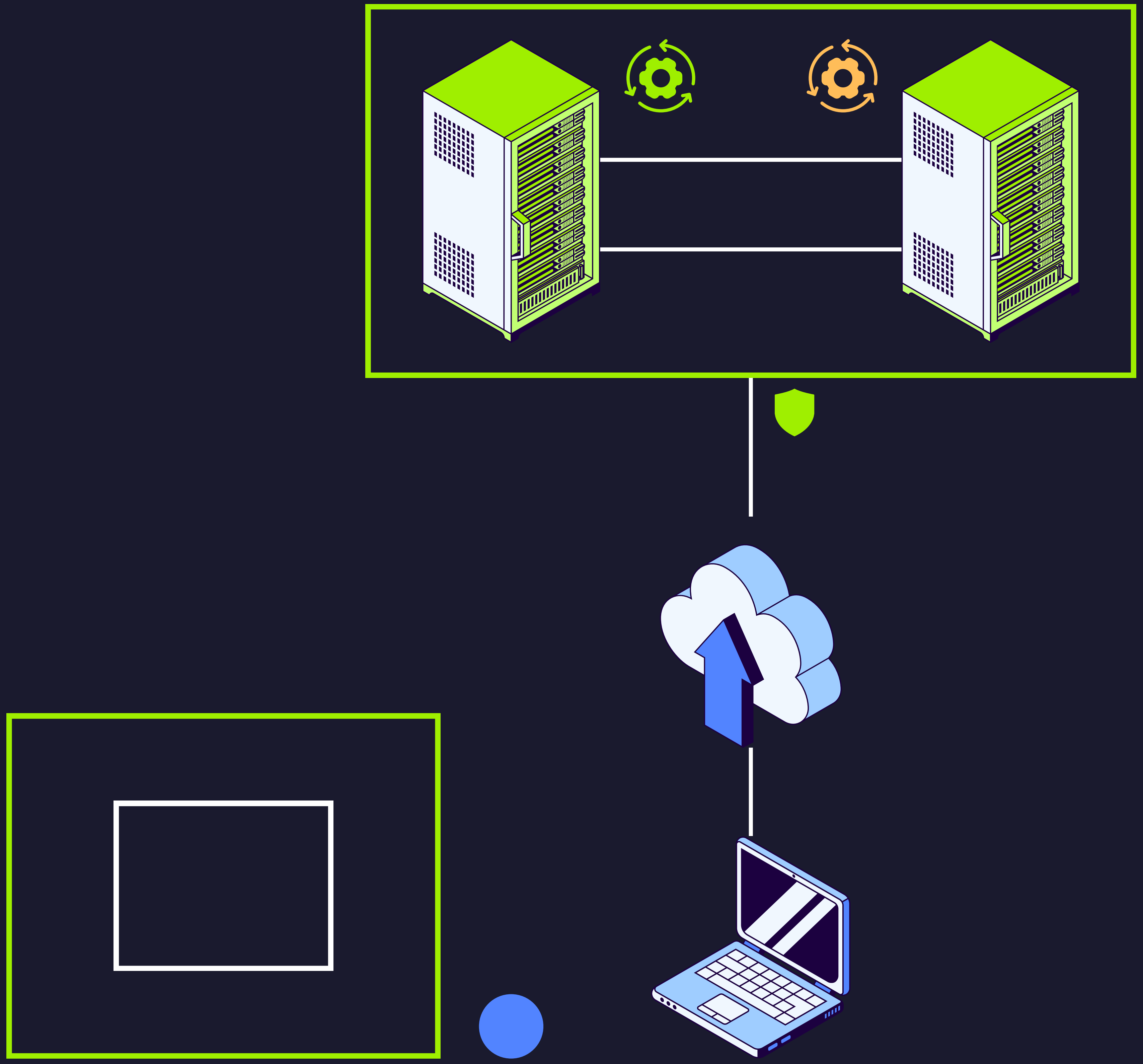
Concurrency:



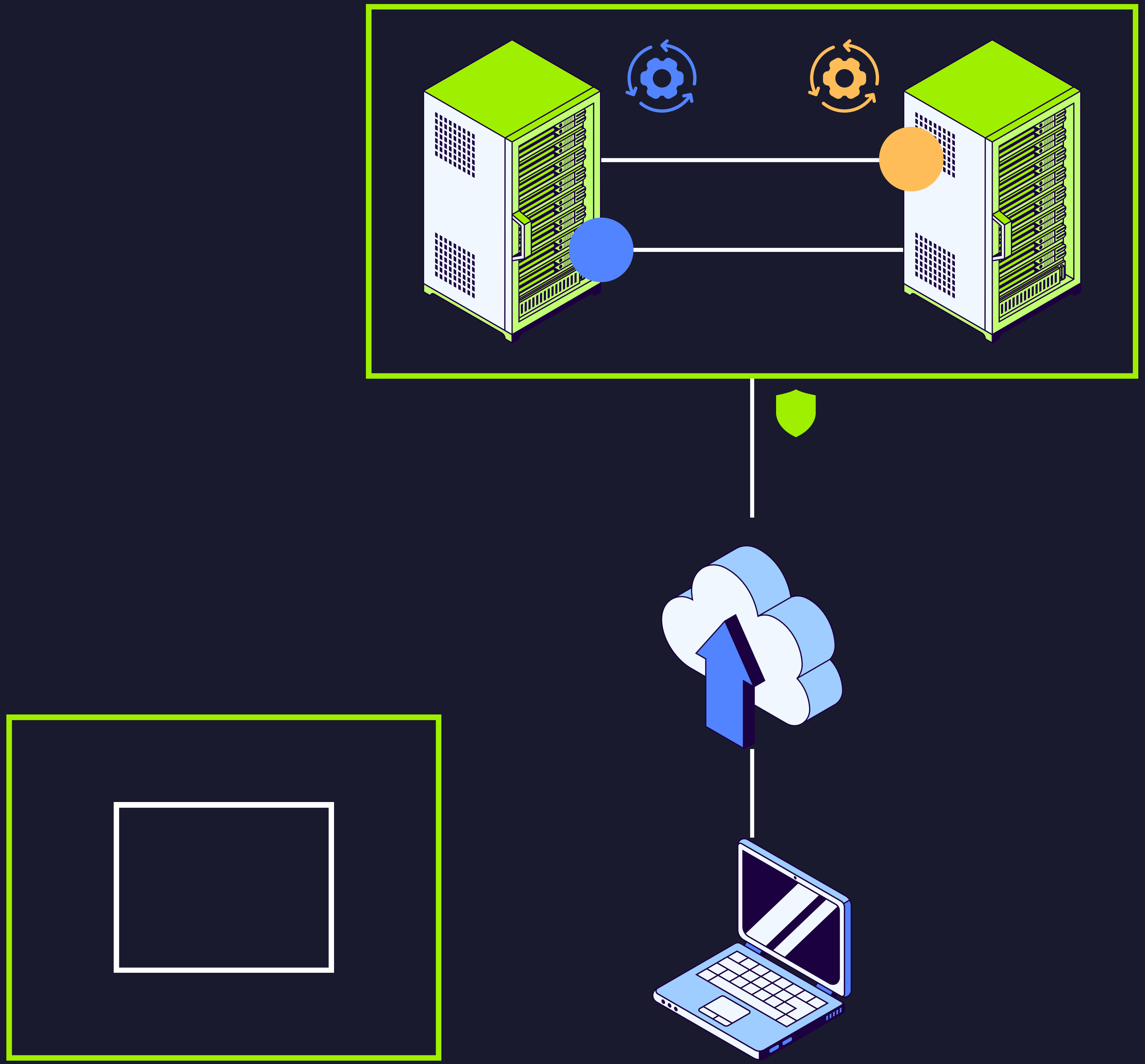
Concurrency:



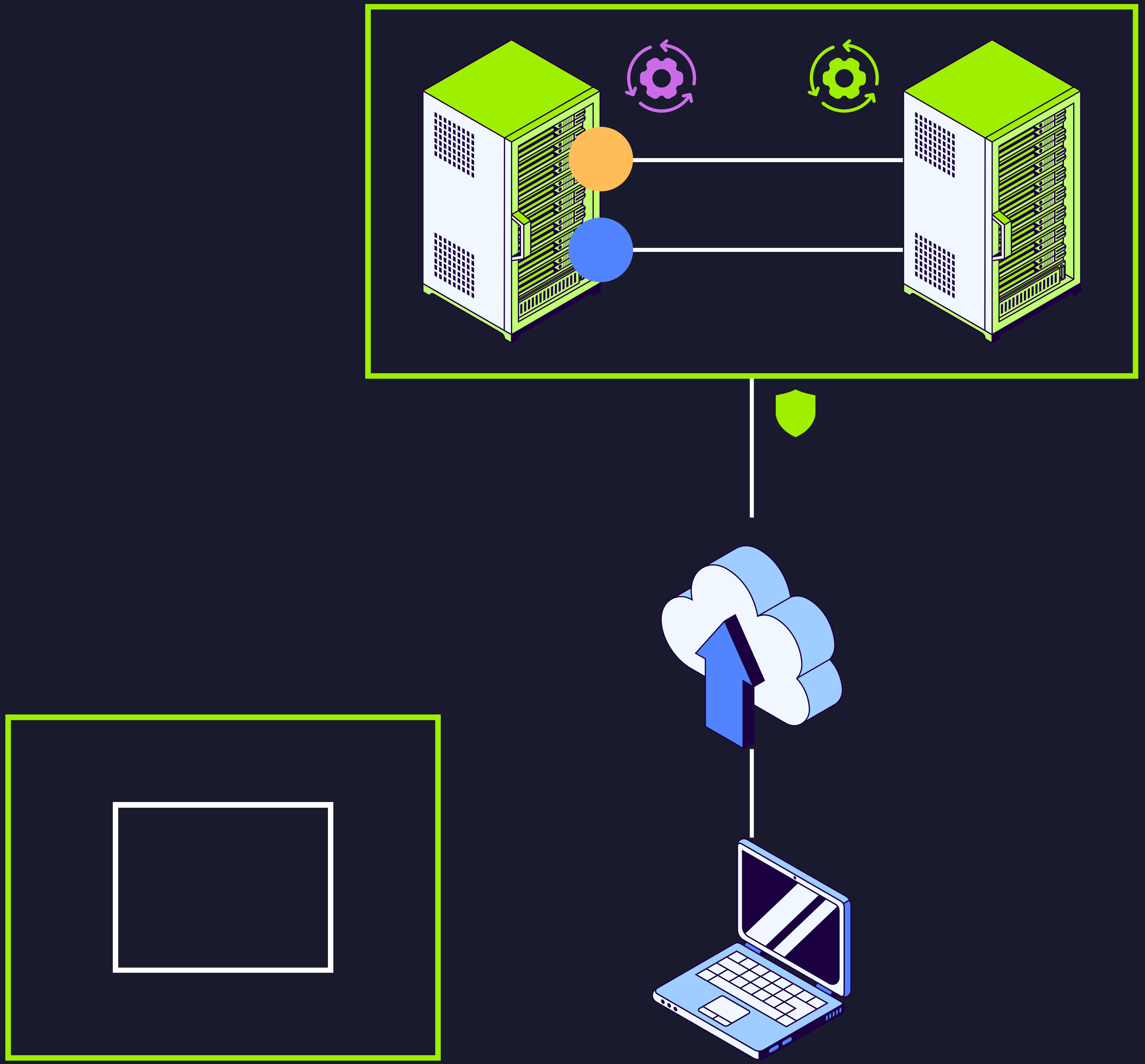
Concurrency:



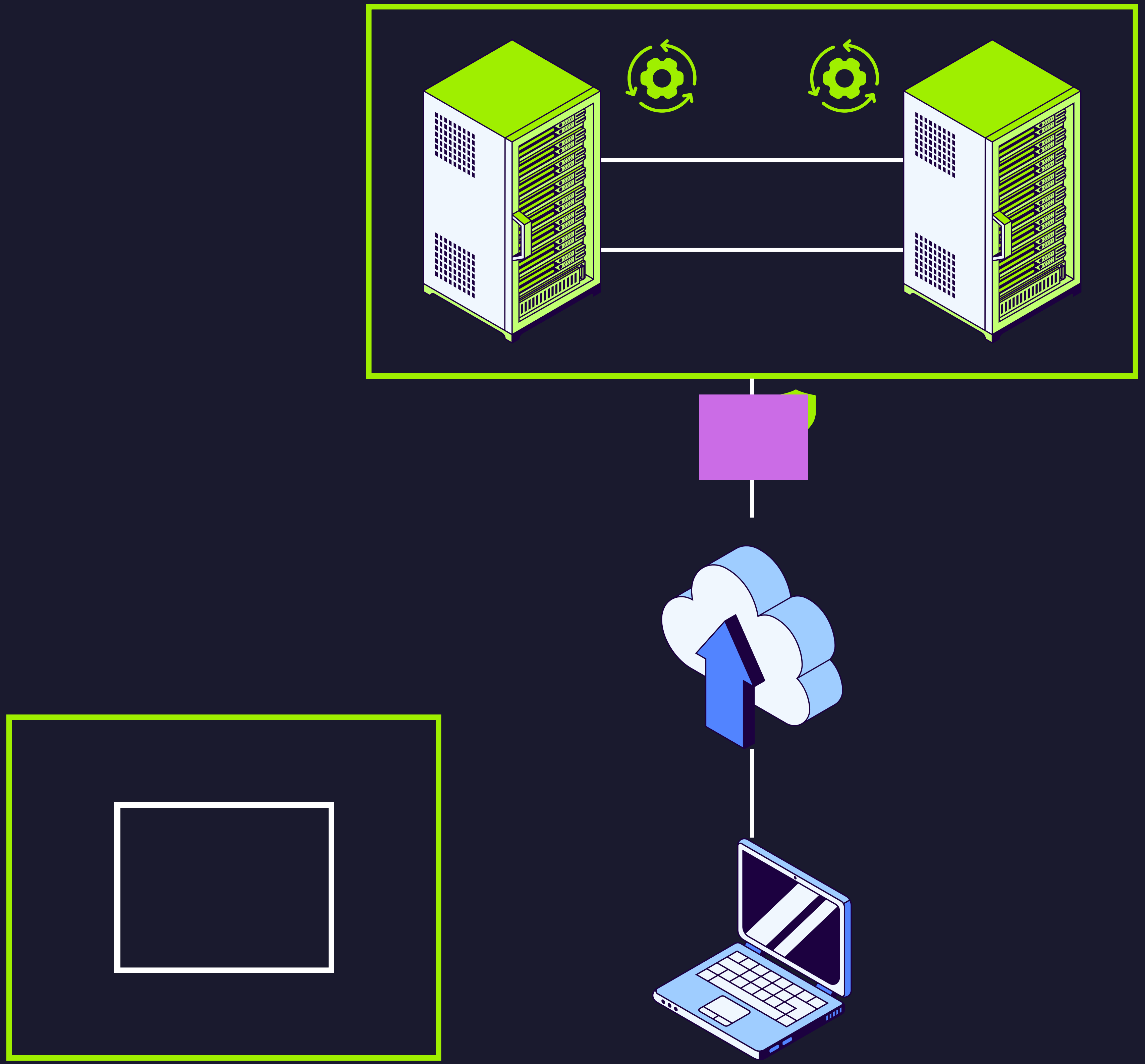
Concurrency:



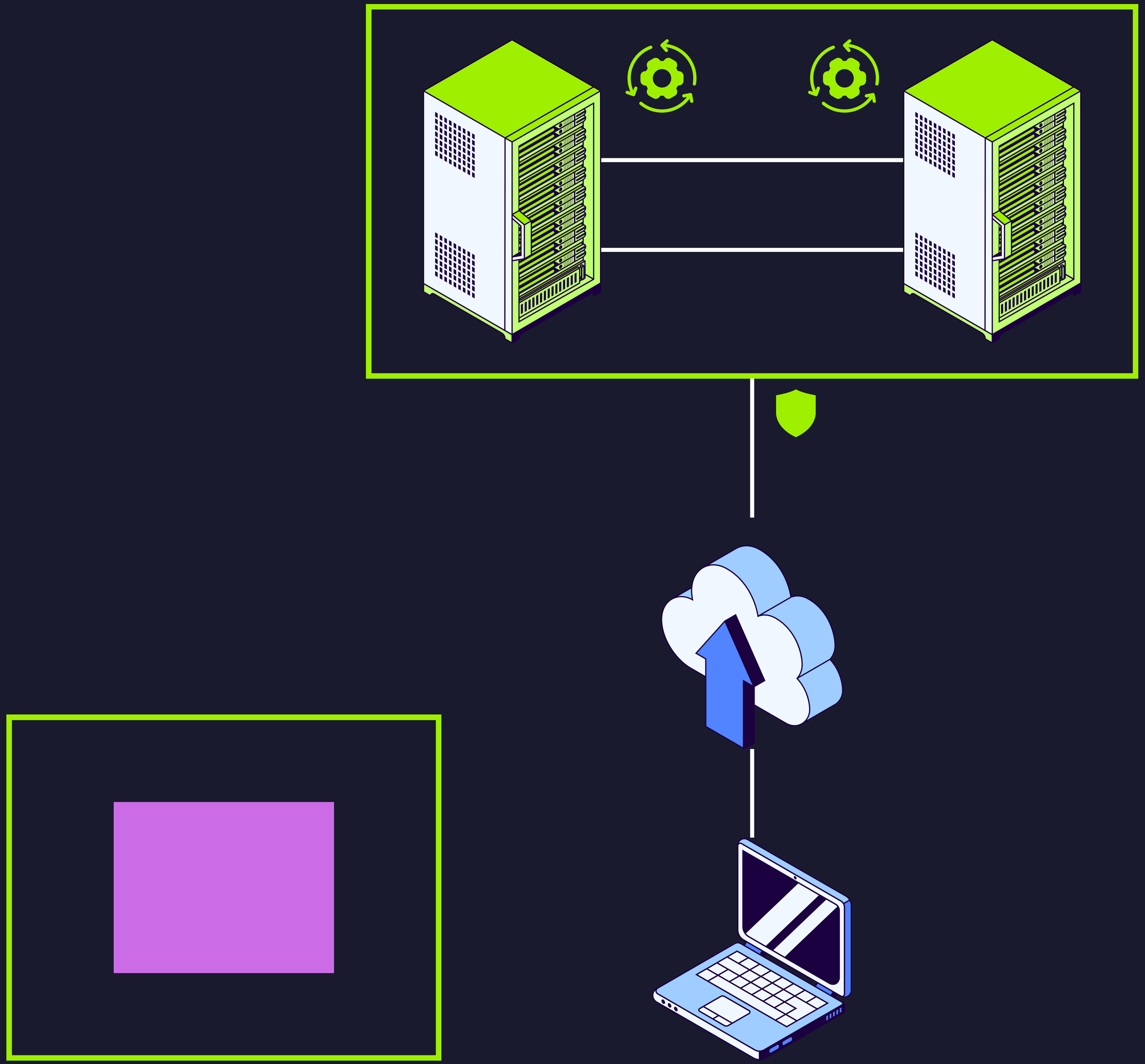
Concurrency:



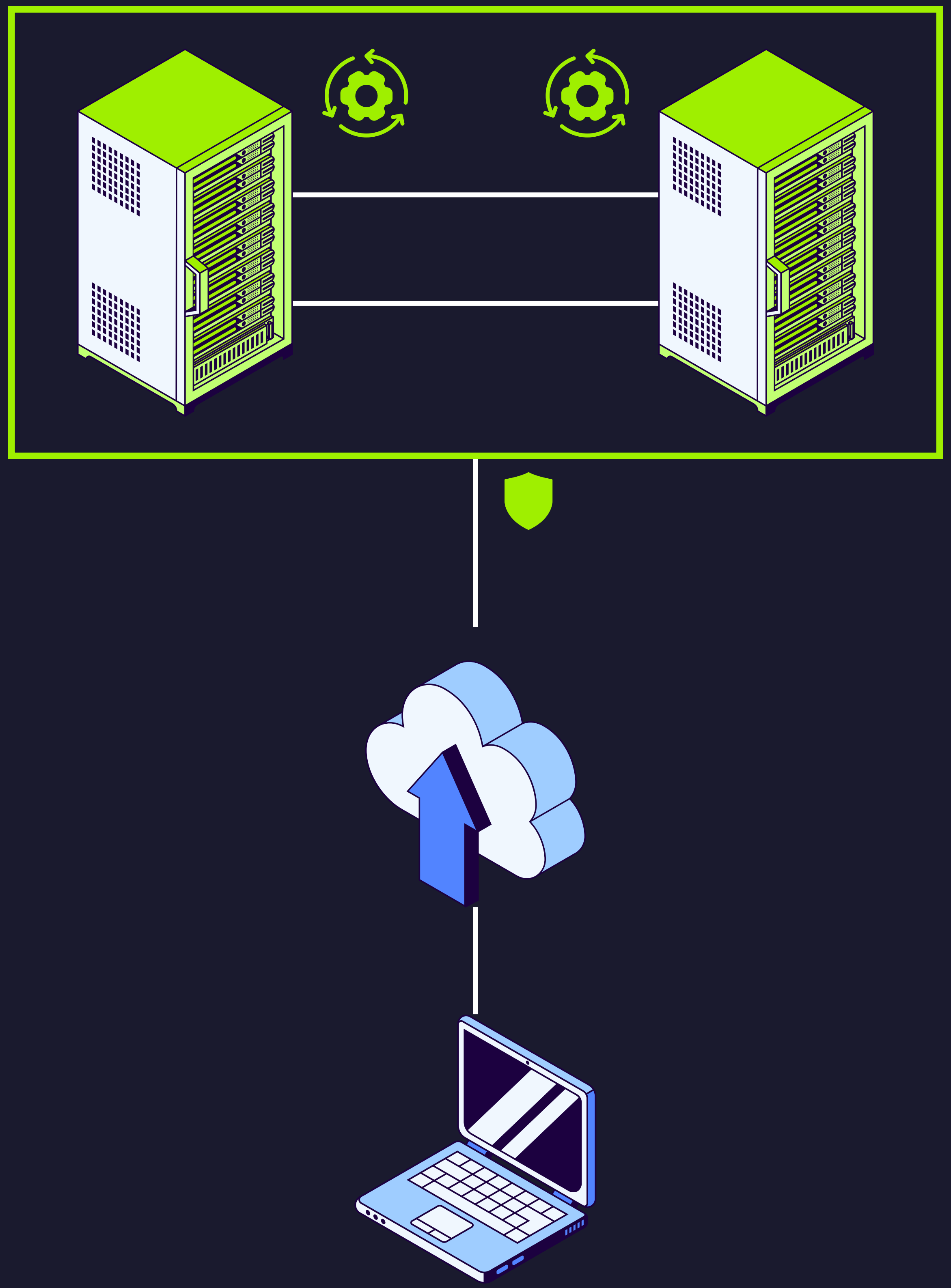
Concurrency:



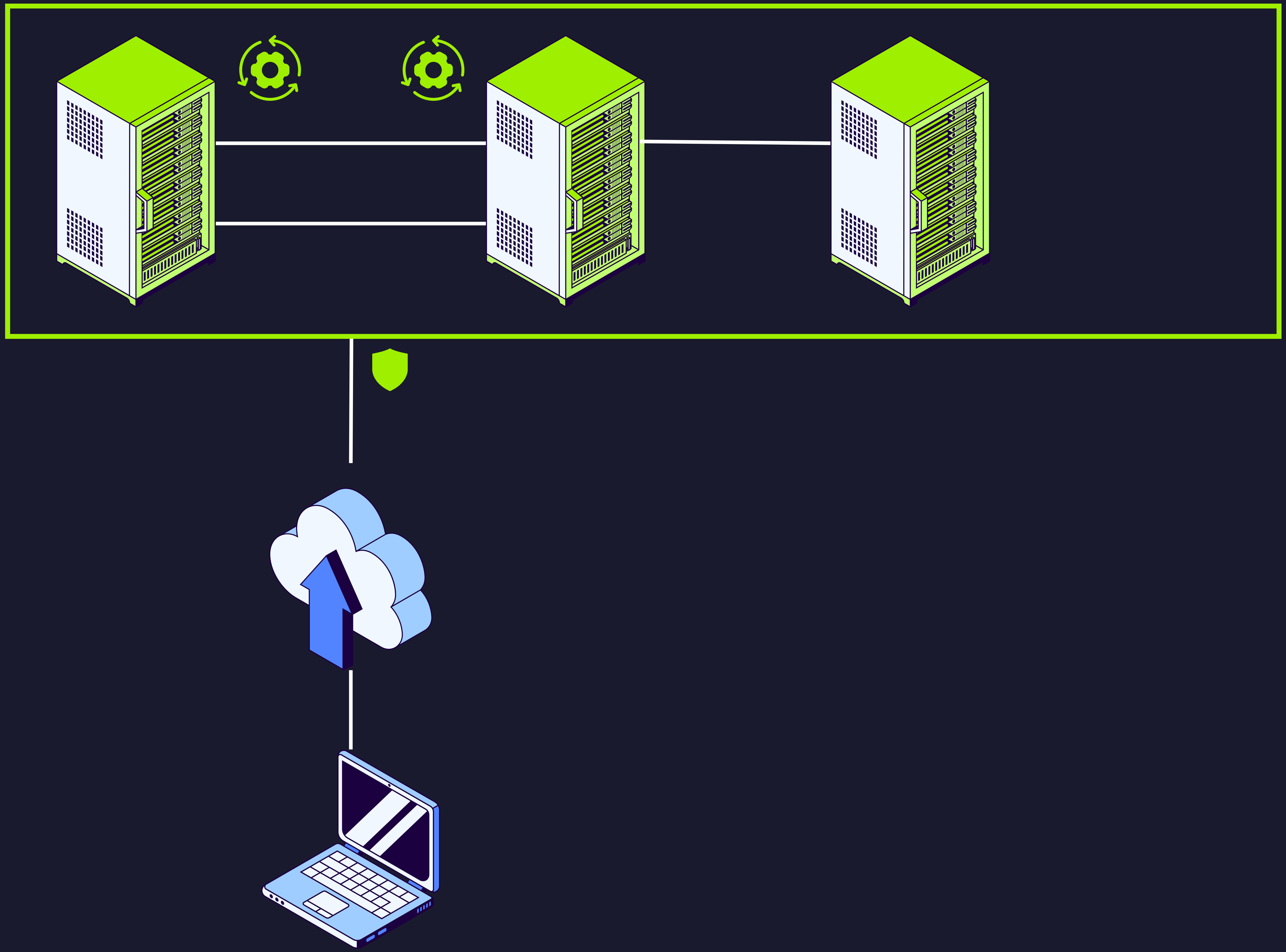
Concurrency:



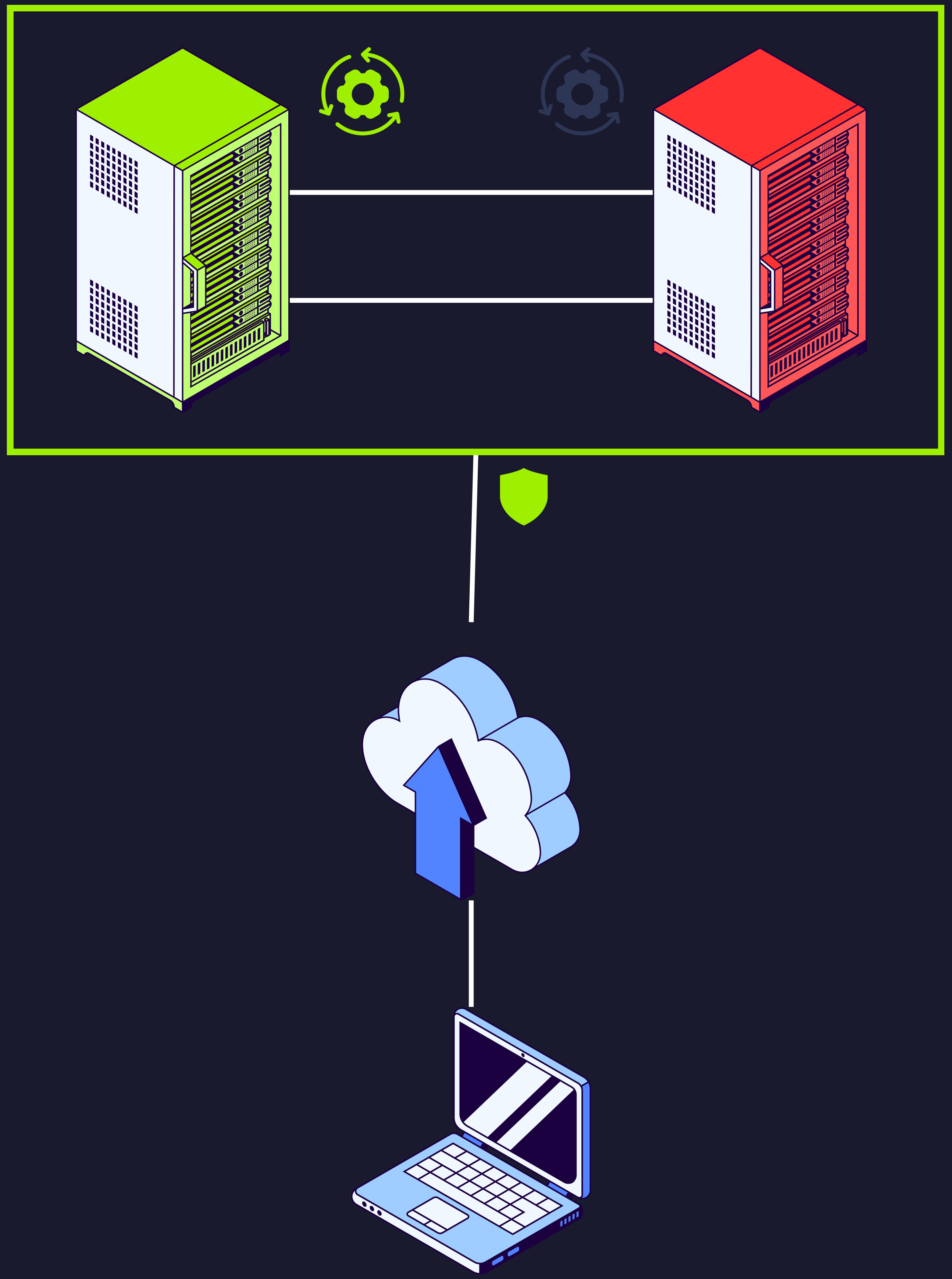
Scalability:



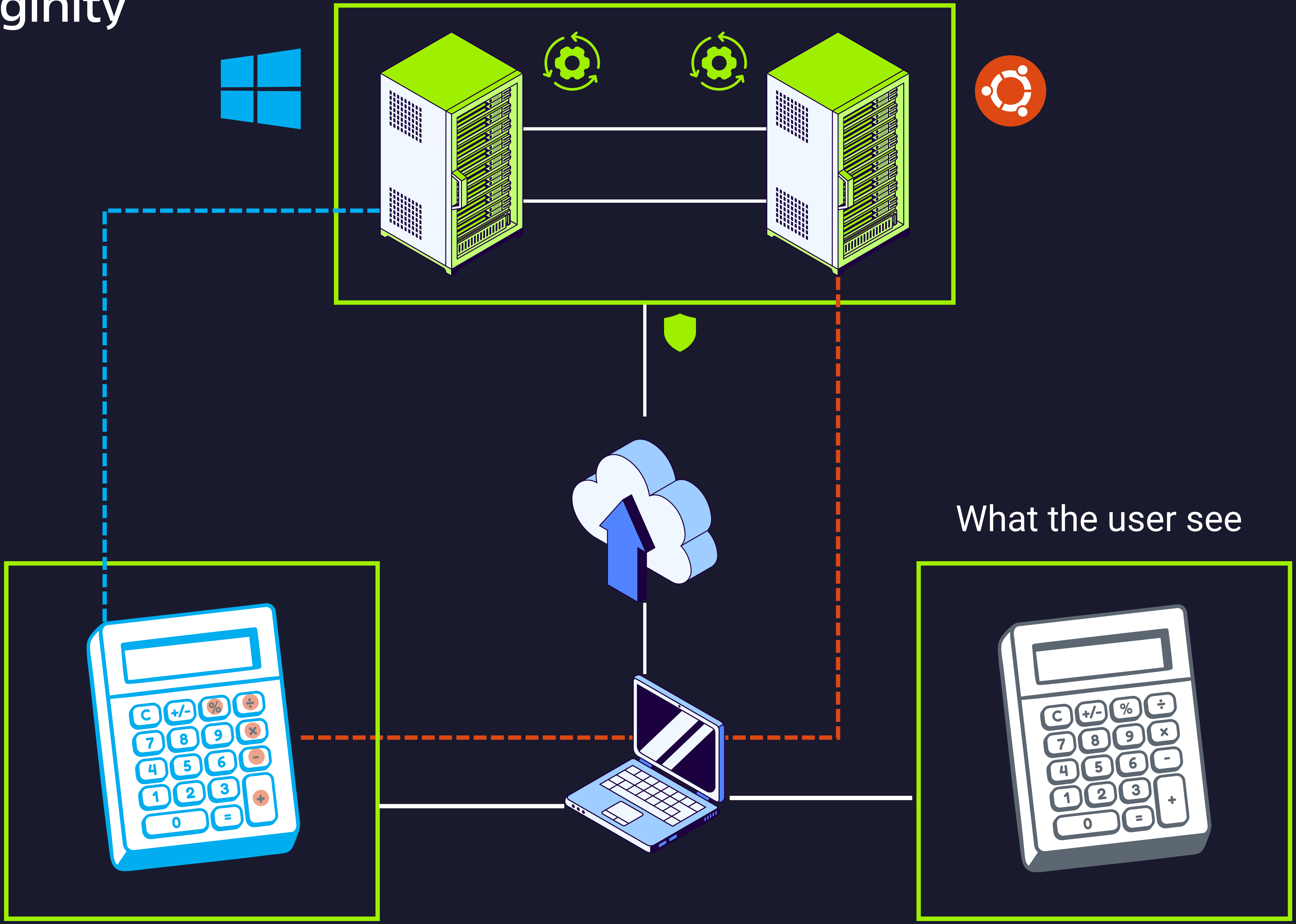
Scalability:



Fault Tolerance



Transparency and heteroginity



Common Architectures

There are various architectural styles, but the most prevalent ones are as follows:

Client-Server

Centralized servers handle requests from clients.

Peer-to-Peer (P2P)

Every node can act as both client and server.

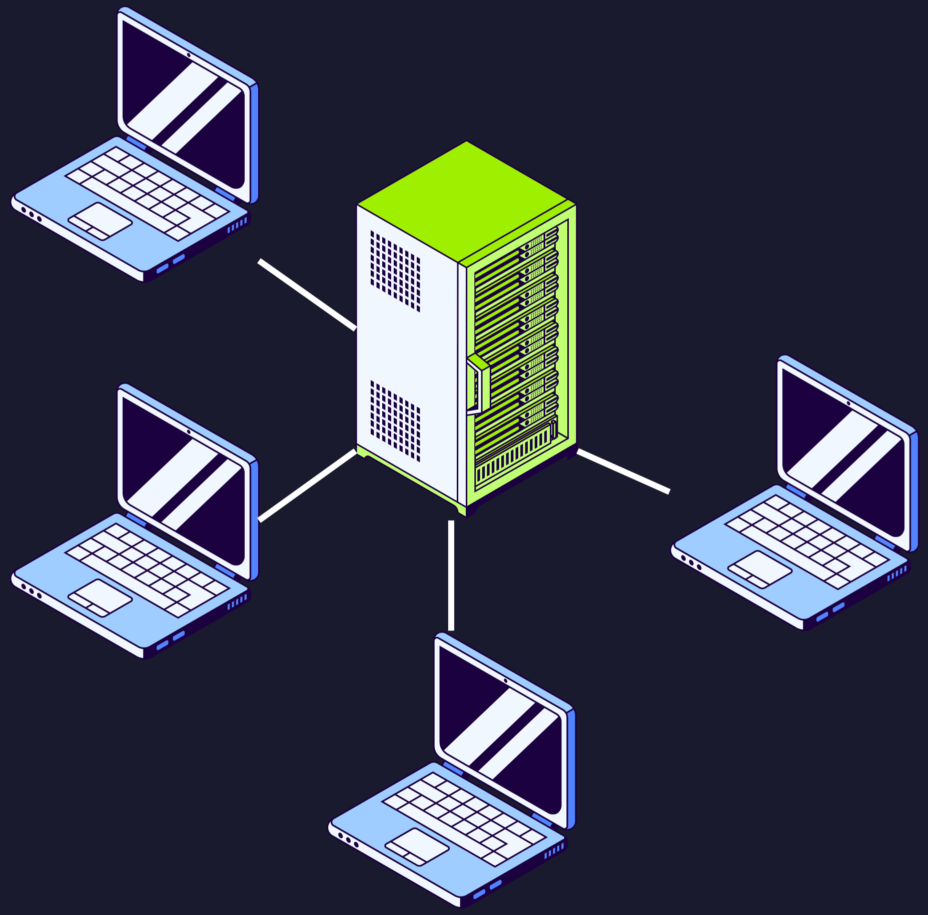
Microservices

Applications split into small, independent services communicating via APIs

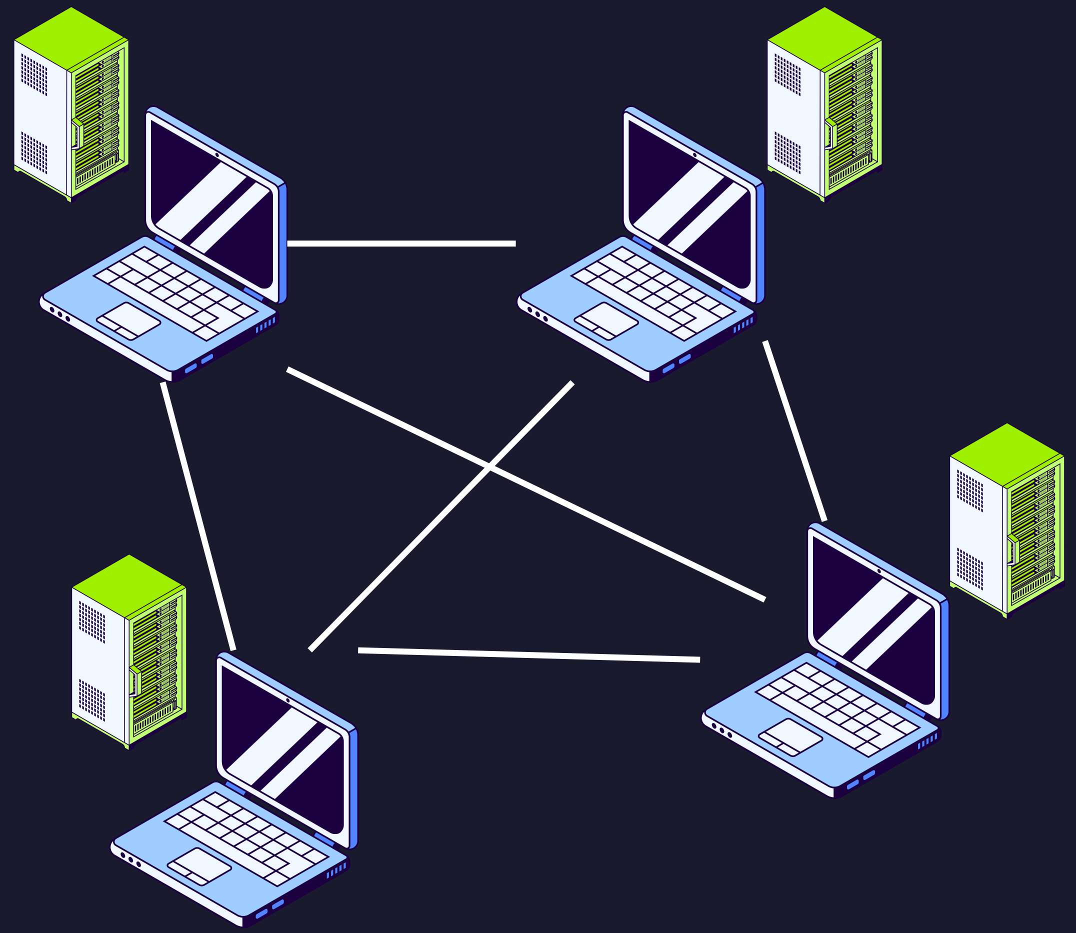
Cloud & Edge Systems

Distributed resources across data centers and edge devices

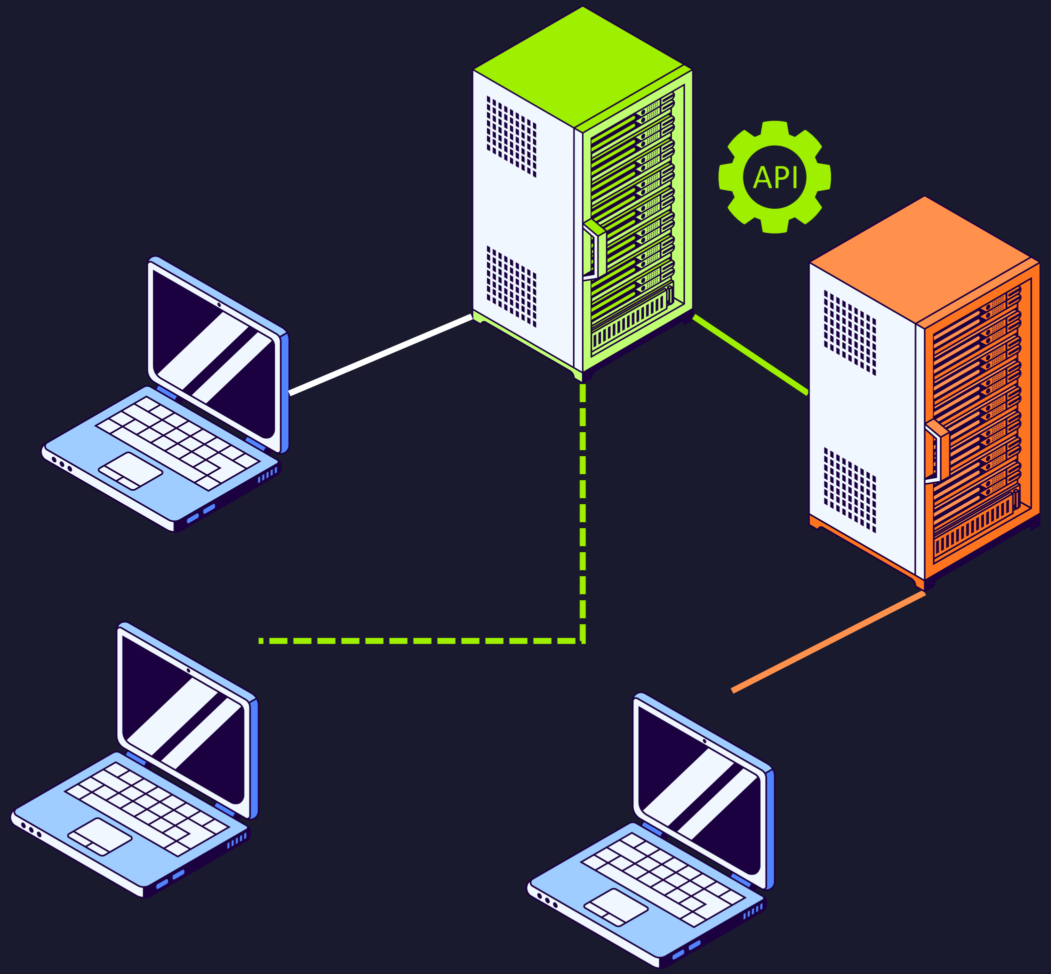
Server-Client Architecture



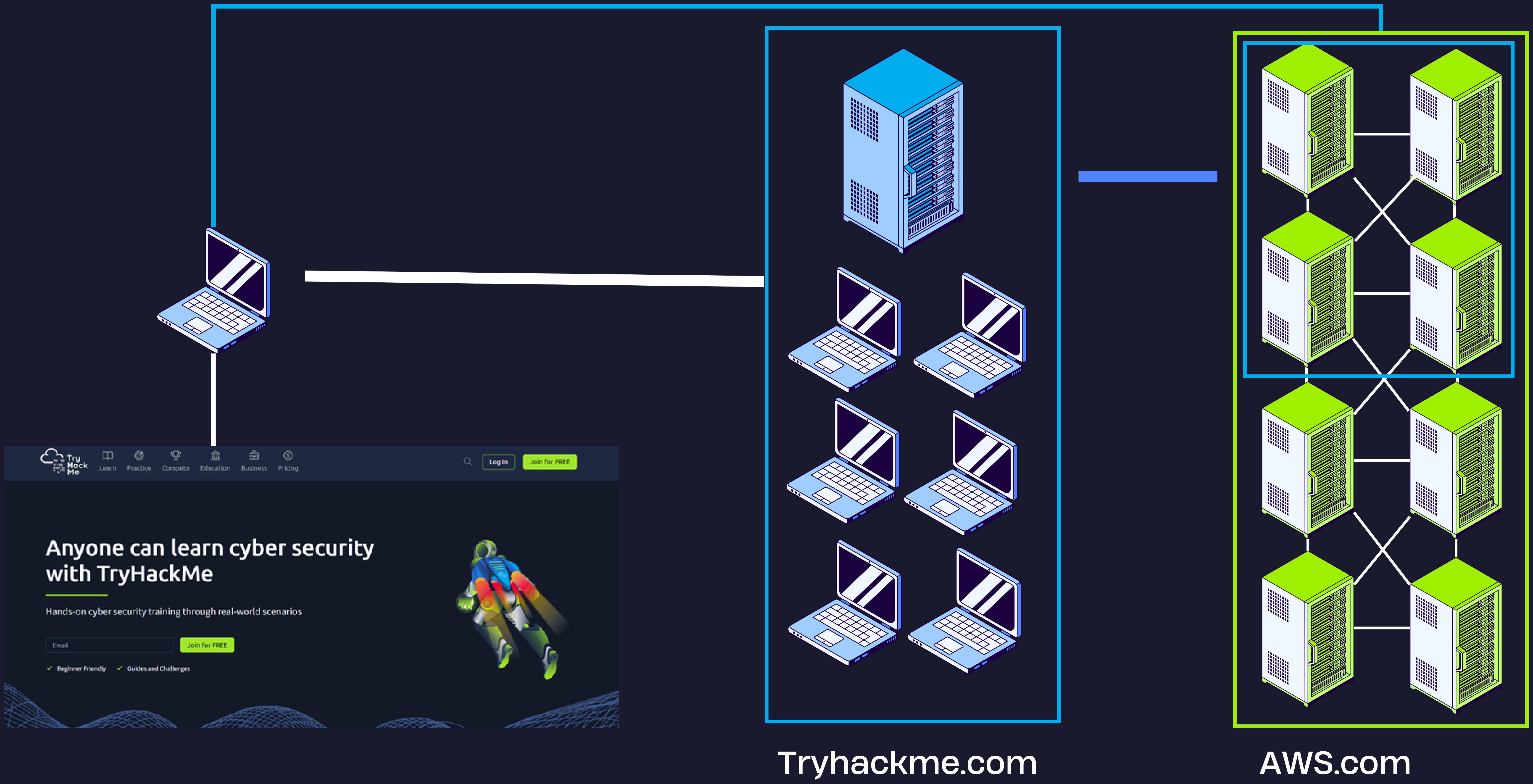
peer-to-peer



Microservices



Cloud & Edge Systems



Anyone can learn cyber security with TryHackMe

Hands-on cyber security training through real-world scenarios

✓ Beginner Friendly ✓ Guides and Challenges

Real-World Examples

Distributed systems play a crucial role in modern applications, enhancing efficiency and connectivity across various sectors.

Online Banking

Offers secure transactions and real-time updates for users.

Cloud Services

Provides scalable storage and computing power accessible from anywhere.

E-commerce Platforms

Facilitates seamless transactions and user experiences through distributed architecture.

Social Media

Enables global connectivity and content sharing across diverse user bases.

Streaming Services

Delivers content to millions of users through distributed networks and resources.

Evolution of Application Integration

Understanding the evolution from monolithic architecture is essential for modern application design and scalability.

Monolithic

Monolithic applications are built as a single, interconnected unit.

Limitations

They face challenges in scalability, maintainability, and deployment processes.

Service-Oriented

Service-Oriented Architecture (SOA) emerged to address these monolithic limitations.

Microservices

Microservices allow for fine-grained, independent service deployment and management.

Scalability

Enhanced scalability is achieved through distributing workloads across multiple services.

Service-Oriented Architecture Explained

Service-Oriented Architecture (SOA) enables flexible, modular systems by connecting independent services through standardized protocols.

Modularity

Systems can be easily modified without affecting other components, enhancing adaptability.

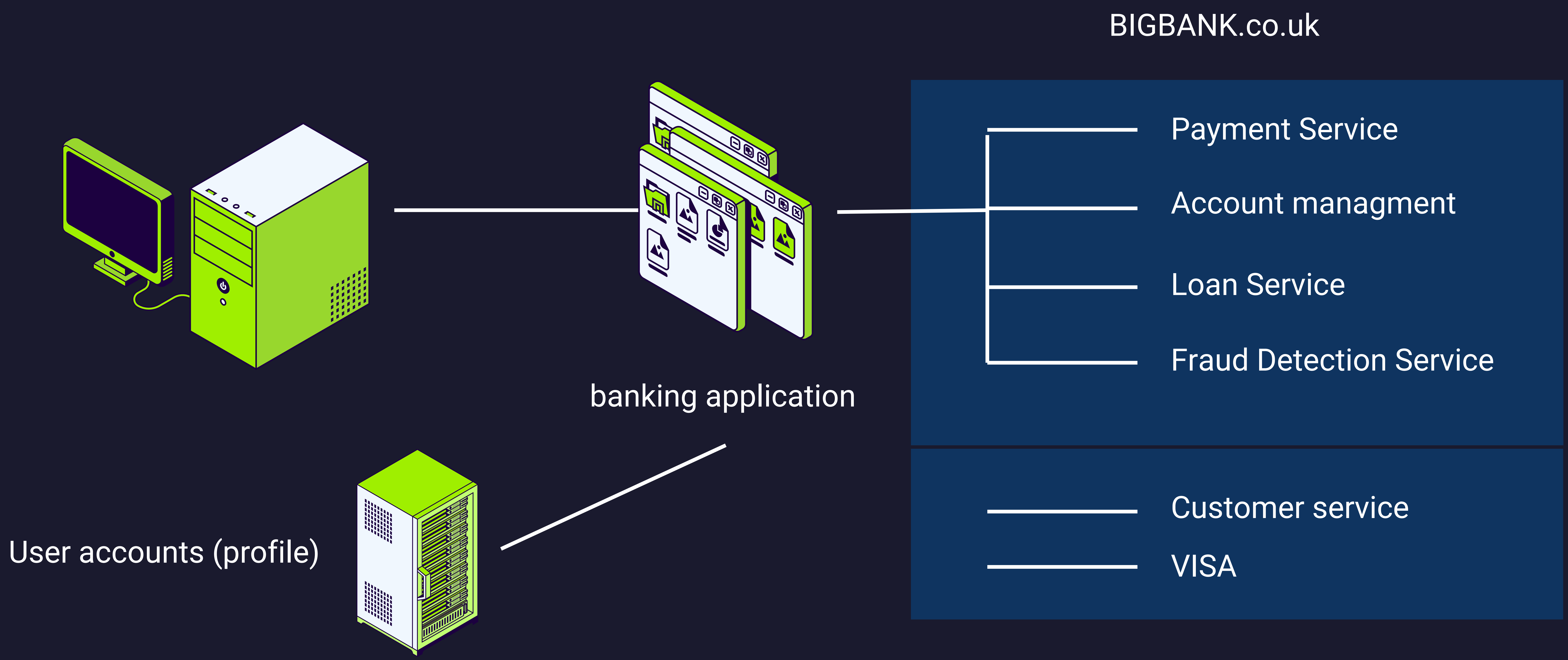
Reusability

Services can be reused across different applications, saving time and resources.

Interoperability

SOA allows diverse systems to communicate seamlessly, promoting integration and collaboration.

Service Oriented Application



Evolution Timeline

2000

Monolithic applications dominated the software development landscape.

2005

Service-Oriented Architecture (SOA) introduced modular application design.

2010

Microservices architecture emerged, enhancing scalability and flexibility.

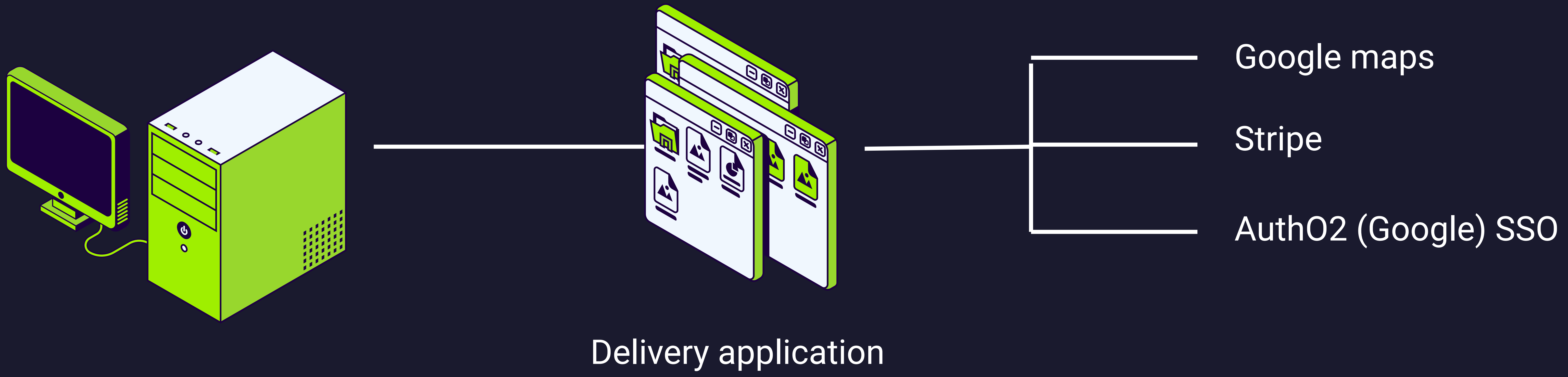
2015

Containerization technologies streamlined deployment of microservices applications.

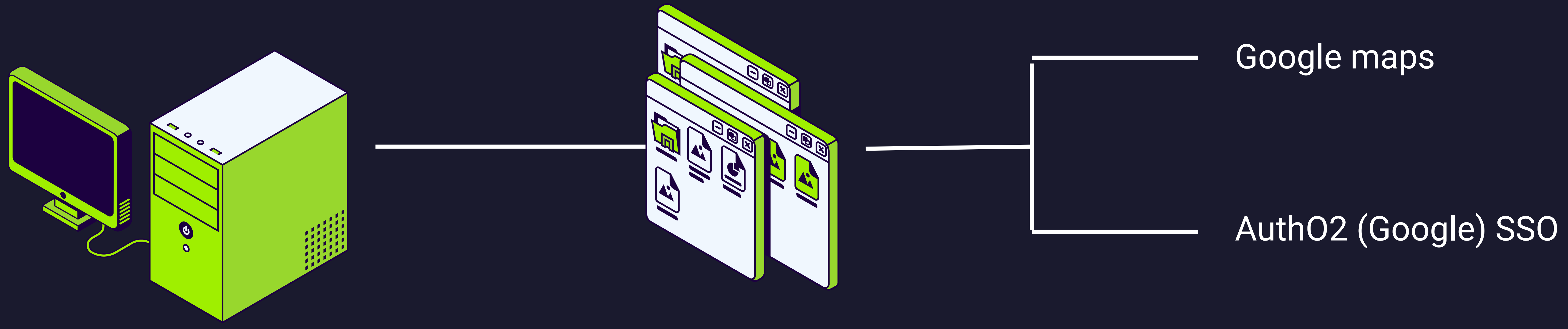
2020

Cloud services became essential for modern software infrastructure.

Microservices



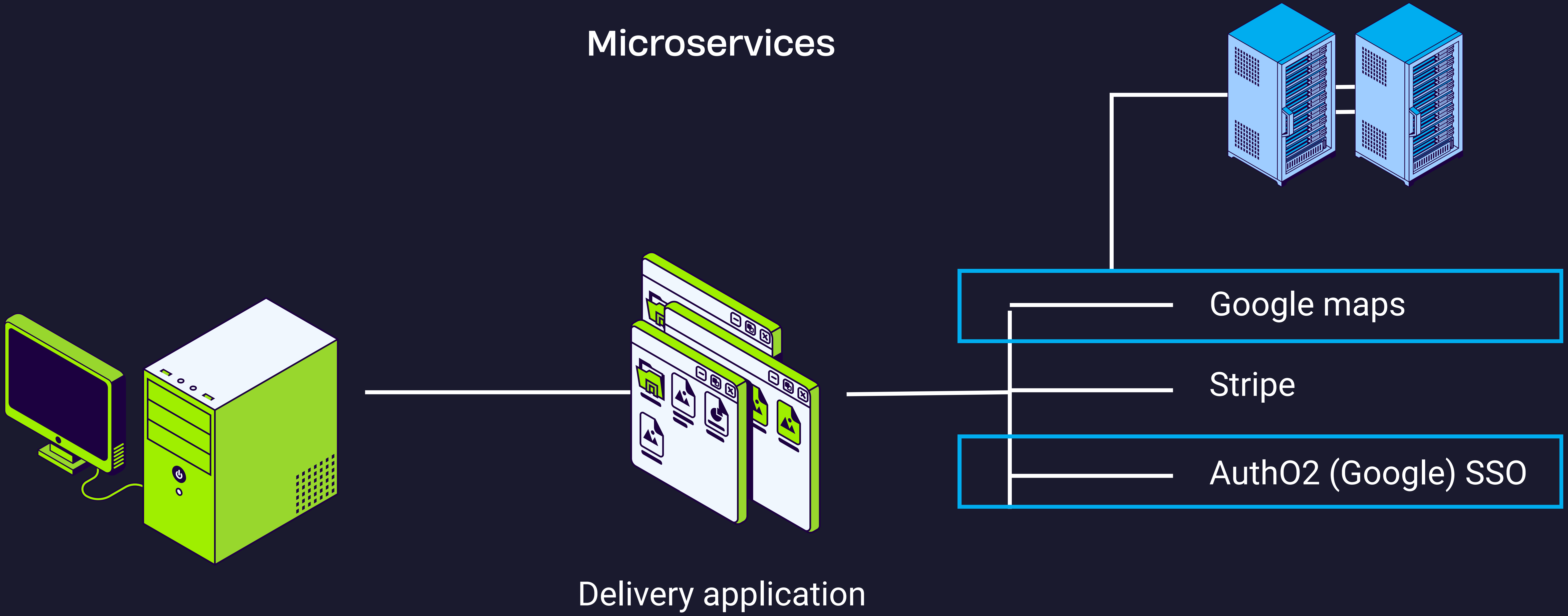
Microservices



tracking application

Reusability

Microservices

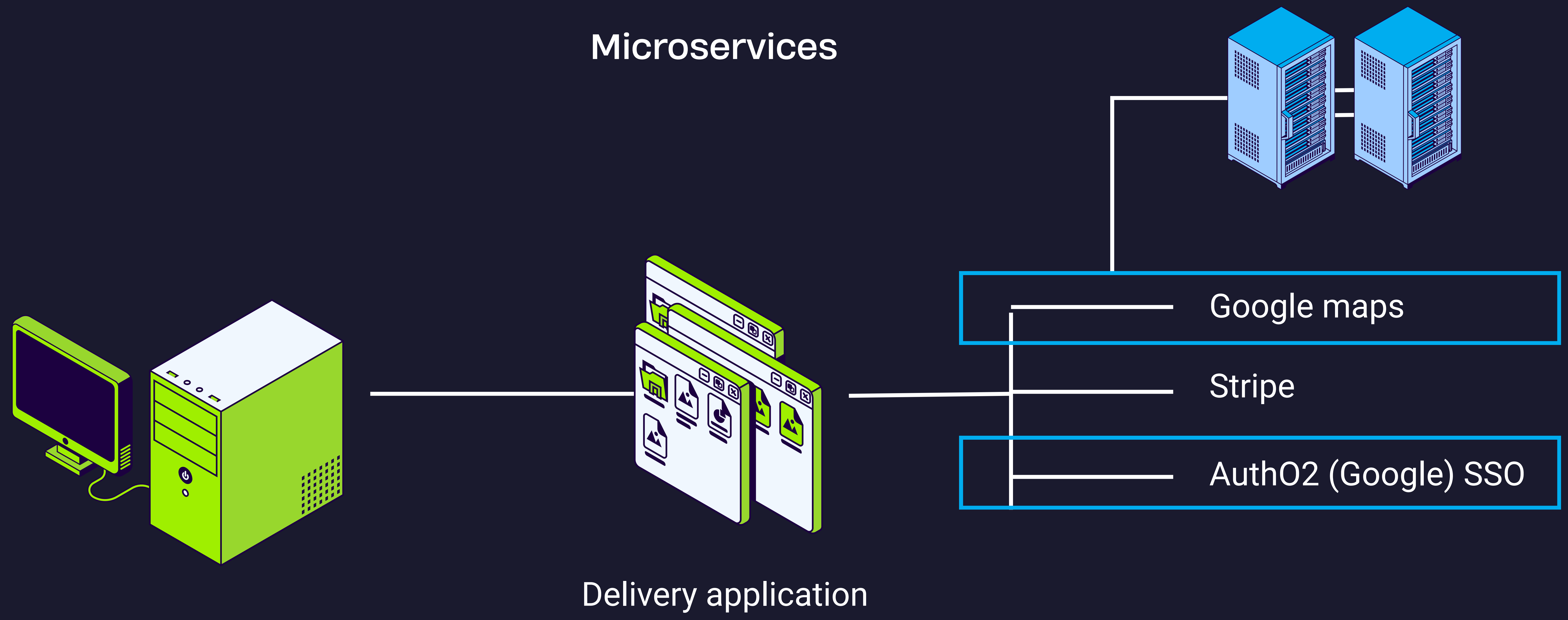


Reusability

Scalability

Resilient

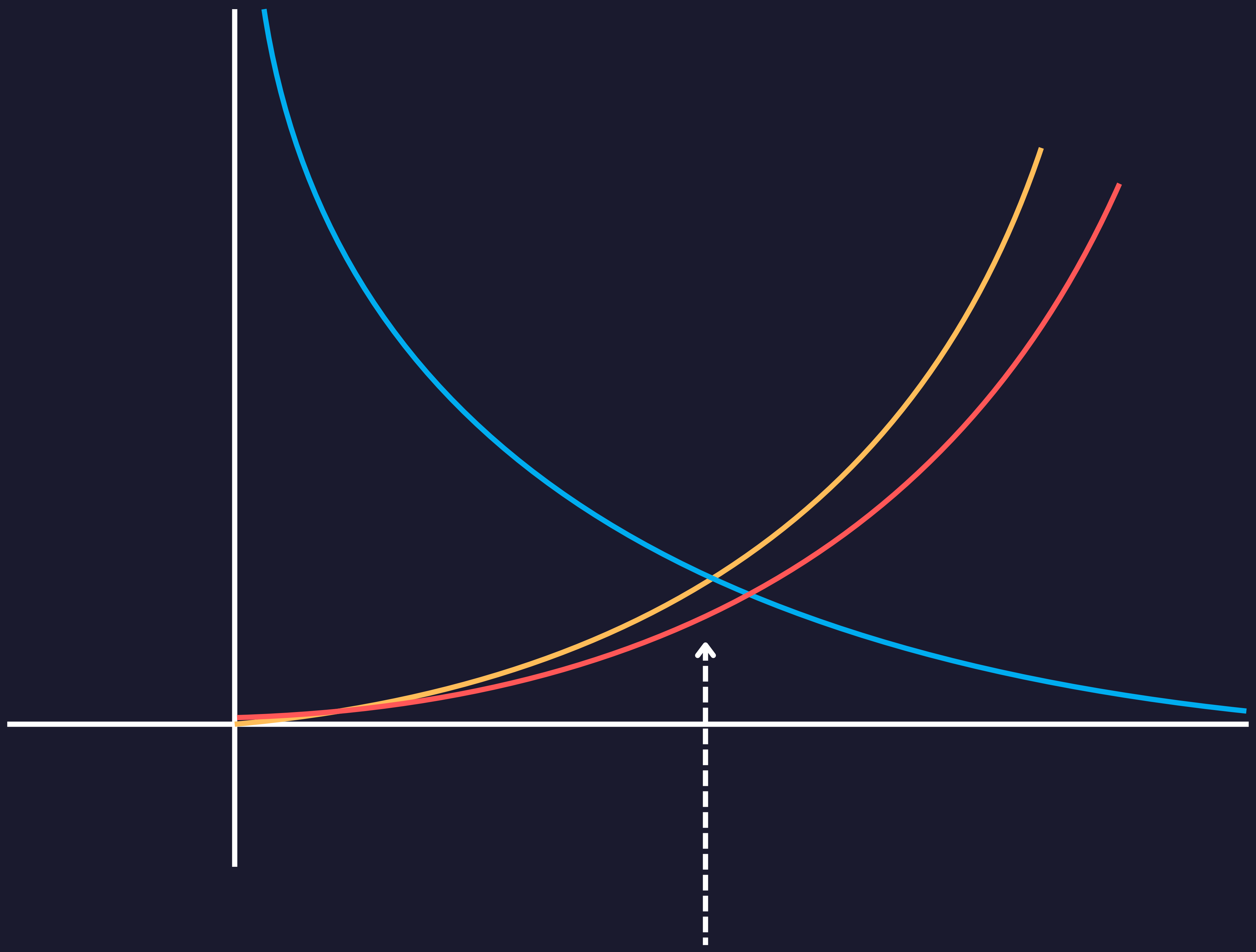
Microservices



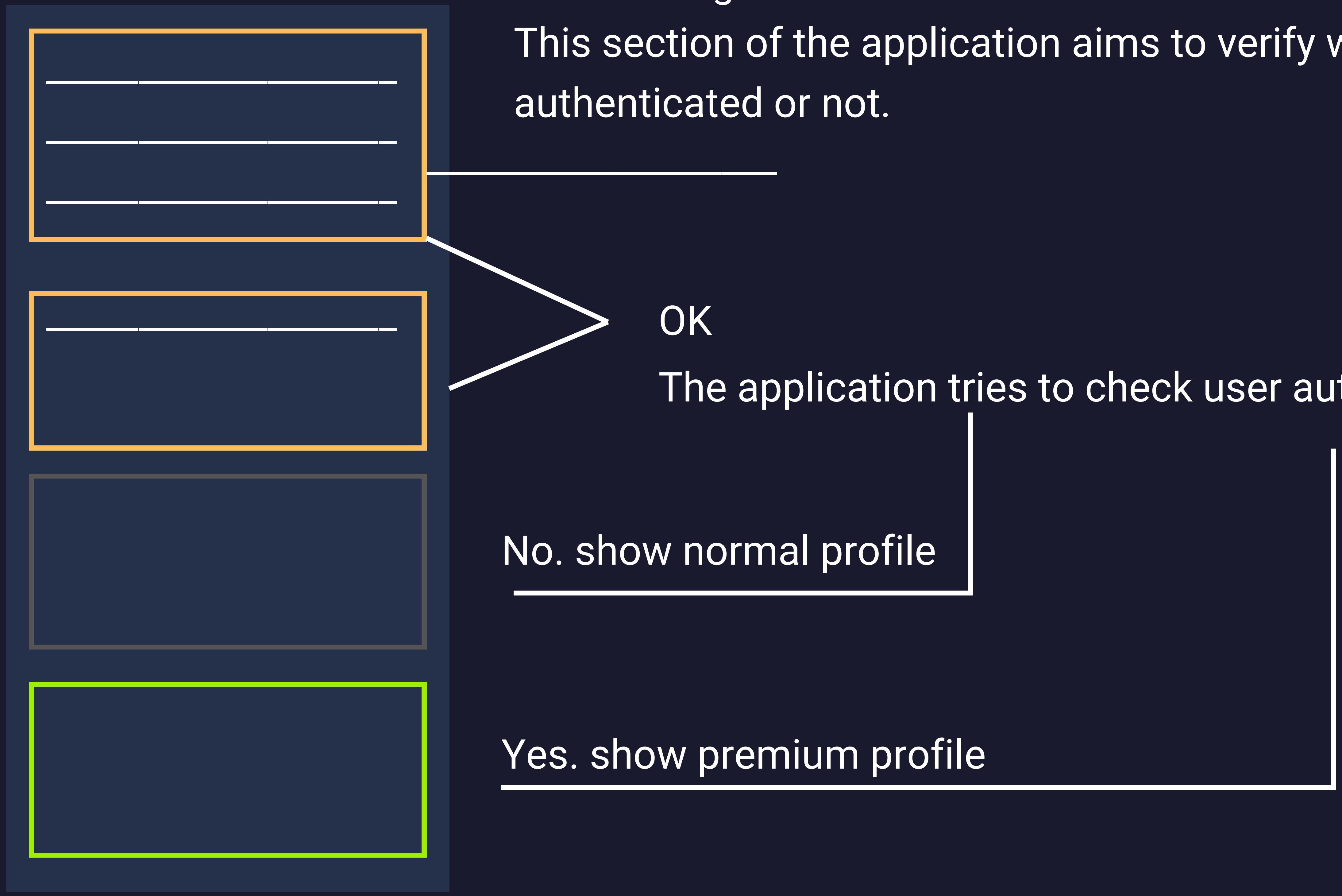
Complexity

Network Overhead

Complexity , security and Threat



Well-designed application



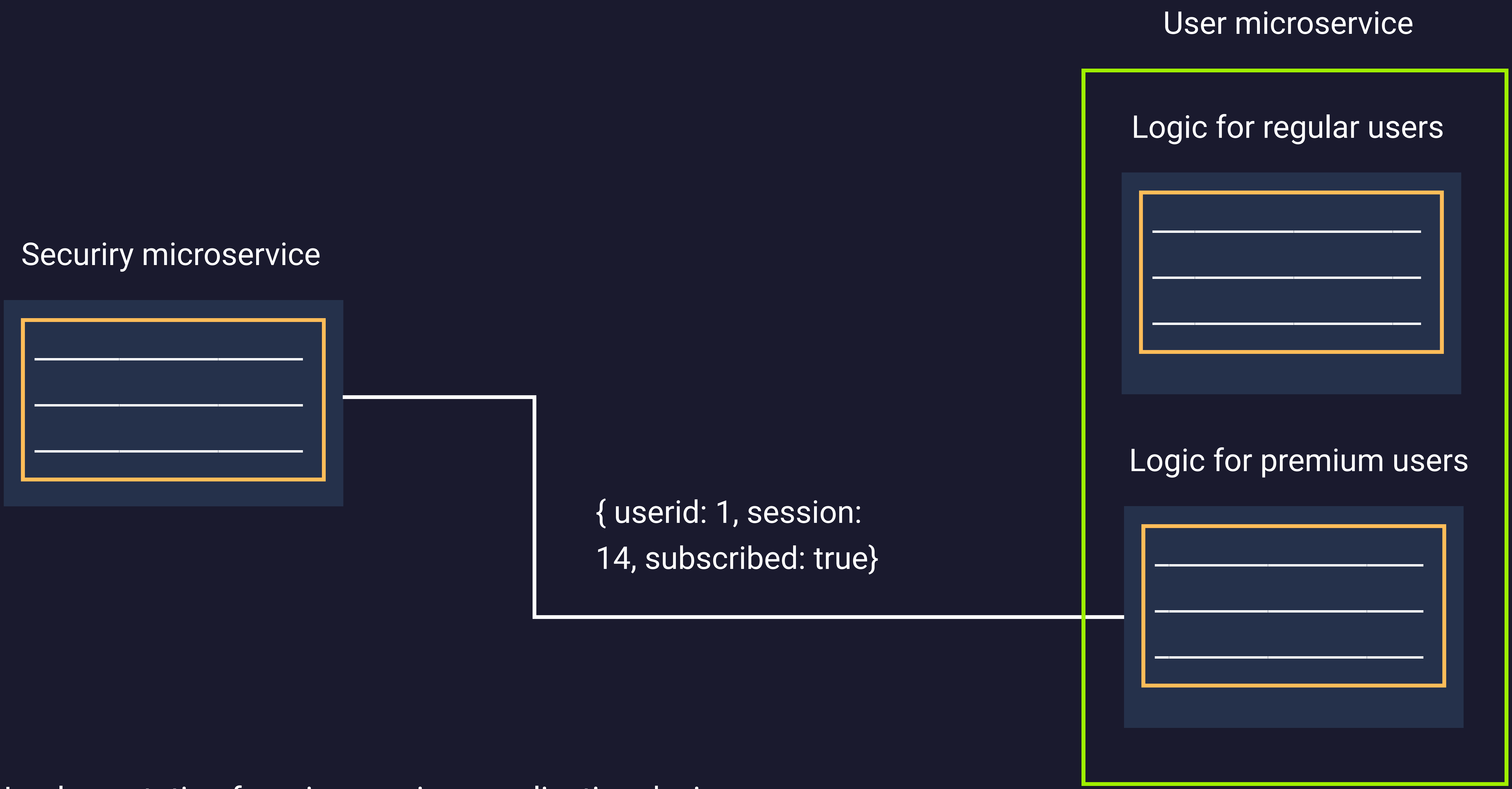
Determining User Authentication Status Within the Same Application
This section of the application aims to verify whether the user is authenticated or not.

OK
The application tries to check user authorization. is it premium or not.

No. show normal profile

Yes. show premium profile

Implementation for monolithic application design. (oldest)



Implementation for microservices application design

Web Services Explained

Web services are crucial **network-accessible** components that use standardized protocols for communication and interoperability.

Interoperability

Web services can operate across various platforms, ensuring seamless integration.

Loose Coupling

Independent components can evolve without affecting the overall system significantly.

Reusability

Existing services can be reused in different applications, promoting efficiency and consistency.

The Role of APIs

APIs enhance software functionality through interoperability and reusability principles.

- Facilitate system integration
- Enable cross-platform communication
- Promote modular software design
- Allow for service reuse
- Support agile development practices