Building and Testing Elixir Containers with GitHub Actions

Denver Elixir Meetup
January 8, 2024

Jake Morrison jake@cogini.com

Cogini

Agenda

- Motivation
- Approach
- Tools
- Example

Cogini

Motivation

"Breaking up the monolith" on a large e-commerce site (Phoenix, RoR, Absinthe)

- Improve reliability by decoupling public services from internal processing
  - Create "services" such as authentication, user profiles, products, cart
  - GraphQL API -> federated GraphQL
- Improve development speed
  - Reduce the risk associated with change
  - Make independent components that can be developed more quickly
- Improve security
  - Compartmentalize access, make audit easier
  - Separate user roles by service
  - GraphQL security

Effective microservices

- Automated testing
    - Ensure services can be updated without breaking clients, supporting incremental deployment
- Fast, easy, and reliable deployment
- Development and QA processes for multiple components
- Observability to identify and debug problems

Cogini

Details

- Test against the deployed OS image: public AP, library/OS updates, support security updates
- Improve speed for better developer experience and efficiency
- Add code quality tools and security scans
- Integrate test results with PR UI, providing actionable feedback
- Support dev and QA for multiple services in local or review environments, plus security workflows

Cogini

Challenges

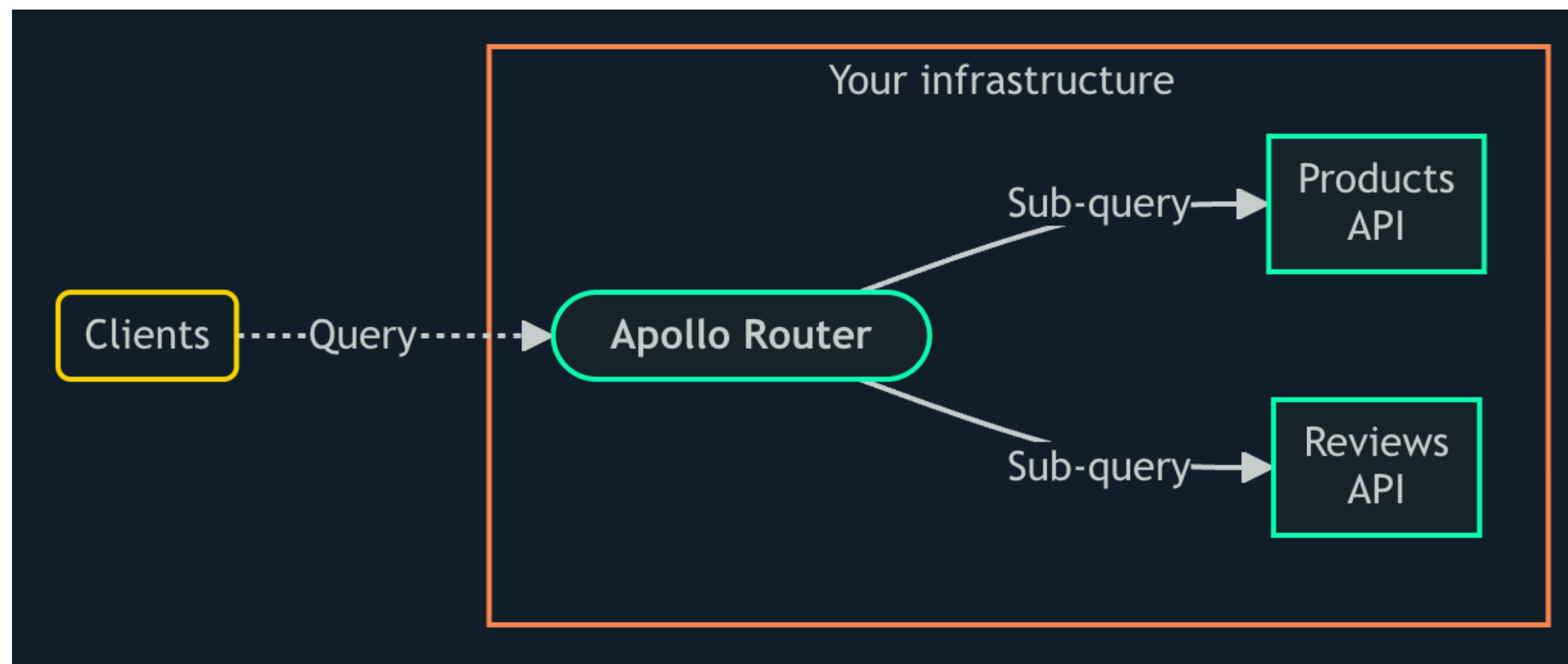
- Splitting up complex GraphQL schemas and untangling application dependencies
- Supporting schema evolution
- Configuring applications, particularly secret management
- Managing code ownership

Cogini

Examples

- phoenix_container_example: CI/CD system based on containerized build and test running in GitHub Actions, deploying to AWS ECS using Terraform

- absinthe_federation_example: federated GraphQL with Apollo Router



Cogini

Testing is key

Testing is the most important part of microservices, particularly when the monolith has become big enough that it's causing problems.

- Not breaking production
- Improving quality
- Supporting change

Cogini

Need for speed

- Slow tests waste developer time and cause task-switching
- Slow tests cause production outages
- Fail fast, giving feedback as early as possible

Cogini

Hierarchy of tests

1. Unit tests with synthetic data
2. Unit tests with data from the db
3. Unit tests for external services with mocks
4. Unit tests for external services in a test environment
5. External tests with data from the db or external service in a test environment
6. External tests that combine data from multiple services, i.e., GraphQL
7. Deeper checks, like load tests, property tests, or security scans
8. Health checks

Plus

- Observability to identify behavior in production and debug
- Feature flags and associated testing
- Everyone tests in production, but only some people admit it

Cogini

Test tools

- Unit tests: Eunit, run in parallel
- Mocks: Bypass vs HTTPoison vs Tesla
- Quality checks: Credo, Dialyzer, code coverage, etc.
- Styler advanced formatter
- Security scanners: mix.audit, hex.audit, Sobelow, Trivy, Grype, GitHub Advanced Security
- Make mocks real
- Postman/Newman or Insomnia

Cogini

Visibility

- Editor integration
- Hound PR integration
- Test summaries, avoid the great wall of text
- Test observability, finding slow and flaky tests

Challenges

- Configuration
- Managing secrets
- Running prod containers in test environment

Cogini

Better containers

- Caching, caching, caching
- Erlang releases
- Configuration

Cogini

Containers

- Building and testing in containers
  - Ease of setup and development
  - Repeatability
  - Easier CI development
- Containerized testing, test containers
  - Run back-end services in containers: db, Redis, Kafka
- Testing app containers
  - Smoke tests
  - API tests
  - Coordinated testing of multiple microservices

GitHub Actions

Summary

Jobs

- Build test image (1.15.7, 26.1.2, ...
- Build prod image (1.15.7, 26.1.2, ...
- Build prod image (1.15.7, 26.1.2, ...
- Build otel image
- Build newman image
- Run tests (1.15.7, 26.1.2, bullsey...
- Run dialyzer (1.15.7, 26.1.2, bulls...
- Security scan code
- Security scan prod image (1.15.7...
- Security scan prod image (1.15.7...
- Run external API tests (1.15.7, 2...
- Run external API tests (1.15.7, 2...
- Deploy to AWS ECS
- Tag prod images as latest
- Test Results

Run details

- Usage
- Workflow file

Re-run triggered 1 minute ago — reachfh 1f142e0 main

Status: Success

Total duration: 1m 46s

Artifacts: –

ci.yml

on: push

Matrix: Build test image — 1 job completed — Show all jobs

Matrix: Build prod image — 2 jobs completed — Show all jobs

Build newman image 11s

Build otel image 12s

Matrix: Run dialyzer — 1 job completed — Show all jobs

Matrix: Run tests — 1 job completed — Show all jobs

Security scan code 37s

Matrix: Security scan prod ima... — 2 jobs completed — Show all jobs

Matrix: Run external API tests — 2 jobs completed — Show all jobs

Deploy to AWS ECS 7s

Tag prod images as latest 8s

Cogini

CI features

- Separate, parallel build of test and prod containers
- Multiple versions of prod containers, e.g., OS version, debug
- Quality checks, Dialyzer, security checks
- External API tests
- GitHub integration with Hound, test results, GitHub Advanced Security
- Pushing containers to GitHub GHCR and AWS ECR
- Deploying to AWS ECS, assets to CDN

Cogini

App features

- OpenTelemetry to AWS X-Ray
- Structured logging with JSON, Uinta
- Production debugging with Observer CLI, Recon, AWS container console
- Clustering, service discovery

Cogini

Dev and QA

- Local dev vs containerized dev
- `docker compose` vs local Kubernetes
- Releases in GHCR
- Review environments
- Development in the cloud
- Kubernetes all the things

Questions?

- Into the code

Cogini

AWS architecture

- VPC with public and private subnets
- Load balancer
- ECS with public web, API, and worker services (or EC2 in Auto Scaling Group)
- RDS database
- CodeDeploy for Blue/Green deployment
- Service discovery for clustering
- Amazon Certificate Manager for certs, Route53 for DNS
- CloudFront CDN
- CloudWatch Logs and X-Ray for observability
- Access to GitHub Ci/CD using OpenID Connect

Cogini