Phocas.

DEVOPS-101

Code. Automate. Deploy. Repeat.

Jared Rowe

Senior DevOps Engineer

Jeremy Bertenshaw

DevOps Engineer

Gift Mkwara

Graduate Engineer

Phocas.

Roadmap

In this session, we're going to cover:

1. Who We Are
2. DevOps Fundamentals
3. Take Home Example Demonstration
4. Q & A

1

Our Journeys into DevOps

Phocas.

Phocas.

Jared Rowe

Senior DevOps Engineer

Phocas.

Graduated with a BSc in Computer Science from UC.

Worked in a variety of roles in Operations and Systems Engineering.

Switched to Development.

Pivoted into Application Performance Monitoring, and then Solutions Architecture.

Currently at Phocas and fully immersed in a dedicated DevOps role.

Jeremy Bertenshaw

DevOps Engineer

Phocas.

Also graduated with a BSc in Computer Science from UC.

From personal projects, deployment was clicking "Go" on Vercel.

Joined Phocas and got introduced into the world of production grade infrastructure and CI/CD.

Tried to mimic this with my personal projects, "Why is this so confusing!"

Gift Mkwara

Graduate Engineer

2

DevOps Fundamentals

Phocas.

What even is DevOps?!

Phocas.

Core Concepts

Phocas.

CI/CD

Infrastructure as Code

Containerisation

Observability

Phocas.

Continuous Integration (CI)

Phocas.

What is CI?

Is the practice of merging code **frequently** into a central repository with confidence.

An **automated** process ensures that the code is:

- Compiled 🛠️
- Tested ✅
- High quality 🏆
- Style conformant 🎉

Does it build?

Phocas.

- ✓ Is the language's compiler happy with the source code?
- ✓ For compiled languages run the compiler against the source code.
- ✓ For interpreted languages run tools like type checkers and linters

Do the tests pass?

Phocas.

- ✓ Run automated tests, *ideally* at all levels
  - Unit
  - Integration
  - End-to-end
- ✓ Report on test coverage.
- ✓ Make failing tests easy to find and diagnose.

Does it meet our standards?

Phocas.

- ✓ Formatting & Style
- ✓ Linting, Code Smells, Best Practices
- ✓ Static Analysis & Security Checks

Phocas.

Continuous Delivery (CD)

Phocas.

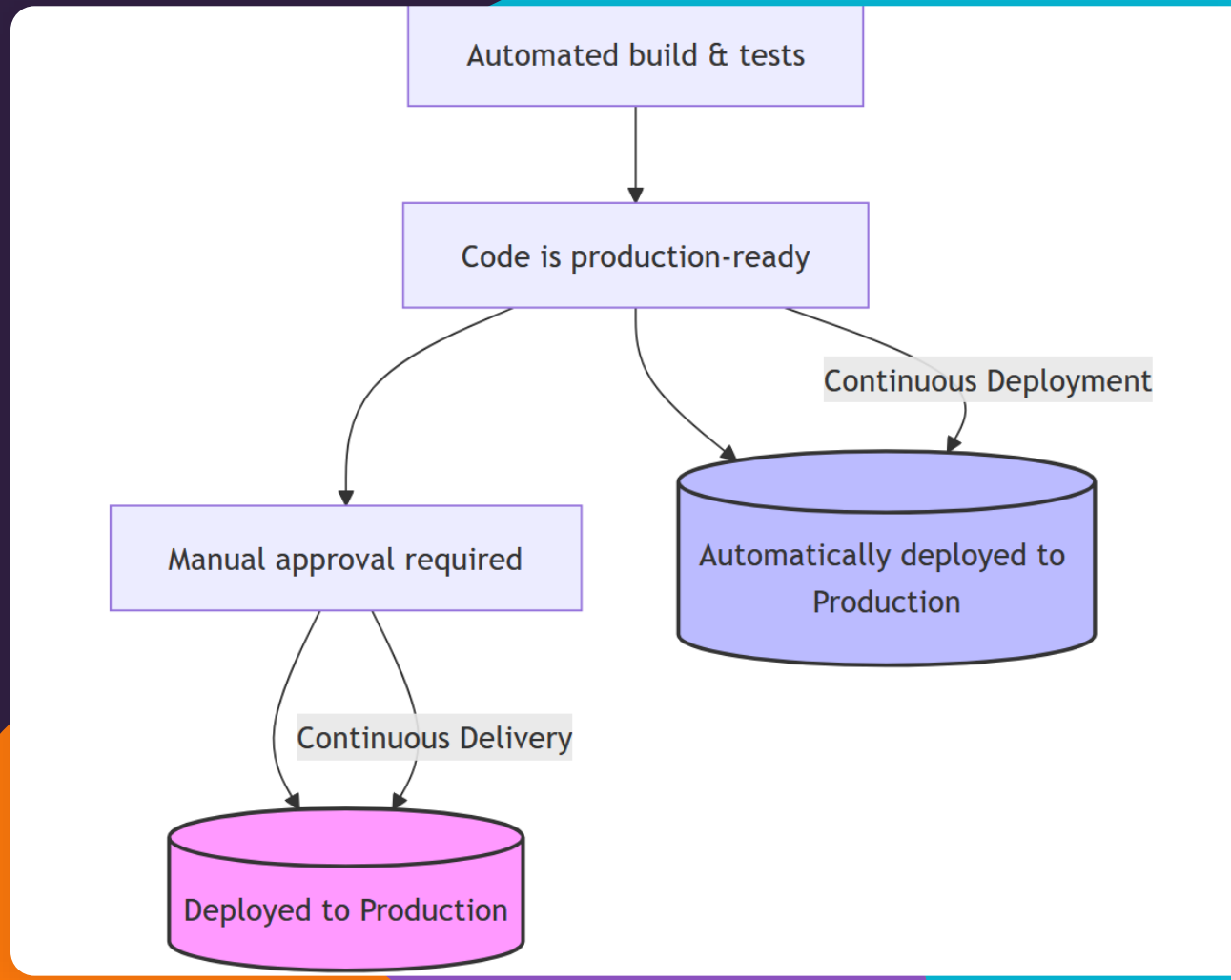
What is CD?

Delivery

All code that is merged into the main branch is ready to be deployed.

Deployment

All code that is merged into the main branch is deployed.



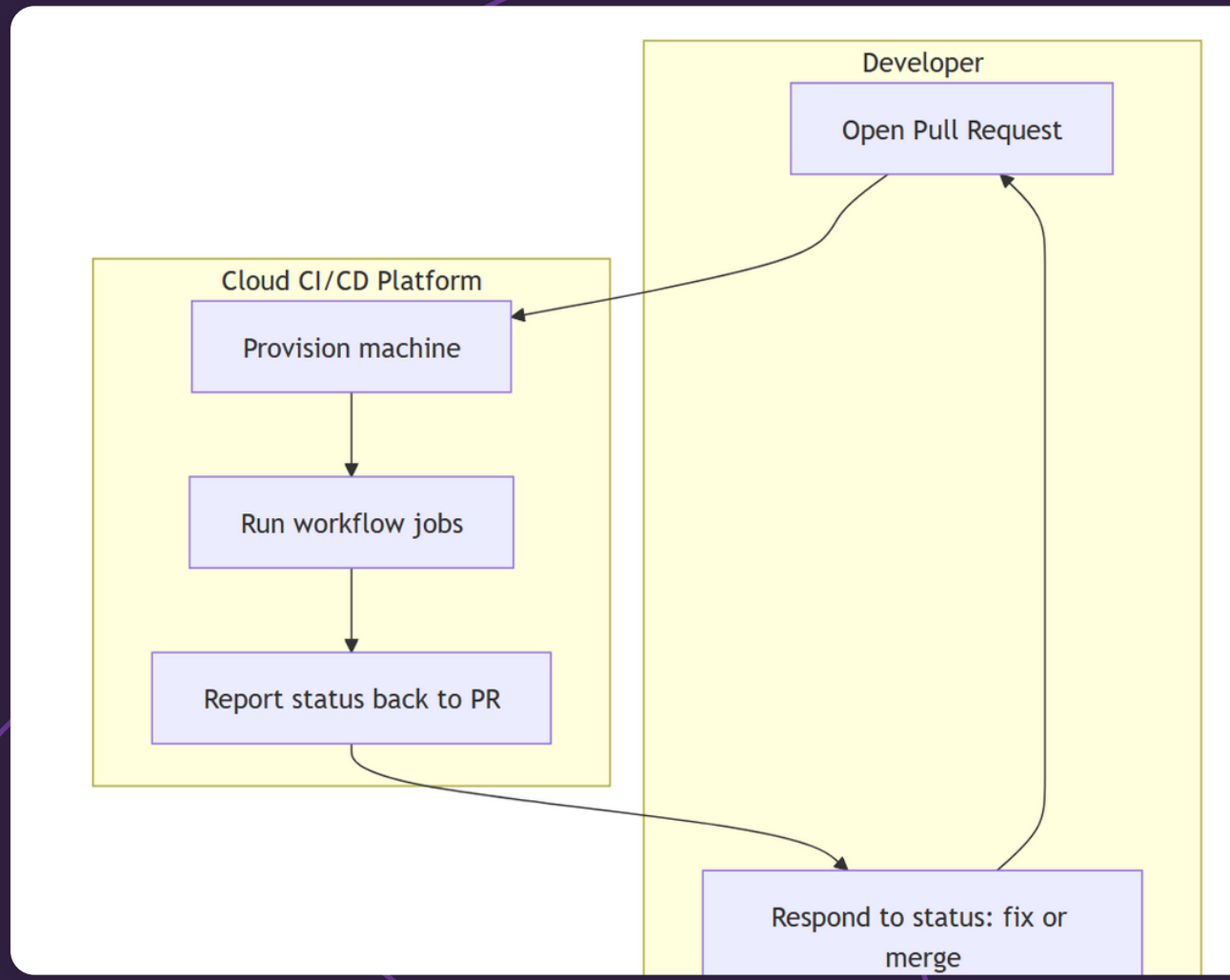
Phocas.

CI/CD Tooling & Platforms

Phocas.

Cloud-based CI/CD platforms

- GitHub Actions
- GitLab CI/CD
- Jenkins
- JetBrains TeamCity
- CircleCI
- TravisCI
- Octopus Deploy



Phocas.

Deployment Environments

Multi-Environment Deployments

Phocas.

Development

- A first pass environment for Engineers to test and validate new changes

Staging/Pre-production/Test

- A production-like environment where internal users and testers can validate changes

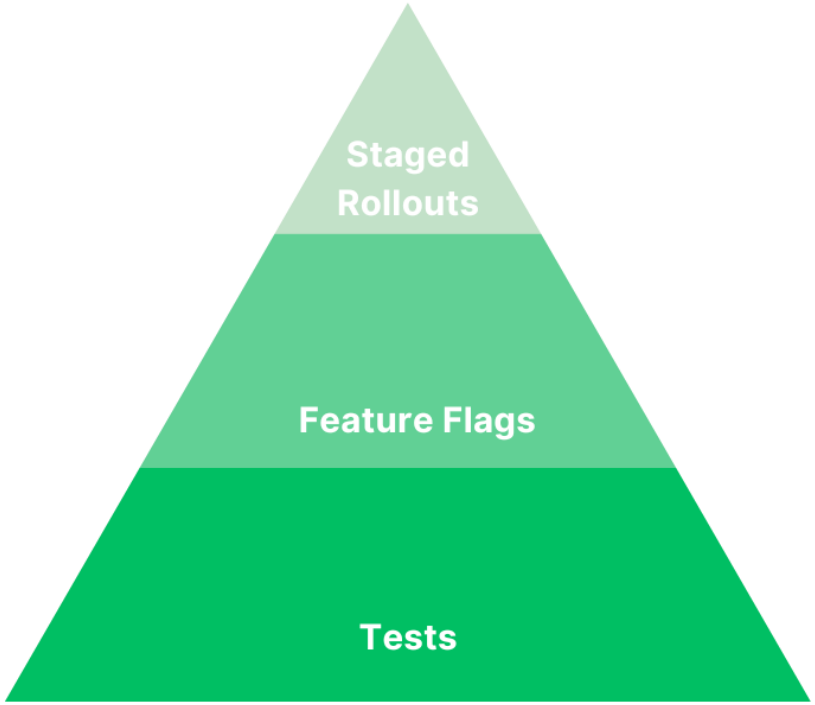
Production

- Where we make our money! 💸 (deliver value to customers)

Phocas.

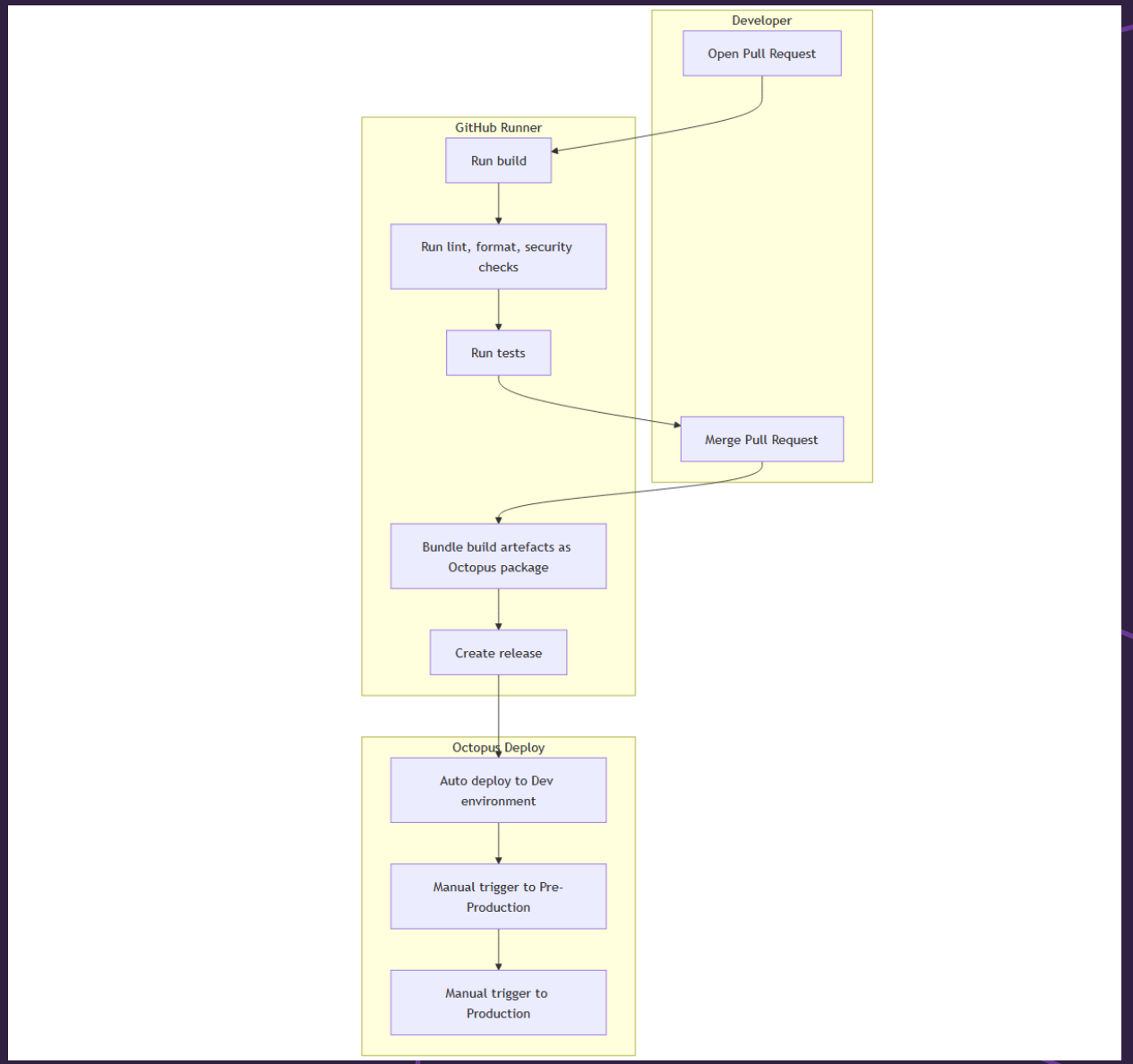
Being Extra Cautious

- ✓ For continuous delivery to work you need a good foundation of tests
- ✓ Feature flags allow us to toggle features at **runtime**
- ✓ Staged/Canary rollouts allow us to release to a **subset** of our users



CI/CD at Phocas

Phocas.



CI/CD Quiz - 1

Phocas.

Why would a company choose continuous delivery over deployment?

A Continuous deployment only works if you're using a cloud-based CI/CD platform

B Continuous delivery ensures bugs will never reach production, while continuous deployment cannot

C Continuous delivery allows manual approval before production, giving more control over releases

D Continuous delivery is only possible if the company hosts its own servers rather than using the cloud

CI/CD Quiz - 1

Phocas.

Why would a company choose continuous delivery over deployment?

A Continuous deployment only works if you're using a cloud-based CI/CD platform

B Continuous delivery ensures bugs will never reach production, while continuous deployment cannot

C Continuous delivery allows manual approval before production, giving more control over releases

D Continuous delivery is only possible if the company hosts its own servers rather than using the cloud

CI/CD Quiz - 2

Phocas.

Which factor has the greatest impact on the success of your CI/CD pipelines

A Picking the right cloud platform

B Running CI/CD jobs in parallel whenever possible

C Using a compiled programming language over an interpreted one

D Having reliable and comprehensive tests

CI/CD Quiz - 2

Phocas.

Which factor has the greatest impact on the success of your CI/CD pipelines

A Picking the right cloud platform

B Running CI/CD jobs in parallel whenever possible

C Using a compiled programming language over an interpreted one

D Having reliable and comprehensive tests

Phocas.

Infrastructure as Code (IaC)

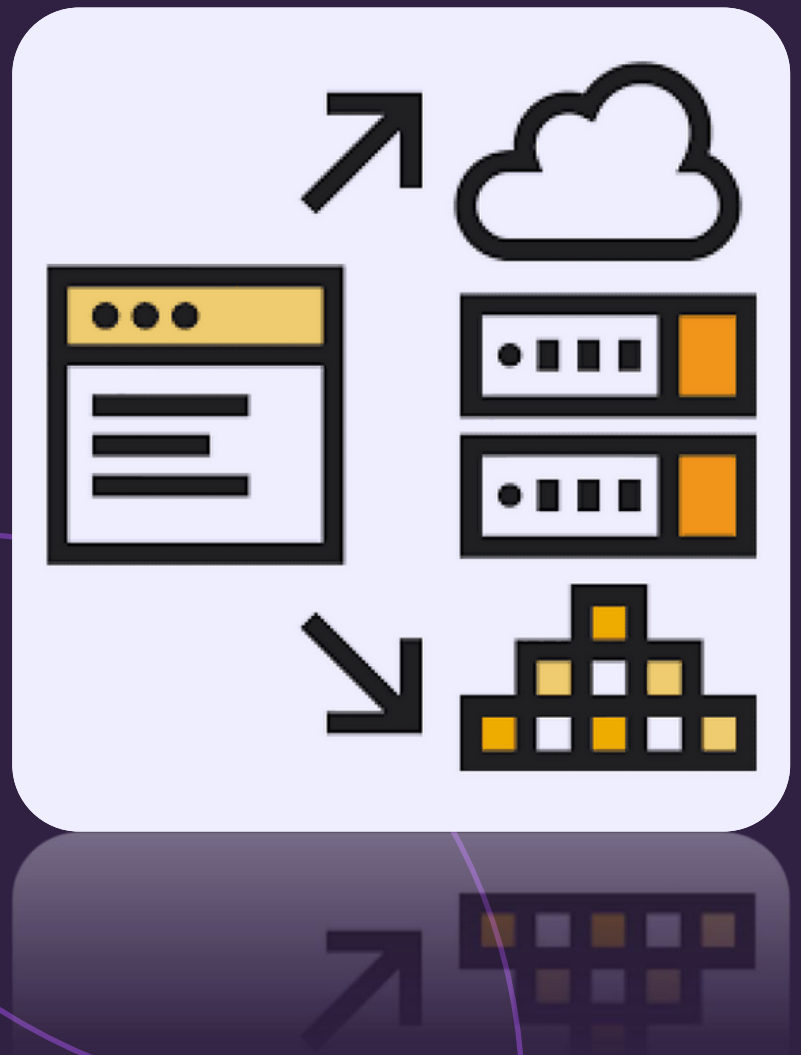
Phocas.

What is IaC?

- IaC is the practice of managing infrastructure (servers, networks, cloud resources) using machine-readable configuration files.

- It treats infrastructure the same way we treat application code — versioned, tested, and deployed automatically.

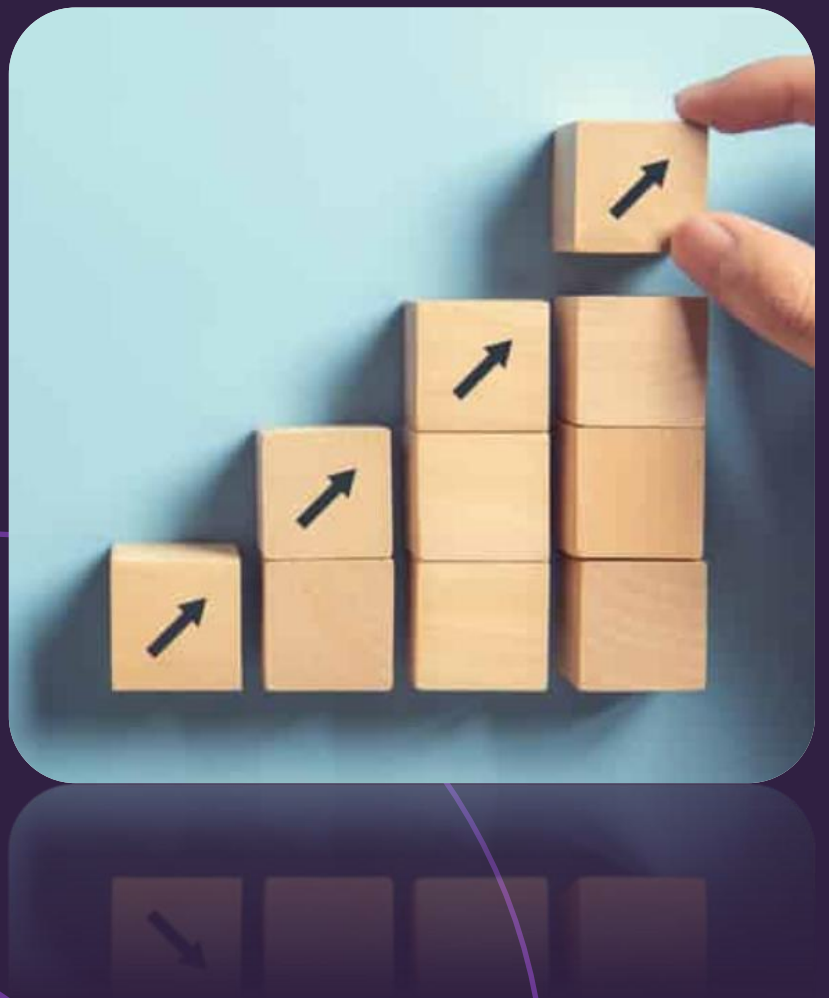
“Think of it as turning a manual setup into repeatable scripts.”



Phocas.

Why It Matters

- **Consistency -** No more “it works on my machine” — environments are identical across dev, test, and prod.
- **Speed -** Infrastructure can be spun up or torn down in minutes.
- **Collaboration -** Dev and Ops teams work from the same source of truth.
- **Scalability -** Easily replicate environments for testing, scaling, or disaster recovery.

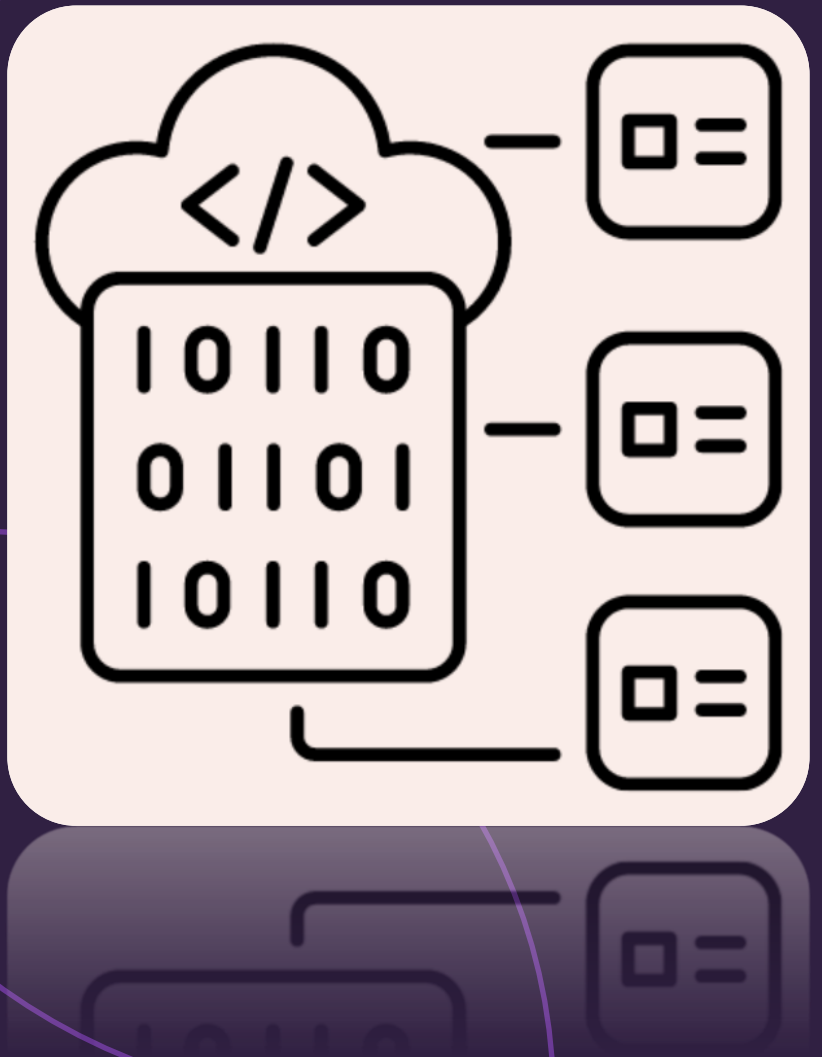


Phocas.

Popular Tools

- **Terraform -** Declarative, cloud-agnostic, widely adopted.
- **Pulumi -** Uses real programming languages (TypeScript, Python, etc.).
- **Ansible -** Great for configuration management and provisioning.
- **CloudFormation -** AWS-native IaC tool for managing resources.

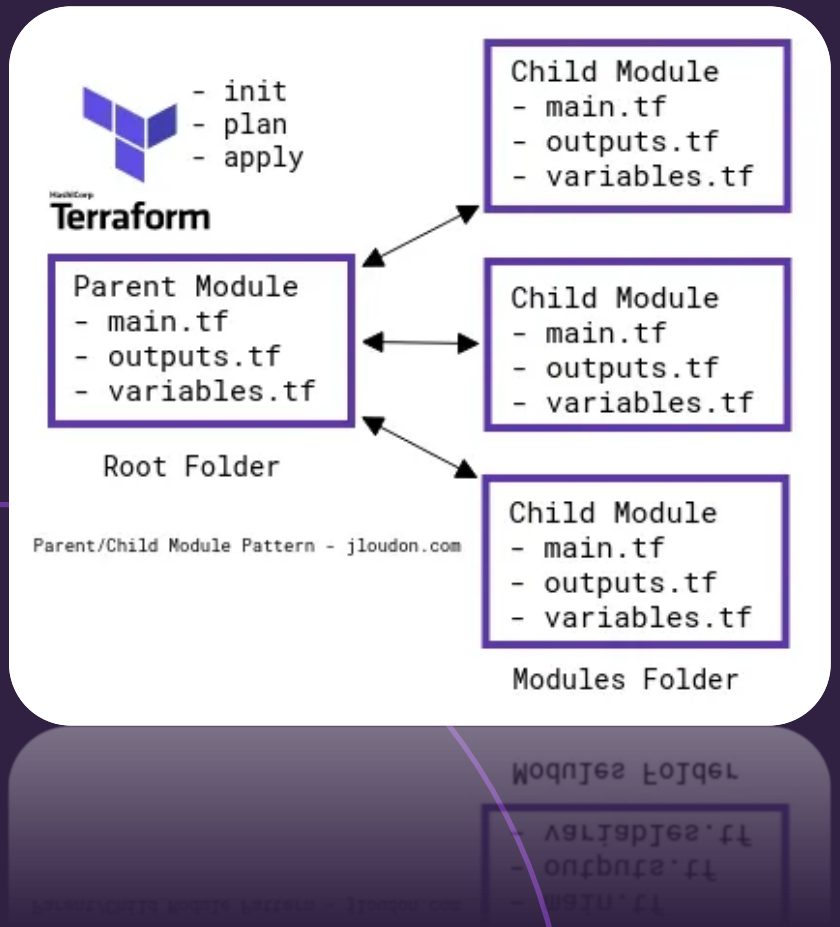
“Each tool has its strengths — choose based on your stack and workflow.”



Phocas.

Best Practices

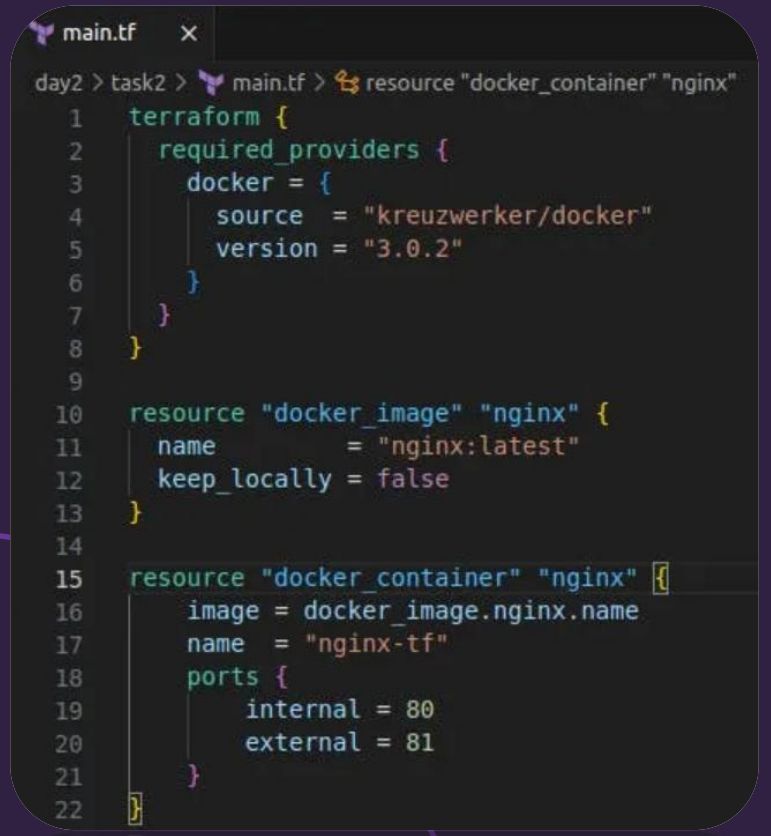
- **Version Control Everything** - Store IaC files in Git for traceability and rollback.
- **Modular Design** - Break infrastructure into reusable components (e.g., VPC modules).
- **Automated Validation** - Use tools like terraform validate or ansible-lint in CI pipelines.
- **Pipeline-Driven Deployment** - Apply changes via CI/CD, not manually.
- **Tag Resources** - Add metadata for cost tracking, ownership, and auditing.



Phocas.

What Does IaC Look Like At Phocas?

- **GitHub** – Source Control, Code Validation & Triggering Deployment.
- **Terraform** – Provisions infrastructure and tracks changes over time.
- **Terragrunt** – Wraps Terraform, simplifying complex Terraform configurations, reduces duplication across environments, and makes multi-environment setups (dev, staging, prod) easier to manage.
- **Atlantis** – Enables Terraform automation via pull requests — bringing Terraform into our CI/CD pipeline.



IaC Quiz - 1

Phocas.

What is the main benefit of using Infrastructure as Code?

A It eliminates the need for cloud providers

B It enables consistent, repeatable infrastructure deployments

C It allows infrastructure to be managed manually

D It improves application performance

IaC Quiz - 1

Phocas.

What is the main benefit of using Infrastructure as Code?

A It eliminates the need for cloud providers

B It enables consistent, repeatable infrastructure deployments

C It allows infrastructure to be managed manually

D It improves application performance

IaC Quiz - 2

Phocas.

Which of the following best describes a key difference between Terraform and AWS CloudFormation?

A Terraform is limited to AWS, while CloudFormation supports multiple cloud providers

B CloudFormation uses a declarative language, while Terraform uses imperative scripts

C Terraform supports multiple cloud providers, while CloudFormation is AWS-specific

D CloudFormation stores state locally, while Terraform stores state in AWS S3 by default

IaC Quiz - 2

Phocas.

Which of the following best describes a key difference between Terraform and AWS CloudFormation?

A Terraform is limited to AWS, while CloudFormation supports multiple cloud providers

B CloudFormation uses a declarative language, while Terraform uses imperative scripts

C Terraform supports multiple cloud providers, while CloudFormation is AWS-specific

D CloudFormation stores state locally, while Terraform stores state in AWS S3 by default

Phocas.

Containerisation

Phocas.

What is Containerisation?

- Containerisation is a method of packaging an application along with its dependencies, libraries, and configuration into a single unit.
- Containers run consistently across environments—whether it’s on your laptop, a test server, or production in the cloud.
- While there are several container runtimes available today, Docker has become the de facto standard.

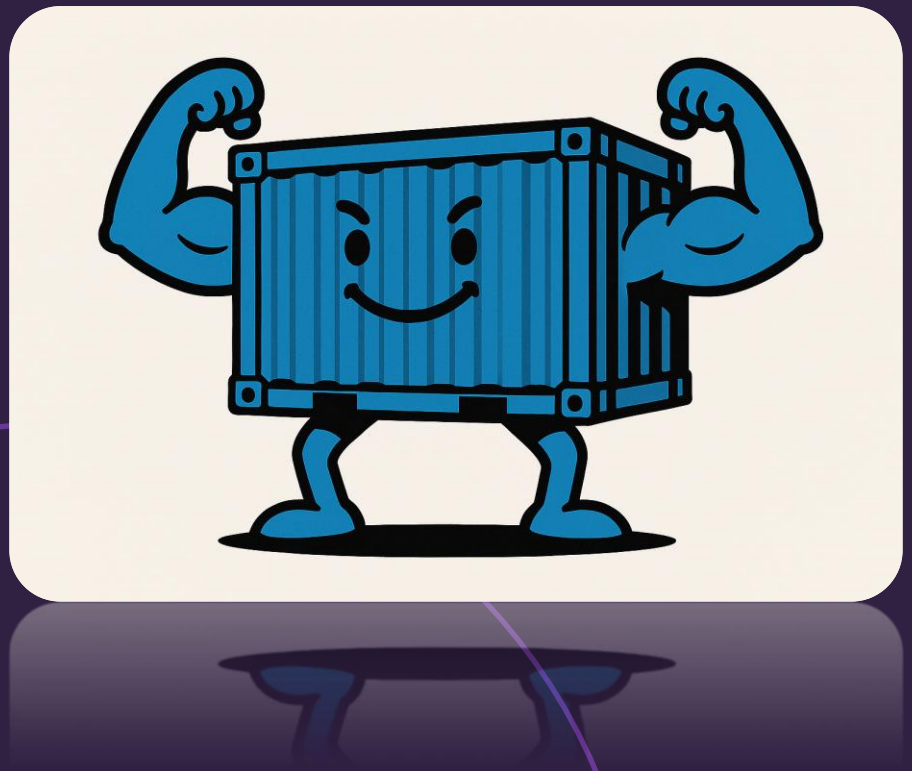
"Think of a container like a shipping container—it doesn't matter what's inside, it can be moved and deployed anywhere reliably."

Phocas.

Why Use Containers?

- **Portability**: Containers run the same regardless of the host system.
- **Speed**: Faster startup and deployment compared to traditional VMs.
- **Efficiency**: Lightweight and share the host OS kernel, reducing overhead.
- **Isolation**: Each container runs independently, reducing risk of conflicts.

"A team can develop a Node.js app locally, test it in staging, and deploy it to production without worrying about environment differences."

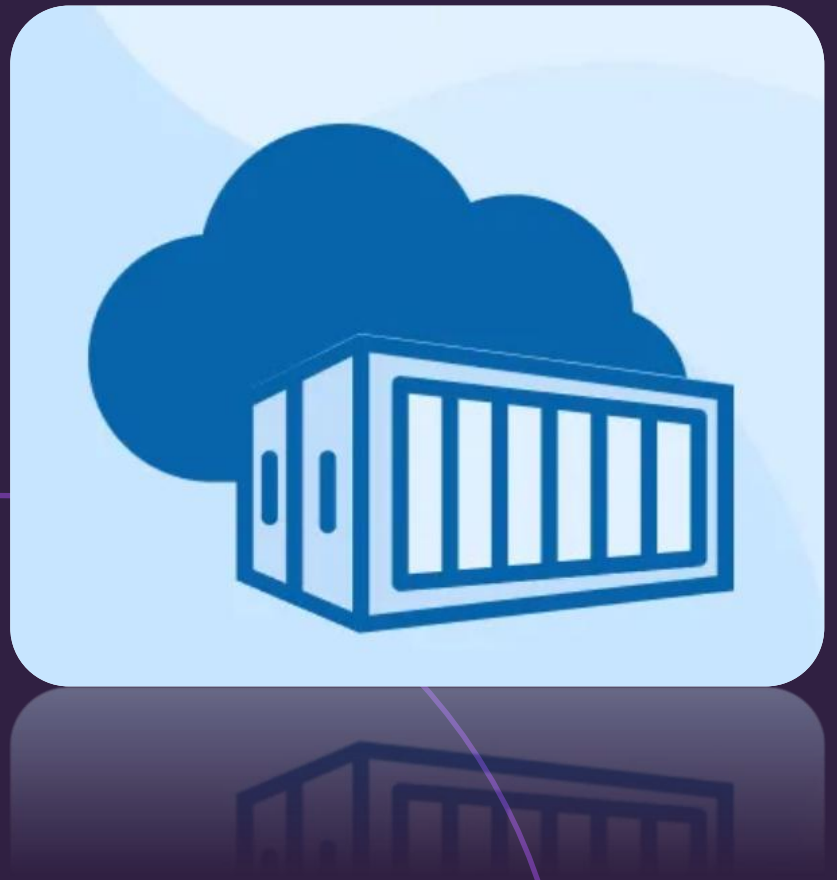


Phocas.

Common Use Cases

- **Microservices**: Each service runs in its own container.
- **CI/CD Pipelines**: Containers simplify testing and deployment.
- **Cloud-native apps**: Designed to scale and run in distributed environments.
- **Dev/Test environments**: Quickly spin up isolated environments.

"Netflix uses containers to deploy thousands of microservices across its infrastructure."



Phocas.

Challenges and Limitations

- **Persistent storage**: Managing stateful data across container restarts can be tricky.
- **Security**: Containers share the host kernel, so isolation isn't as strong as VMs.
- **Complexity**: Orchestration (e.g., Kubernetes) adds a learning curve.
- **Not ideal for legacy apps**: Older systems may rely on host-specific configurations.

"Containers are powerful, but not a one-size-fits-all solution. Evaluate based on your app's needs."

Phocas.

Containers vs Virtual Machines

Feature	Container	Virtual Machine
Startup Time	Seconds	Minutes
Resource Usage	Lightweight	Heavy
Isolation	Process-level	Full OS-level
Portability	High	Moderate

"Containers are ideal for lightweight, scalable workloads. VMs are better for full isolation and legacy systems."

Phocas.

Orchestration

- Tools like **Kubernetes**, **Docker Swarm**, and **OpenShift** help manage containers at scale.

- They handle scheduling, scaling, networking, and health checks.

"Without orchestration, managing hundreds of containers manually would be a nightmare."

Phocas.

How Do We Use Containerisation At Phocas?

- Docker for local development and testing.
- CI/CD Workflows in GitHub.
- Amazon ECS (Elastic Container Service) for our long running services.
- Amazon EKS (Elastic Kubernetes Service) for hosting Users, Permissions, Identity, and Access Management services.

Phocas.

Key Takeaways

- Containerisation is a tool—not a silver bullet.
- It requires thoughtful adoption based on application needs and team maturity.

"Use containers where they shine—portability, speed, and scalability—but don't force them where they don't fit."

Containerisation Quiz - 1

Phocas.

Your team is building a real-time trading platform with strict latency requirements.
What is a potential drawback of using containers?

A Containers are not compatible with high-performance computing environments

B Containers cannot be deployed on-premises

C Containers require a GUI to operate

D Containers introduce slight overhead and unpredictable latency due to shared kernel resources

Containerisation Quiz - 1

Phocas.

Your team is building a real-time trading platform with strict latency requirements.
What is a potential drawback of using containers?

A Containers are not compatible with high-performance computing environments

B Containers cannot be deployed on-premises

C Containers require a GUI to operate

D Containers introduce slight overhead and unpredictable latency due to shared kernel resources

Containerisation Quiz - 2

Phocas.

What is a key difference between containers and virtual machines?

A Containers include a full operating system, while VMs do not

B Containers are heavier and slower to start than VMs

C Containers share the host OS kernel, while VMs run separate OS instances

D Containers require more memory than VMs

Containerisation Quiz - 2

Phocas.

What is a key difference between containers and virtual machines?

A Containers include a full operating system, while VMs do not

B Containers are heavier and slower to start than VMs

C Containers share the host OS kernel, while VMs run separate OS instances

D Containers require more memory than VMs

Phocas.

Observability

Phocas.

What is Observability?

- Observability is the ability to understand the internal state of a system based on the data it produces — such as logs, metrics, and traces — without needing to modify or directly inspect the system itself.

- Monitoring tells you *what* happened; observability helps you understand *why*.

- It helps us understand *why* things break, not just *that* they broke.

- Essential for debugging modern, distributed systems.

“Observability is how well you can understand a system from the outside, without having to ship new code to do so.”
— *Charity Majors, CTO of Honeycomb*

Phocas.

Observability vs Monitoring

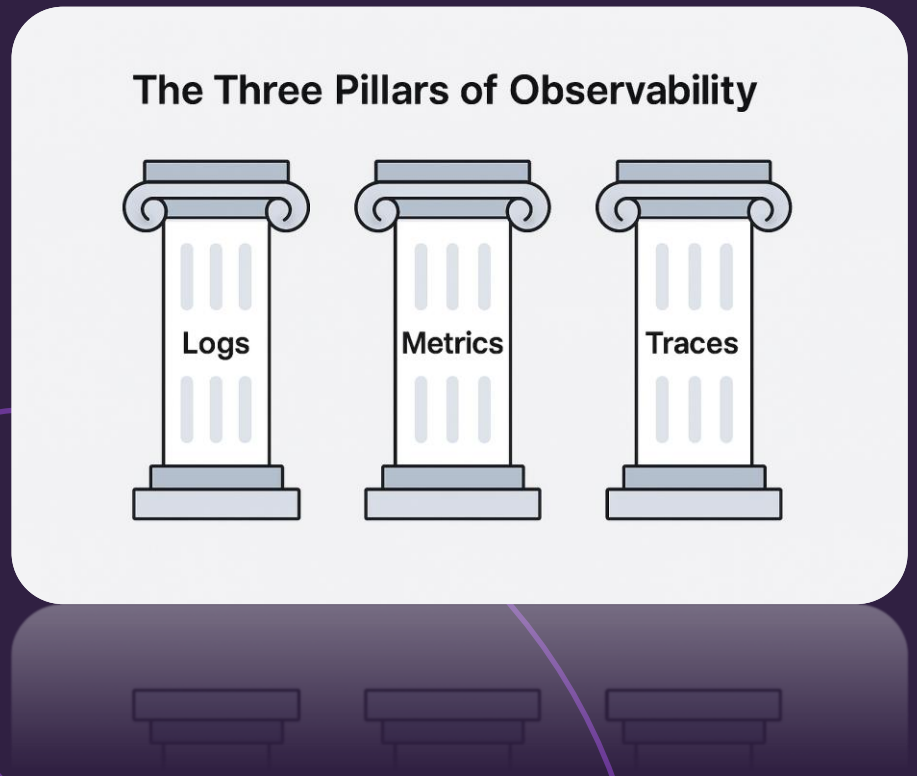
Observability	Monitoring
Explores Unknowns	Detects Known Issues
Ad-hoc Investigation	Predefined Alerts
Dynamic Querying	Static Dashboards

Phocas.

The Three Pillars

- **Logs** – What happened?
- **Metrics** – How is it performing?
- **Traces** – Where did it go wrong?

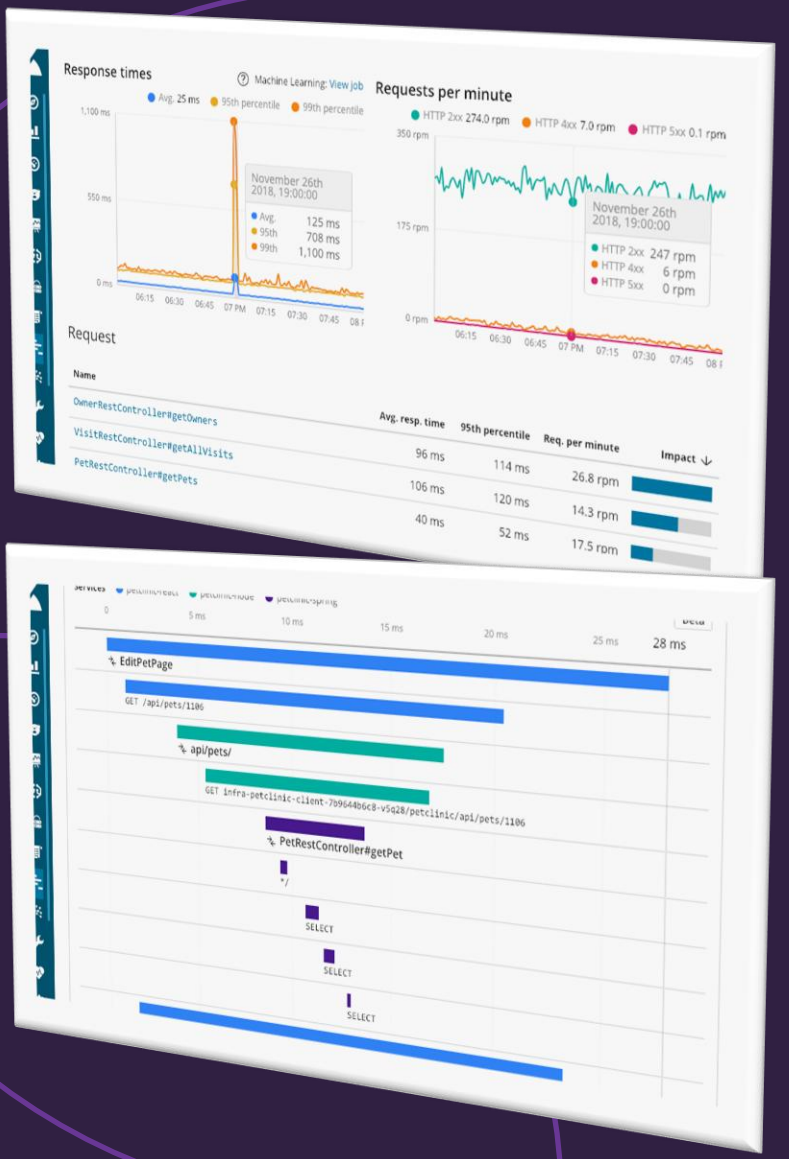
Together, they help you understand system behavior.



Phocas.

Tooling Examples

- **Metrics -** Prometheus, Grafana
- **Logs -** Elastic (ELK - Elasticsearch, Logstash, and Kibana) Stack, Loki
- **Traces -** Jaeger, OpenTelemetry
- **Application Performance Monitoring (APM) -** AppDynamics, Dynatrace, New Relic
- **All-in-One -** Datadog, Honeycomb



Phocas.

What Does Observability Look Like at Phocas?

- **Metrics -** AWS CloudWatch, Elastic
- **Logs -** AWS CloudWatch, Elastic
- **Traces -** OpenTelemetry, Elastic
- APM - Elastic, Raygun

Phocas.

Best Practices

- **Instrument early** – Add telemetry during development.
- **Use structured logs** – Log in key-value pairs (e.g., JSON), include request IDs, user IDs, and timestamps.
- **Correlate data with trace IDs** – Link logs, metrics, and traces using unique identifiers (e.g., trace IDs), connect the dots.
- **Automate alerts** – Focus on meaningful thresholds and anomalies.
- **Focus on actionable insights** – Build dashboards that answer real questions, prioritising user impact, not just system health.
- **Foster a Culture of Curiosity** – Encourage teams to explore telemetry data, not just react to alerts. Use observability as a learning tool during postmortems and retrospectives.

Phocas.

Looking Ahead

- **OpenTelemetry adoption** – Open Source standard, unifies observability tooling, easier to instrument code and is platform agnostic.
- **AI-powered insights** – Using machine learning to detect anomalies, predict failures, and suggest root causes, helps move teams from being reactive to proactive.
- **Observability-as-Code** – Defining dashboards, alerts, and telemetry configurations in version-controlled code, bringing observability into CI/CD pipelines, ensuring consistency across environments, and makes changes auditable and repeatable.
- **Shift-left observability** – Embedding observability earlier in the development lifecycle — during coding, testing, and staging. It helps catch performance and reliability issues before they reach production, improving release confidence and reducing firefighting.

Phocas.

Key Takeaways

- Observability is a mindset
- It enables faster, smarter debugging
- It’s critical for modern DevOps success

Observability Quiz - 1

Phocas.

Which of the following is not one of the three pillars of observability?

A Logs

B Alerts

C Metrics

D Traces

Observability Quiz - 1

Phocas.

Which of the following is not one of the three pillars of observability?

A Logs

B Alerts

C Metrics

D Traces

Observability Quiz - 2

Phocas.

How does observability differ from traditional monitoring?

A Observability only works with cloud-native systems

B Monitoring is proactive, observability is reactive

C Observability helps explore unknown issues, monitoring detects known ones

D They are exactly the same

Observability Quiz - 2

Phocas.

How does observability differ from traditional monitoring?

A Observability only works with cloud-native systems

B Monitoring is proactive, observability is reactive

C Observability helps explore unknown issues, monitoring detects known ones

D They are exactly the same

3

Take Home Example Demonstration

Phocas.

4

Q & A

Phocas.

GitHub URL:

https://github.com/GiftMk/key-value-converter

Phocas.

Phocas.

Thank you