Microservices + DevOps

Charlie Garrod

Michael Hilton

School of Computer Science

institute for SOFTWARE RESEARCH

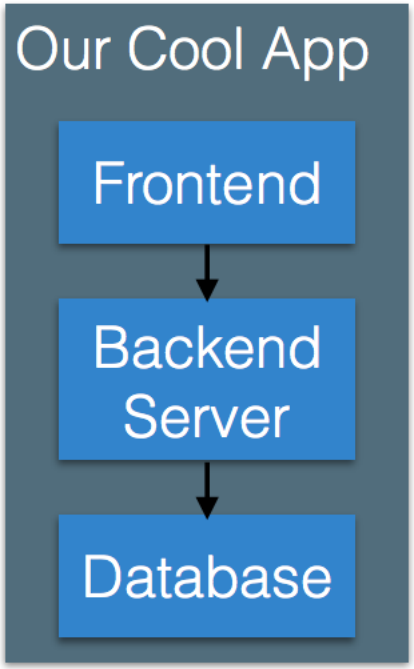
Administrivia

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

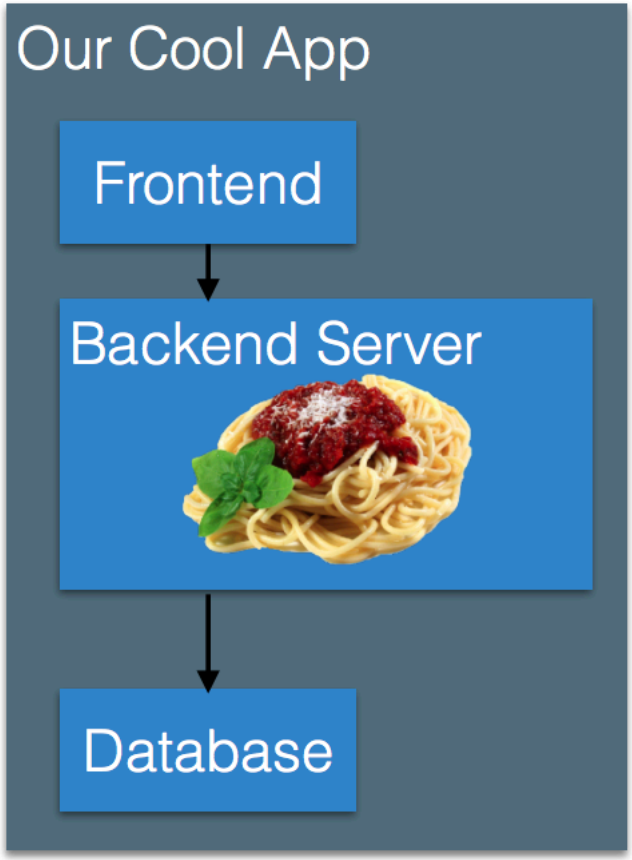
Last Time:

- Architectural styles

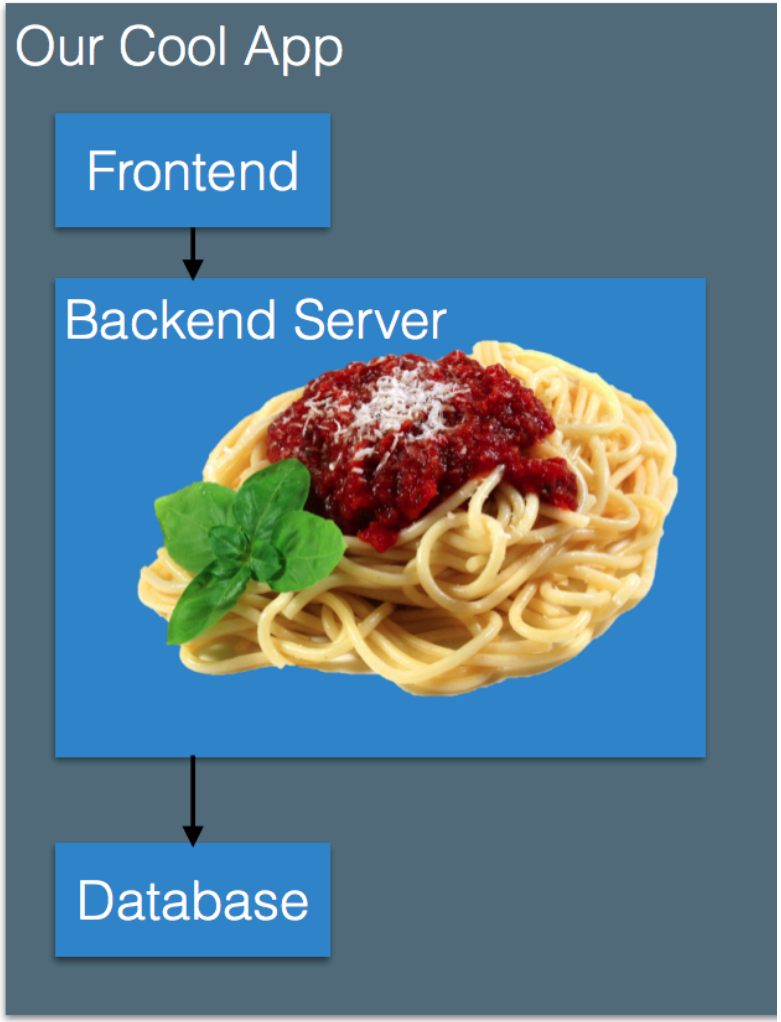
Simple Layers App



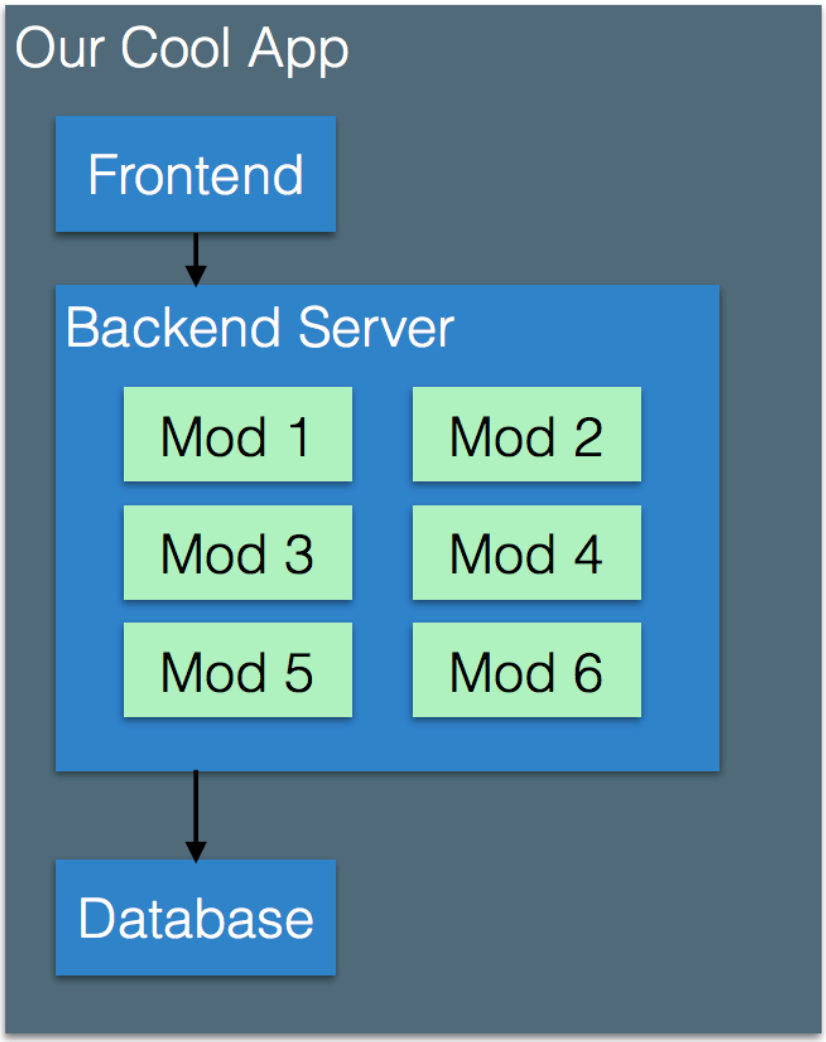
More functionality



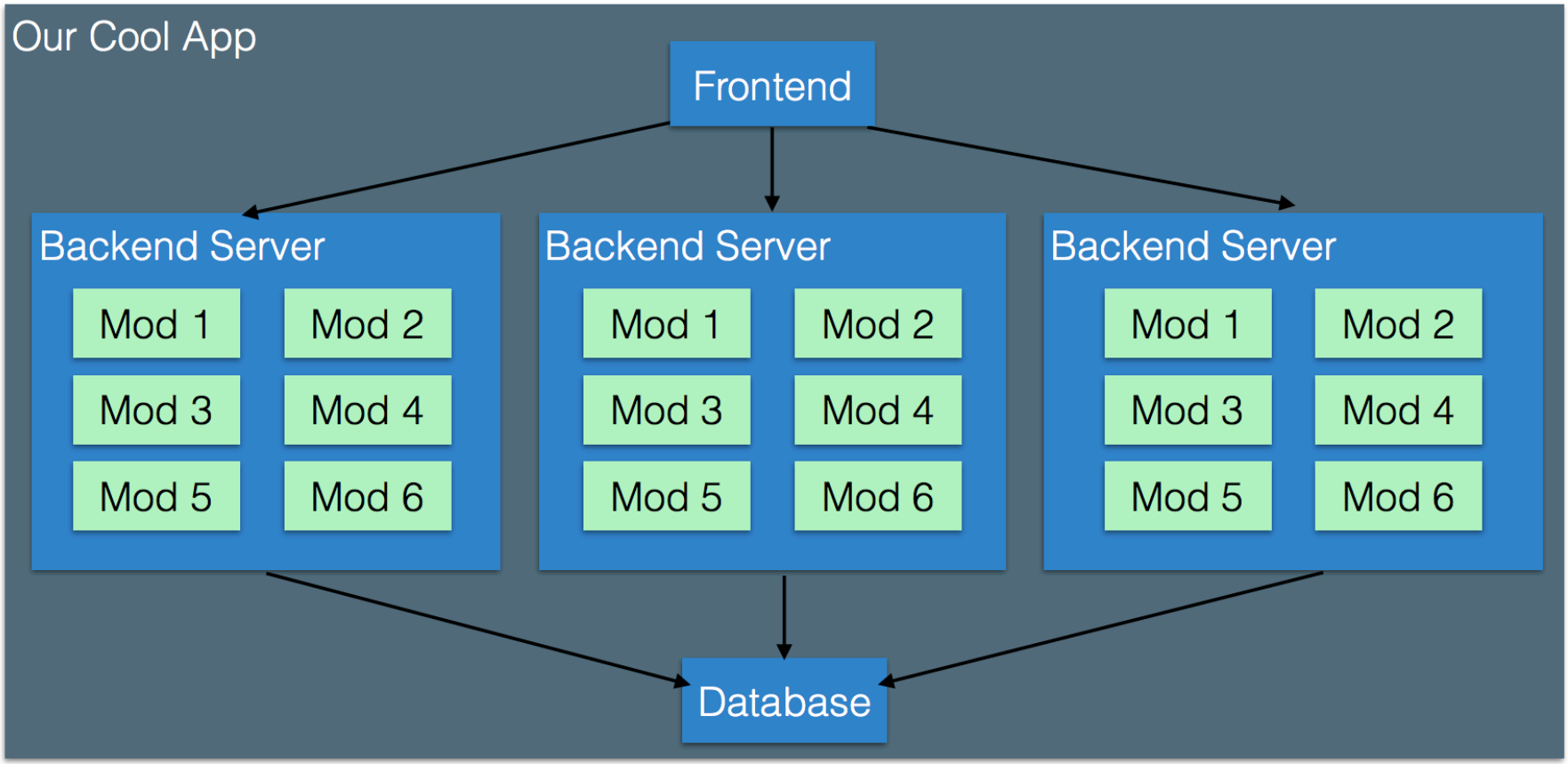
Even more functionality



Organize our backend

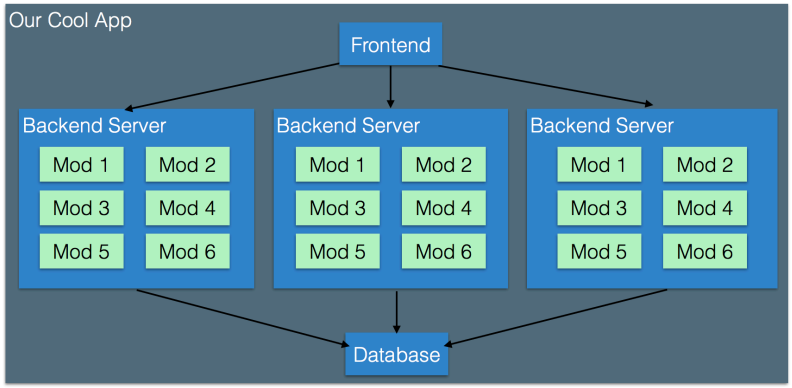


How to scale?

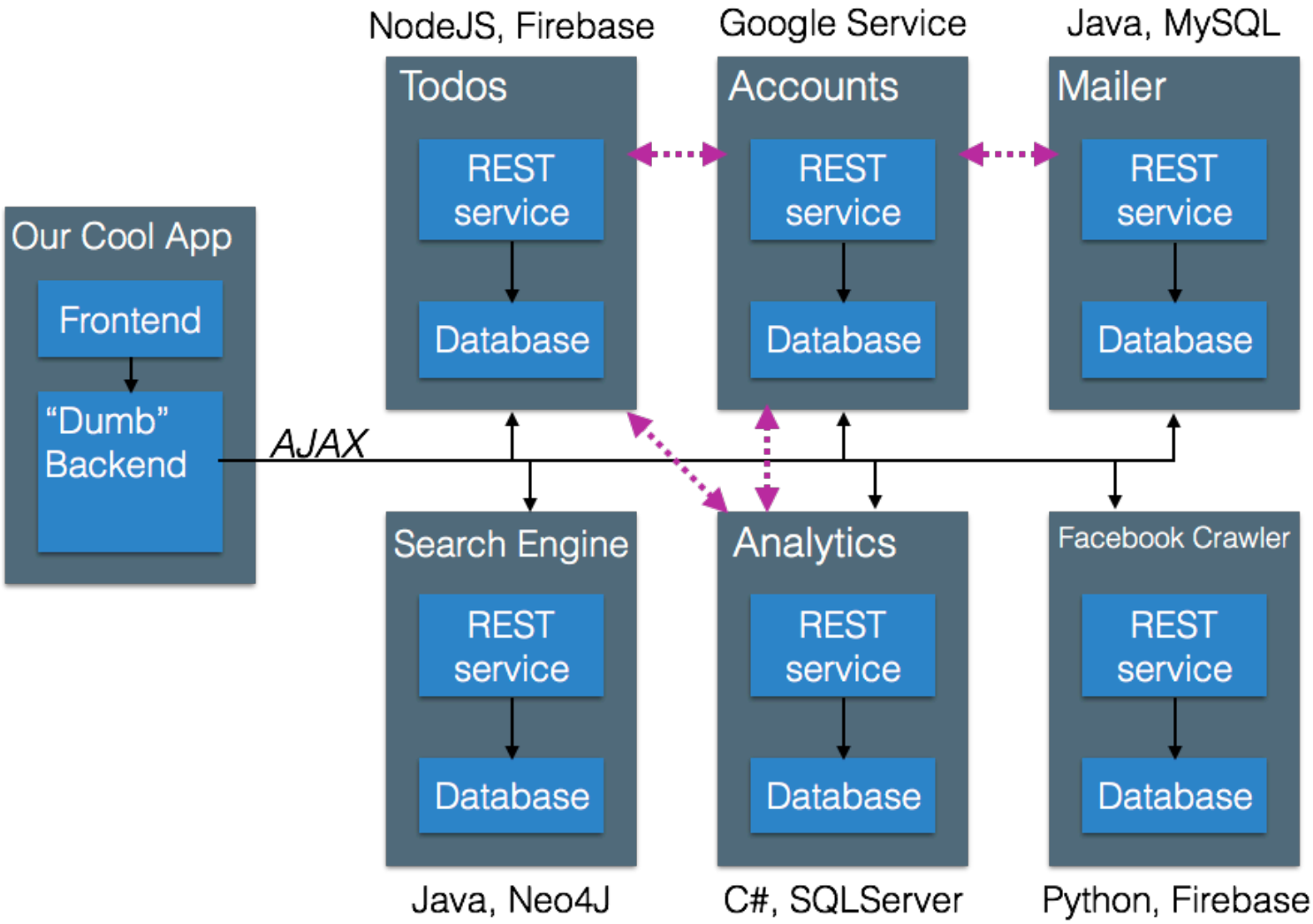

Our Cool App
Frontend
Backend Server
Mod 1
Mod 2
Mod 3
Mod 4
Mod 5
Mod 6
Backend Server
Mod 1
Mod 2
Mod 3
Mod 4
Mod 5
Mod 6
Backend Server
Mod 1
Mod 2
Mod 3
Mod 4
Mod 5
Mod 6
Database

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

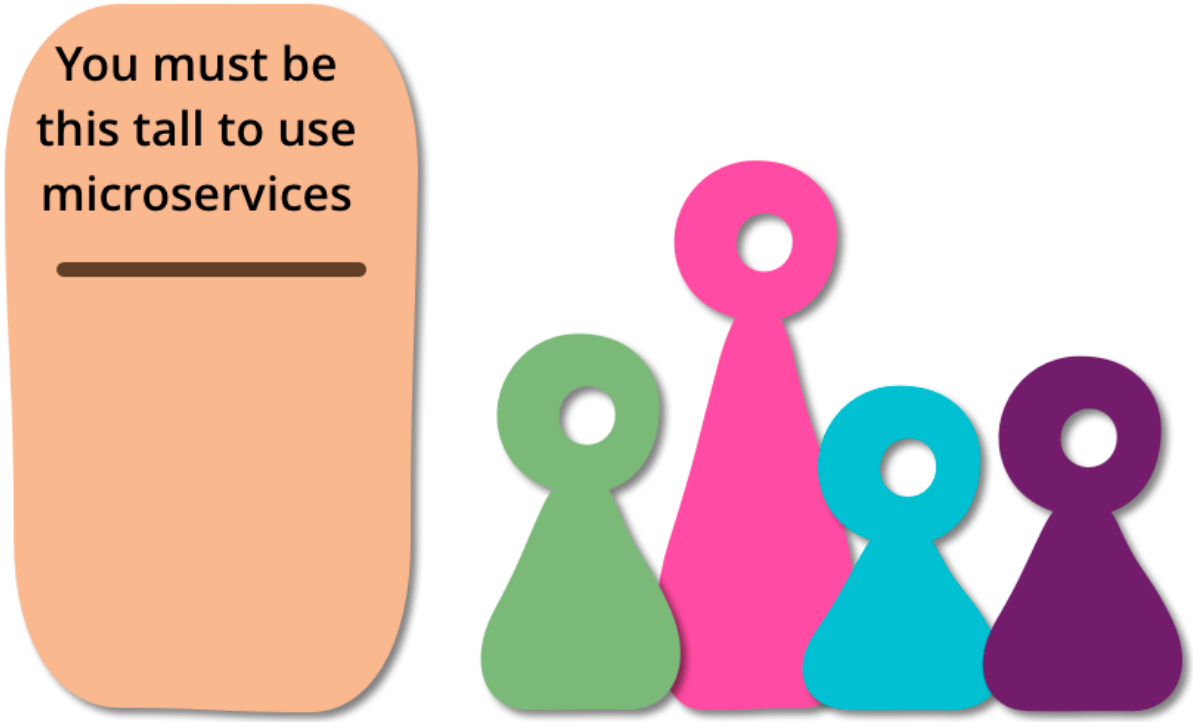


Microservices should be:

- 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




MICROSERVICES...
...SO HOT RIGHT NOW
memegenerator.net

Why are microservices such a big deal?

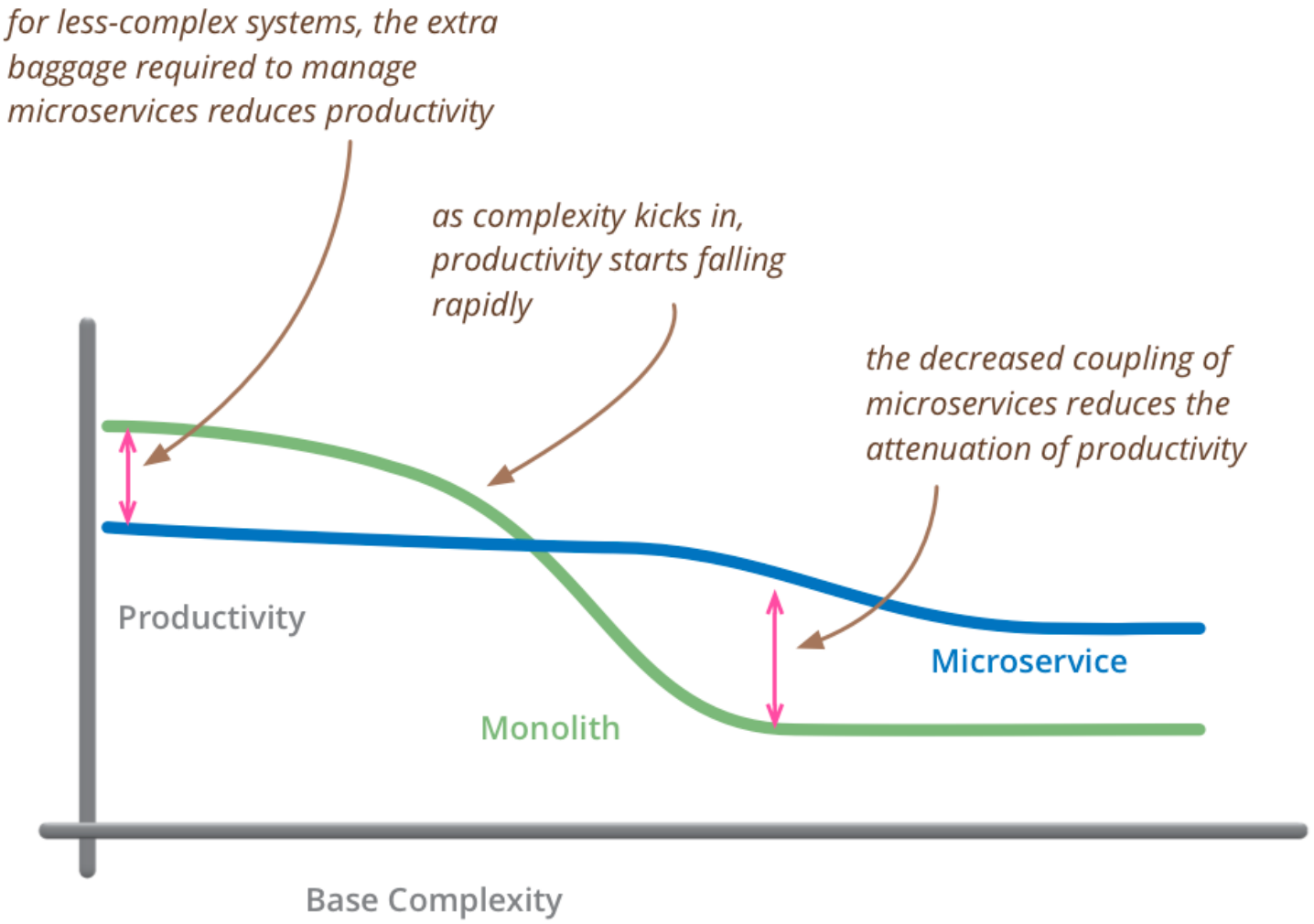
Impact on development practices

- 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



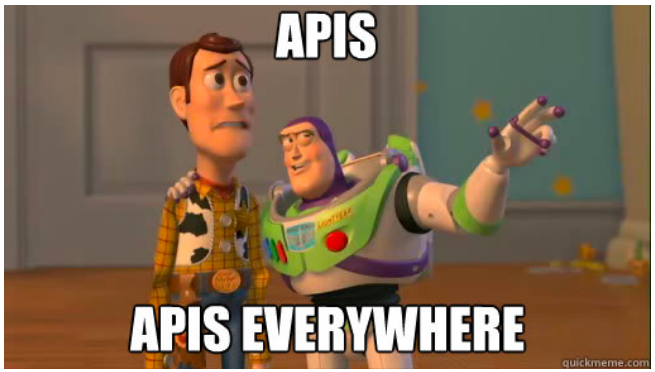
but remember the skill of the team will outweigh any monolith/microservice choice

Microservice costs

- Distribution
- Eventual Consistency
- Operational complexity

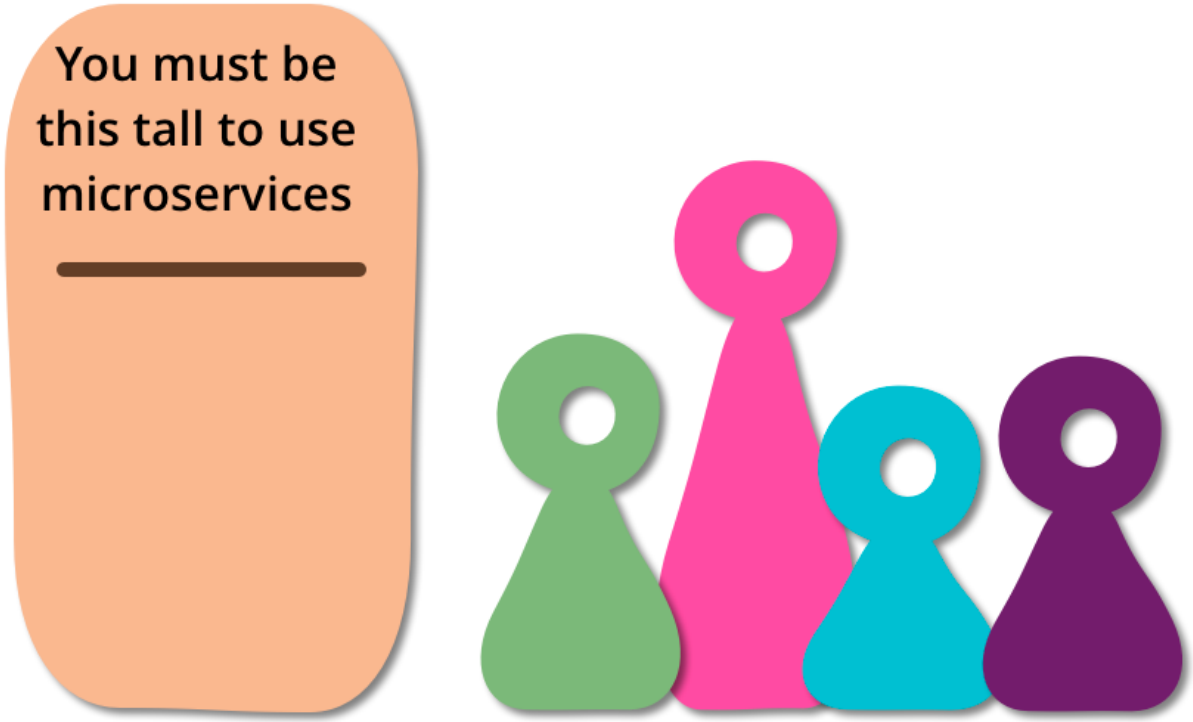
Discussion of Microservices

- 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**



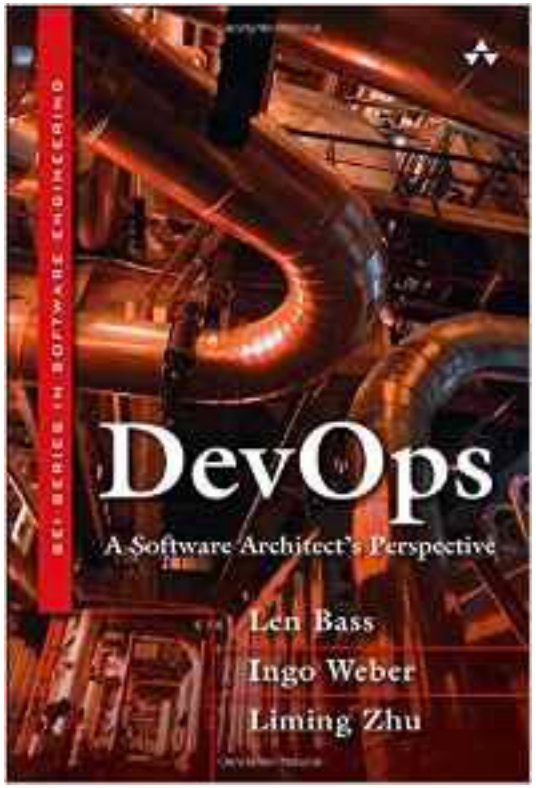
DEVOPS

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

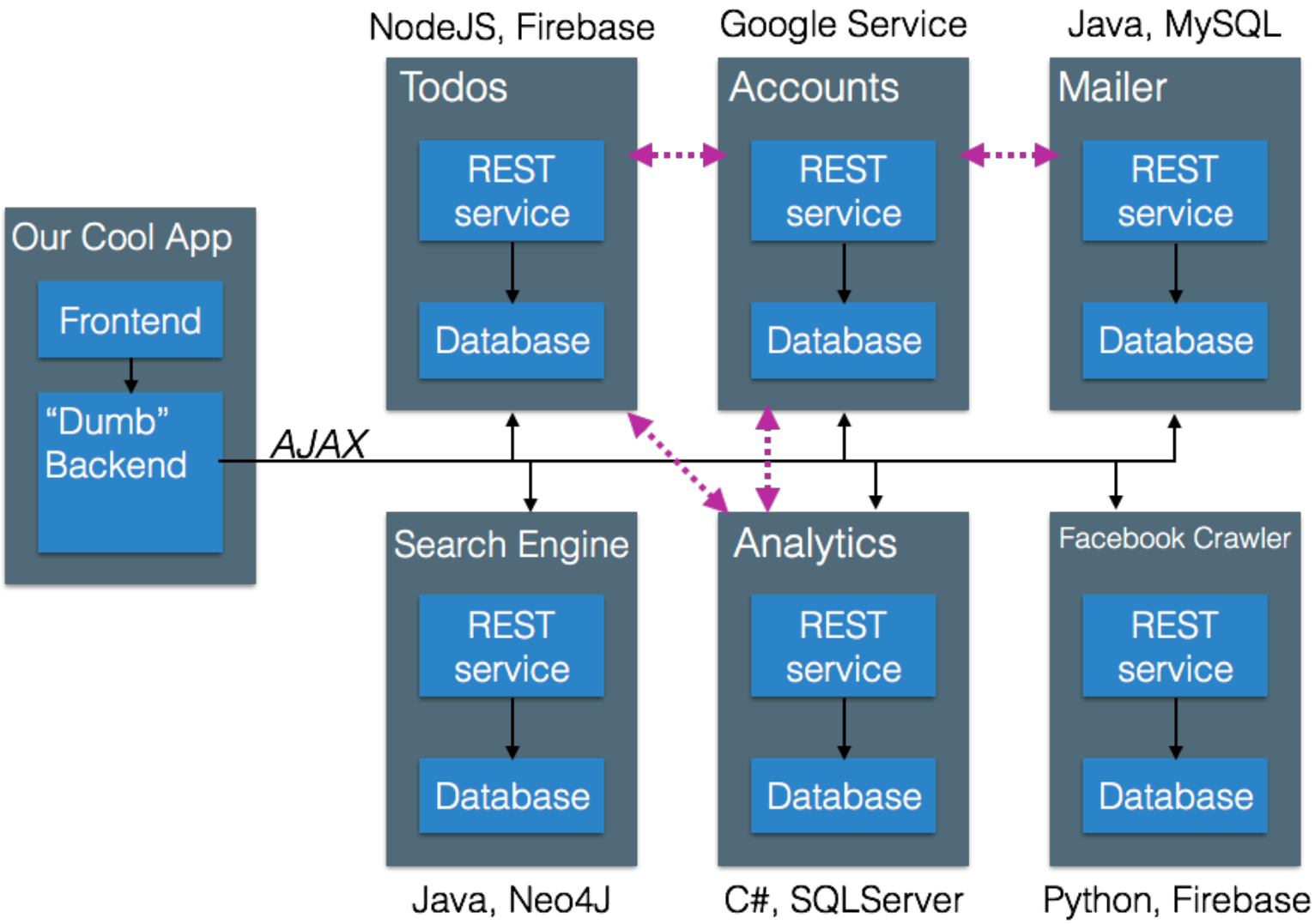
- "DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality."



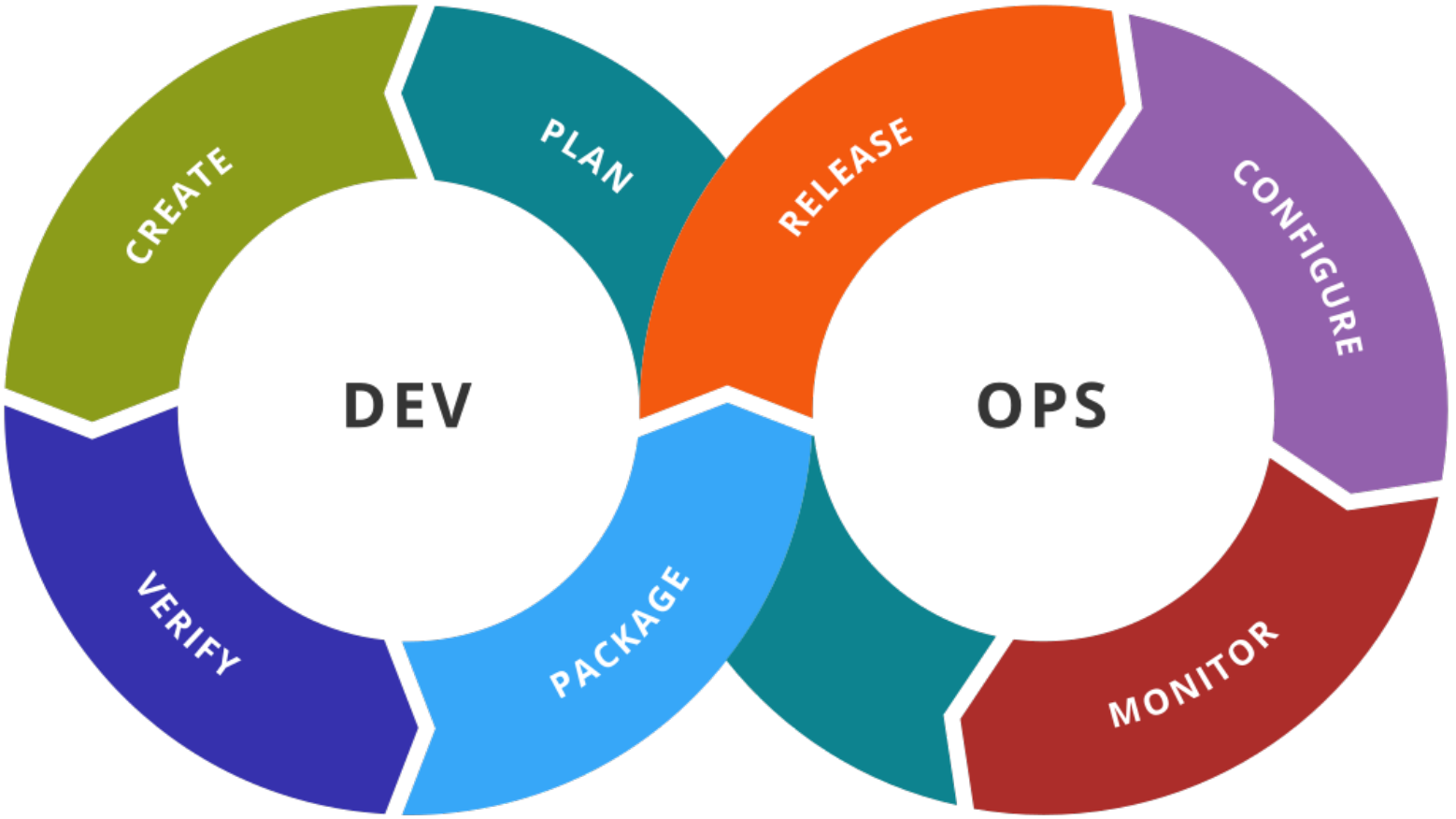
What are implications of DevOps?

- 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

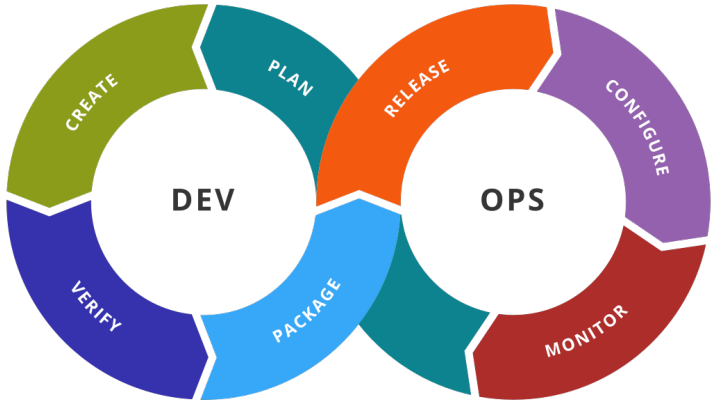


DevOps Toolchain



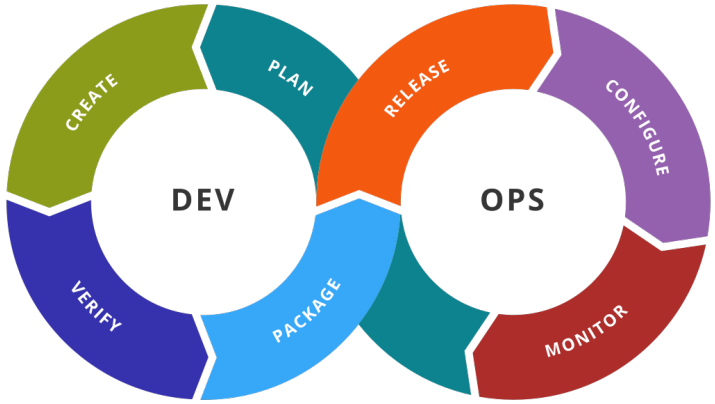
DevOps Toolchain

- 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



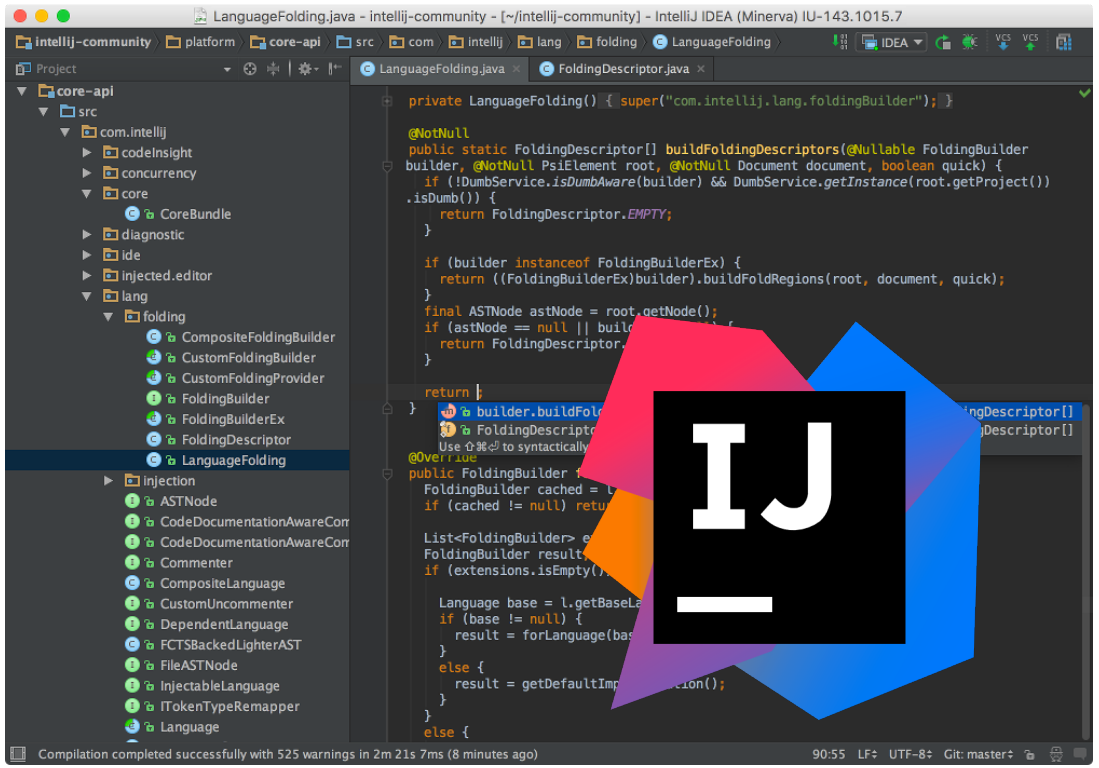
DevOps Toolchain continued

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



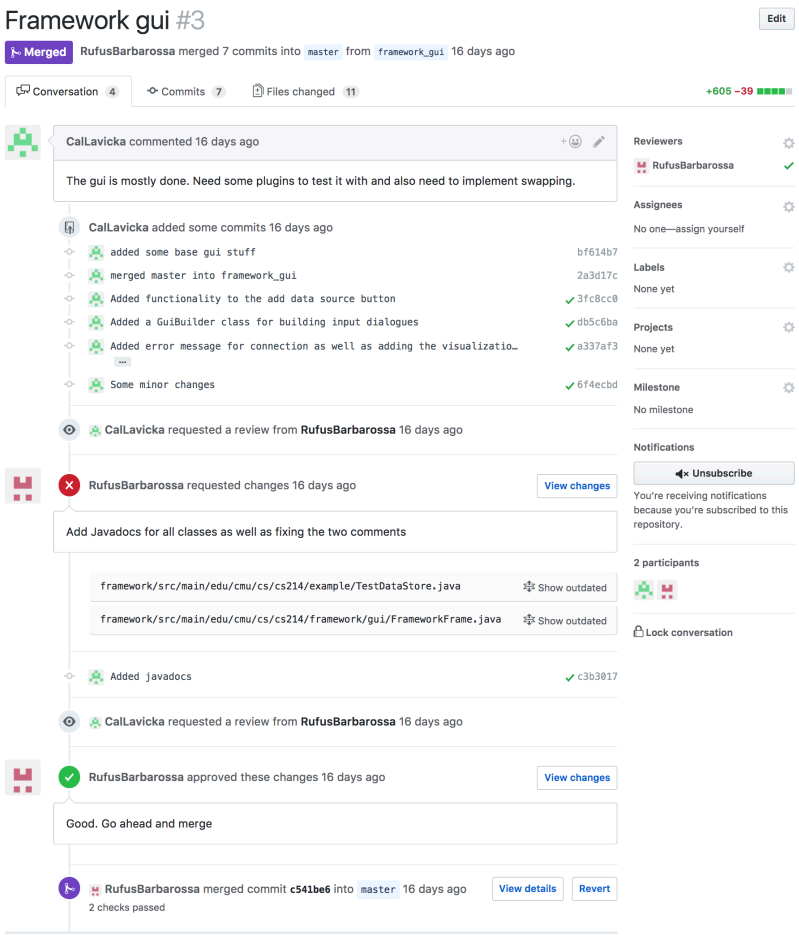
DevOps Toolchain - Code

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



DevOps Toolchain - Code

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



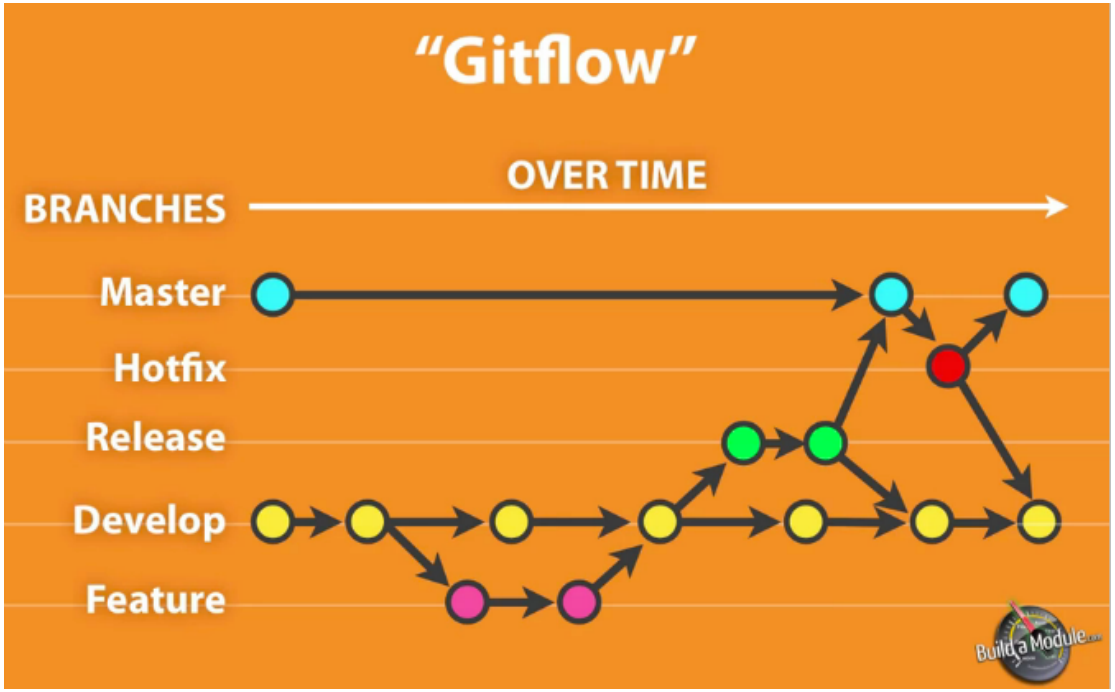
More on Code Review in 17-313

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

DevOps Toolchain - Test

- Continuous testing tools that provide feedback on business risks

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