

Kroger.com

Herding Cats at Mach 1

Stephen Shary

github.com/STeveShary

Twitter: @StephenShary

[linkedin.com/in/stephenshary](https://www.linkedin.com/in/stephenshary)

Myself

- Technical Lead
- Full-stack Developer
- Kroger Customer Facing Website
 - JVM (Java, Clojure)
 - JS (ES6).
 - AngularJS 1.x -> React
 - A bit of Golang
 - Databases: Cassandra, Oracle, Redis, Neo4j, MongoDB

What We Will Cover

- Show our growth and Delivery
- Discuss team structure
- Our architecture
- Culture patterns and anti-patterns

Our Start: 2013

- Major push to overhaul site.
- Add E-com (ClickList)
- Add Pharmacy
- Add Shopping List

Our Start: 2013

- Started with 3 teams: Site, List, Pharmacy.
- ~20 people
- Sharepoint -> AngularJS ~1.0.6
- Embedded Tomcat Server
- Java 1.7
- Completed 2 production releases

Now: 2017

- 30+ teams
- 185+ Developers
- 60+ applications
- AngularJS -> React
- New PAAS.
- Prod: ~150 release per month. (1,800 per year)
- Non-Prod: ~1,200 releases per month. (14,000 per year)

Ohh yeah, don't forget

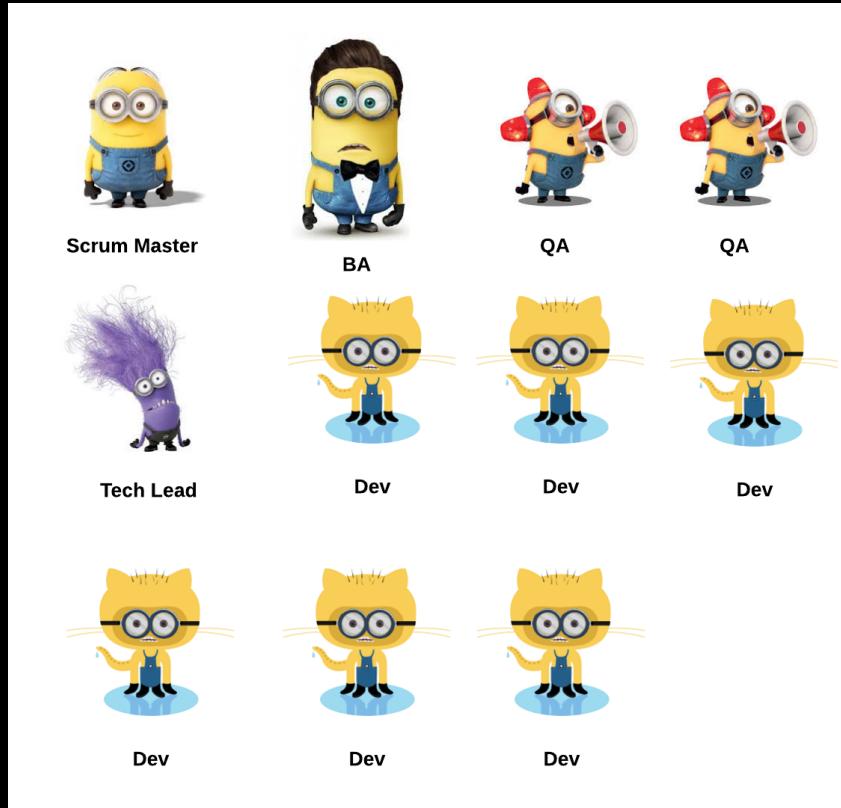
- 24/7 uptime.
- Billion Dollar E-com site.
- Over 1 Billion Coupons per year.
- Tens of millions of registered users.

Teams



Teams: Structure

- “two pizza team”
 - Scrum Master
 - Business Analyst
 - 1 to 2 Testers
 - Tech Lead
 - 4 to 8 Devs



Team Traits

- Own their apps.
- Own their support.
- Choose their release schedule.
- Choose their tech stack.
- Choose their name.

(Some)Team Names

Star Lord

Fury Road

Pharmasaurus Rex

Groot

Ironman

Batman

Punisher

Ronin

Thor

Odin

Night's Watch

Pinky

Pied Piper

Voltron

Good Eats

Old Monks

Brute Squad

Cat Rodeo

Swat Kats

Misfit Toys

Night Fury

Sons of Anarchy

Damaged Sentinels

Our Office

- Google Maps: “Kroger at the Landings”



Team Types

- Two types of teams:
 - Vertical (Owns a domain)
 - Horizontal.
- Inspired by Spotify Model

Vertical Team

- Owns the whole stack.
 - Front End: HTML, CSS, JS
 - Backend: BFF, Web Apps
 - Database
- Owns whole SDLC
 - Dev, Test, Deploy, Support

Horizontal Team

- Provides services to vertical teams.
- Owns common apps
- 4 Teams
 - Ronin - Echo
 - Punisher - Core Engineering
 - Shazam - Core Infrastructure
 - Damaged Sentinels - Core Architecture

YEAH....

HOW DOES IT REALLY WORK?

ONE MONOLITH



TO RULE THEM

ALL

JK

LOL



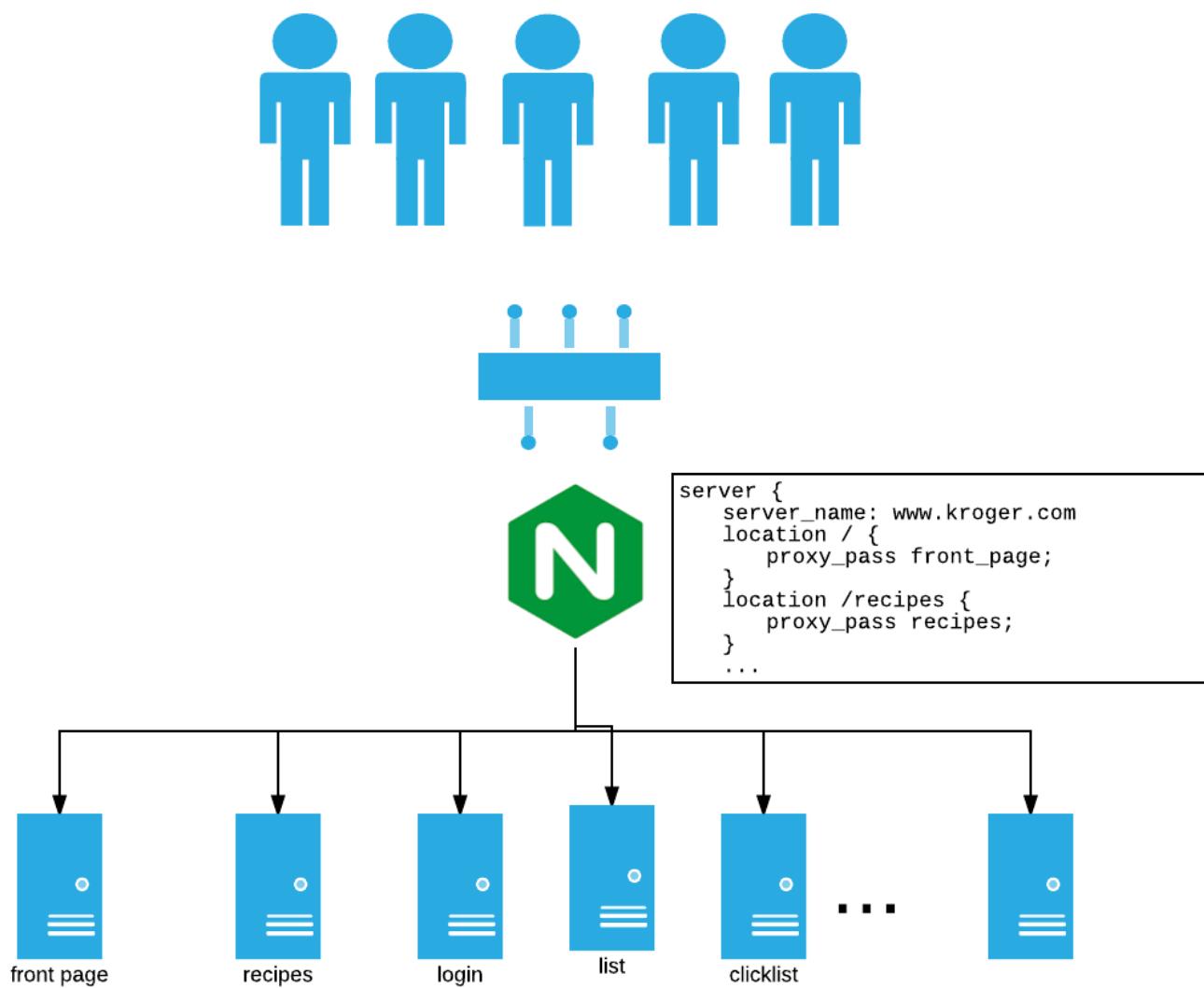
Our Architecture

- Applied Conway's Law

Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations.

But It's Single Site!

- We have architecture that helps.
 - Reverse Proxy
 - We use Nginx
 - Full CI/CD pipeline with 4,000+ e2e tests.



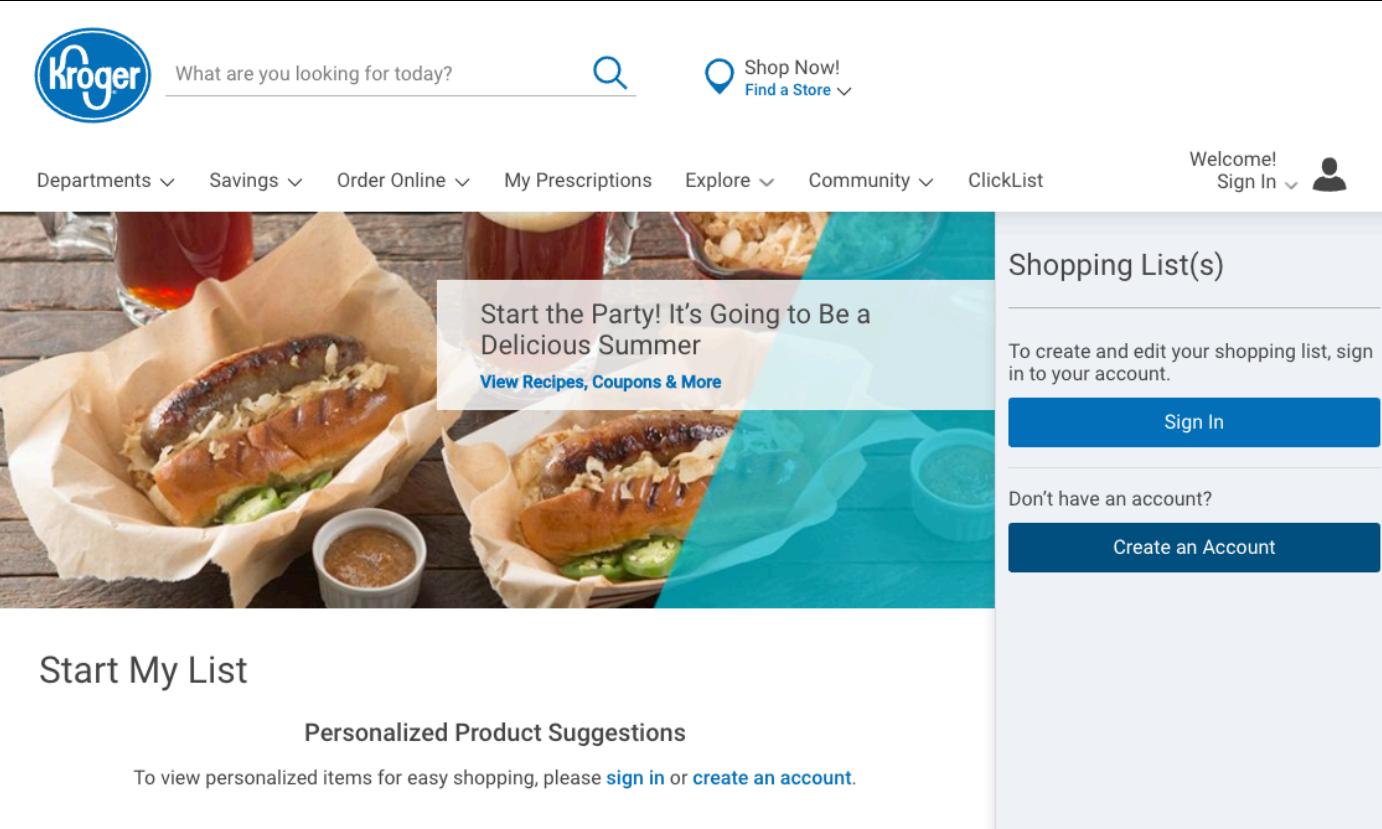
Orchestration

- Software Define Networking
- HA clustering with auto-restart
- Auto-scaling
- Allows
 - Blue/Green
 - Dark

Yeah... But what about

- Compatibility?
- Consistency?

Home Page



The screenshot shows the Kroger website homepage. At the top left is the Kroger logo. To its right is a search bar with the placeholder "What are you looking for today?". Next to the search bar is a magnifying glass icon. To the right of the search bar is a "Shop Now!" button with a location pin icon, followed by "Find a Store" and a dropdown arrow. On the far right of the top header is a "Welcome!" message, a "Sign In" link, and a user profile icon.

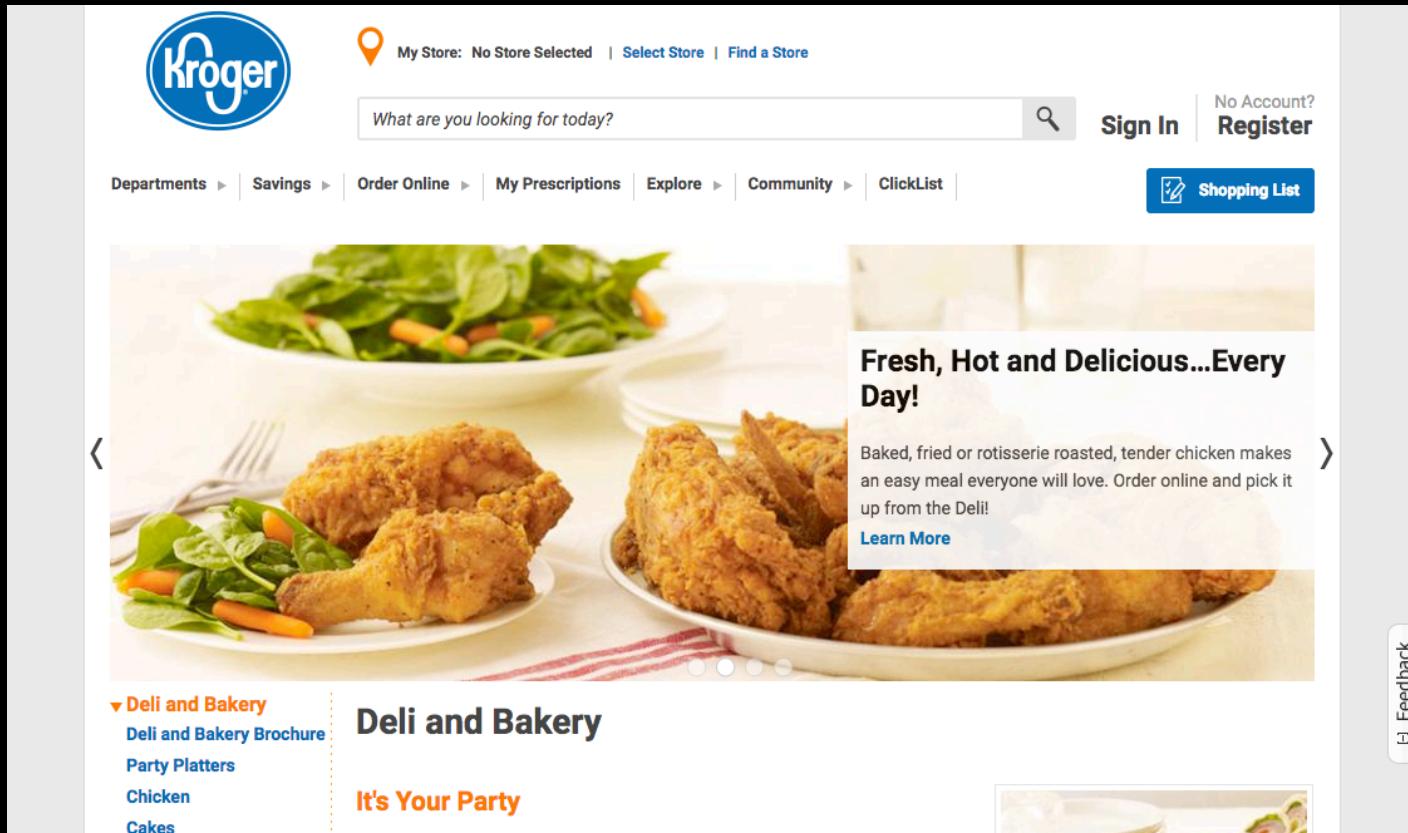
Below the header, there is a navigation menu with links: Departments, Savings, Order Online, My Prescriptions, Explore, Community, and ClickList.

The main content area features a large image of a hot dog and a drink, with a callout box overlaid containing the text "Start the Party! It's Going to Be a Delicious Summer" and a "View Recipes, Coupons & More" link.

On the right side, there is a sidebar titled "Shopping List(s)". It contains a message: "To create and edit your shopping list, sign in to your account." Below this are two buttons: a blue "Sign In" button and a white "Create an Account" button.

At the bottom left, there is a large, bold "Start My List" button. Below it is a section titled "Personalized Product Suggestions" with a note: "To view personalized items for easy shopping, please [sign in](#) or [create an account](#)".

Hey.... Is that different?



The screenshot shows the Kroger website homepage. At the top, there's a navigation bar with the Kroger logo, a store locator, a search bar containing "What are you looking for today?", and links for "Sign In" and "Register". Below the navigation is a horizontal menu with links for "Departments", "Savings", "Order Online", "My Prescriptions", "Explore", "Community", and "ClickList". To the right of this menu is a "Shopping List" button. The main content area features a large image of a meal consisting of fried chicken and a salad. To the right of the image is a promotional text block: "Fresh, Hot and Delicious...Every Day!" followed by a description of baked, fried, or rotisserie roasted chicken, and a "Learn More" link. At the bottom left, there's a sidebar for "Deli and Bakery" with links to a brochure, party platters, chicken, and cakes. The bottom right corner has a "Feedback" link.

My Store: No Store Selected | Select Store | Find a Store

What are you looking for today?

Sign In | No Account? Register

Departments ► Savings ► Order Online ► My Prescriptions Explore ► Community ► ClickList ► Shopping List

Deli and Bakery

▼ Deli and Bakery
Deli and Bakery Brochure
Party Platters
Chicken
Cakes

It's Your Party

Fresh, Hot and Delicious...Every Day!

Baked, fried or rotisserie roasted, tender chicken makes an easy meal everyone will love. Order online and pick it up from the Deli!

[Learn More](#)

Feedback

Culture points

- Bring the pain forward.
- Monolith first.
- Reduce risk.
- Automate Everything.
- Make the best path the easiest.

Bring the Pain Forward

- Fix the problem.
- Don't mask the symptoms.
- What is pain?
 - Paperwork
 - Downtime
 - Manual Testing
 - Process ceremonies

Monolith First

- Don't make micro services a buzzword
- Split when the team is too big.
- Split when the app crosses two domains.
- Split when the pipeline is too slow.
- Split when you don't know all the code in one app.

Reduce Risk

- Small changes are better.
- Practice deployments.
- Spend time on backwards compatibility.
- Expect to roll back.
- Toggles.
- QA in Prod (Dark)

Automate Everything

- Bring the pain forward
 - Deployments
 - Tests
 - Workflows
 - Logging -> Metrics -> Feedback
 - Provisioning
 - Timesheets?... Expense Reports?

Make the Best Option the Easiest

- Developers are lazy.
- We choose the wrong tool to get done.
- Recommend, don't mandate.
- Look at adoption rates.

How to fail

- Make it manual (automation doesn't pay off)
- Deploy together
- Make deployments = downtime
 - Over night deployments
- The process requires meetings
- The beatings will continue until stability improves.

Questions?

Thank You

Stephen Shary

github.com/STeveShary

Twitter: [@StephenShary](https://twitter.com/@StephenShary)

[linkedin.com/in/stephenshary](https://www.linkedin.com/in/stephenshary)