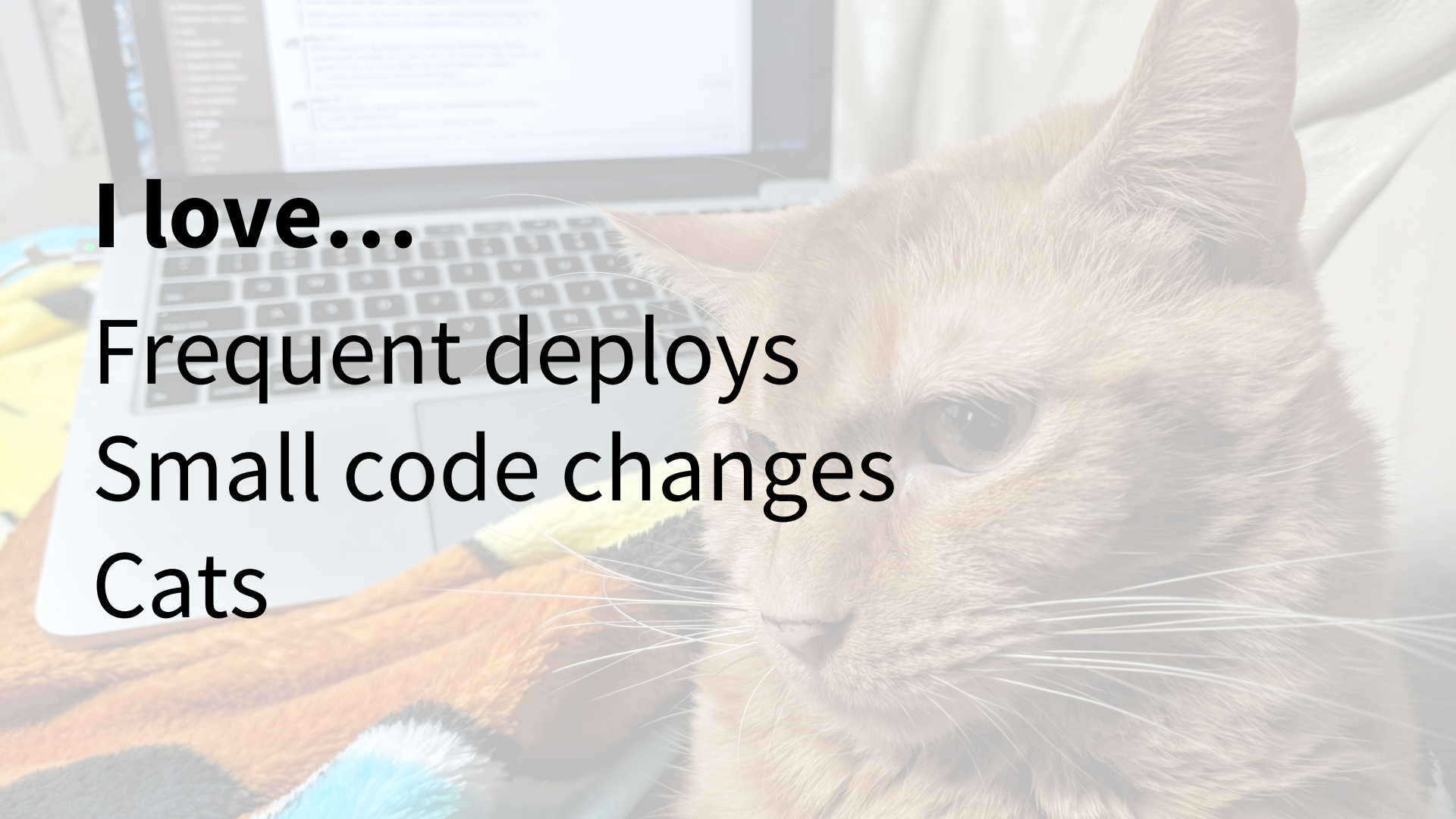
Slaying the Long-Lived Branch

Kat Fairbanks

Director of Engineering
Everything But The House



I love...

Frequent deploys

Small code changes

Cats

I hate…

Risky situations

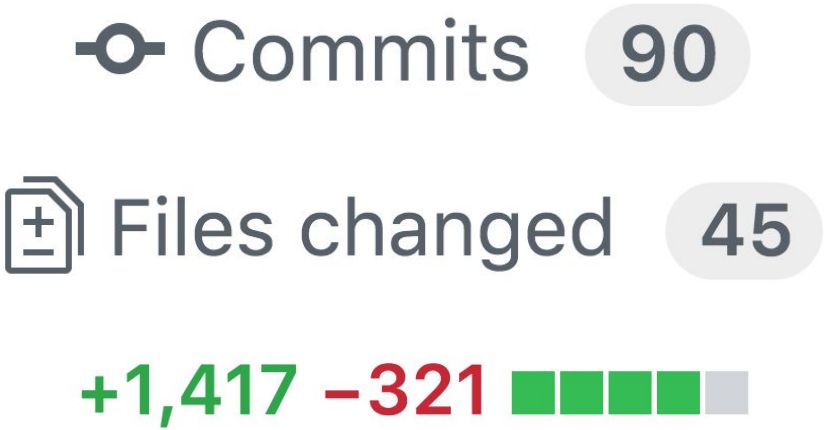
Long-lived branches

Long-lived branches

- Work done over an extended period of time (several days - weeks)
- Has a significant number of changes

1 developer
1 designer
3 weeks

1 QA resource
1 reviewer (me)



View all items from Antiques, Collectibles, Décor & More sale

Arita Japan Porcelain Tea Set with an Andrea by Sadek Bowl

● 6 MINUTES LEFT

October 6th 2018 @ 8:08pm EST

CURRENT BID

$10 10 Bids

PLACE BID

DELIVERY OPTIONS

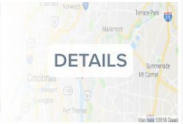
More Info

Ship To Me
$17.31

Deliver to **45202** EDIT ZIP

Pick It Up
Free

Cincinnati, OH
Wednesday, October 10, 1-6pm



EBTH
EVERYTHING BUT THE HOUSE

THE PREMIER ESTAT

Sell With Us | Following 1 | My Info

Search our collection of unique items

OPULAR | BOUTIQUES | FEATURED

CURRENT BID
$10

TIME LEFT
6M 22S

How do you want to get this item if you win?

Ship To Me
$17.31
105 E Fourth St Floor 9 Cincinnati, OH 45202
EDIT

Pick It Up
Free
Cincinnati, OH
Wednesday, October 10, 1-6pm
DETAILS

CONTINUE

es, Décor & More sale

ain Tea Set with an
Bowl

October 6th 2018 @ 8:08pm EST

$10 10 Bids

PLACE BID

DELIVERY OPTIONS

More Info

Ship To Me
$17.31
Deliver to 45202
EDIT ZIP

Pick It Up
Free
Cincinnati, OH
Wednesday, October 10, 1-6pm
DETAILS

EBTH
EVERYTHING BUT THE HOUSE

THE PREMIER ESTAT

Sell With Us | Following 1 | My Info

Search our collection of unique items

POPULAR | BOUTIQUES | FEATURED

es, Décor & More sale

ain Tea Set with an
Bowl

October 6th 2018 @ 8:08pm EST

PLACE BID

More Info

EDIT ZIP

Pick It Up Cincinnati, OH
Free Wednesday, October 10, 1-6pm

DETAILS

CURRENT BID
$10

TIME LEFT
6M 22S

Your Bid

$12

Current Bid + $2 bid increments

Auto Bid (optional)

$

Maximum you're willing to bid

Pick It Up - FREE
Cincinnati, OH – Wednesday, October 10, 1-6pm
EDIT

VISA Visa ending in 1234
EDIT

☐ I agree to pay **$12** for this item if I win.

PLACE BID

By bidding, you acknowledge that this item is not returnable and sold "as is". Failure to pickup within the designated timeframe will result in shipment at the buyer's expense or forfeiture without reimbursement. Terms & Conditions

Changes included — a few refactors

- Refactored state management
- Refactored shipping/quoting components

Changes included — several features

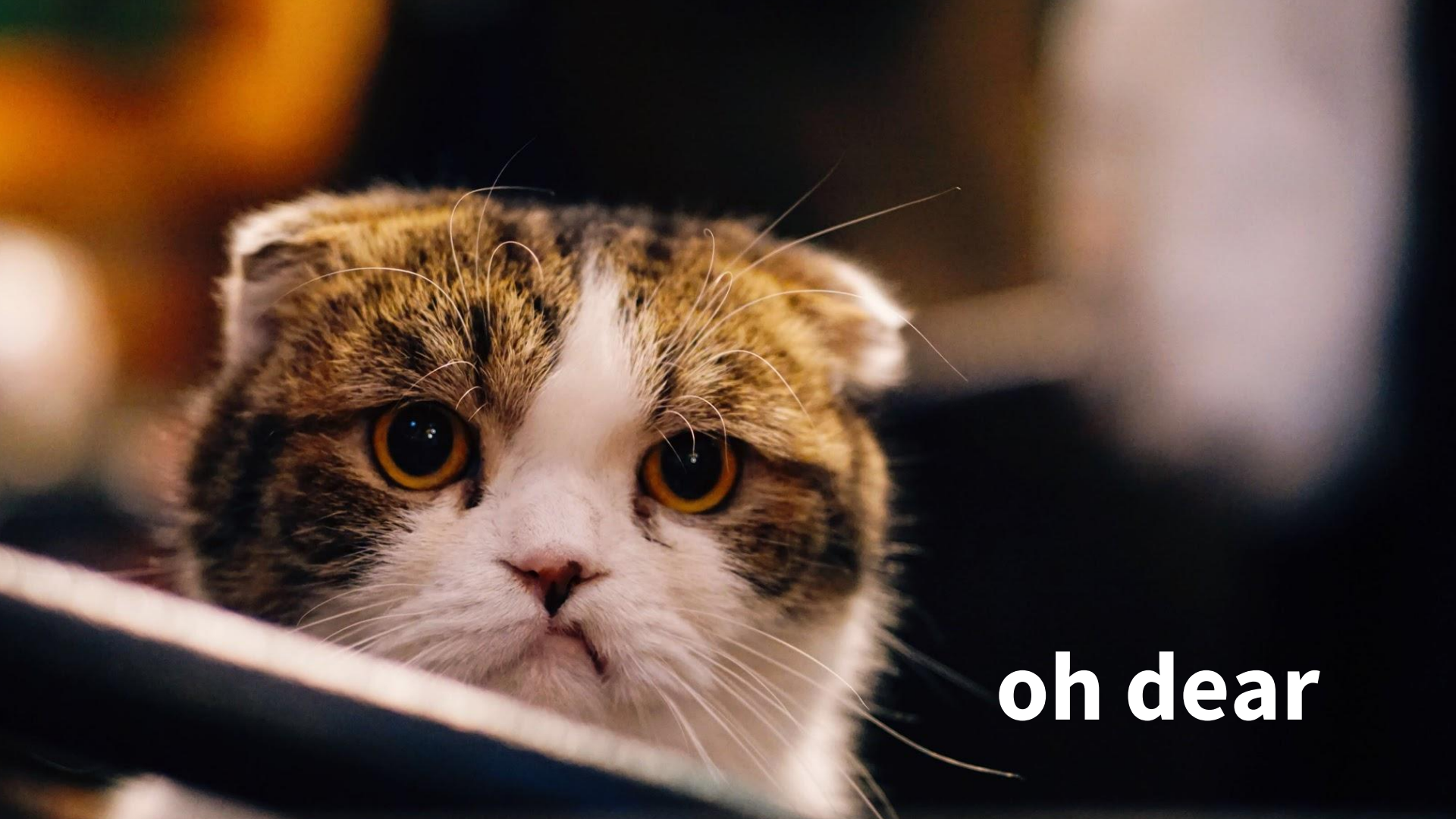
- Users are now required to select a shipping address
- Change address selection from a dropdown to a dedicated page of the bidding modal
- Do not allow selection of “undeliverable” addresses, prompt user to add an address if no shippable addresses are on the account

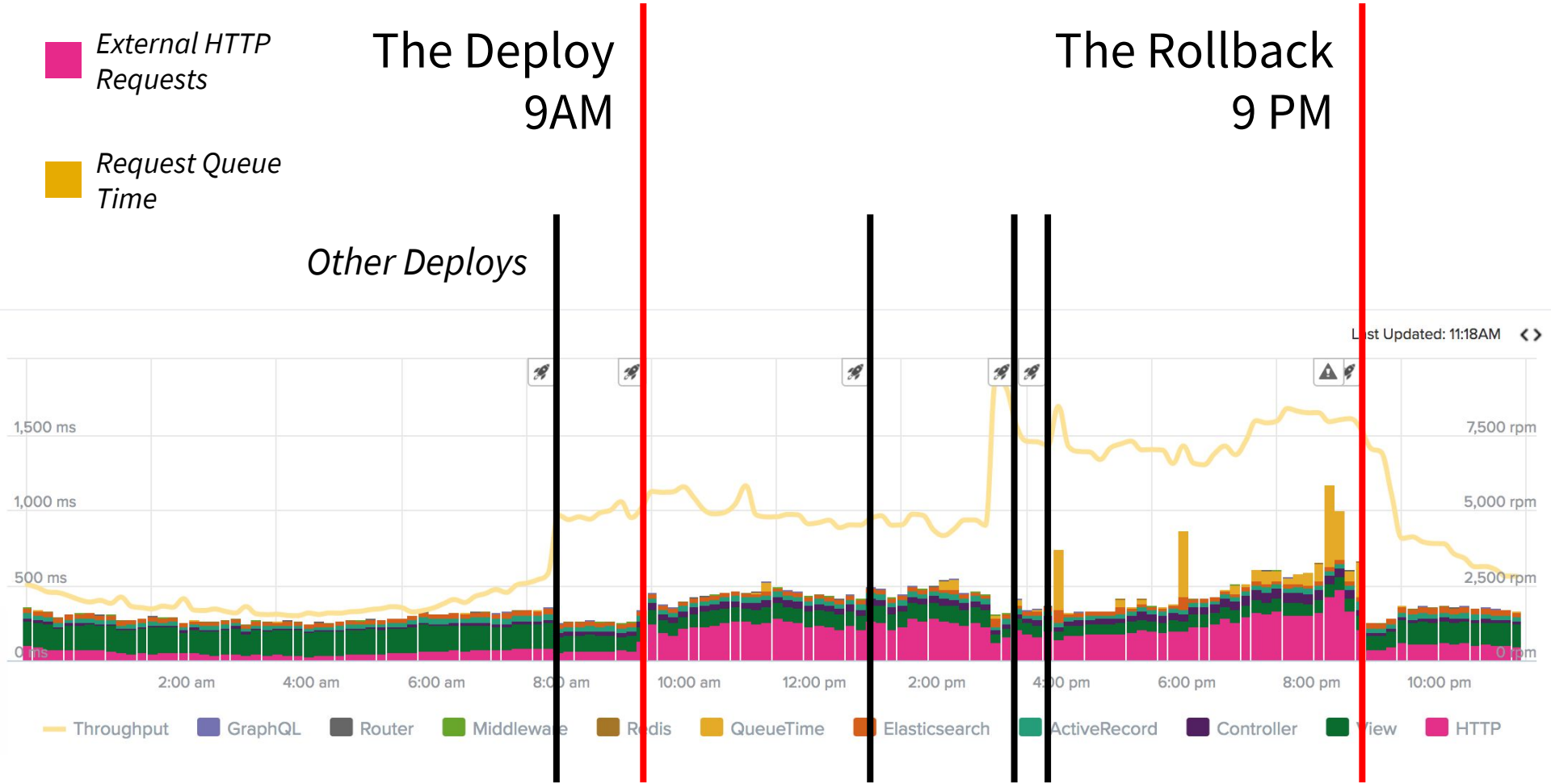
Changes included — several features

- Show a message if max bid goes over $5000 (we do not accept credit cards over $5k)
- Updates autocheckout to support shipping address preference
- Group invoices by address (We batch invoices by certain properties, like sales and fulfillment preferences)


Ready...
Aim...
Deploy!

We are seeing a huge spike in traffic to the site, starting around 3:30 PM. There was a deploy around that same time, not sure if it is related.

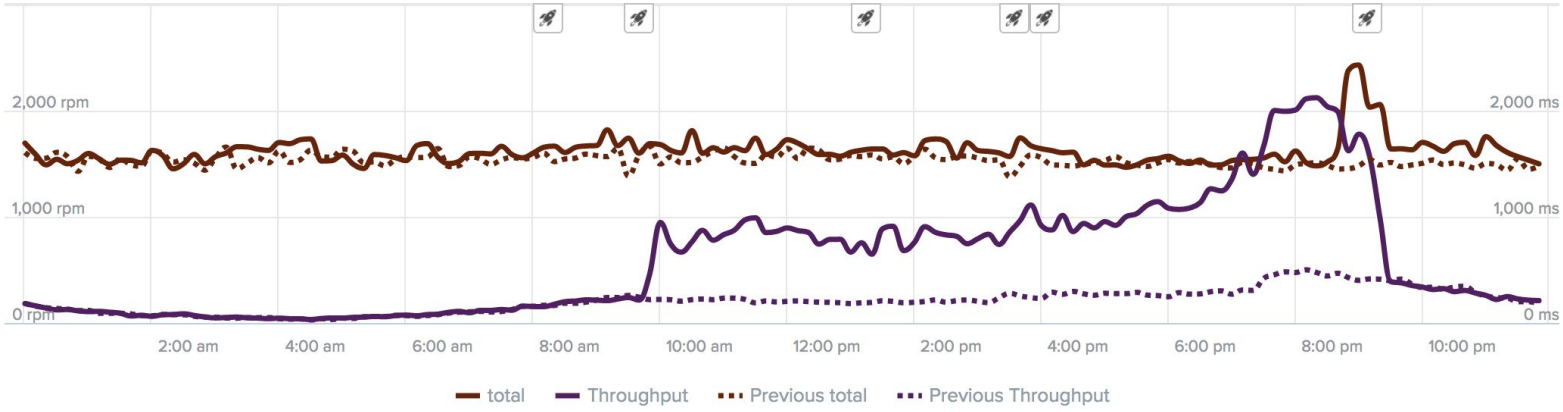

oh dear


External HTTP Requests
Request Queue Time
The Deploy
9AM
The Rollback
9 PM
Other Deploys
Last Updated: 11:18AM
1,500 ms
1,000 ms
500 ms
0 ms
7,500 rpm
5,000 rpm
2,500 rpm
0 rpm
2:00 am
4:00 am
6:00 am
8:00 am
10:00 am
12:00 pm
2:00 pm
4:00 pm
6:00 pm
8:00 pm
10:00 pm
Throughput
GraphQL
Router
Middleware
Redis
QueueTime
Elasticsearch
ActiveRecord
Controller
View
HTTP

TL;DR

Number of requests to our shipping quotes almost tripled after this pull request was deployed at 9:30am.

Servers were keeping up — until we hit peak traffic hours.



Problems encountered

Difficult to give feedback

- Most developers already hate reviewing code
- Large pull requests are even more painful to review
- It is easy to miss important details
- I felt guilty asking for changes because we would need to test the entire feature again

Reduces communication

Working on large, long lived changesets; a practice I like to call “continuous isolation”

— Benji Weber

Reduces communication

Missing timely feedback from teammates

Reduces communication

The team is missing communication from *you*

Reduces communication

Missing timely feedback from production

How do we
fight back?

Remove distractions

- It is tempting to add small changes to a Pull Request, but it can snowball quickly
  - Refactoring code should be done separately from feature changes — refactoring should not change any functionality
  - Small bug fixes can be separated from the feature work
- There is a tradeoff between managing several Pull Requests and maintaining clear communication with your team

Developing in production

Case Study at GitHub - Upgrading GitHub from Rails 3.2 to 5.2

https://githubengineering.com/upgrading-github-from-rails-3-2-to-5-2/

- The project took a year and a half to complete
- There was no dedicated team - engineers did what they could during their free time
- They used conditional statements to write code for different versions of Rails

Developing in production

```
if GitHub.rails_3_2?
  # 3.2 code (i.e. production a year and a half ago)
elsif GitHub.rails_4_2?
  # 4.2 code
else
  # all 5.0+ future code, ensuring we never
  # accidentally fall back into an old version
  # going forward
end
```

Developing in production

Case Study at GitHub - Upgrading GitHub from Rails 3.2 to 5.2

https://githubengineering.com/upgrading-github-from-rails-3-2-to-5-2/

Do incremental upgrades. Each minor version of Rails provides the deprecation warnings for the next version. By upgrading from 3.2 to 4.0, 4.0 to 4.1, etc we were able to identify problems in the next version early and define clear milestones.

Developing in production

Like how GitHub used conditionals to activate different code, conditionals can be used to show/hide new features

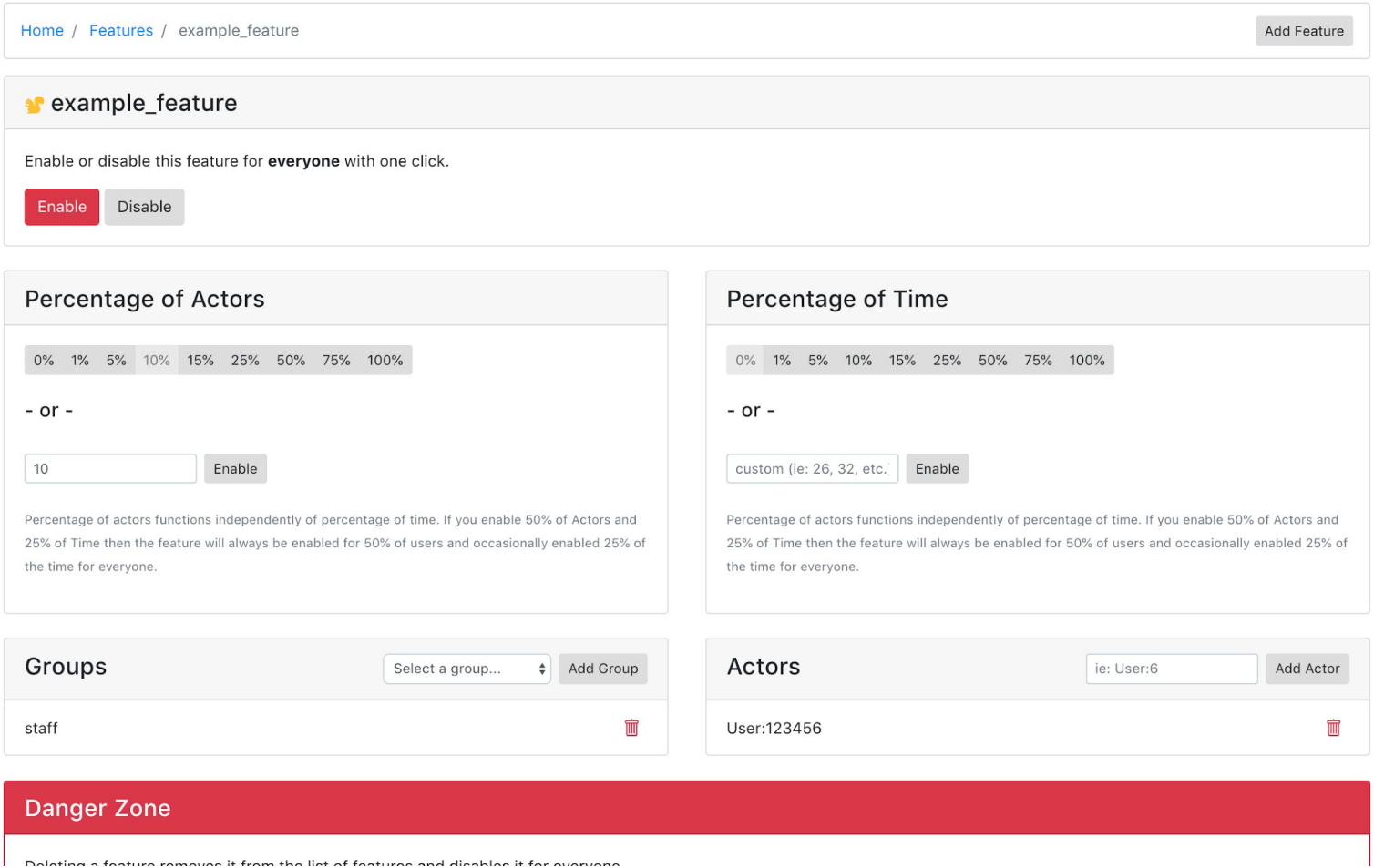
```
use_new_sort = false

if use_new_sort
  new_sort_items(items)
else
  sort_items(items)
end
```

Developing in production

We can take this one step further using runtime configuration – allowing us to turn features on and off without changing anything in our code. These are called **Feature Flags** or **Feature Toggles**.

```
if FeatureFlags.enabled?(:new_item_sort)
  new_sort_items(items)
else
  sort_items(items)
end
```



https://github.com/jnunemaker/flipper

Feature flags

An **actor** is any kind of object that is passed into Flipper, usually a **User** or an **Item** at EBTH. We can enable features by:

By actor — 5% of users will have the feature enabled

By time — 5% of calls will have the feature enabled

Group — Enable for a predefined set of actors

Actor — Enable for specific actors

Feature Flag
OFF

RECENT PURCHASES

INVOICE	SALE	DATE	# OF ITEMS	SHIPPING STATUS	TOTAL
#B-3971587	Toys, Collectibles, Décor & More	October 27, 2017	1 item	**Delivered** Courier: UPS Tracking #: 1Z557VY3P298378634	$32.91

Feature Flag
ON

RECENT PURCHASES

ITEM(S)	INVOICE	DATE	STATUS	TOTAL
 Vintage Otagiri Mercantile Company Porcelain Tableware	#B-3971587	October 27, 2017	**Delivered** Courier: UPS Tracking #: 1Z557VY3P298378634	**$32.91**

Uses for feature flags

- **Releasing features**
- **A/B testing and other experiments**
  - See how different pages perform by hooking up feature flags and user analytics
- **Permission toggles**
  - We are using feature flags to enable sending server timings down with each request - so we only send it to developers when needed

What about our example?

We had feature flags for the visual components, but the part of code that broke the site was where we were preloading data for the bidding modal.

Using a Feature Flag to enable this for some users would have given us not only a **better way to control server load** — but we would also get to see how it **impacted user behavior**.

But doesn’t this add complexity?

Adding conditionals may seem like we are adding more code paths, making the code more complex. But the alternative still has the conditional — it’s just hiding on your computer.

However, for almost every feature flag you add you will have to **set aside time to clean up** unused code after a feature is released or an experiment is over.

Words of wisdom — from Martin Fowler

Toggle tests should only appear at the **minimum amount of toggle points** to ensure the new feature is properly hidden.

Release toggles are a useful technique and lots of teams use them. However they should be your **last choice** when you're dealing with putting features into production.

Your first choice should be to **break the feature down** so you can safely introduce parts of the feature into the product.

Breaking down the example

- Users are now required to select a shipping address
- Change address selection from a dropdown to a dedicated page of the bidding modal
- Do not allow selection of “undeliverable” addresses, prompt user to add an address if no shippable addresses are on the account
- Show a message if max bid goes over $5000 (we do not accept credit cards over $5k)
- Updates autocheckout to support shipping address preference
- Group invoices by address (We batch invoices by certain properties, like sales and fulfillment preferences)

Sometimes long-lived branches are OK

Sometimes the best way to communicate is by submitting a large chunk of work in one go. It makes the **entire scope** of the change **immediately understandable**. It also ensures that the **entire feature is complete** — important for open source projects where you might not have good follow up.

As with everything, there are tradeoffs and you have to find the right balance for your project.

Wrapping it all up

Thank you!

Kat Fairbanks
kat.fairbanks@ebth.com
github.com/katzenbar/talks

Resources

- Long-Running Branches Considered Harmful https://blog.newrelic.com/culture/long-running-branches-considered-harmful/
- The Pitfalls of Feature Branching https://rollout.io/blog/pitfalls-feature-branching/
- The Top 5 Use Cases for Feature Flags https://rollout.io/blog/top-5-use-cases-feature-flags/
- Learning from Pain https://benjiweber.co.uk/blog/2018/09/12/learning-from-pain/
- Upgrading GitHub from Rails 3.2 to 5.2 https://githubengineering.com/upgrading-github-from-rails-3-2-to-5-2/
- The (written) unwritten guide to pull requests https://www.atlassian.com/blog/git/written-unwritten-guide-pull-requests
- Best Practices for Large Features https://drivy.engineering/best-practices-for-large-features/
- FeatureToggle https://martinfowler.com/bliki/FeatureToggle.html
- Feature Toggles (aka Feature Flags) https://martinfowler.com/articles/feature-toggles.html

Photos

- https://unsplash.com/photos/j_Ch0mwBNds by Henry Hustava
- https://unsplash.com/photos/6x-hVXXiBxs by Matthew Henry
- https://unsplash.com/photos/Yl2Bz4jbm44 by Clem Onojeghuo
- https://unsplash.com/photos/0woyPEJQ7jc by Glenn Carstens-Peters
- https://unsplash.com/photos/tEMU4lzAL0w By FuYong Hua
- https://unsplash.com/photos/5DIFvVwe6wk By Linus Sandvide
- https://unsplash.com/photos/bfGomtF3Uvc by Andrii Podilnyk
- https://unsplash.com/photos/mTkXSSScrzw by Leone Venter
- https://unsplash.com/photos/Qmox1MkYDnY by Mikhail Vasilyev