



ChaosConnect



Connect 4, But It's a Chaotic Massive Multiplayer

The background of the slide is a dark purple gradient with a complex network of white lines and dots, resembling a social or data network. The lines connect various nodes, some of which are highlighted with larger, brighter dots. The overall effect is a sense of interconnectedness and complexity.

Distributed Connect4?

How More Than Two Players Can Play Connect Four

The Game



Hey **Marcel!**

Logout

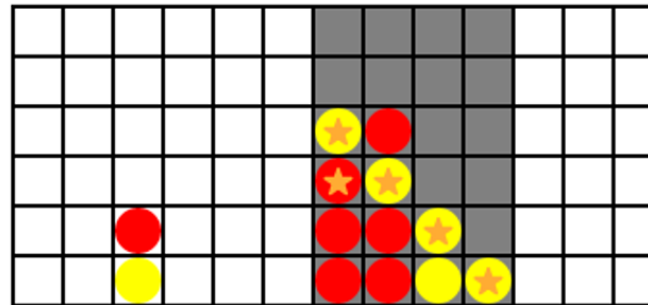
Switch Faction



Marcel with 7 points
Mr. Crowley with 0 points



Chef with 6 points
Superman with 0 points



Screenshots



Choose Your Faction



Play as Guest Login Register

Username

Password

Start 



Hey Marcel! [Go back](#)

How to Play

The general rules are similar to Connect 4, but some changes were made to incorporate the massive multiplayer aspect:

- Clicking on a column will place a piece above it
- After a random delay it will fall down, allowing you to place your next piece
- If four pieces of the same faction are connected horizontally, vertically or diagonally the involved players score points
- Columns automatically get disabled and later cleared once a score was made
- When more players join, the playing field adapts and automatically grows or shrinks in size



Hey Marcel! [Go back](#)

Set New Display Name

New display name



Set New Password

New password

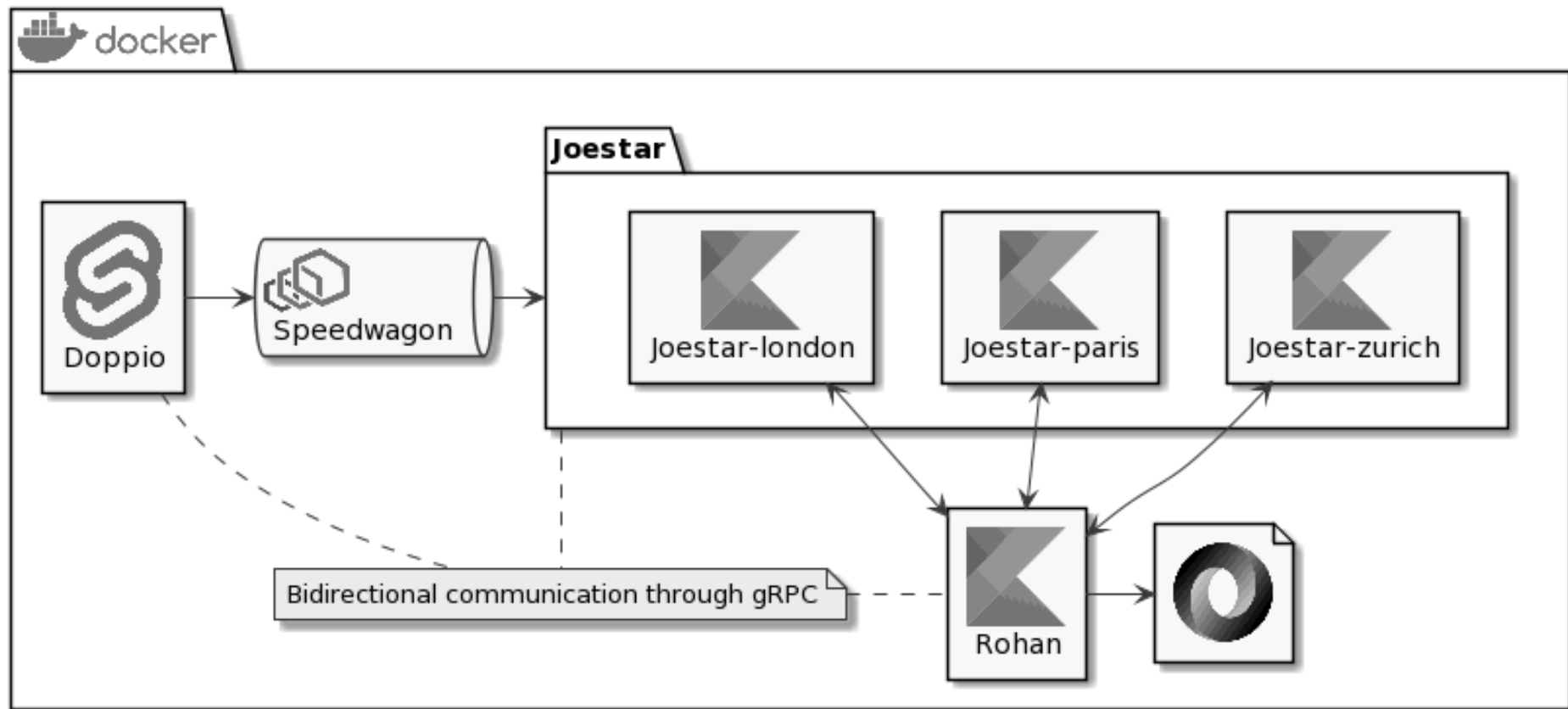


Software Architecture

How It's Built



Big Picture



Components

Name	Role	Description
Doppio	Frontend	Svelte -based web client
Speedwagon	Load balancer	envoy -based load balancing reverse proxy
Joestar	Scaling backend	Micronaut server for user authentication and caching
Rohan	Central backend	Micronaut server for central storage and processing of the game and its users

The service names are all a reference to the popular anime JoJo's Bizarre Adventure.

Why This Architecture?

- Requirement For Horizontal Scaling => Joestar
- Single Source of Truth For Game Update => Rohan
- As Few Services as Possible => No Database
- The Simplest Effective Architecture

Technologies & Frameworks

Using Bleeding Edge In Production

Communication

Requirements

- Strictly Typed
- Real Time Updates
- Web to Server
- Server to Server
- Open Source



- Protobuf
- Streaming
- gRPC-web
- Micronaut gRPC
- Cloud Native

Doppio

- Frontend Needs To Display The Game State
- No Need For Complex Forms Or Logic
- We Decided On Svelte
 - Simple And Modern Web Framework
 - Supports TypeScript
 - Built In State Management



Speedwagon

- Ideally Native Docker Support
- Needs To Support Load Balancing
- Needs To Support gRPC
- We Decided on envoy
 - Support gRPC-web
 - Sample Config Available



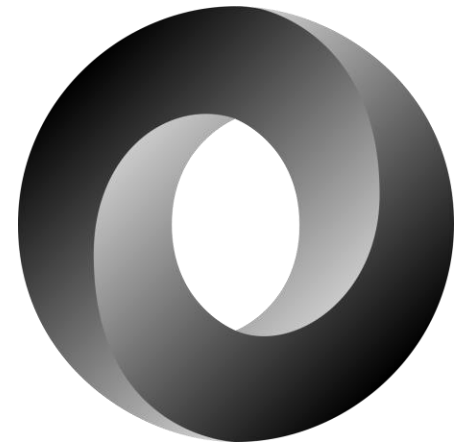
Joestar & Rohan

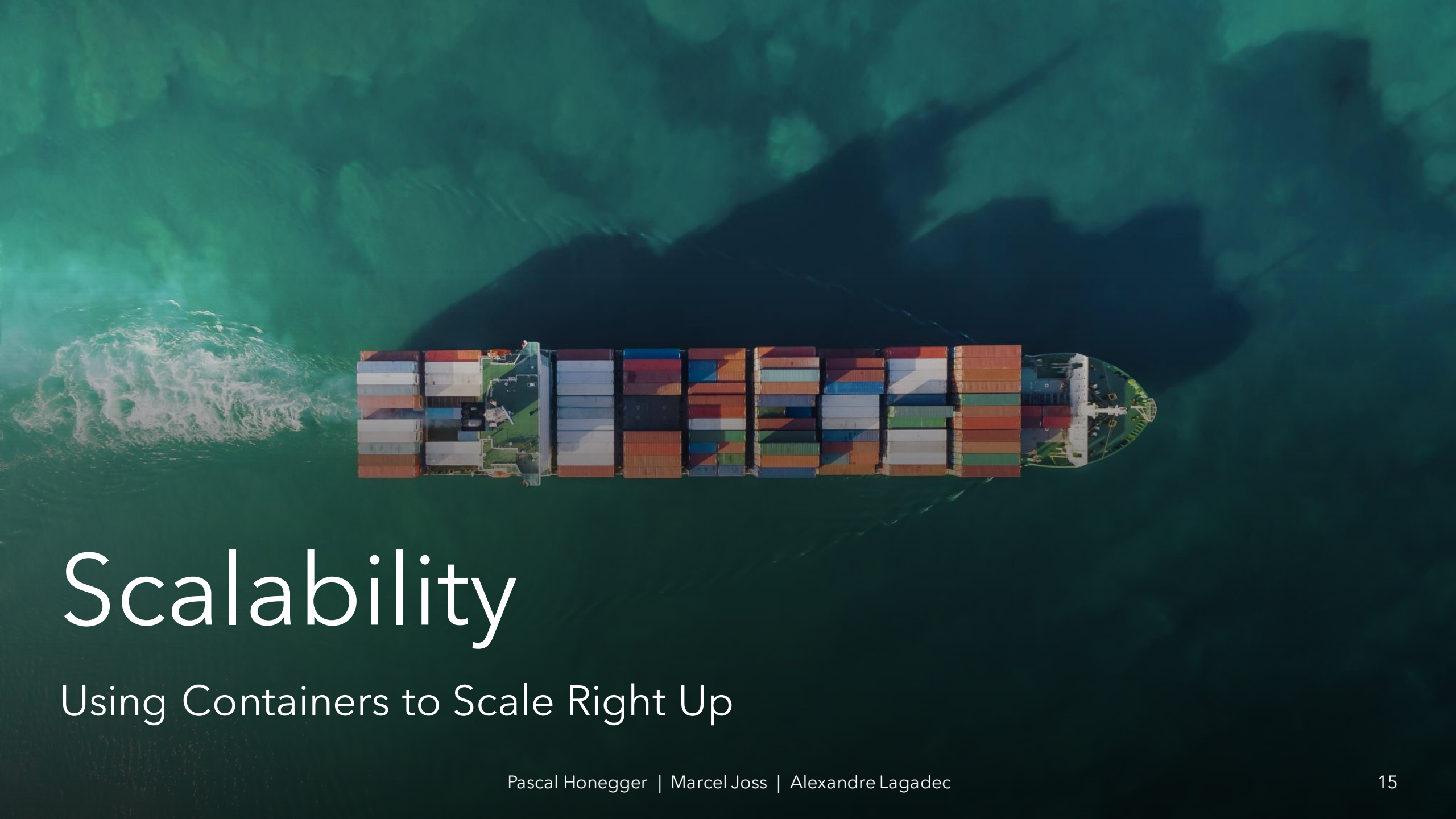
- Needs To Support gRPC Client & Server
- As Little Overhead as Possible
- We Decided on Micronaut & Kotlin
 - Modern Web Framework
 - Supports All Requirements Out of the Box
 - Easy To Use, Test And Deploy



Persistent Storage

- Rohan Holds Users And Their Scores In Memory
- Regularly Written To a JSON Document
- Serialized Using *Kotlinx.Serialization*
 - Very Fast And Extensible
 - Format Could Easily Be Changed If Desired
- Keep It Simple, Stupid





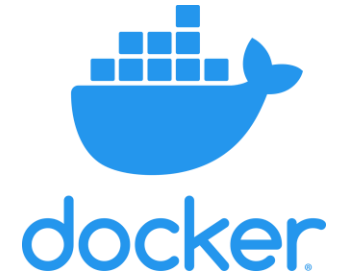
Scalability

Using Containers to Scale Right Up

Speedwagon: **envoy** Reverse Proxy

- Single Entry Point For API Requests
- We Configured **envoy** To Use Round-Robin Load Balancing
- Only Consider Servers Which Fulfill a Readiness Check
 - A Starting Joestar Isn't Ready
 - A Joestar Without a Valid Rohan Connection Isn't Ready

Docker



- Every Service Is Dockerized
- Mostly Used For Production or Code Generation
- All Docker Images Are Optimized For File Size
- Our Images Implement Docker Healthchecks
- Minimal Configuration Required To Start Your Own Instance



Security

Signed JSON Web Tokens Ensure Secure Operation

Password Hashing

- Password Hash Stored In JSON File
- Only For Registered Users
- We Chose Argon2
 - State of The Art - Won 2015 Password Hashing Competition
 - Simple Java API Available
 - Insanely Secure

Symmetric JWT

- We Couldn't Use The Micronaut-Security Package
 - gRPC Support Still Work In Progress
- Official gRPC Authentication Relies on Google As JWT Provider
 - We Didn't Want To Have a Vendor Lock-In
- Simple Do-It-Yourself Solution
 - Joestar Signs And Verifies JWT With Symmetric Secret
 - Doppio Sends Secret In Authorization Header



Hosting

From GitHub To Cloud

Pascal Honegger | Marcel Joss | Alexandre Lagadec

Cloud Provider **Linode**

We Chose [Linode](#) as a Cloud Provider Because They Provide Fair Pricing And a Free Initial Credit

 **RUNNING**

Summary

1 CPU Core	50 GB Storage
2 GB RAM	0 Volumes

IP Addresses

172.104.253.151



2a01:7e01::f03c:92ff:fe01:be6e/128



Plan: Linode 2GB | **Region:** Frankfurt, DE | **Linode ID:** 26692155 | **Created:** 2021-05-07 06:05

Continuous Integration & Delivery

- Every Push Runs GitHub Actions
 - Compile & Test All Components
 - Build Docker Container
- Publish Images on GitHub Release
 - Follows Semantic Versioning (e.g. cc-rohan:1.0.1, cc-doppio:1)
 - Publicly Available
- Latest Version Automatically Installed on Linode

Let's Play!

<https://chaos.honegger.dev/>



Questions?

For Details See <https://github.com/PascalHonegger/ChaosConnect>