

# Microservices + DevOps

Charlie Garrod

**Michael Hilton**

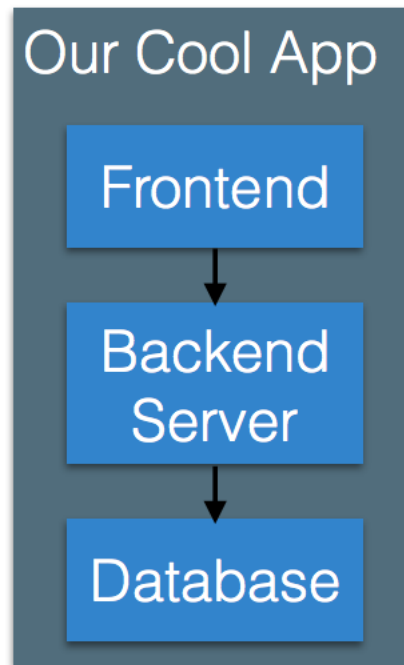
# Administrivia

- Homework 6 checkpoint – Monday Dec 4<sup>th</sup>
- Final Exam Review: Dec 13<sup>th</sup>, 2-4pm Wean 5409
- Final Exam: Dec 15<sup>th</sup>, 5:30-8:30pm Wean 7500

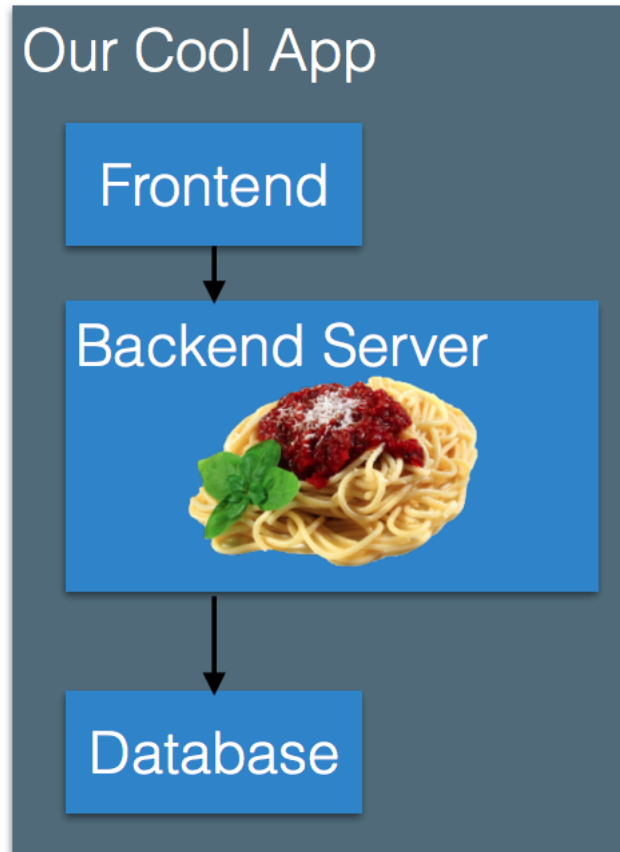
## Last Time:

- Architectural styles

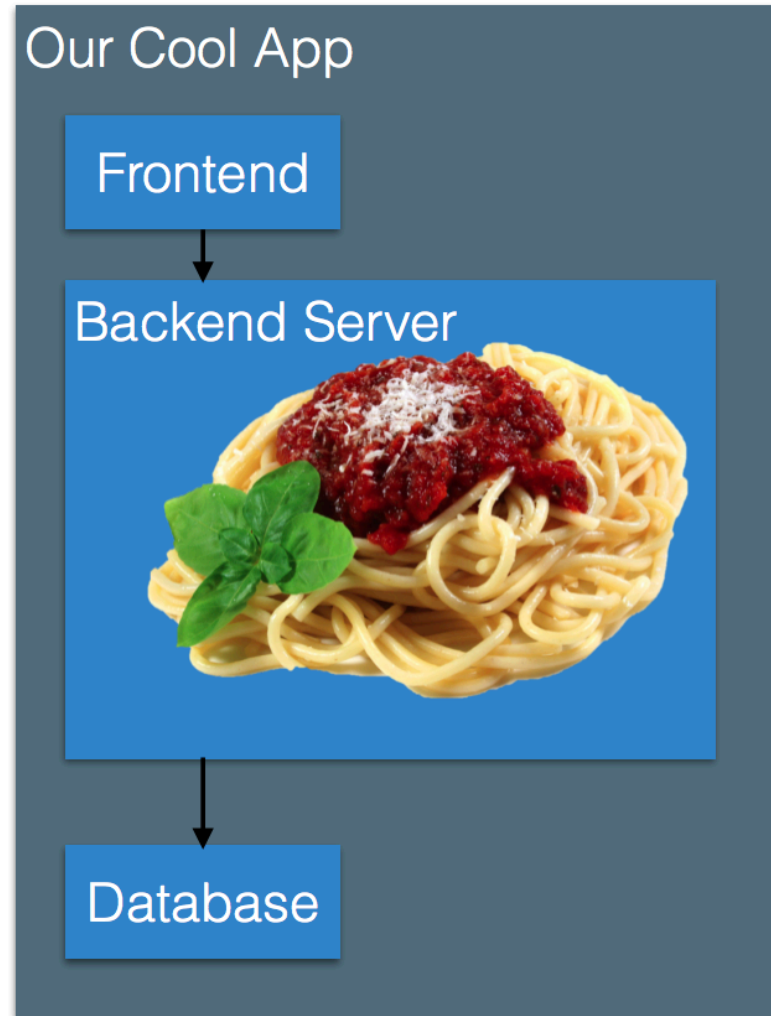
# Simple Layers App



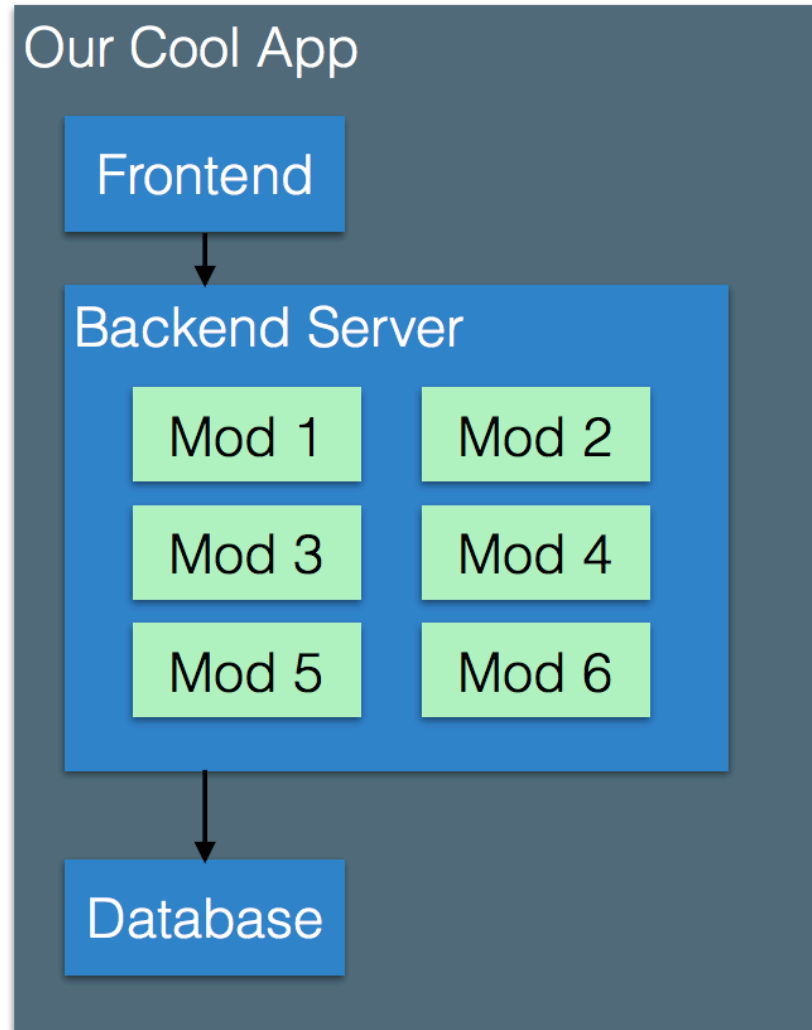
# More functionality



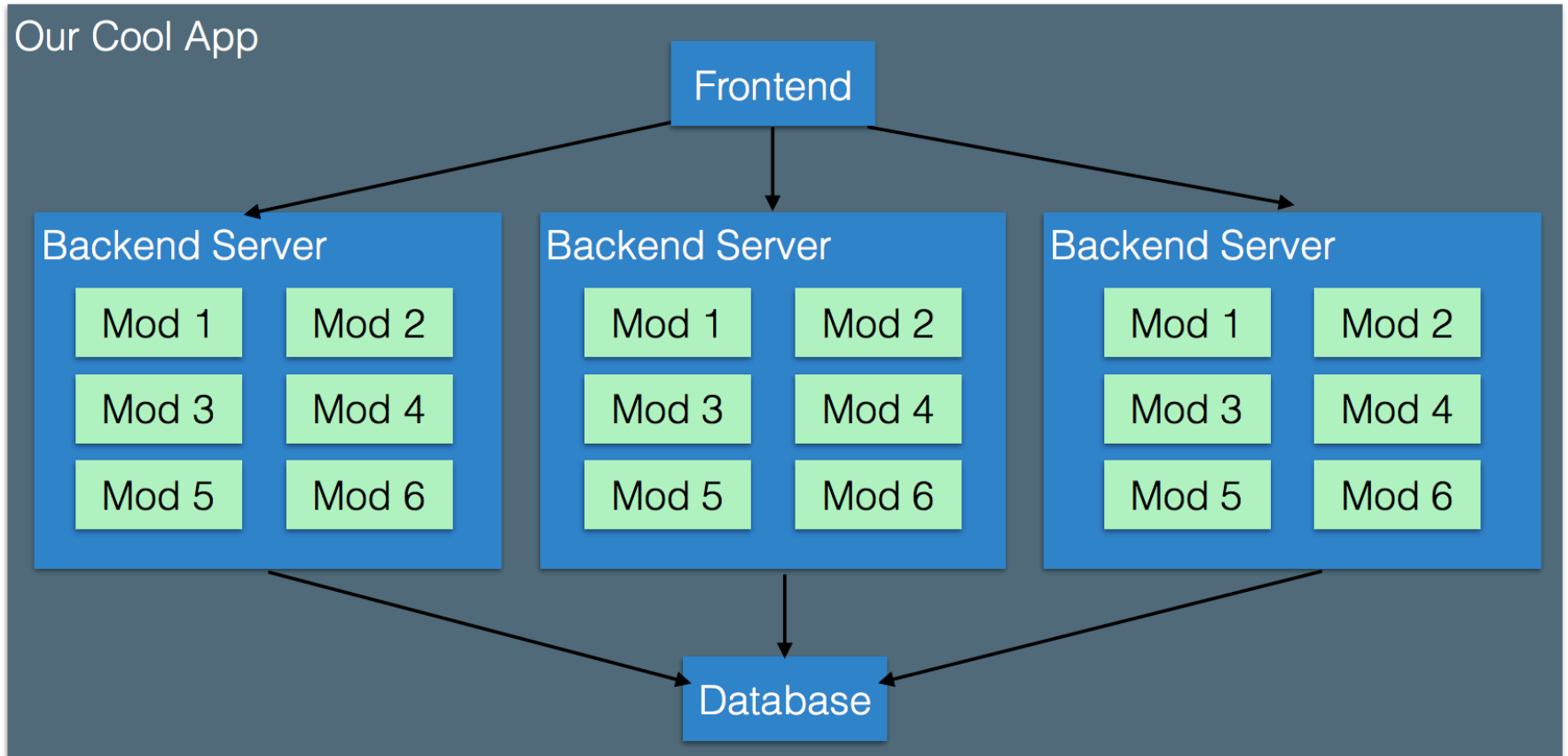
# Even more functionality



# Organize our backend



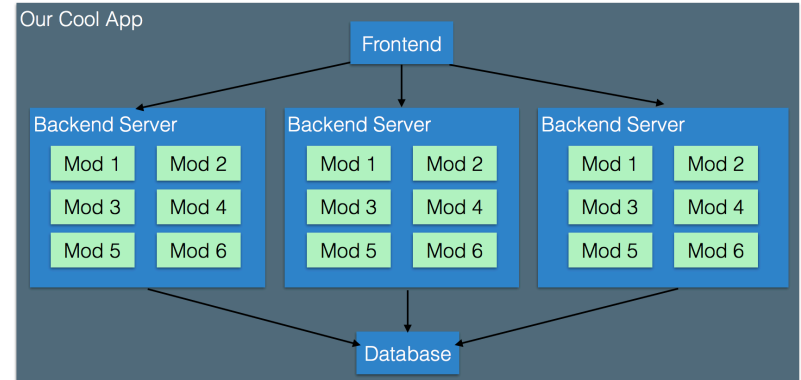
# How to scale?



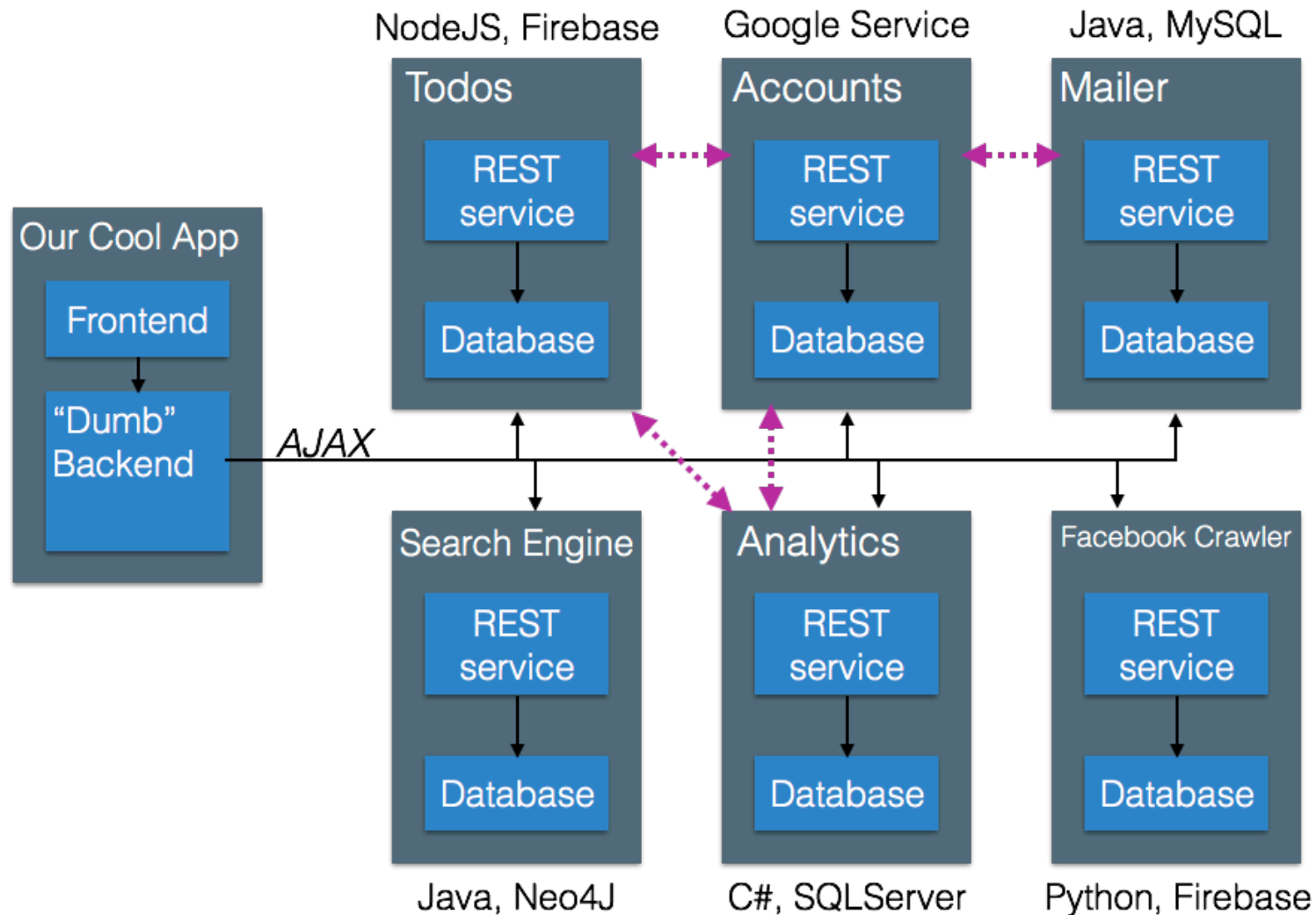


# Monolith

- What happens when we need 100 servers?
- What if we don't use all modules equally?
- How can we update individual models?
- Do all modules need to use the same DB, language, runtime, etc?



# Microservices



# Microservices should be:

- Modelled around business domain
- Culture of automation
- Hide implementation details
- Decentralized governance
- Deploy independently
- Design for failure
- Highly observable

# Microservice prerequisites

- Rapid Provisioning
- Basic Monitoring
- Rapid Application Deployment
- Devops Culture

You must be  
this tall to use  
microservices

---





Why are microservices such a big deal?

**NETFLIX**

**amazon**

# Impact on development practices

- Amazon transitioned to “two-pizza” teams
- “Full Stack” developers
- “Devops” as a prereq
- Live testing and rollback
- Migrating from “monolith to microservices” is popular, but comes at a cost

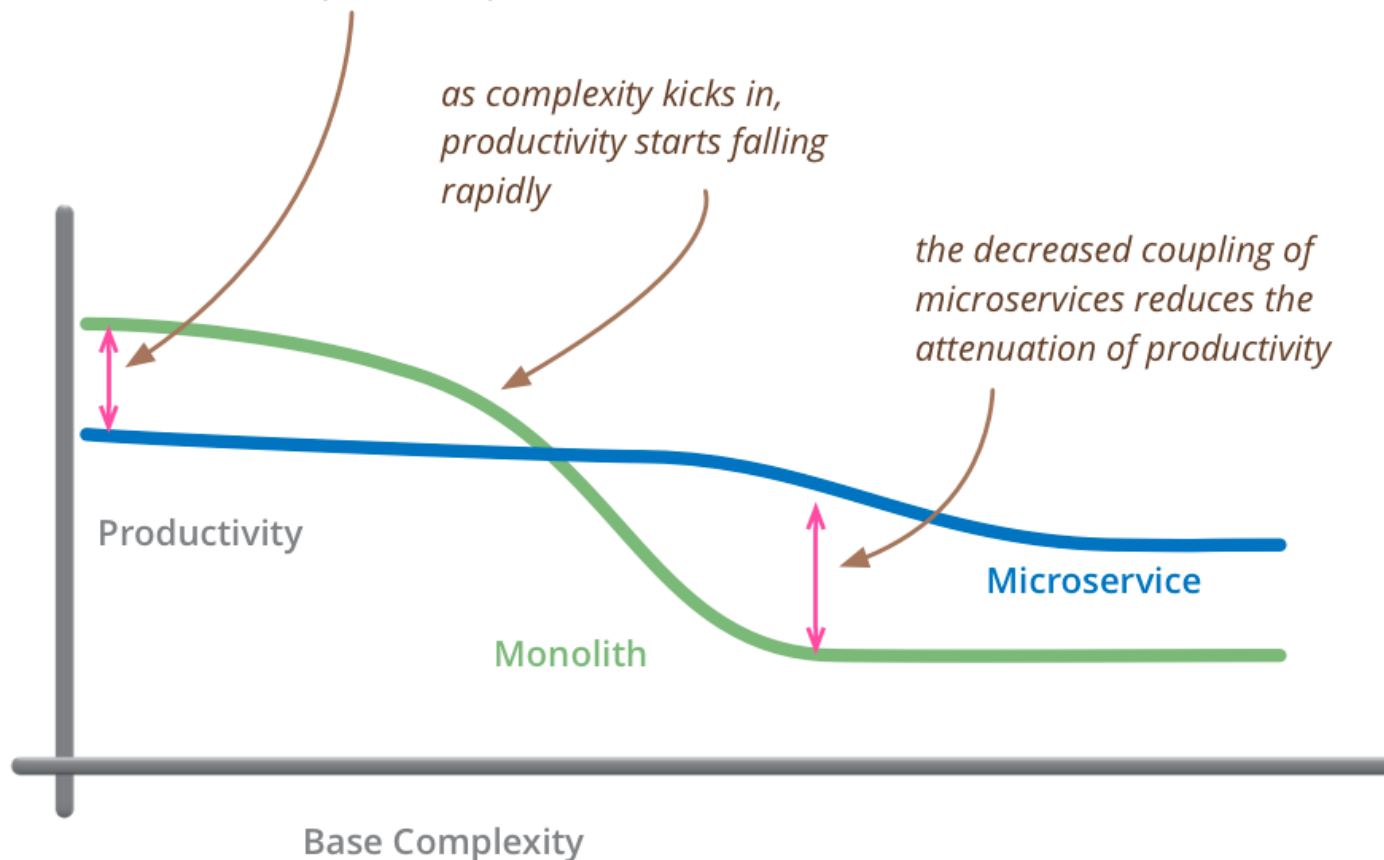
# Microservices benefits

- Strong Module Boundaries
- Independent Deployment
- Technology Diversity



# Microservices overhead

*for less-complex systems, the extra baggage required to manage microservices reduces productivity*



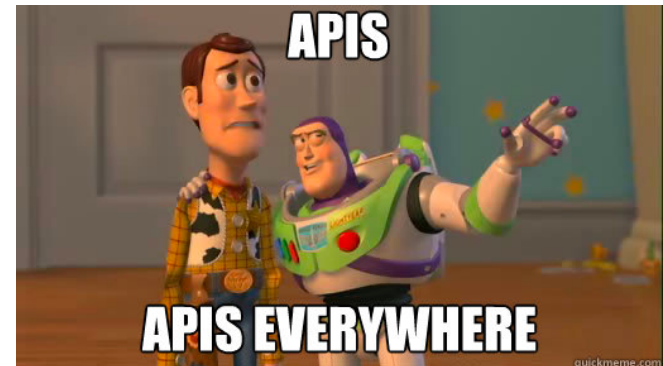
*but remember the skill of the team will outweigh any monolith/microservice choice*

# Microservice costs

- Distribution
- Eventual Consistency
- Operational complexity

# Discussion of Microservices

- Are they really “new”?
- Do microservices solve problems, or push them down the line?
- What are the impacts of the added flexibility?
- Good Architecture doesn’t fix poor low level design problems
- Beware “cargo cult”
- “If you can’t build a well-structured monolith, what makes you think microservices is the answer?” – Simon Brown
- Leads to more API design decisions



# Microservice prerequisites

- Rapid Provisioning
- Basic Monitoring
- Rapid Application Deployment
- **Devops Culture**

You must be  
this tall to use  
microservices

---



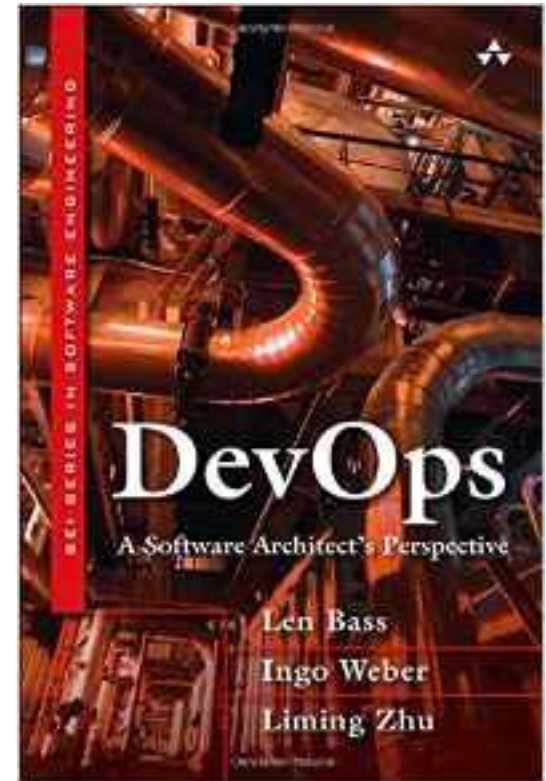
# DEVOPS

# Why DevOps?

- Developers and Operations don't have the same goals
  - Devs want to push new features
  - Ops wants to keep the system available (stable, tested, etc.)s
- Poor communication between Dev and Ops
- Limited capacity of operations staff
- Want to reduce time to market for new features
- Reduce “Throw it over the fence” syndrome

# DevOps Definition

- “DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality.”

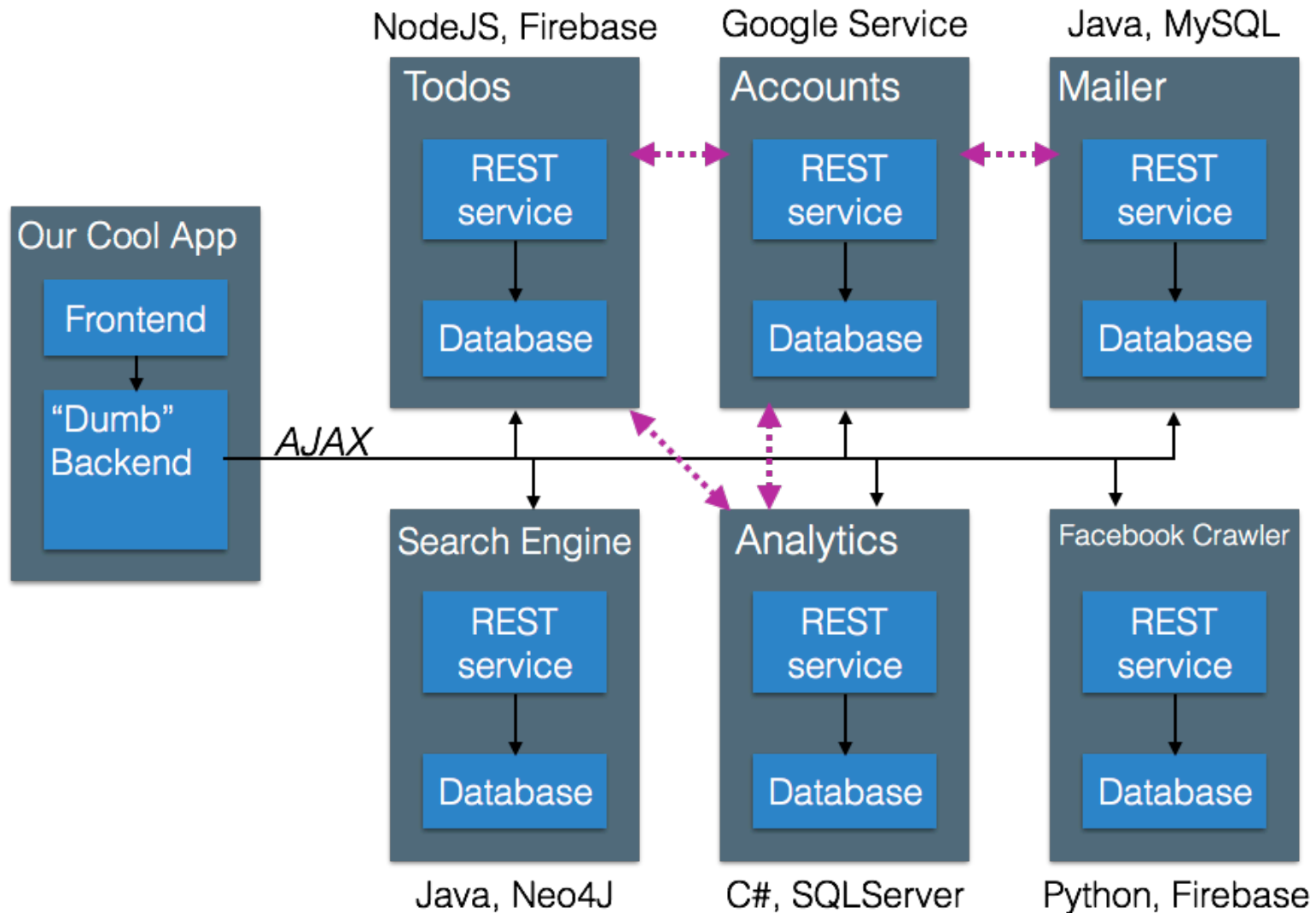


# What are implications of DevOps?

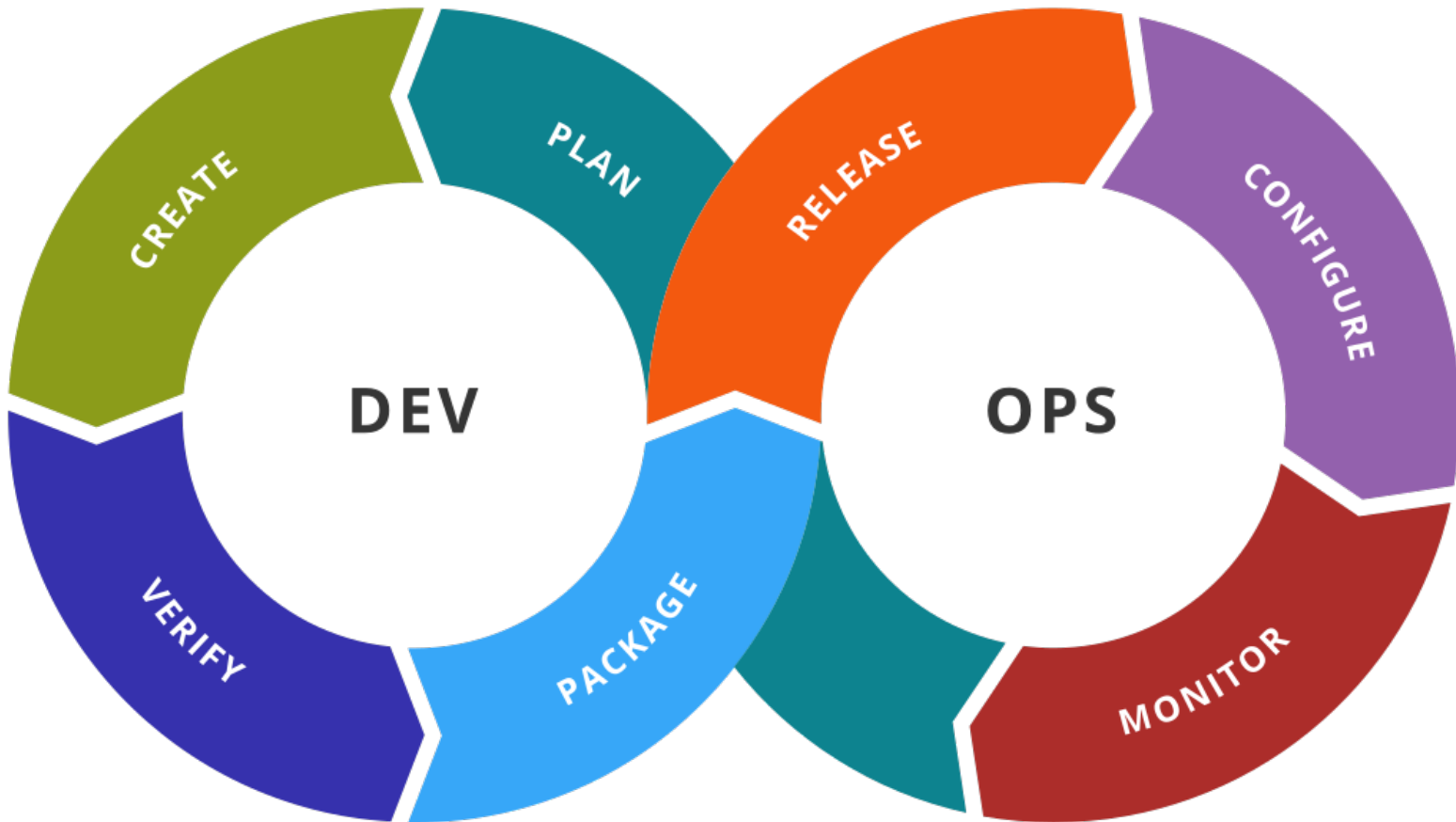
- Quality of the code must be high
  - Testing
- Quality of the build & delivery mechanism must be high
  - Automation & more testing
- Time is split:
  - From commit to deployment to production
  - From deployment to acceptance into normal production
- Goal-oriented definition
  - May use agile methods, continuous deployment (CD), etc.
  - Likely to use tools
- Achieving it starts before committing



# Microservices rely on DevOps

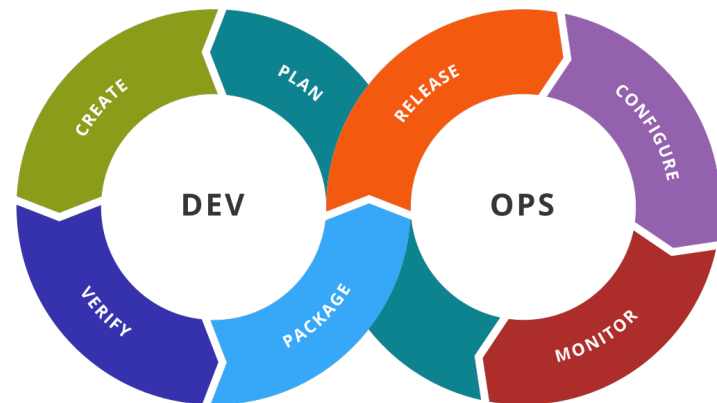


# DevOps Toolchain



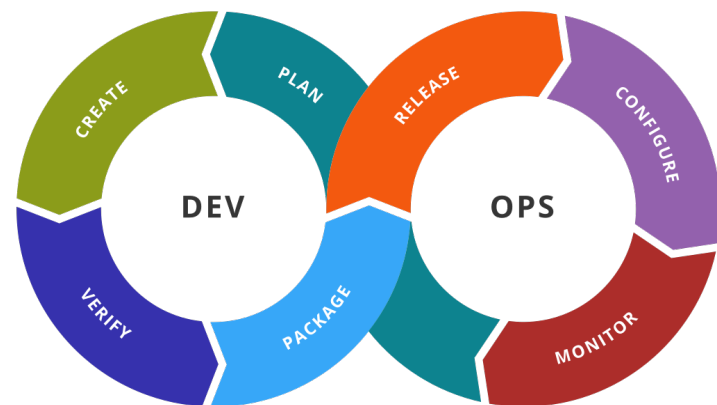
# DevOps Toolchain

- Code — code development and review, source code management tools, code merging
- Build — continuous integration tools, build status
- Test — continuous testing tools that provide feedback on business risks
- Package — artifact repository, application pre-deployment staging



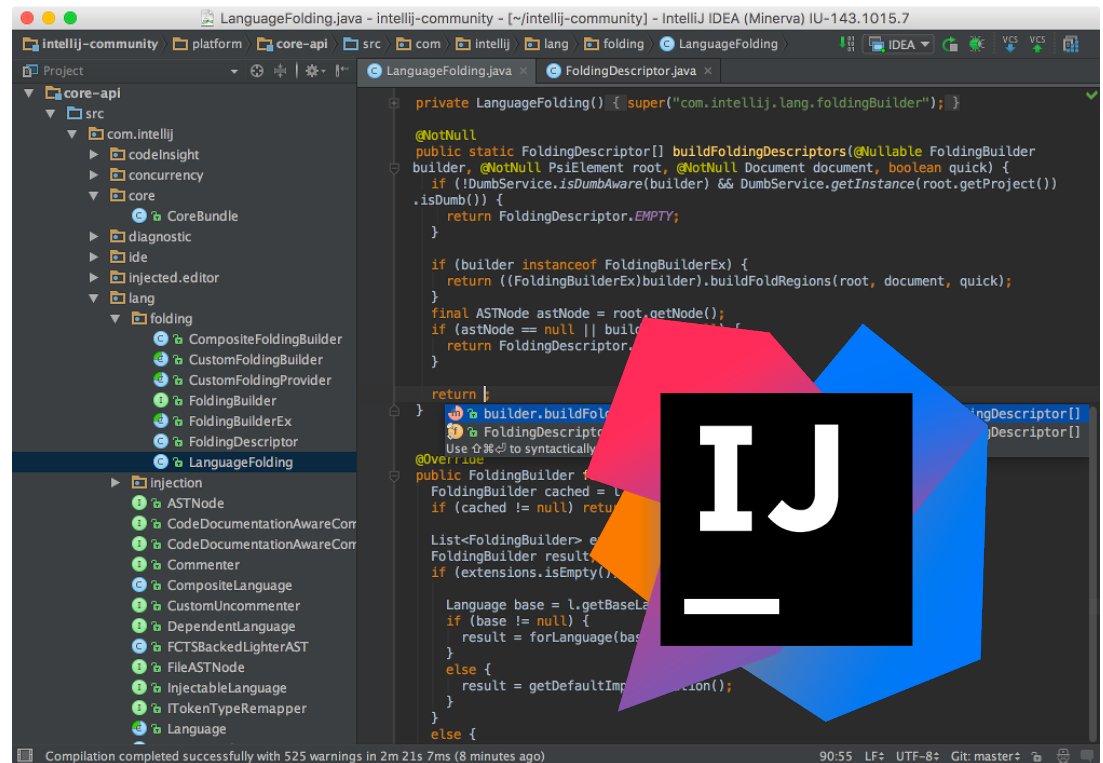
# DevOps Toolchain continued

- Release — change management, release approvals, release automation
- Configure — infrastructure configuration and management, Infrastructure as Code tools
- Monitor — applications performance monitoring, end-user experience



# DevOps Toolchain - Code

- **Code development** and review
- Source code management tools
- Code merging



# DevOps Toolchain - Code

- Code development and **review**
- Source code management tools
- Code merging

Framework gui #3 Edit

**Merged** RufusBarbarossa merged 7 commits into master from framework\_gui 16 days ago

Conversation 4 Commits 7 Files changed 11 +605 -39

CalLavicka commented 16 days ago

The gui is mostly done. Need some plugins to test it with and also need to implement swapping.

Reviewers

RufusBarbarossa ✓

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

Unsubscribe

You're receiving notifications because you're subscribed to this repository.

2 participants

Lock conversation

CalLavicka added some commits 16 days ago

- added some base gui stuff b1614b7
- merged master into framework\_gui 2a3d17c
- Added functionality to the add data source button 3fc8cc0
- Added a GuiBuilder class for building input dialogues db5c6ba
- Added error message for connection as well as adding the visualizatio... a337af3
- Some minor changes 6f4ecbd

CalLavicka requested a review from RufusBarbarossa 16 days ago

RufusBarbarossa requested changes 16 days ago View changes

Add Javadocs for all classes as well as fixing the two comments

framework/src/main/edu/cmu/cs/cs214/example/TestDataStore.java Show outdated

framework/src/main/edu/cmu/cs/cs214/framework/gui/FrameworkFrame.java Show outdated

Added javadocs c3b3017

CalLavicka requested a review from RufusBarbarossa 16 days ago

RufusBarbarossa approved these changes 16 days ago View changes

Good. Go ahead and merge

RufusBarbarossa merged commit c541be6 into master 16 days ago View details Revert

2 checks passed

More on Code Review in 17-313

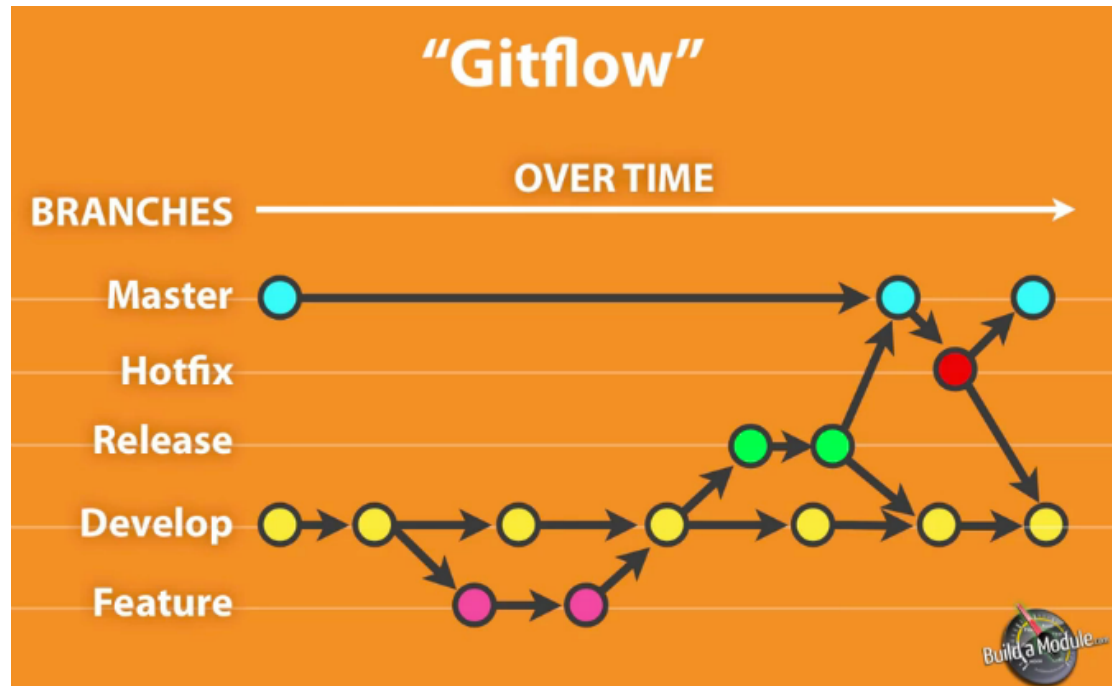
# DevOps Toolchain - Code

- Code development and review
- **Source code management tools**
- Code merging



# DevOps Toolchain - Code

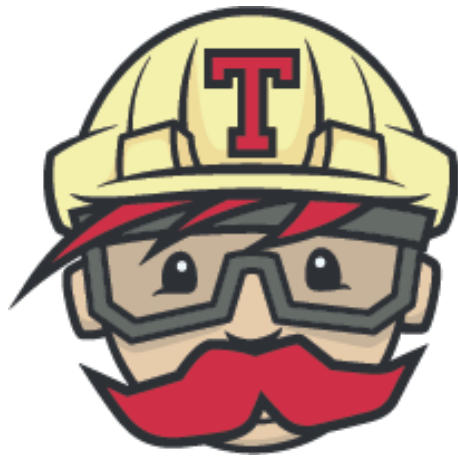
- Code development and review
- Source code management tools
- **Code merging**





# DevOps Toolchain - Build

- **Continuous integration tools**
- Build status



# Travis CI

# DevOps Toolchain - Test

- Continuous testing tools that provide feedback on business risks

- Continuous testing tools that provide feedback on business risks

Testing tools must  
have **tests** to be  
valuable!!