Robust R Deployments: Building a Pipeline from RStudio to Production

David Maguire, Senior Data Scientist

dv01

R is bad for production

dv01

Misconceptions

- R code is ad hoc and untested
- R code is not scalable
- No package & dependency management

R is bad for production

dv01

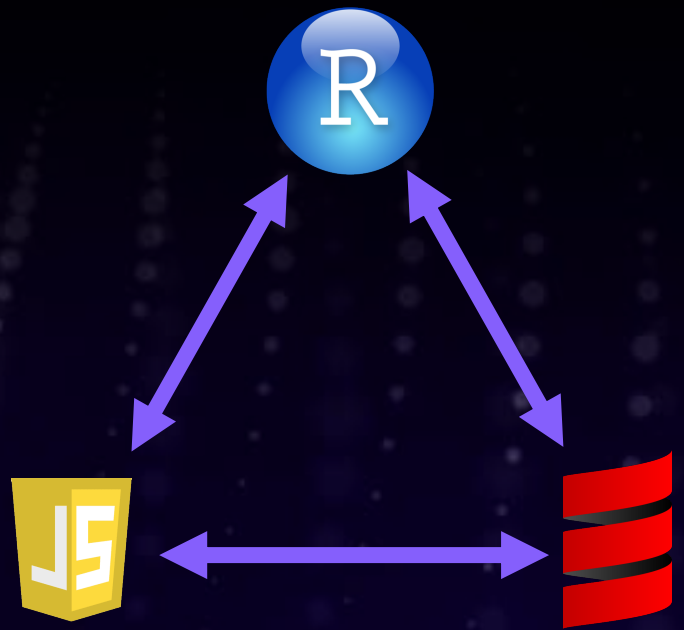
Misconceptions

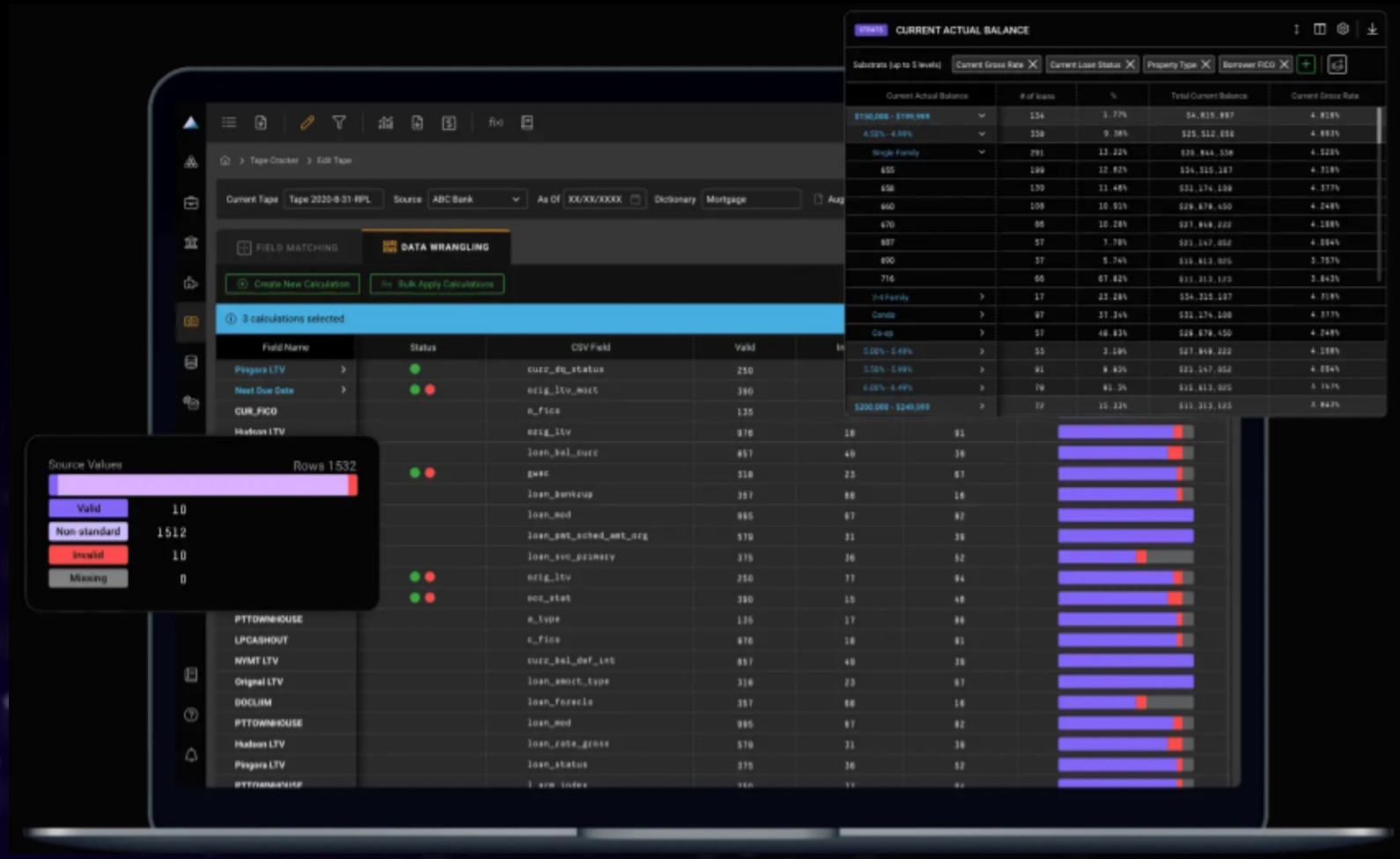
- R code is ad hoc and untested
- R code is not scalable
- No package & dependency management


R is great
for
production

Tape Cracker: Customer Facing Machine Learning

- Web application powered by machine learning in R
- Models available 24/7
- Instant results





dv01

Raising R microservices to thrive in the wild

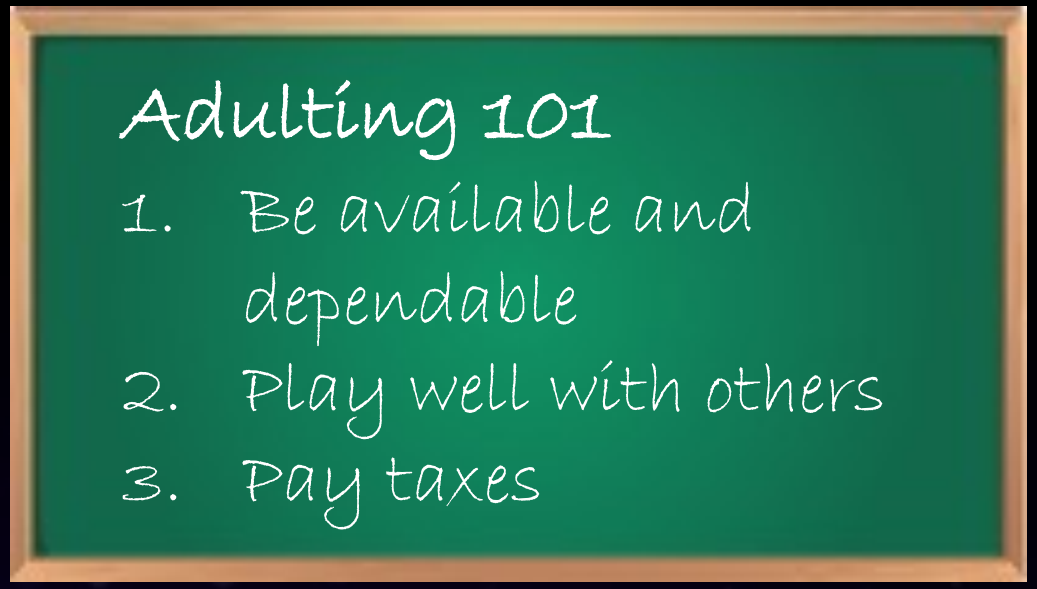
Raising R microservices to thrive in the wild

dv01

https://www.cncf.io/phippy/

Raising R microservices to thrive in the Wild

dv01



https://www.cncf.io/phippy/

Raise R microservices to thrive in the wild!

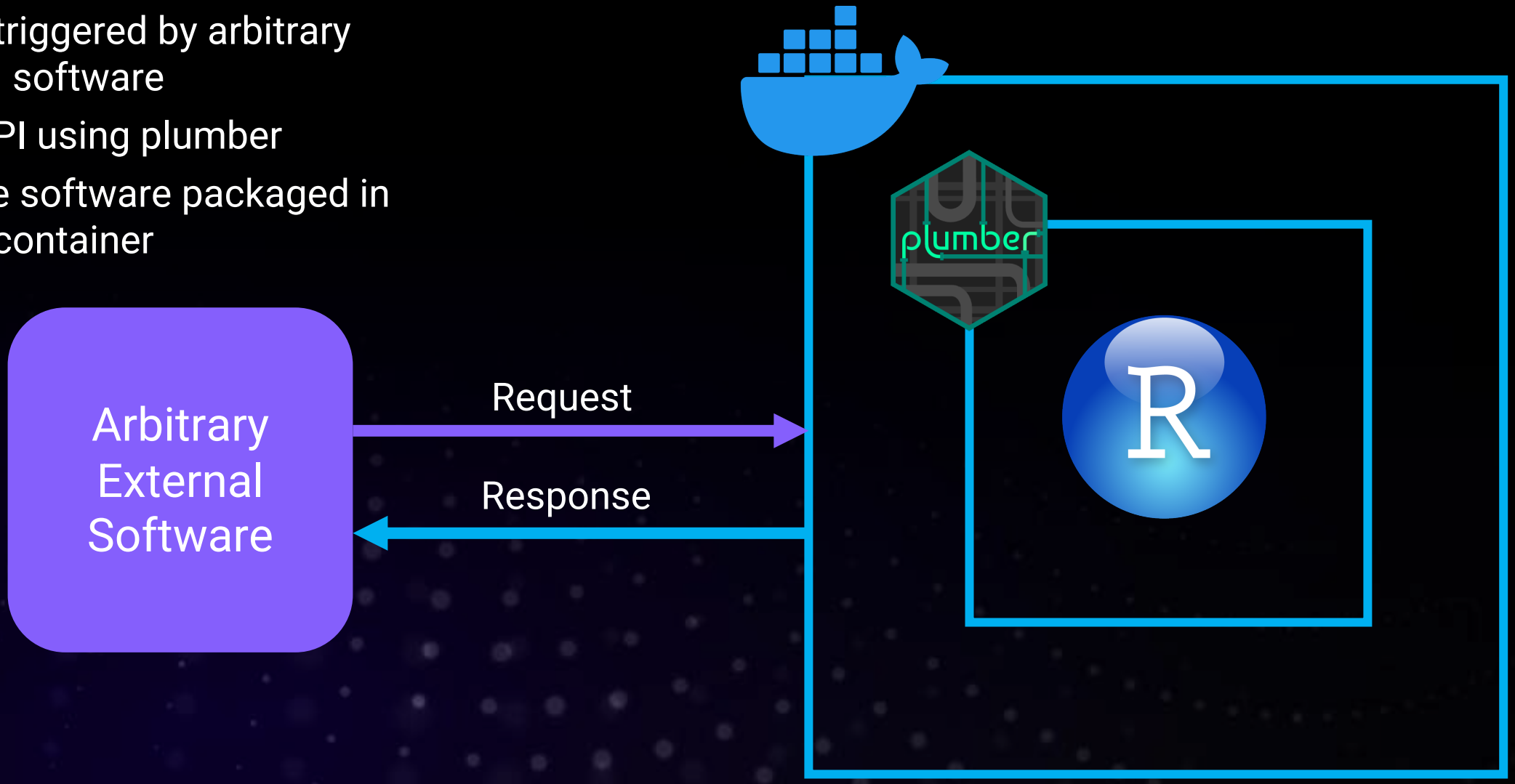
dv01

https://www.cncf.io/phippy/

R microservice <- plumber + docker

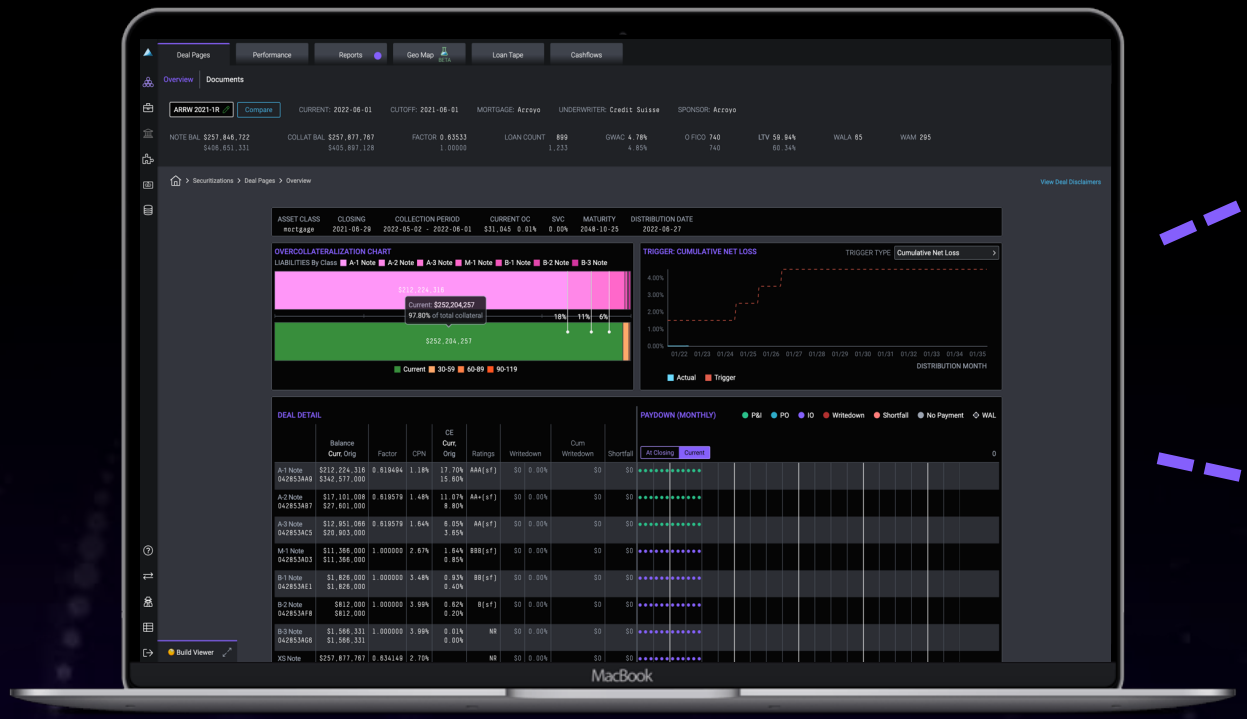
dv01

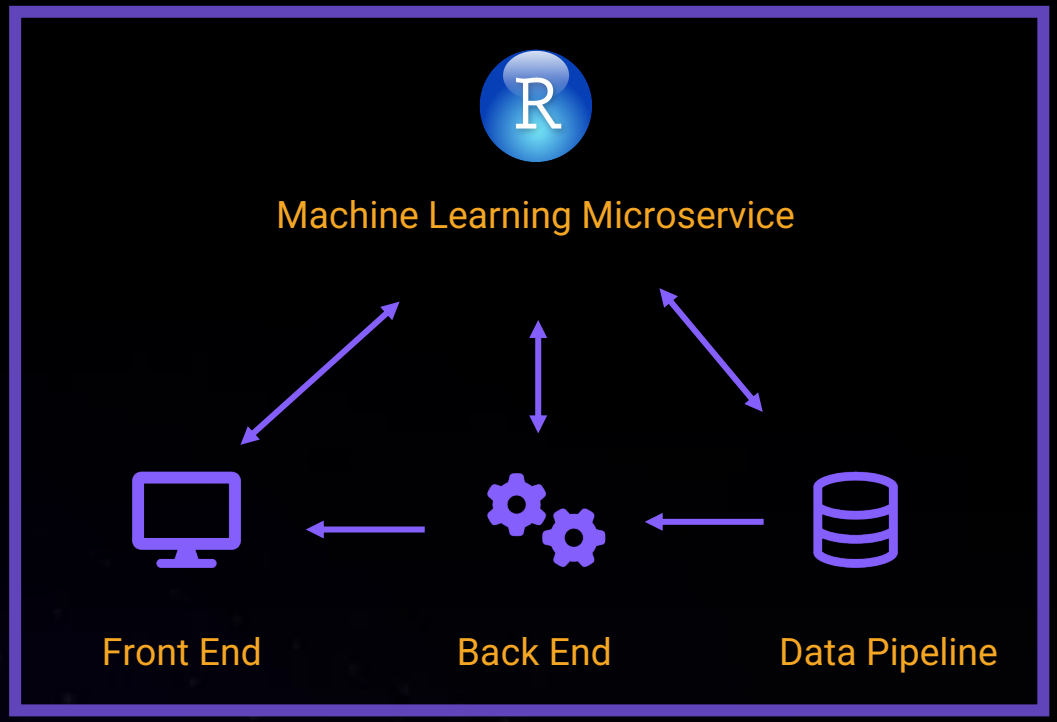
- R code triggered by arbitrary external software
- REST API using plumber
- Portable software packaged in docker container



What is the microservices wild?

dv01





Benefits of Microservice Architecture

dv01

Software engineer builds end-to-end system

Data scientist builds machine learning models

DevOps Principles for Data Science

dv01

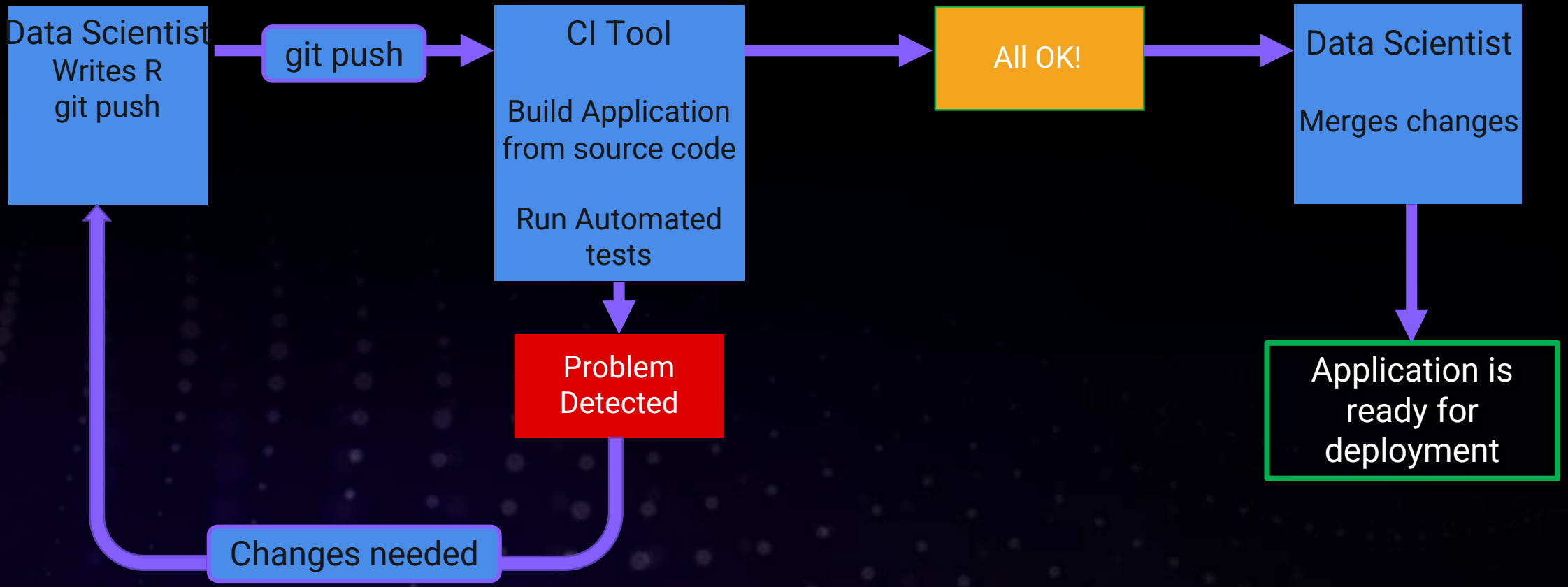
Continuous Integration

Build and test R programs

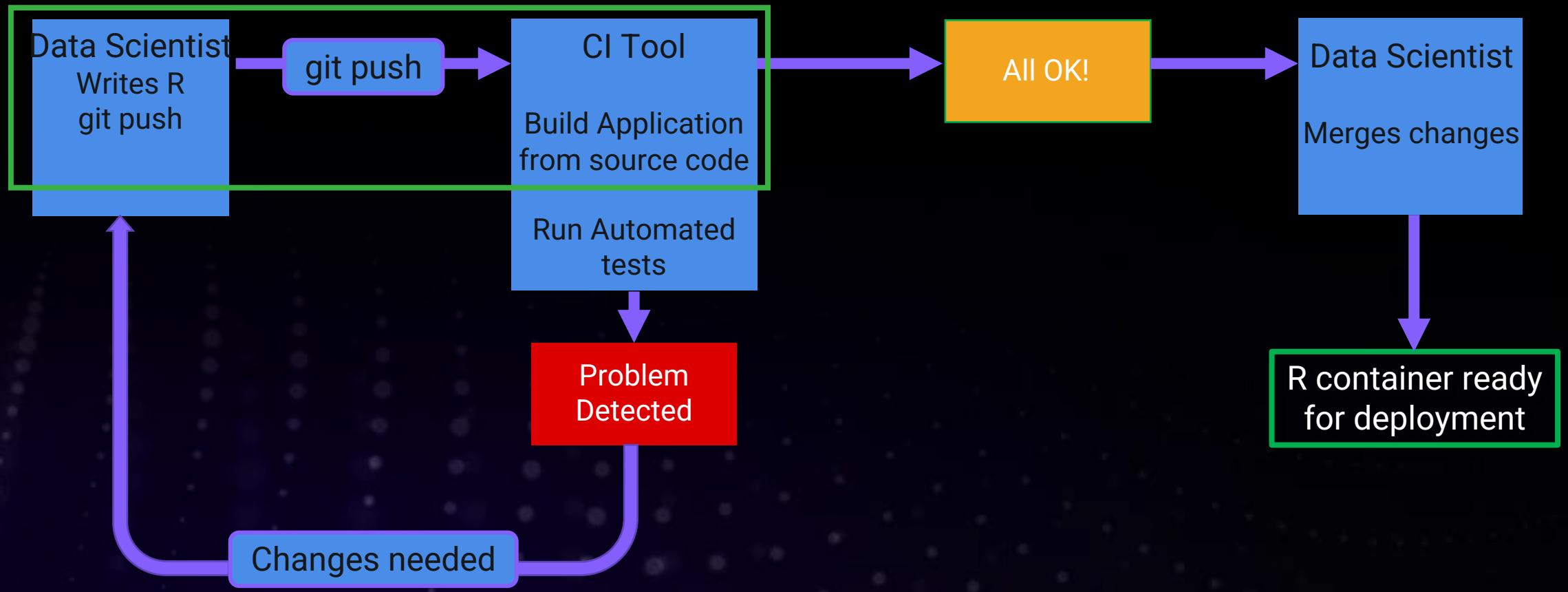
Continuous Deployment

Release validated software into production

Continuous Integration (“CI”)



Continuous Integration (“CI”)



CI Build

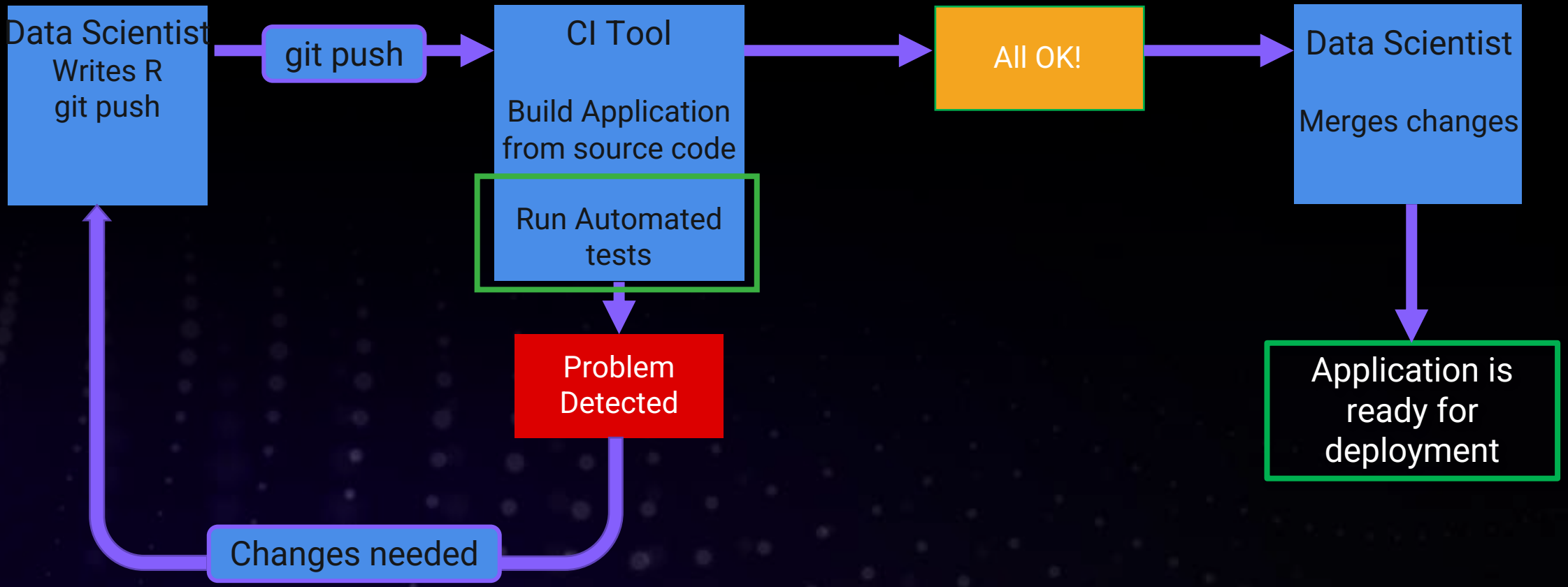
dv01

Data scientist writes R code

CI pipeline triggered for each commit

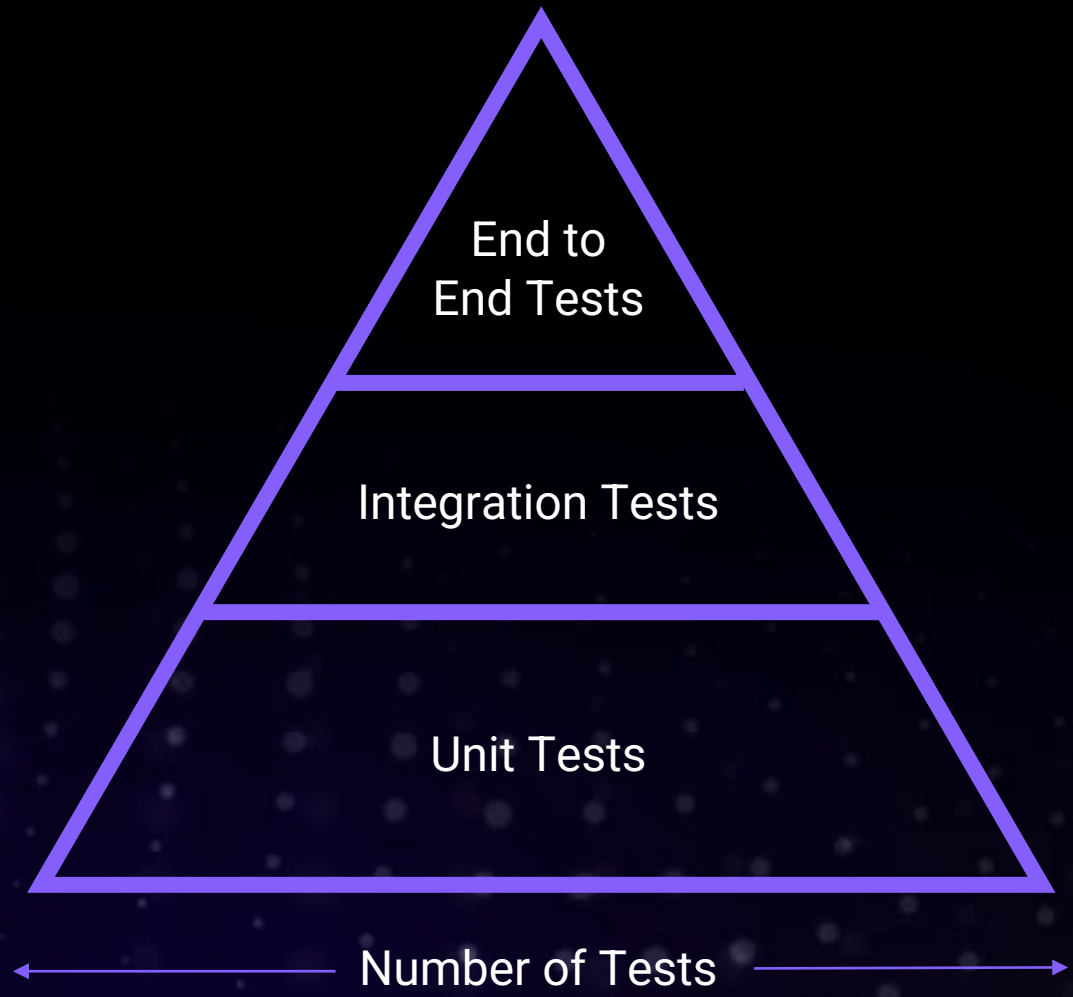
Container pushed to image registry and ready for tests

Continuous Integration (“CI”)

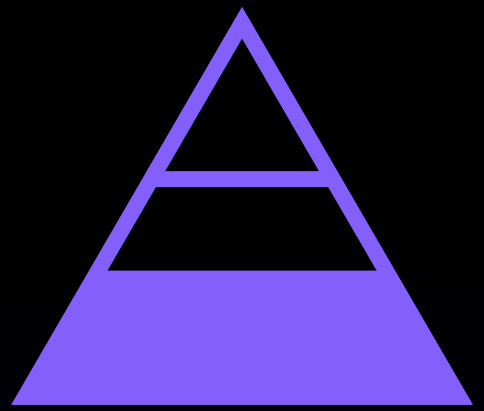


CI Tests

dv01



CI: Unit Tests

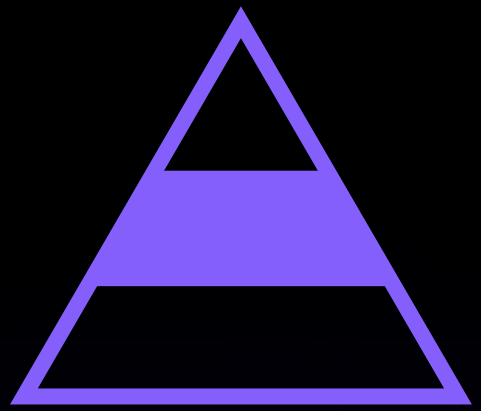


Test individual functions and lower level code

```
1  library(testthat)
2
3  add_20 <- function(x) x + 20
4
5  test_that("add_20() adds 20", {
6      expect_equal(add_20(10), 30)
7  })
```

CI: Integration Tests

dv01

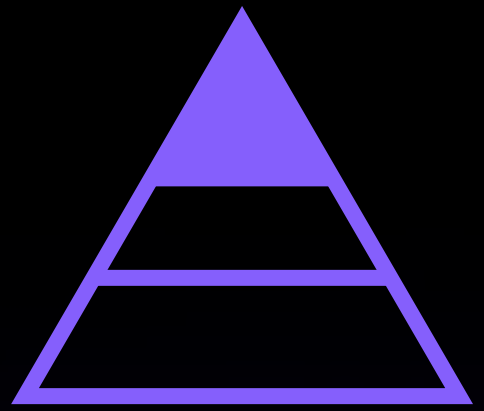


Test R container's ability to interact with other services

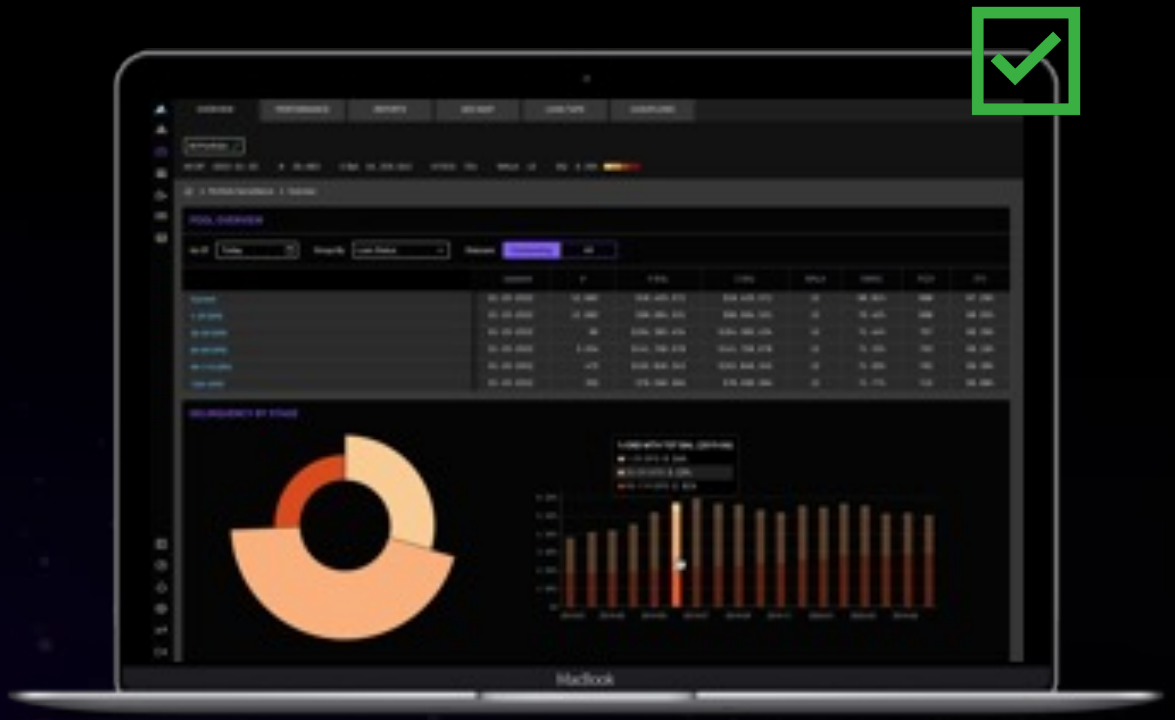
test that

CI: End-to-End Tests

dv01

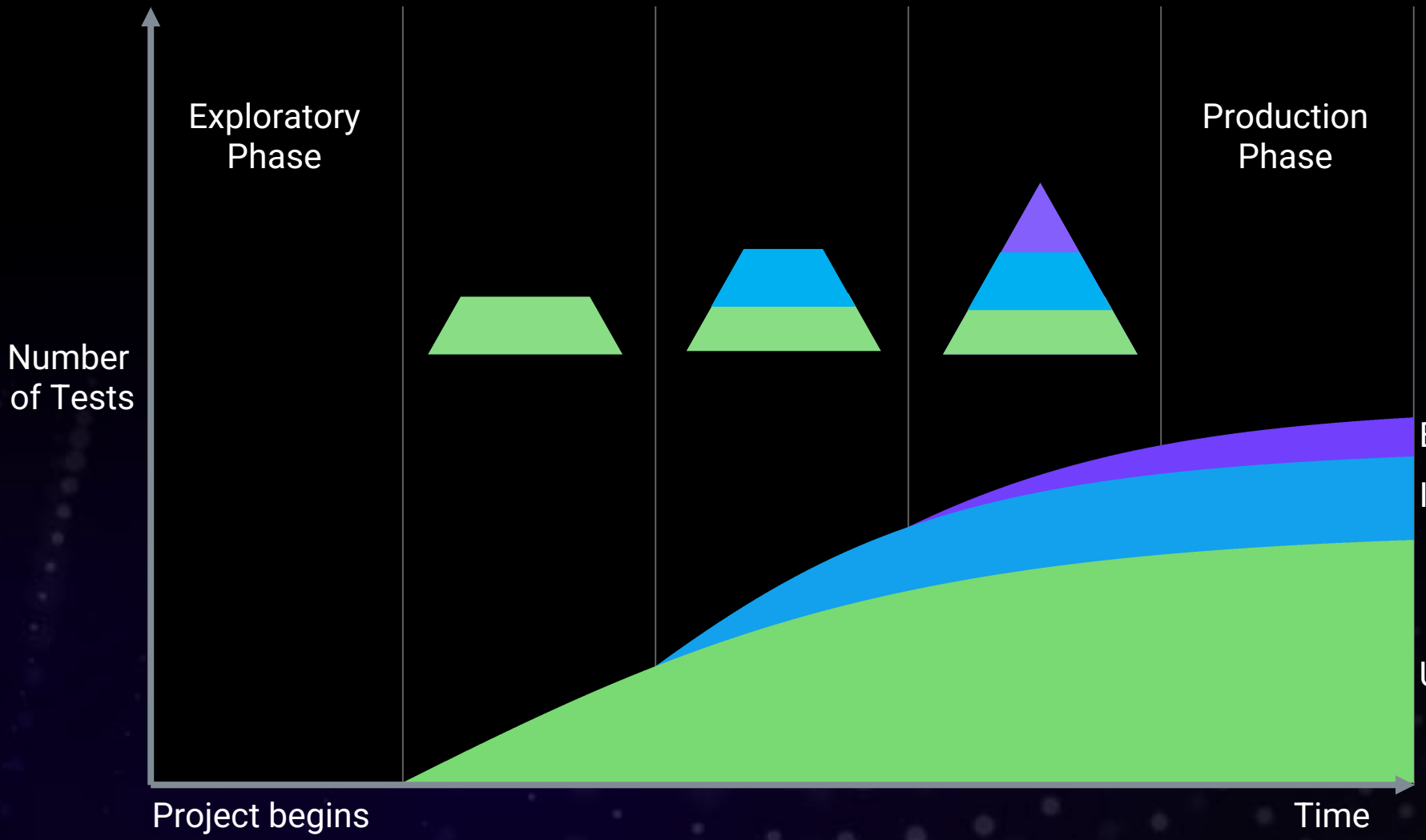


- Full microservice application tested as a whole
- R container tested indirectly



Write tests at the right time

dv01



- CI tests save time in production phase
- Premature test development wastes time

Continuous Deployment

dv01

CI Pipeline

Container Registry

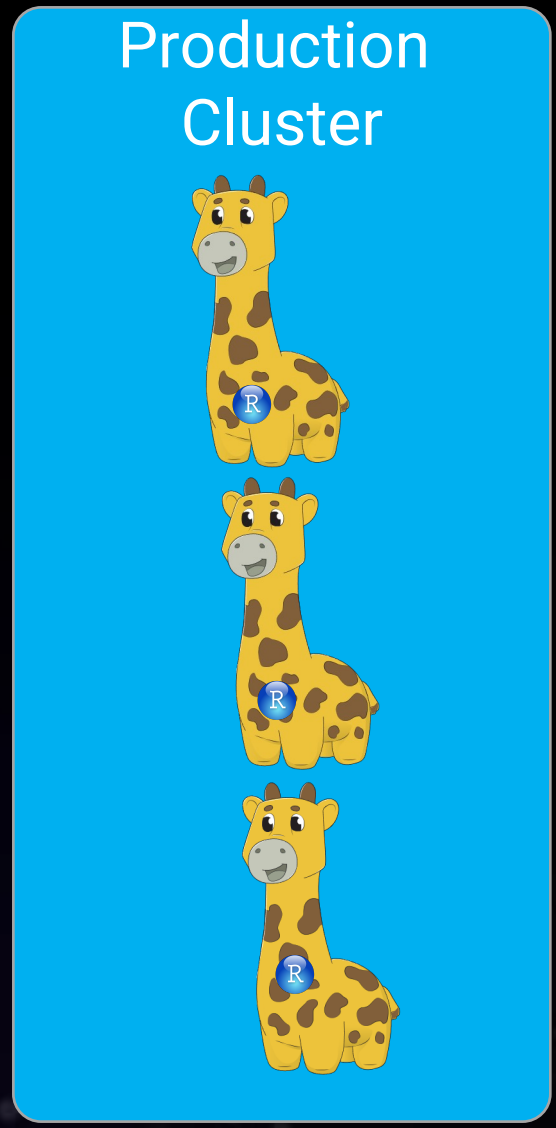
R container ready for deployment

Kubernetes

Staging → Release → Production

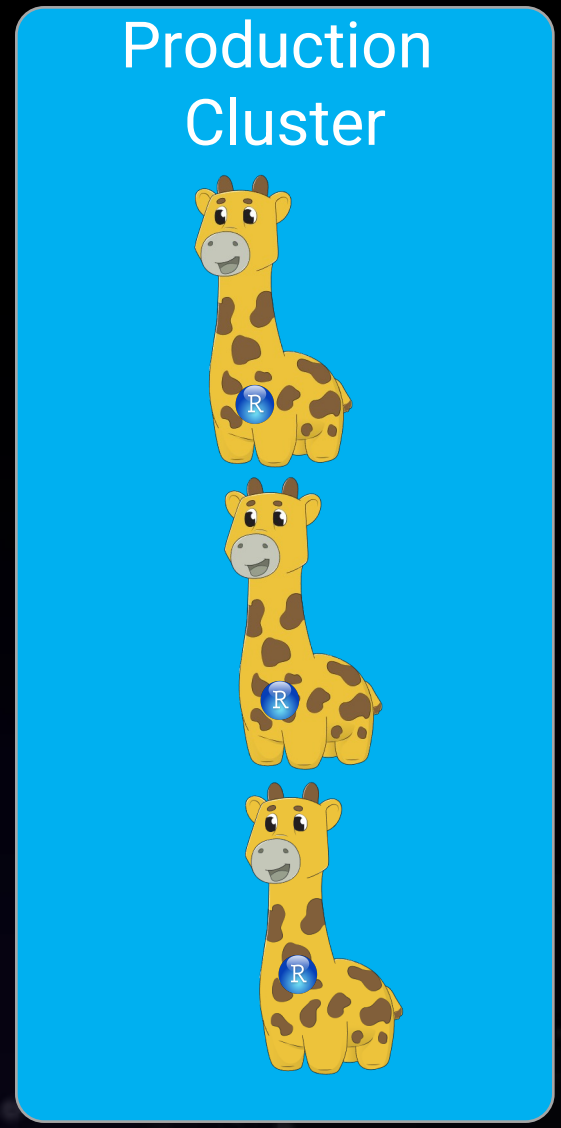
R microservices in Kubernetes

dv01



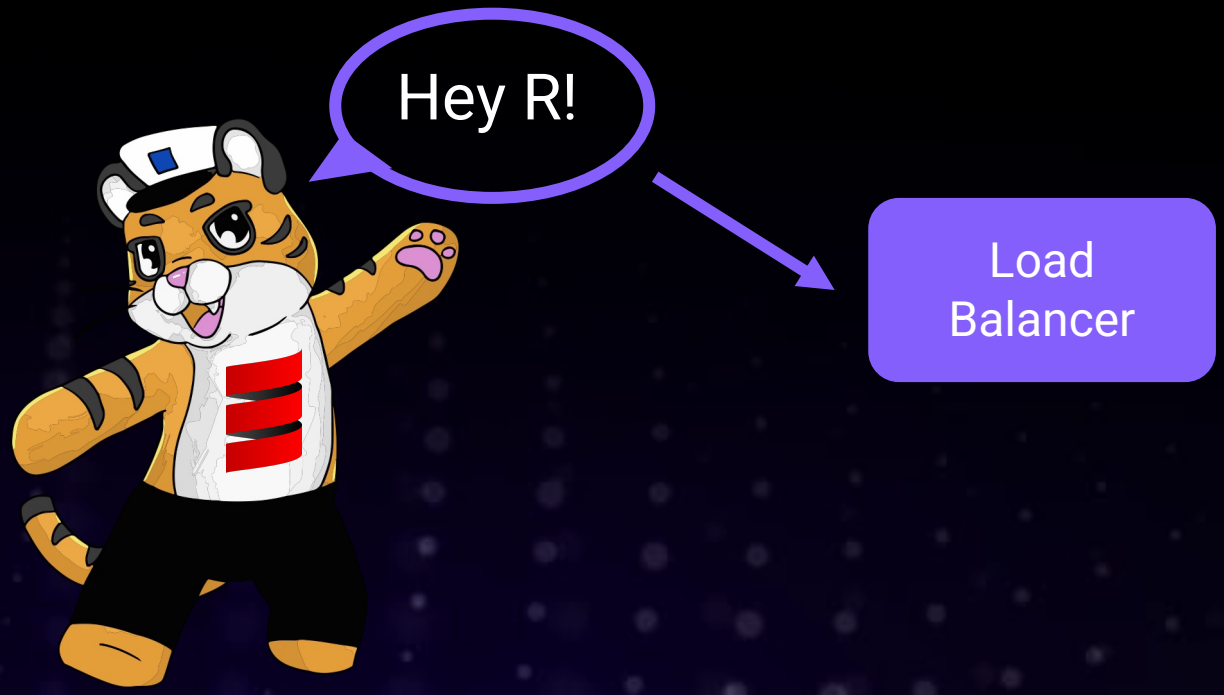
R microservices in Kubernetes

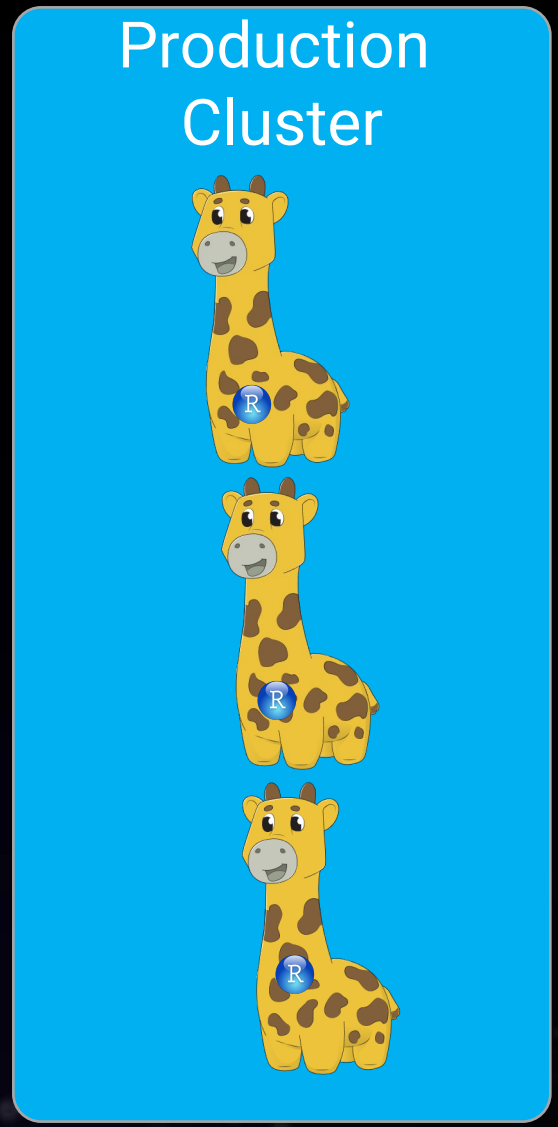
dv01



R microservices in Kubernetes

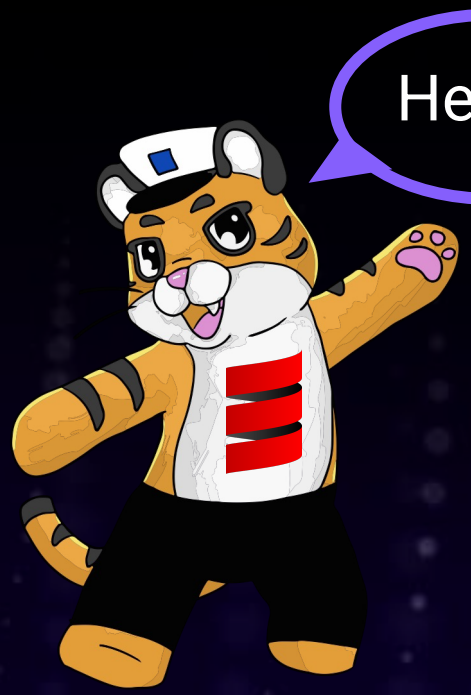
dv01





R microservices in Kubernetes

dv01

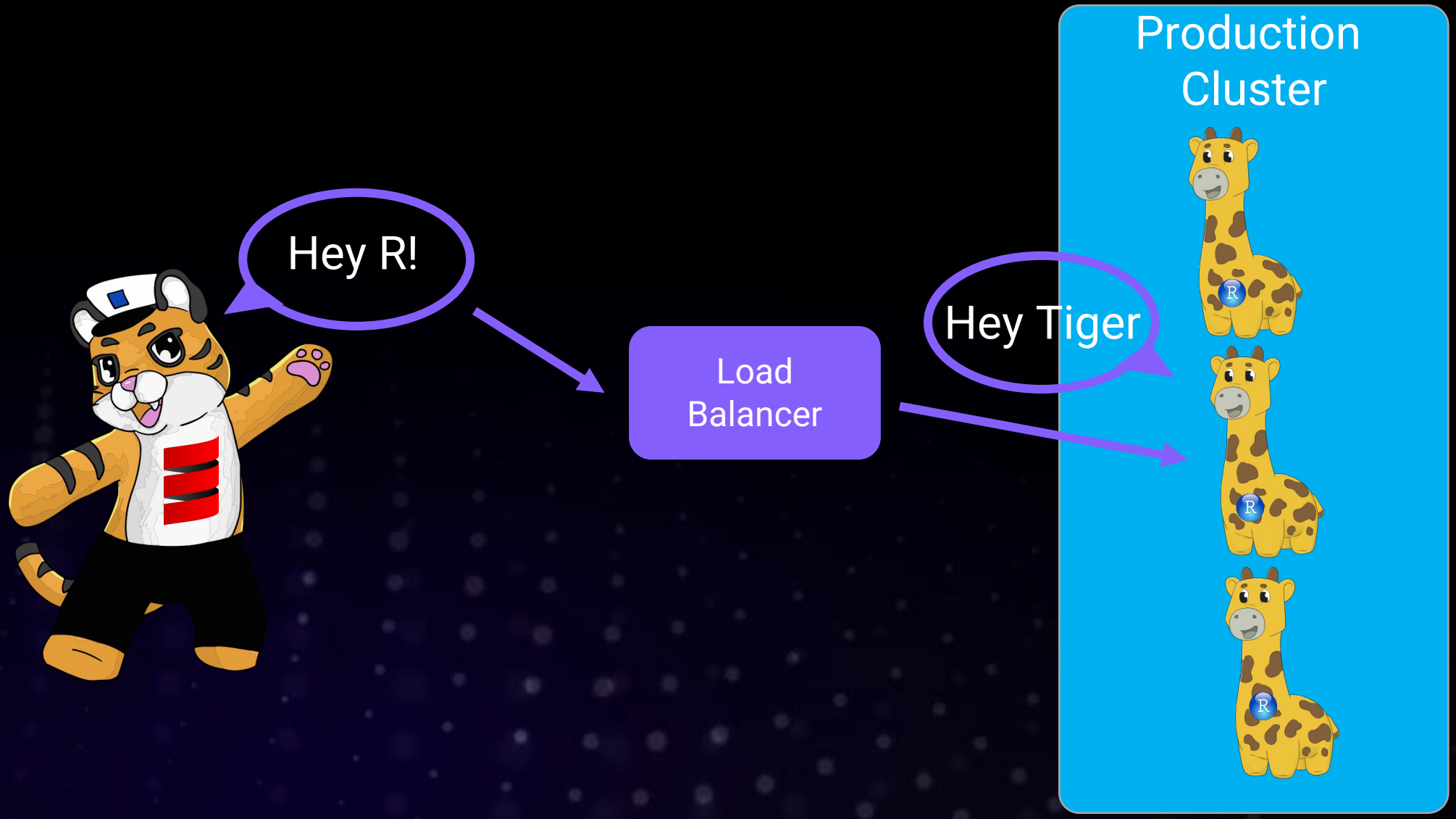


Load Balancer

Production Cluster

R microservices in Kubernetes

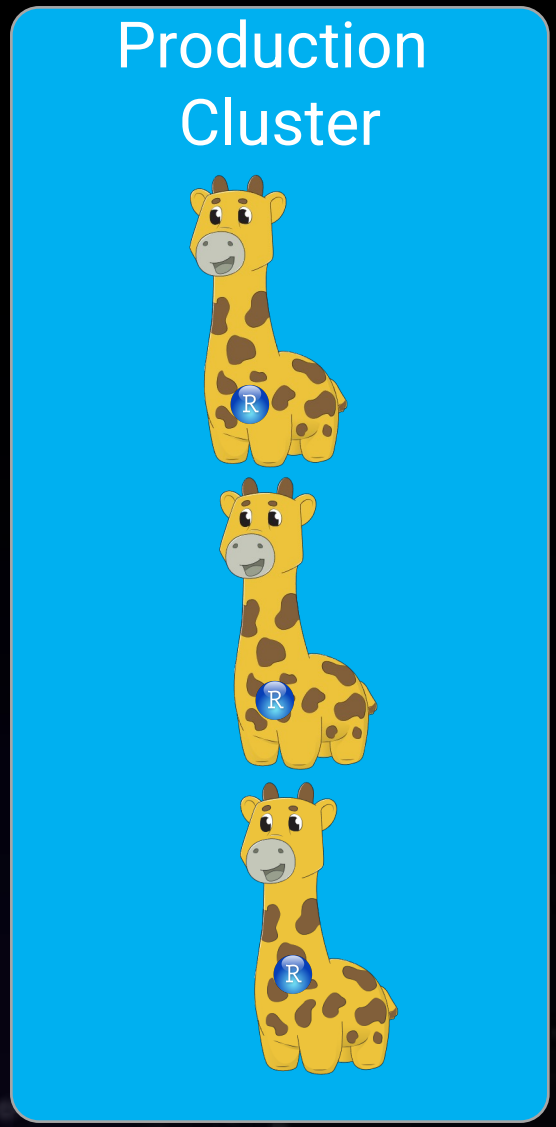
dv01



R microservices in Kubernetes

dv01

- ReplicaSet of microservice
  - 3 copies always present
  - More copies created when traffic is high
  - If one replica fails, it is immediately replaced with a working copy



Conclusion

dv01

- R is great for production (with CI/CD)
  - **Continuous Integration** ensures R packages are robust and well vetted
  - **Continuous Deployment** automates the release of updates
- R microservices can be built with plumber and docker
- **Plumber** creates REST APIs from R code
- **Docker** creates portable software packages
- **Kubernetes** manages resilient compute cluster

dv01

Questions?

David Maguire, Sr Data Scientist

dmaguire@dv01.co

dv01.co