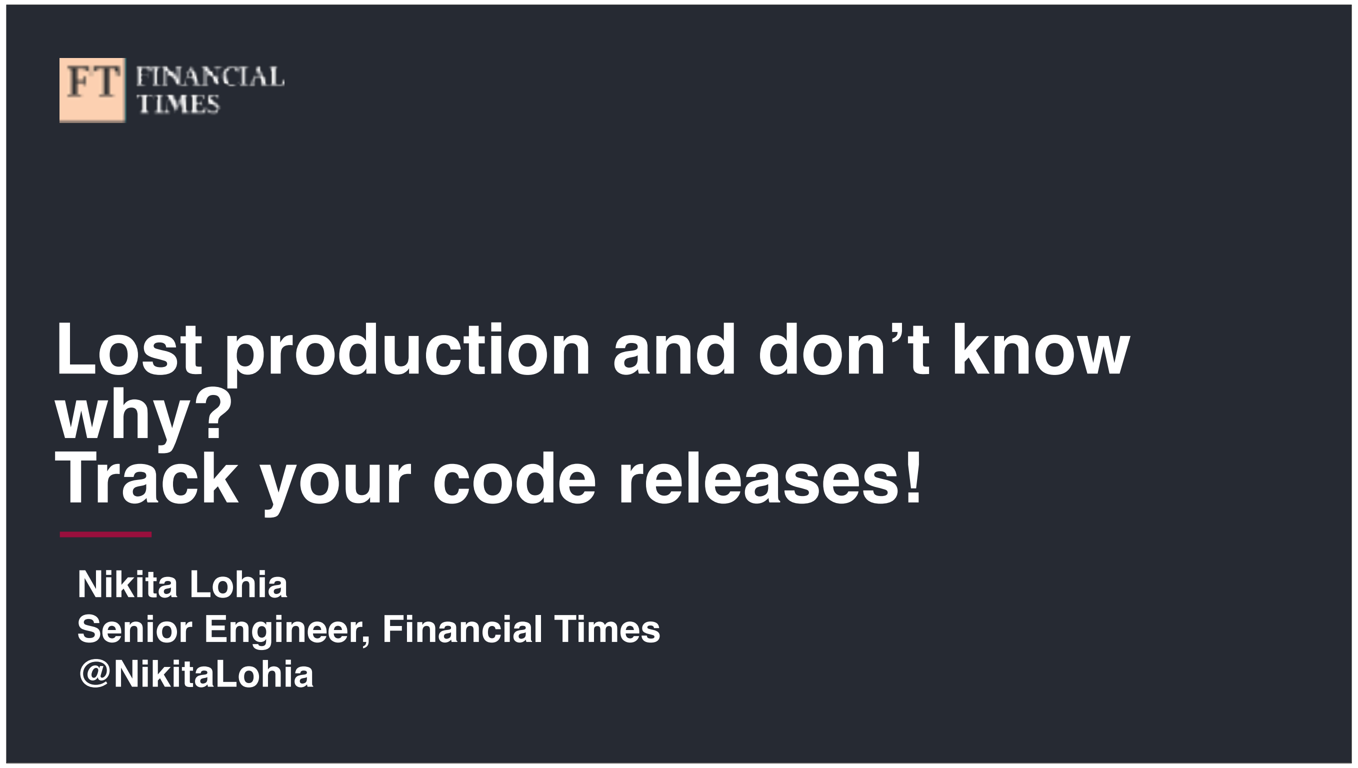
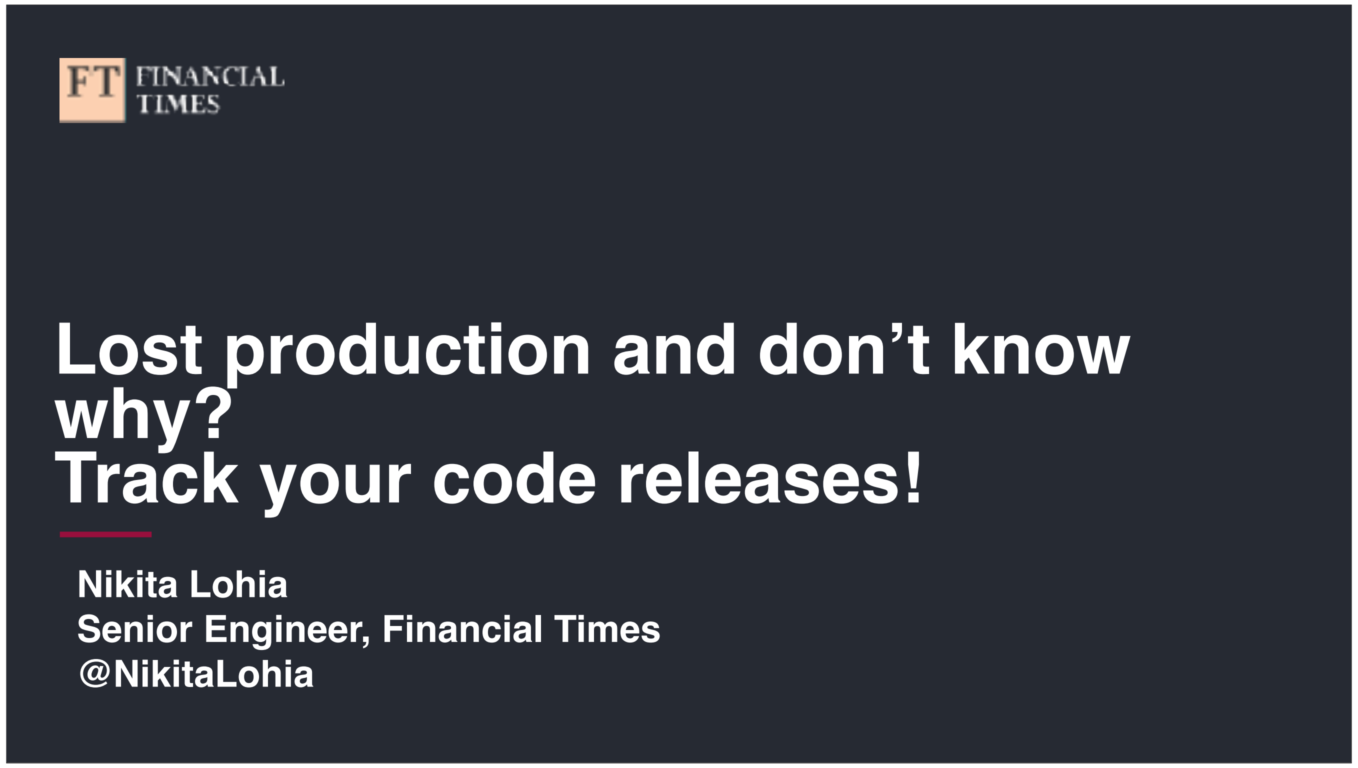


FT FINANCIAL TIMES

Lost production and don’t know why? Track your code releases!

Nikita Lohia
Senior Engineer, Financial Times
@NikitaLohia

HOLDING SCREEN - click once at start for timings!



Lost production and don’t know why? Track your code releases!

Nikita Lohia
Senior Engineer, Financial Times
@NikitaLohia

HOLDING SCREEN - click once at start for timings!

Thank you for joining us today. I am Nikita and I am a senior engineer at the FT. I want to talk about how we track code releases at the FT.

Making the case for cloud only

How we persuaded the Financial Times Technology department to move everything to the cloud.

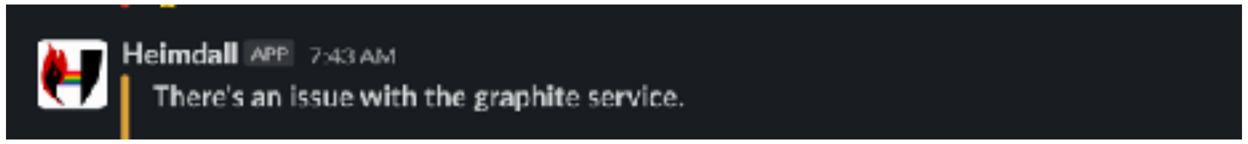
https://medium.com/ft-product-technology/making-the-case-for-cloud-only-92f382ff8dd9

The Financial Times is one of the world’s leading business and financial news organisations.

As of 2020, we are a fully cloud hosted company. Mark Barnes, my colleague at the FT has written an excellent blog on our journey to Cloud-only.

The Problem

When something goes wrong, the first question you want answered is - what changed?



Heimdall APP 7:43 AM

There's an issue with the graphite service.

@NikitaLohia

FT

At the FT, we have around 500 microservices being managed by multiple different teams. These teams work in different timezones, releasing code at different times during the day.

At any point in time, we want to be able to consistently and confidently answer the question - what changed recently and

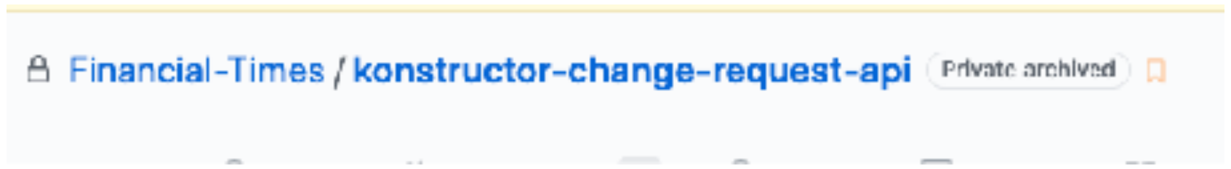
if that change causes a disruption how quickly can we fix it?

This is not only important for the teams developing these services, but also important for our Operations team - watching over our entire estate 24/7.

How can they associate an alert firing to a recent release?

So we set ourselves a challenge - we need to track **all** code releases happening in our tech estate.

Our first attempt ..

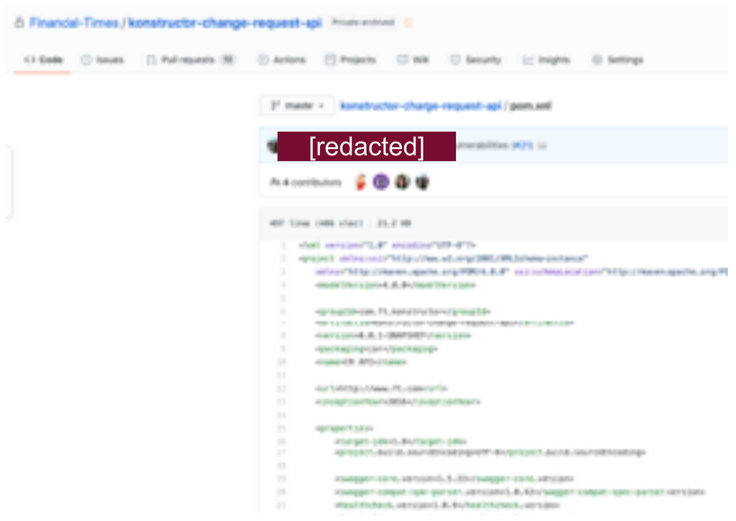

Financial-Times / konstructor-change-request-api Private archived

Our legacy Change Request API(CR API)

@NikitaLohia

Change and Release Management at the FT was not a new concept. Our legacy Change Request API, fondly called CRAPI (/ˈkræpɪ/) was tracking releases, both automated and manual, and storing them in a third party change management tool. There were a few reasons why it wasn't a huge success and why it didn't quite work anymore….

Our first attempt ..



Financial-Times / konstructor-change-request-api

master · konstructor-change-request-api / pom.xml

[redacted]

```
[illegible]
```

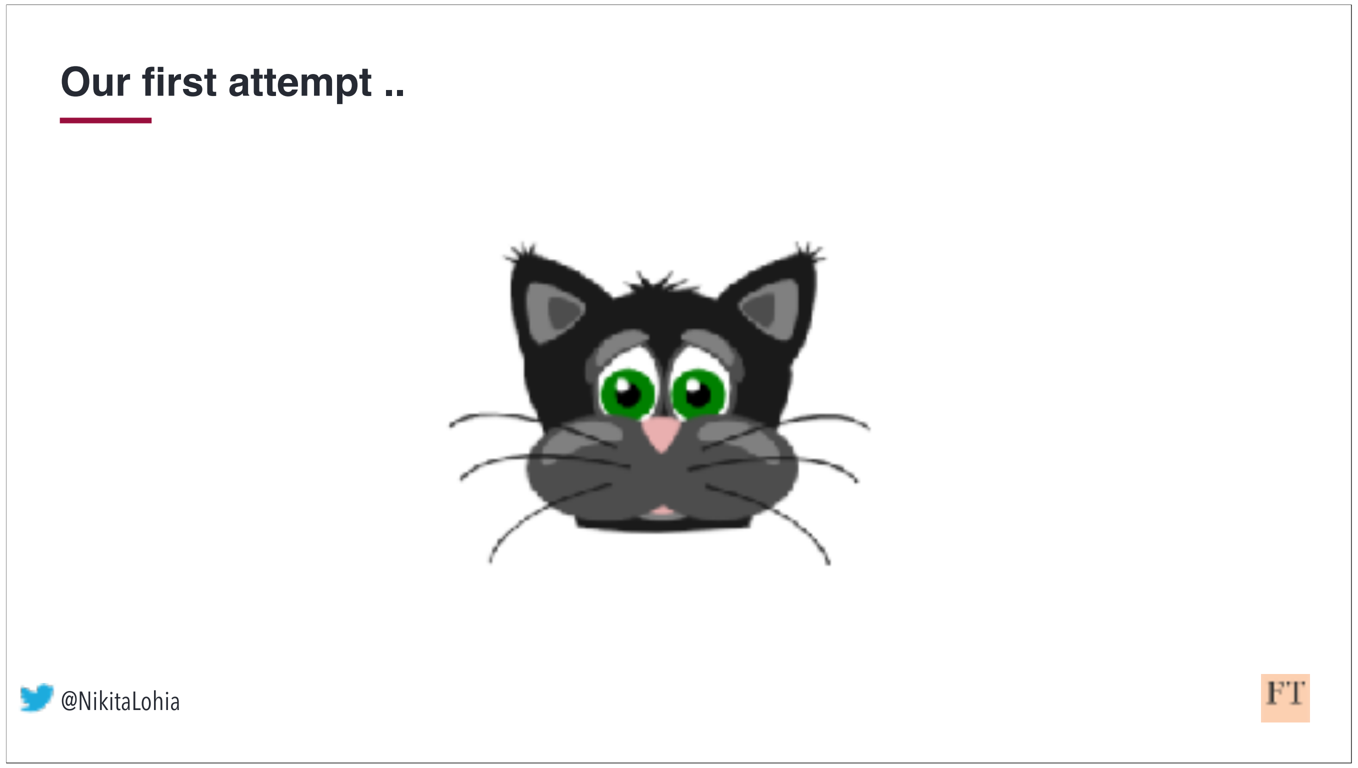
Our legacy Change Request API(CR API)

@NikitaLohia

Legacy CR API was written in Java around 5 years ago, people who wrote it had long moved on. No-one in the team quite understood it anymore.

It was not easy to use and very error prone.

Users also needed to specify a start and an end date for the change, thus needing to call the API twice.

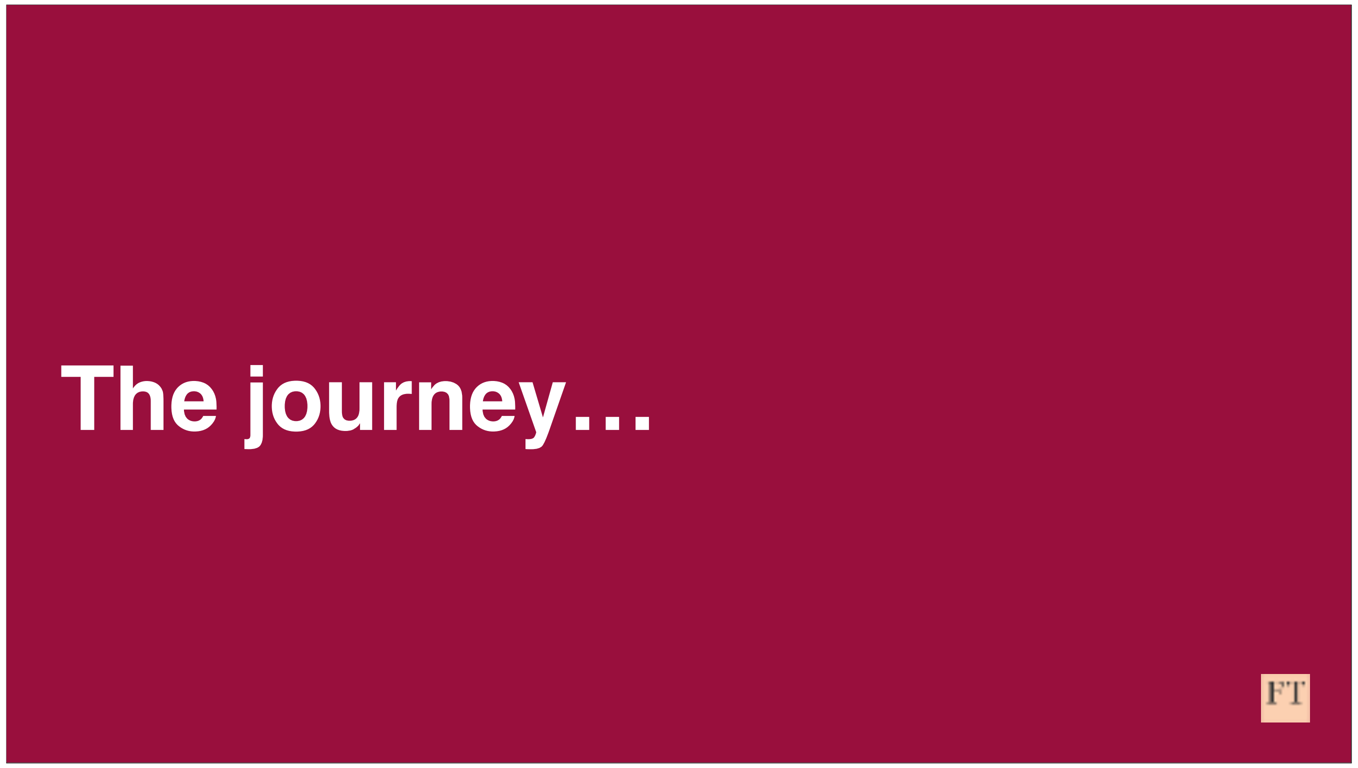


Our first attempt ..

@NikitaLohia

FT

- As a result, teams were often discouraged to add it to their production pipelines since it would constantly break/delay their builds.



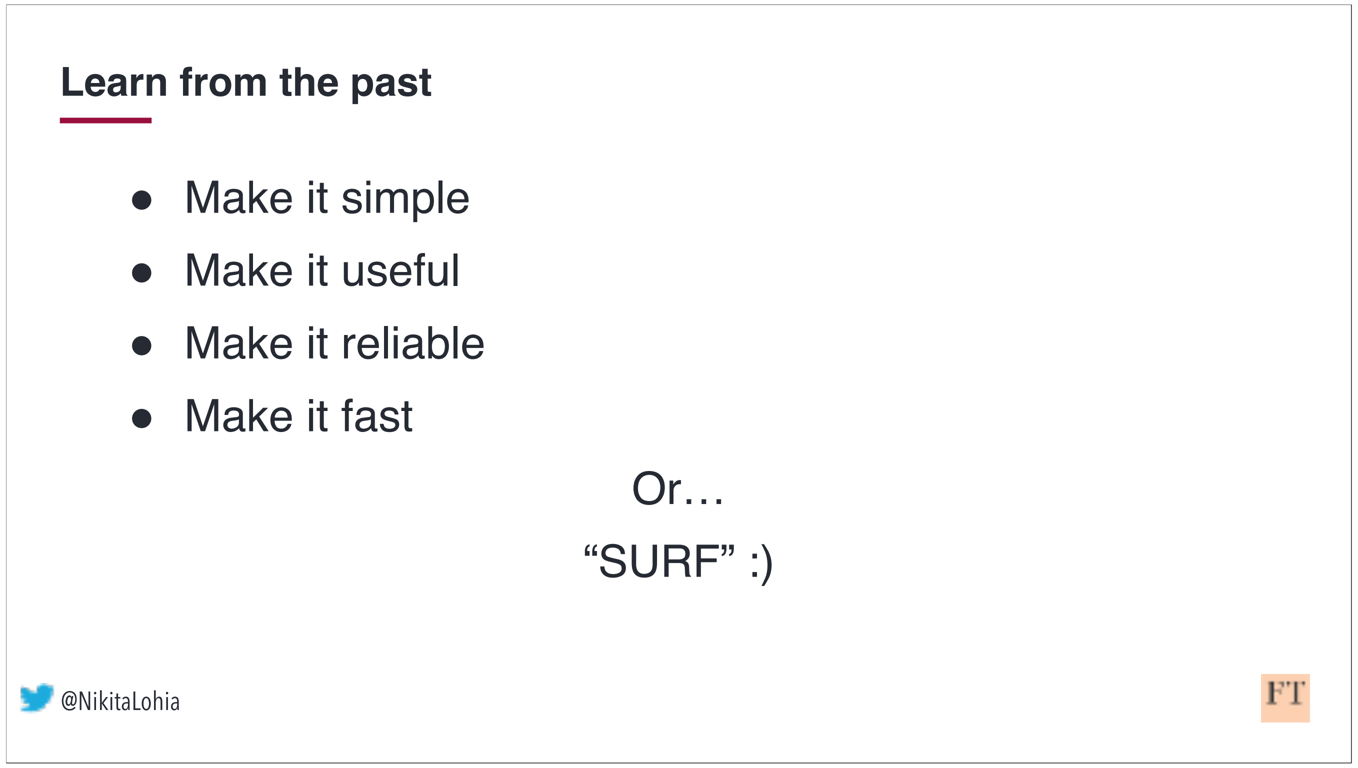
The journey…

Lets go through our journey from migrating from this legacy , rarely used CR API to a fully integrated, automated release log process and the lessons we learnt along the way.

Learn from the past

To start off - we did some digging ..

We spent some time talking to the handful of people who were using legacy CR API. We asked them what about it works and what does not, what can we use and what can we throw away..



Learn from the past

- Make it simple
- Make it useful
- Make it reliable
- Make it fast

Or…

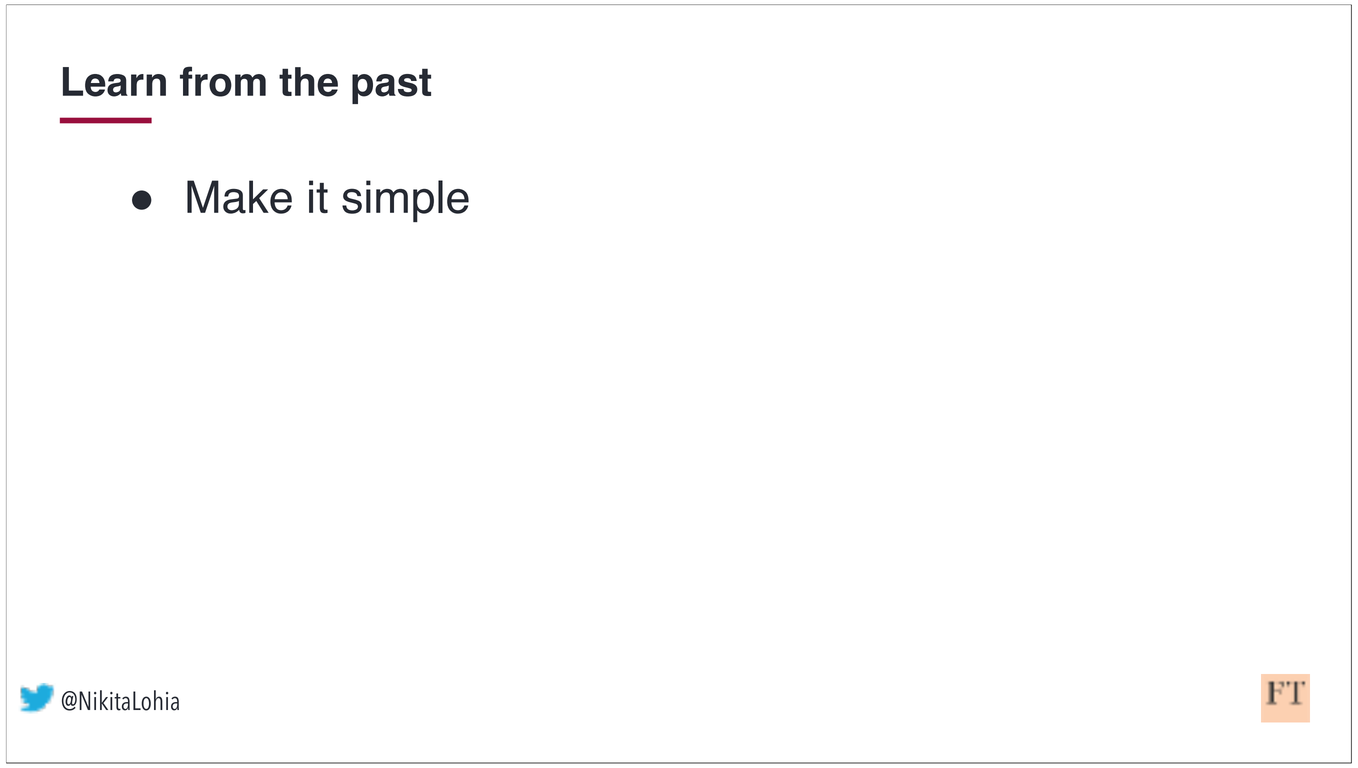
“SURF” :)

@NikitaLohia

FT

We summarised all of that feedback and consolidated that into these 4 targets.

Since we love us some acronyms, why not .. lets call it SURF :)



Learn from the past

- Make it simple

@NikitaLohia

FT

So - how did we address these goals? How did we simplify sending Change logs at the FT?

Learn from the past - Make it simple

For our end users, we reduced the amount of work they need to do

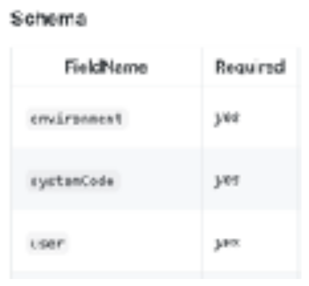
@NikitaLohia

FT

Legacy CR API depended on its users for a lot of information. We decided to make the service, now rebranded as Change API, do the work.

Instead of requiring 10 different fields, we asked Change API users to only provide 3 mandatory values :

Learn from the past - Make it simple



Schema

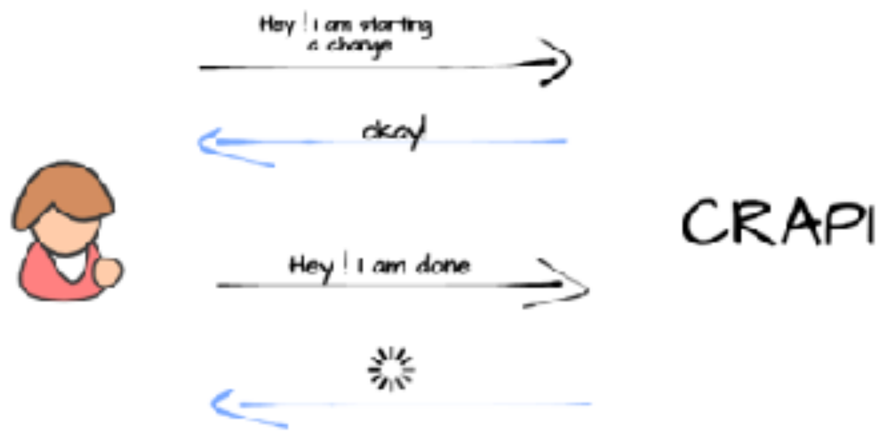
FieldName	Required
environment	yes
systemCode	yes
user	yes

@NikitaLohia

FT

- The service or the system being changed. We annotate every service at the FT with a unique id, called a systemcode, to identify it.
- The environment in which the service is being changed. i.e is it a production or a dev release and finally
- The person who is making the change

Learn from the past - Make it simple



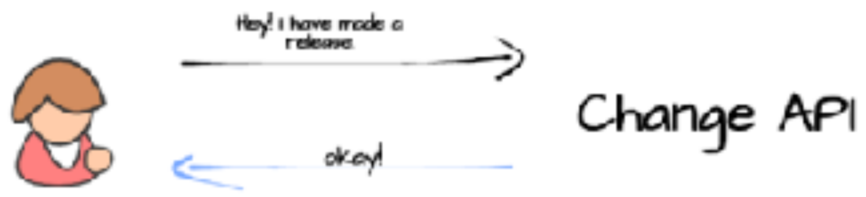
@NikitaLohia

Our legacy Change Request API(CR API)

FT

During our user research, not many users saw the benefit of logging the start and the end date for a release. Rightly so, since with microservices, most of the code releases only take a few minutes. When designing change API, we removed the need for the users to tell Change API the start date of the release. So instead of having to make two separate API calls…

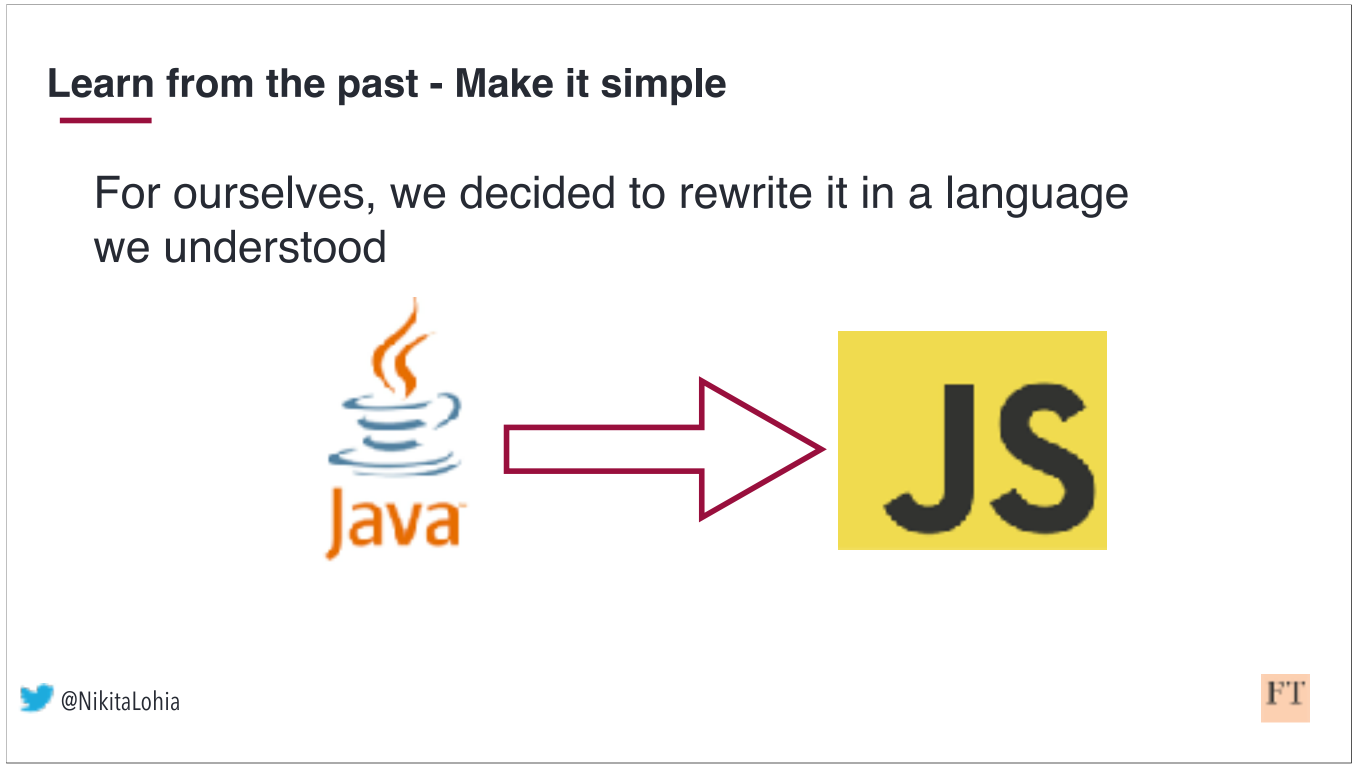
Learn from the past - Make it simple



@NikitaLohia

FT

they only need to call it once, after the change was done.



Learn from the past - Make it simple

For ourselves, we decided to rewrite it in a language we understood

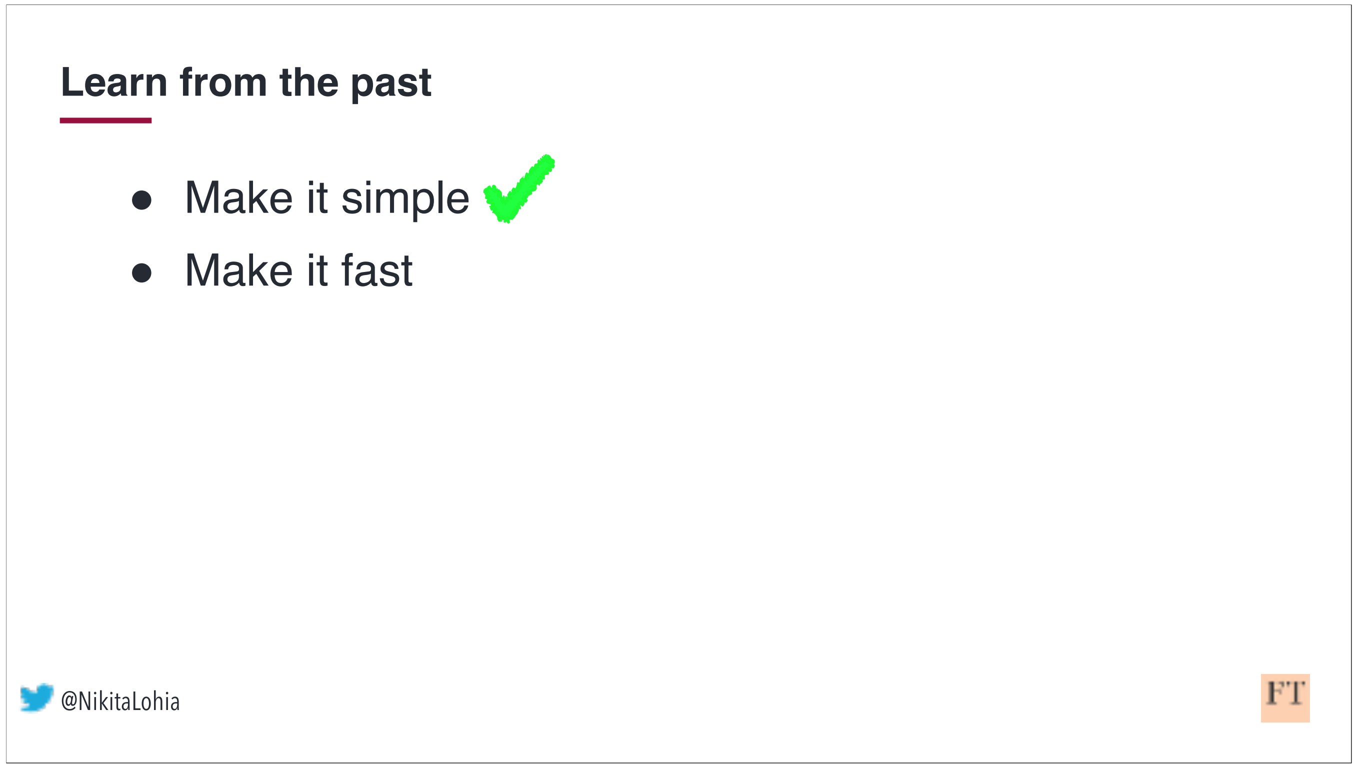
Java → JS

@NikitaLohia

FT

Also,

Since no-one in the team had a lot of experience with Java, we decided to rewrite it in JS to better support it.



Learn from the past

- Make it simple ✓
- Make it fast

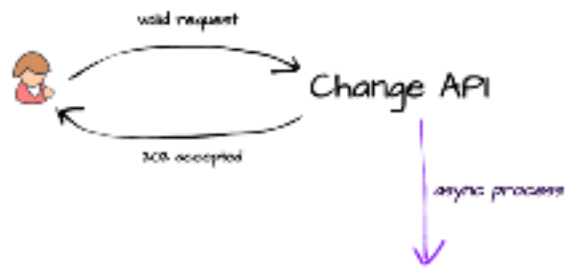
@NikitaLohia

FT

How did we solve the problem of avoiding delays on team’s deployment pipelines?

Learn from the past - Make it fast

We made API calls asynchronous



@NikitaLohia

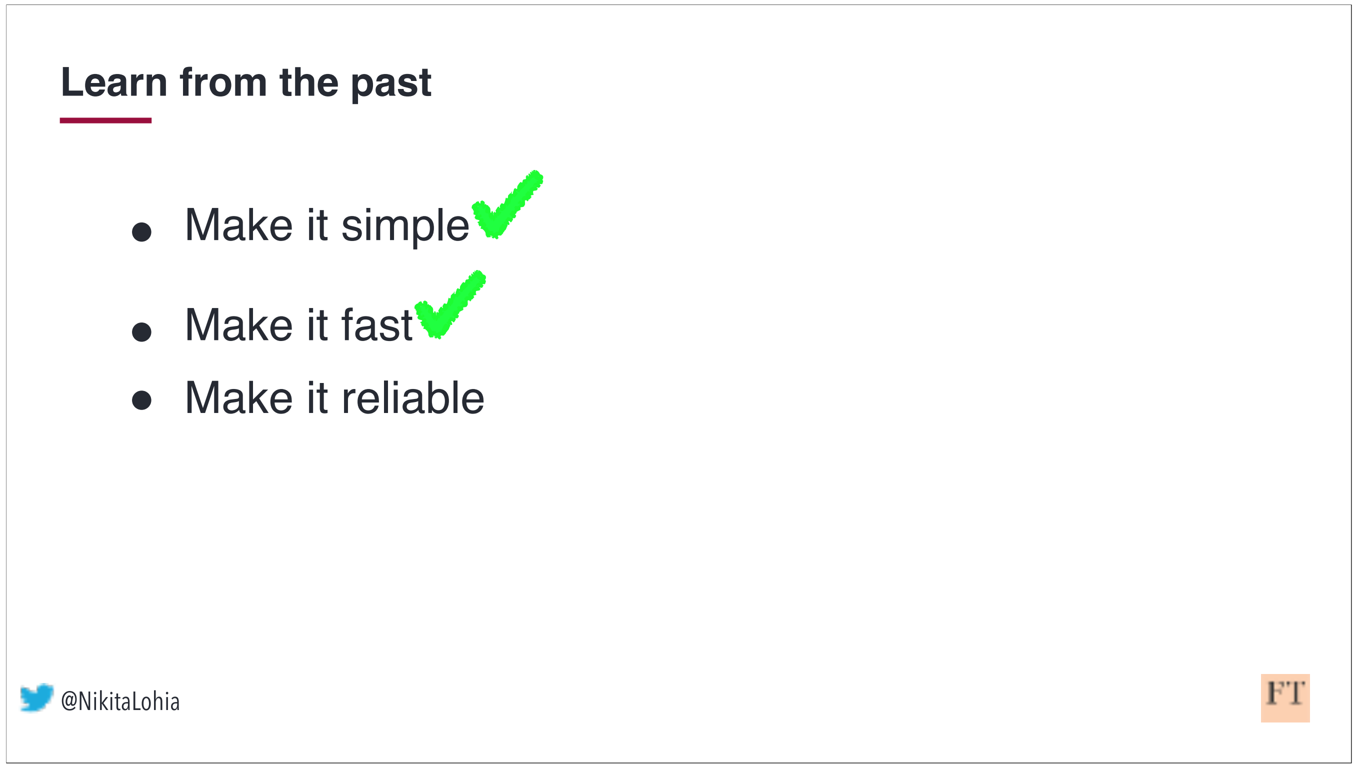
FT

We made Change API async.

As long as the user was sending Change API the 3 mandatory fields and it was in a valid format, Change API would almost immediately send an “Accepted” response. An invalid client request would return a “bad request” instead.

Rest of the release log processing happened asynchronously. As far as the user is concerned, the Change log request is completed within milliseconds, if that.

By doing just these 2 things, we had already solved half of our problems.



Learn from the past

- Make it simple ✓
- Make it fast ✓
- Make it reliable

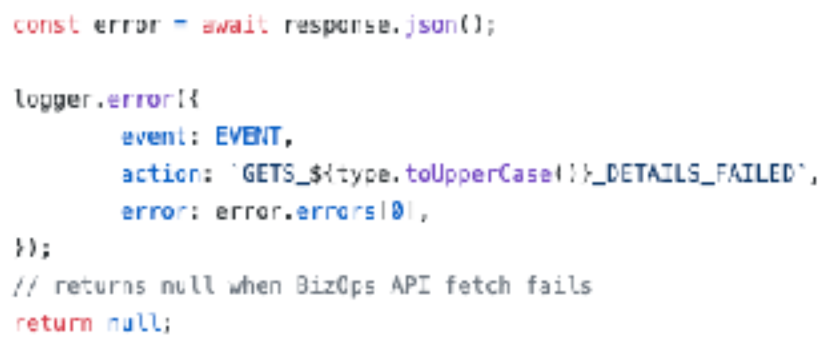
@NikitaLohia

FT

How did we build trust in Change API ?

Learn from the past - Make it reliable

Don’t throw server side errors



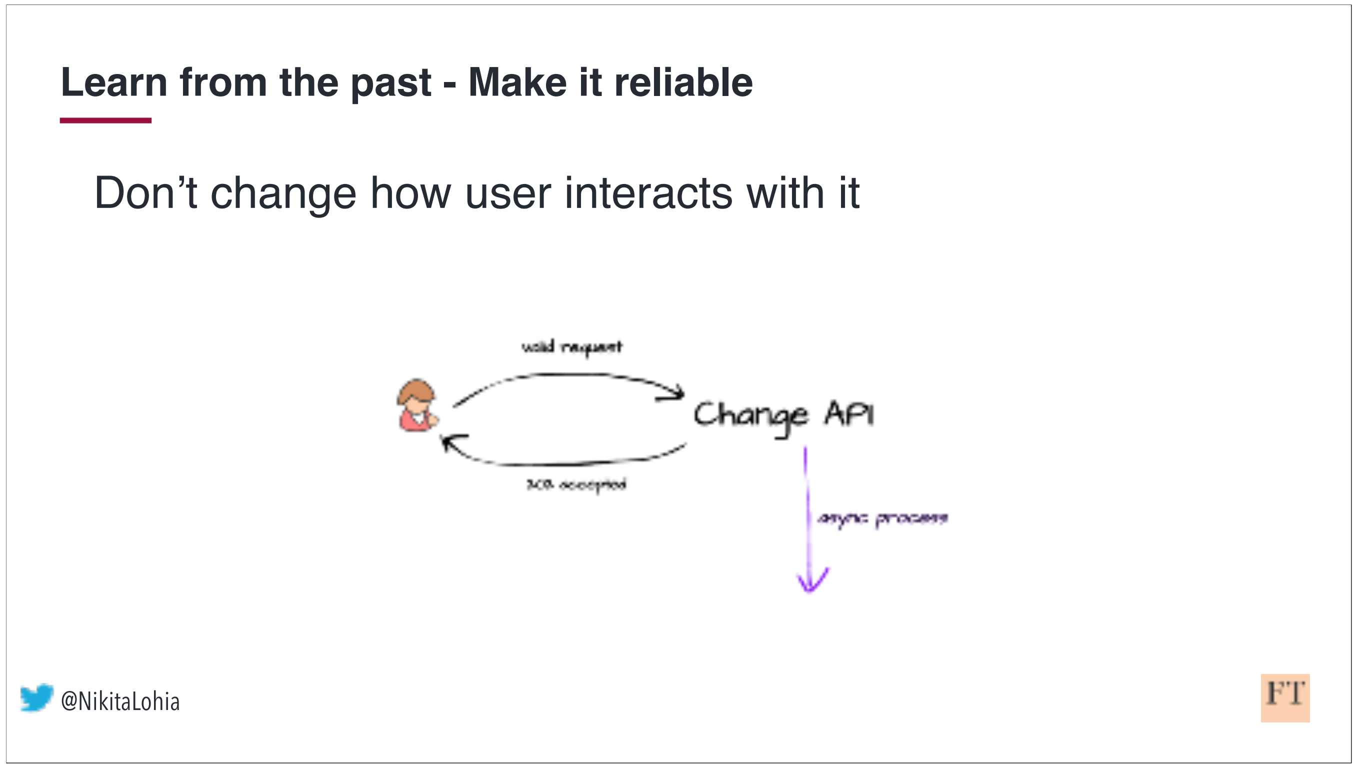
```
const error = await response.json();

logger.error({
        event: EVENT,
        action: `GETS_${type.toUpperCase()}_DETAILS_FAILED`,
        error: error.errors[0],
});
// returns null when BizOps API fetch fails
return null;
```

@NikitaLohia

FT

Another important lesson we learnt was to ensure that server side errors do not cause a bad user experience. We decided to simply not throw any server side errors. There wasn’t anything a user could do about them anyway. They would just get annoyed at us for breaking their pipeline.

We instead, silently log all the errors and let the request complete successfully. It was upto Change API to handle retries under the hood.



Learn from the past - Make it reliable

Don’t change how user interacts with it

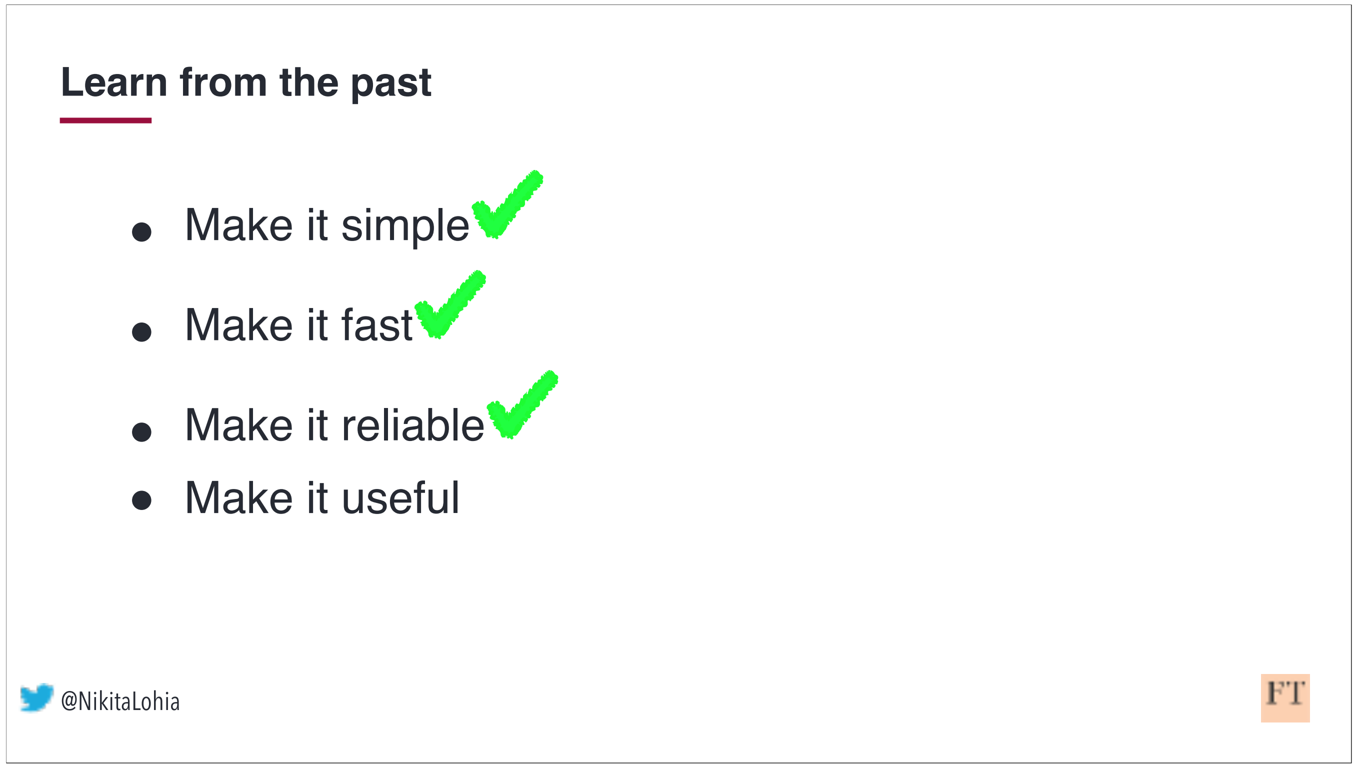
valid request
Change API
202 accepted
async process

@NikitaLohia

FT

Over time, we made a LOT of improvements and modifications to Change API - ALL without changing user interaction with it.

This provided our users some much needed assurance that adding Change API to their deploy pipelines will not stop their deployment, even in case of an error.



Learn from the past

- Make it simple ✓
- Make it fast ✓
- Make it reliable ✓
- Make it useful

@NikitaLohia

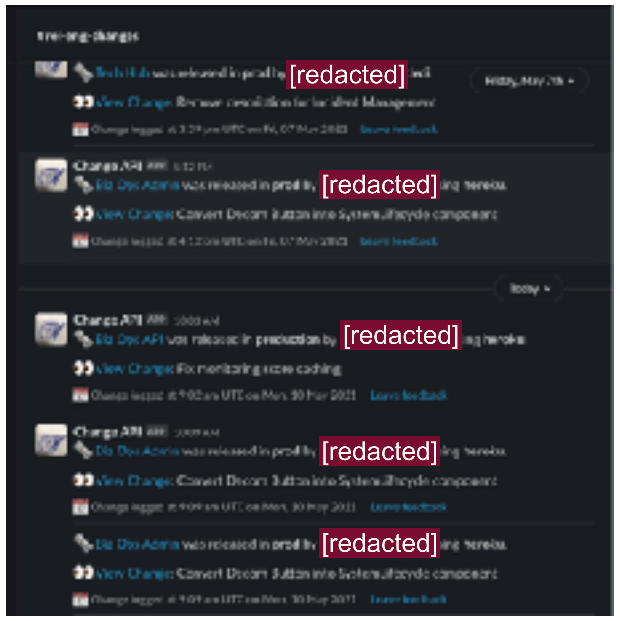
FT

And finally - how did we highlight the benefit of Change API?

Learn from the past - Make it useful

Real-time notifications

of *relevant* change logs


[illegible] [redacted] [illegible]

[illegible]

[illegible]

Change API [illegible]

[illegible] [redacted] [illegible]

[illegible]

[illegible]

Change API [illegible]

[illegible] [redacted] [illegible]

[illegible]

[illegible]

Change API [illegible]

[illegible] [redacted] [illegible]

[illegible]

[illegible]

[illegible] [redacted] [illegible]

[illegible]

[illegible]

@NikitaLohia

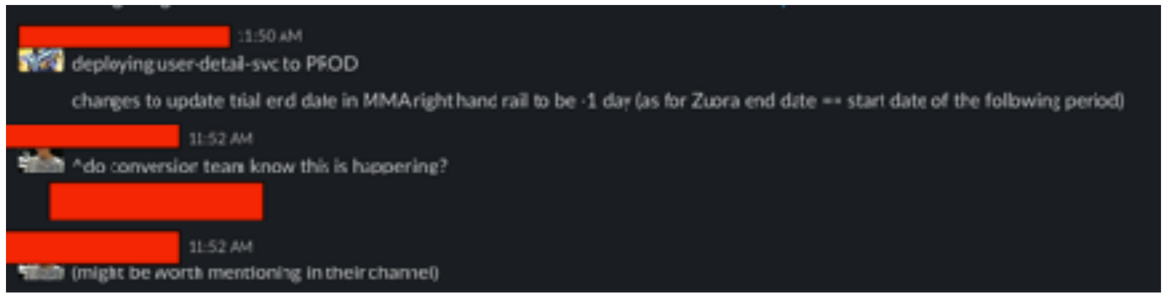
FT

Legacy CR API didn't get a lot of adoption because our engineering teams did not see any benefit in it.

We gave Change API users the ability to get notifications of their releases.. in real-time. **All** production releases would, by default, go to a single slack channel.

A user could also provide a Slack channel of their choosing and Change API would dutifully send a release notification to it.

Learn from the past - Make it useful



[redacted] 11:50 AM
deploying user-detail-svc to PROD
changes to update trial end date in MMA right hand rail to be -1 day (as for Zuora end date == start date of the following period)
[redacted] 11:52 AM
^do conversion team know this is happening?
[redacted]
[redacted] 11:52 AM
(might be worth mentioning in their channel)

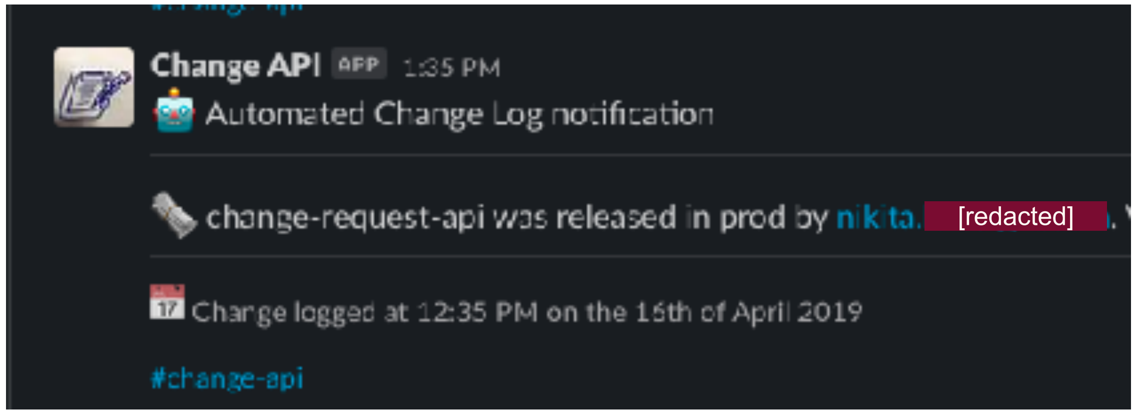
@NikitaLohia

Before Change API

FT

We went from having to manually inform people that a release is happening…

Learn from the past - Make it useful



Change API APP 1:35 PM

Automated Change Log notification

change-request-api was released in prod by nikita.[redacted]

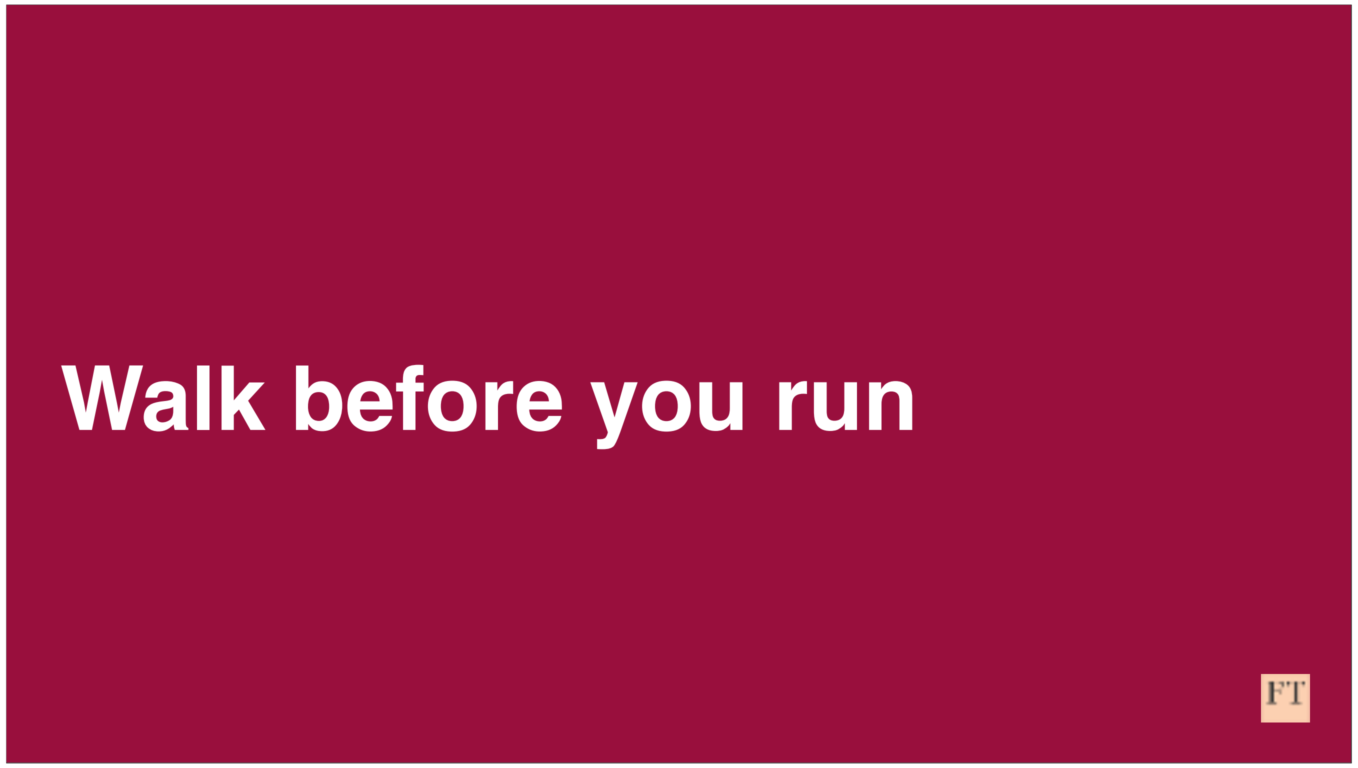
Change logged at 12:35 PM on the 15th of April 2019

#change-api

@NikitaLohia Change API MVP(Minimal Viable Product) FT

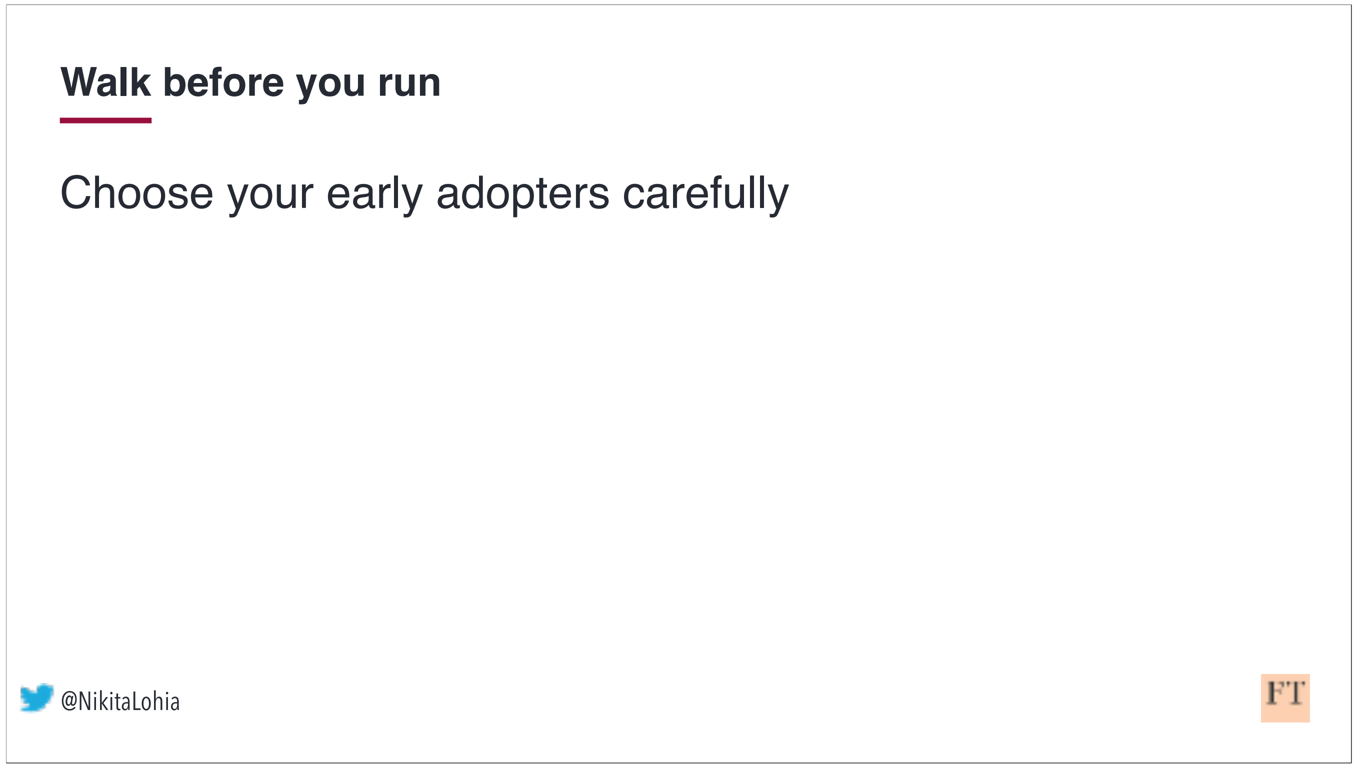
.. to an automated notification model, where Change API would send release logs automatically on Slack.

It looks very bare bones now, but trust me , it gets better :)



Walk before you run

In other words, start small.



Walk before you run

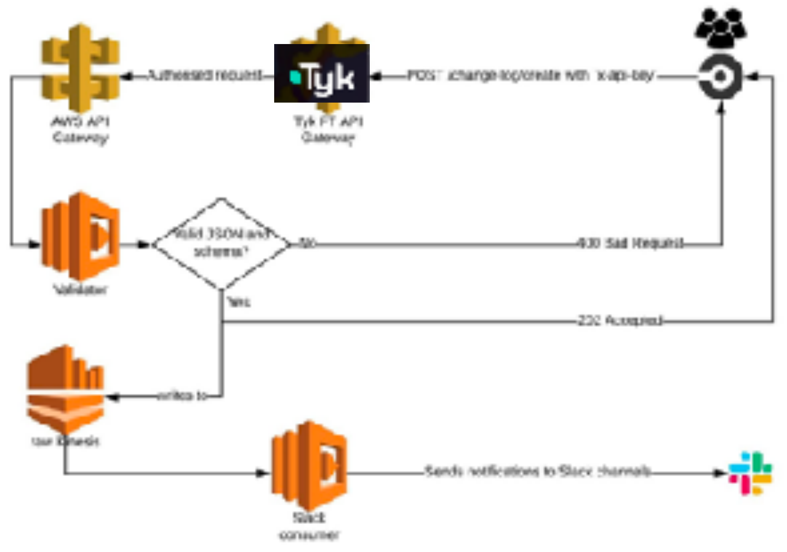
Choose your early adopters carefully

@NikitaLohia

FT

Change and Release Management is big topic. If you think about it, external auditors, compliance team, dev teams, Project Managers, and Operations support team: they are all potential users of Change & Release Management

We picked a handful of engineering teams from different parts of the business using Github and CircleCI as our early adopters since they already had a deployment pipeline in place and we simply had to hook Change API into it.



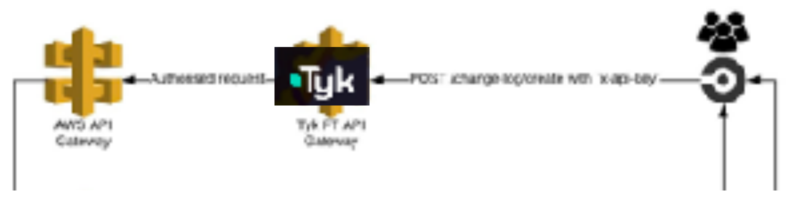
@NikitaLohia

Change API MVP

FT

In its infancy, all change API did was accept a valid JSON payload which sent a message on Slack upon successful execution.

Lets break this down a little…



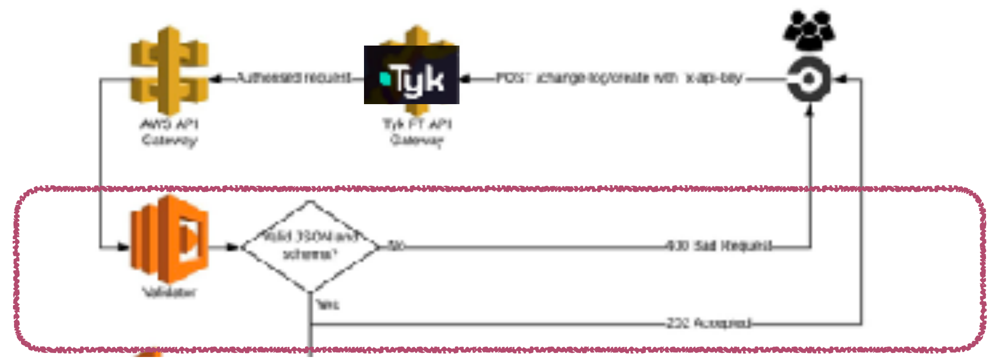
@NikitaLohia

Change API MVP

FT

We asked teams using CircleCI to add a single POST request to their circleCI workflow. This request would contain a JSON body and an API key header.
This request would first get authenticated by our API Gateway platform which would check the validity of the API key.

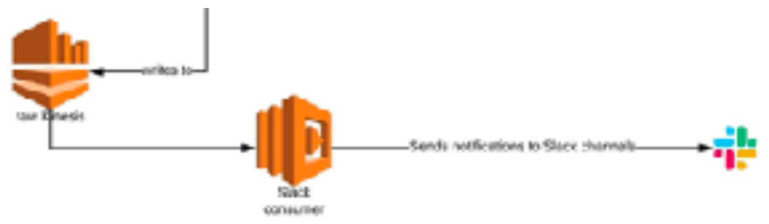
If the authentication fails, the user gets an authentication error.



Change API MVP

If a request is successfully authenticated, a lambda function, called Validator here would validate this request.
- if the validation fails, it would send a 400 bad request to the user, ending the user journey.

If the validation is successful,
- It would immediately send a 202 response to the user, ending the user journey.



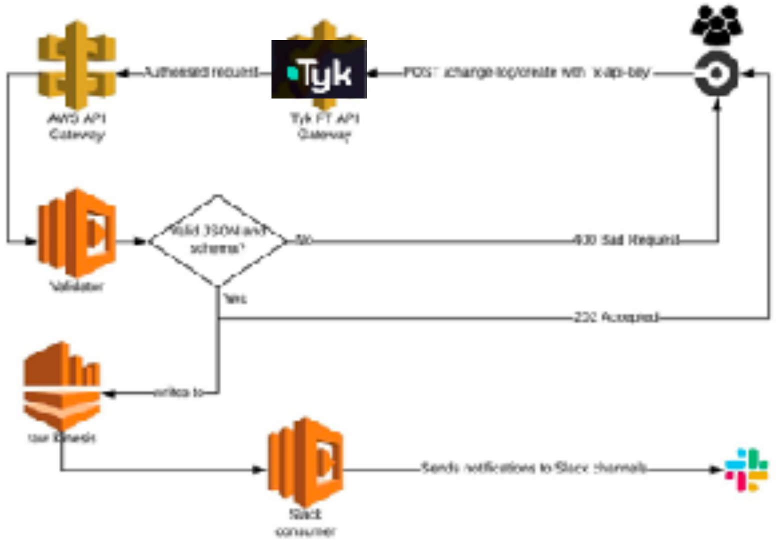
@NikitaLohia

Change API MVP

FT

Async, the lambda would then also write to a data stream, in our case, AWS Kinesis…

Another lambda function, called Slack consumer here, would then read this payload and send a message to slack.



@NikitaLohia

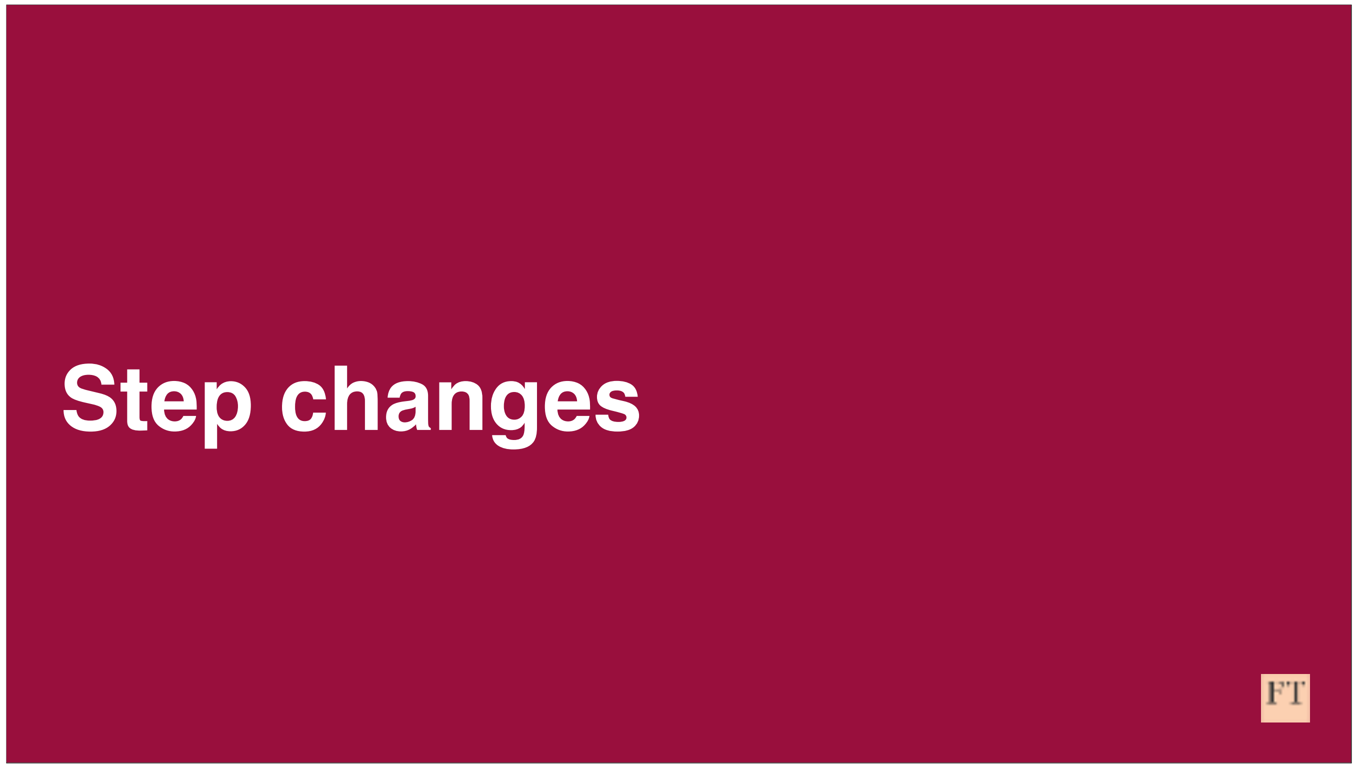
Change API MVP

FT

I mentioned earlier about giving users a consistent and reliable experience with Change API - I cannot stress the importance of this enough. Reliable behaviour of Change API was what drove its adoption rate initially. It was fire once and forget. Through all the changes that came later, this part of change API was never modified. Note the striking absence of a database. That came later too…

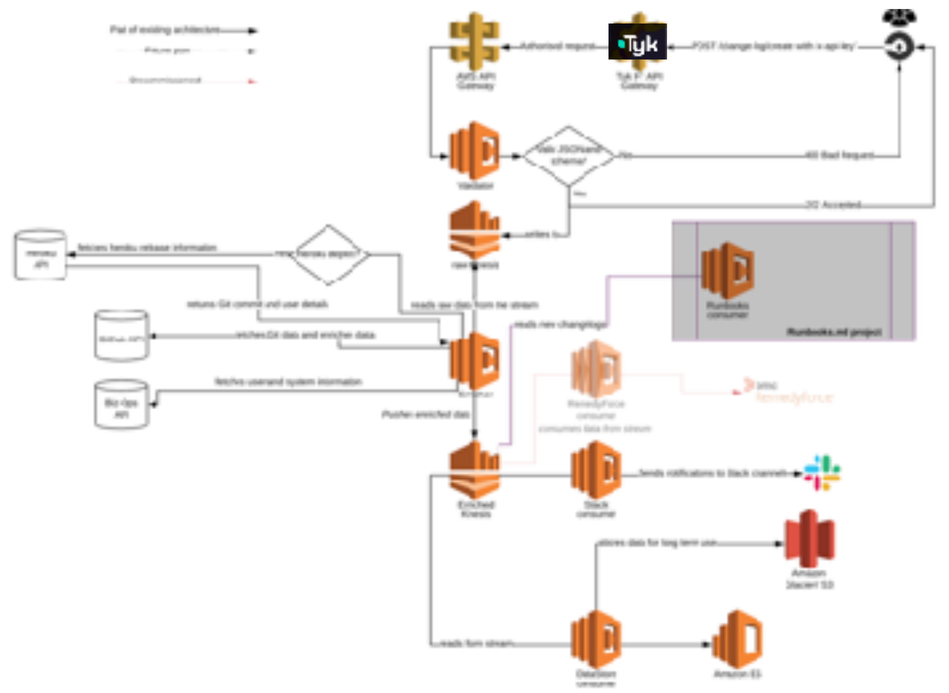
The idea was that we could simply add or remove more features to our steady stream of real-time release logs to continuously BUT iteratively improve Change API.

Within 3 months of starting this project, and a small team of 3 engineers, we produced a Change API MVP. It was a huge success.



Step changes

We expanded our early adopters list and we kicked Change API development into high-gear.

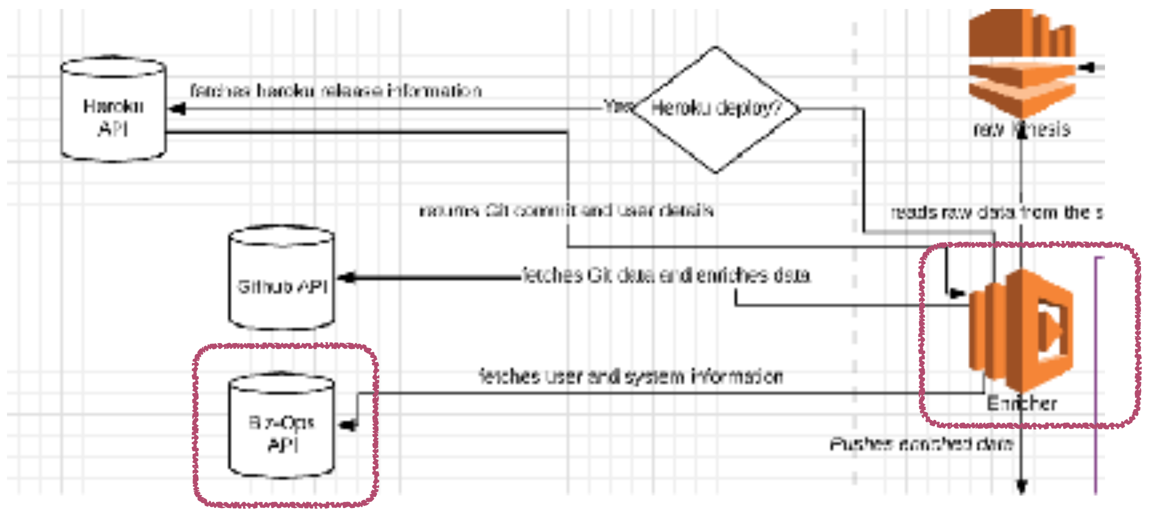


@NikitaLohia

Change API now

FT

We integrated Change API with other services. This is what Change API architecture looks like now. Allow me to highlight some interesting areas..


Heroku API
fetches heroku release information
Yes
Heroku deploy?
raw Kinesis
returns Git commit and user details
reads raw data from the s
Github API
fetches Git data and enriches data
Biz-Ops API
fetches user and system information
Enricher
Pushes enriched data

@NikitaLohia

FT

Change API now

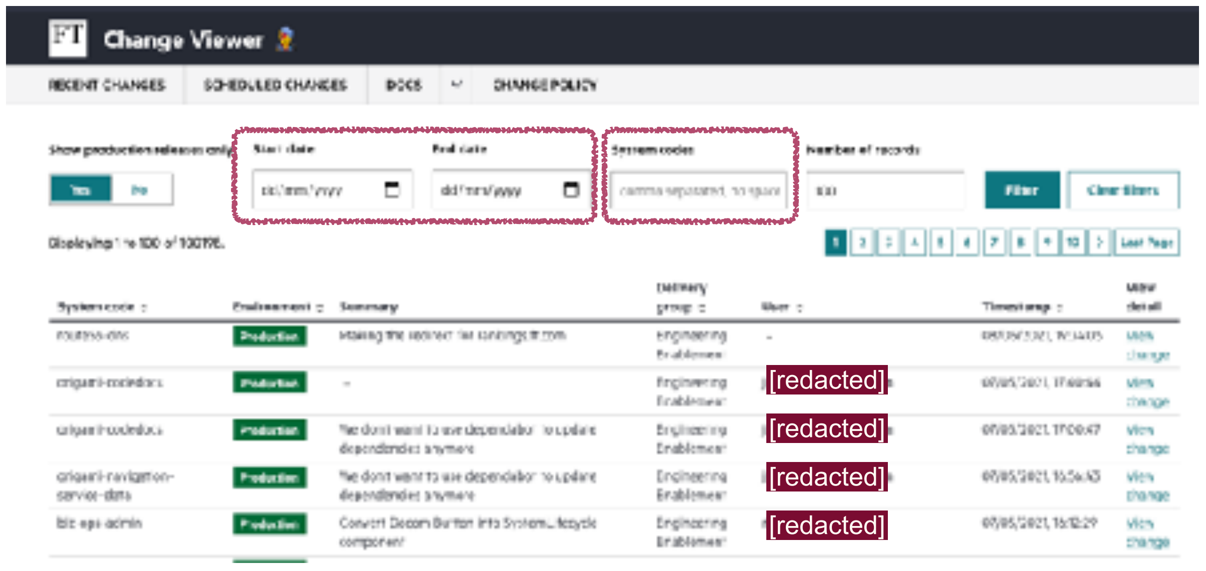
We added “enrichments”..

A lambda function, called enricher here would interact with the GitHub API to fetch code commit and Pull Request information.

The enricher would also talk to our system registry API and fetch additional information about the service being released.

we added many such enrichments…

It’s worth reminding you here that all this happens async. The user journey has long ended after sending a valid request payload .



FT Change Viewer

RECENT CHANGES | SCHEDULED CHANGES | DOCS | CHANGE POLICY

Show production releases only: Yes / No
Start date: dd/mm/yyyy
End date: dd/mm/yyyy
System codes: comma separated, no space
Number of records: 100
Filter | Clear filters

Displaying 1 to 100 of 100196.

1 2 3 4 5 6 7 8 9 10 > Last Page

System code	Environment	Summary	Delivery group	User	Timestamp	View detail
[illegible]	Production	[illegible]	Engineering Enablement	-	[illegible]	View change
origami-codedocs	Production	-	Engineering Enablement	[redacted]	07/05/2021, 17:49:56	View change
origami-codedocs	Production	We don't want to use dependabot to update dependencies anymore	Engineering Enablement	[redacted]	07/05/2021, 17:00:47	View change
origami-navigation-service-data	Production	We don't want to use dependabot to update dependencies anymore	Engineering Enablement	[redacted]	07/05/2021, 16:56:43	View change
biz-ops-admin	Production	Convert Decom Button into SystemLifecycle component	Engineering Enablement	[redacted]	07/05/2021, 16:12:29	View change

@NikitaLohia

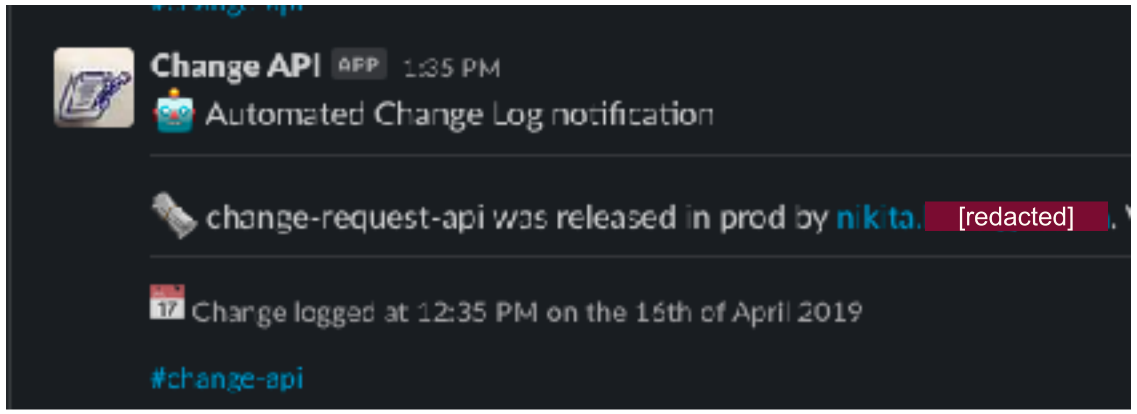
Search and view capability of historical releases

FT

We also build a change log database.

This is Change Viewer which allows users to search for historical releases logged by Change API. It allows filtering by specific days, or system codes for example.

Change API MVP…



Change API APP 1:35 PM

Automated Change Log notification

change-request-api was released in prod by nikita.[redacted].

Change logged at 12:35 PM on the 15th of April 2019

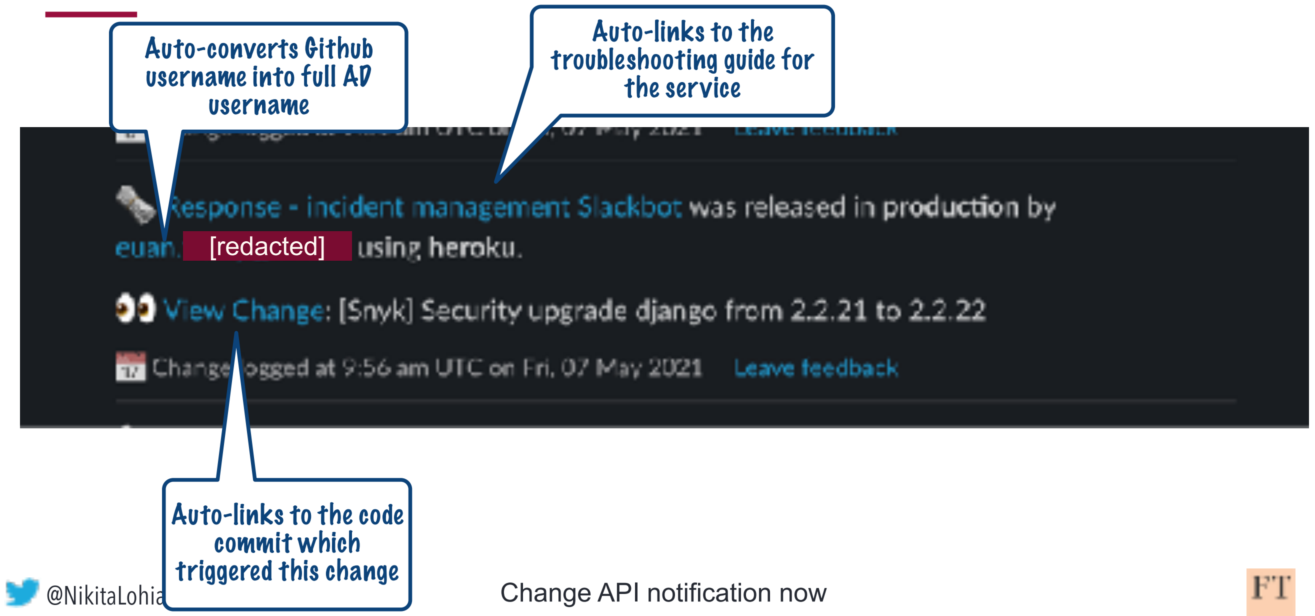
#change-api

@NikitaLohia

FT

Lets remind ourselves what a release notification looked like before..

Enrichments


Auto-converts Github username into full AD username
Auto-links to the troubleshooting guide for the service
Auto-links to the code commit which triggered this change

Response - incident management Slackbot was released in production by euan. [redacted] using heroku.

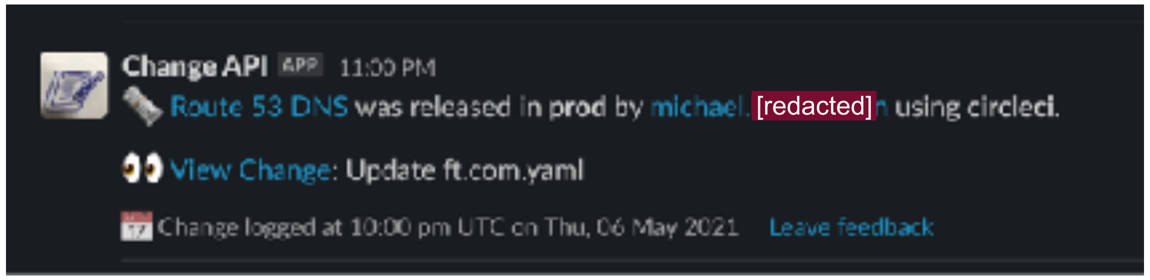
View Change: [Snyk] Security upgrade django from 2.2.21 to 2.2.22

Change logged at 9:56 am UTC on Fri, 07 May 2021 Leave feedback

And now, after all the enrichments this is what a Change API release notification on Slack looks like..

Thanks to the enrichments, the message contains the full name of the person who made the change, along with a link to the code commit, and a troubleshooting guide for the service, just in case the release causes a downtime.

Enrichments - BONUS!



Change API APP 11:00 PM
Route 53 DNS was released in prod by michael.[redacted]n using circleci.

View Change: Update ft.com.yaml

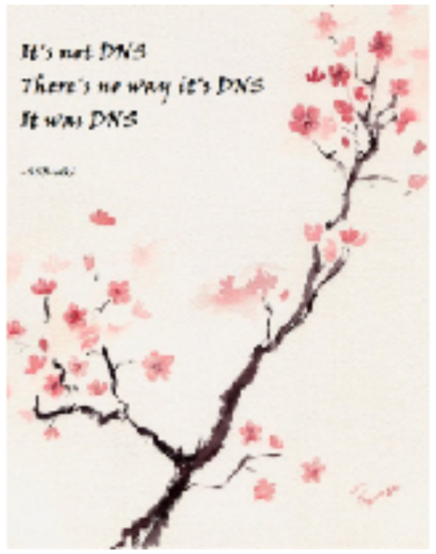
Change logged at 10:00 pm UTC on Thu, 06 May 2021 Leave feedback

@NikitaLohia

FT

We are now even logging changes made to DNS which until a couple of years ago, was a pipe dream !

Enrichments - BONUS!



@NikitaLohia

FT

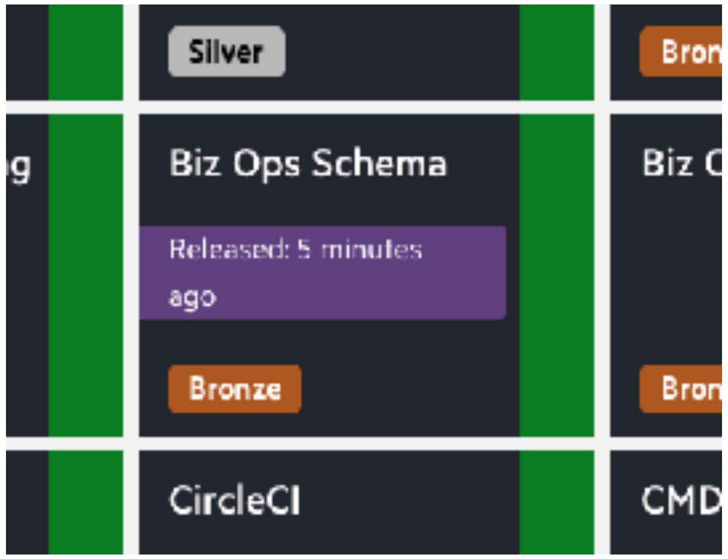
Any sys admins or devops engineers here? You probably have seen this Haiku before..

Would anyone care to guess how many DNS records does the FT have? Think of a number in your head…

Last I checked, we had upwards of 6500 DNS records !

From my experience, if what's wrong is not obviously evident - it really is *always* DNS

Enrichments



@NikitaLohia

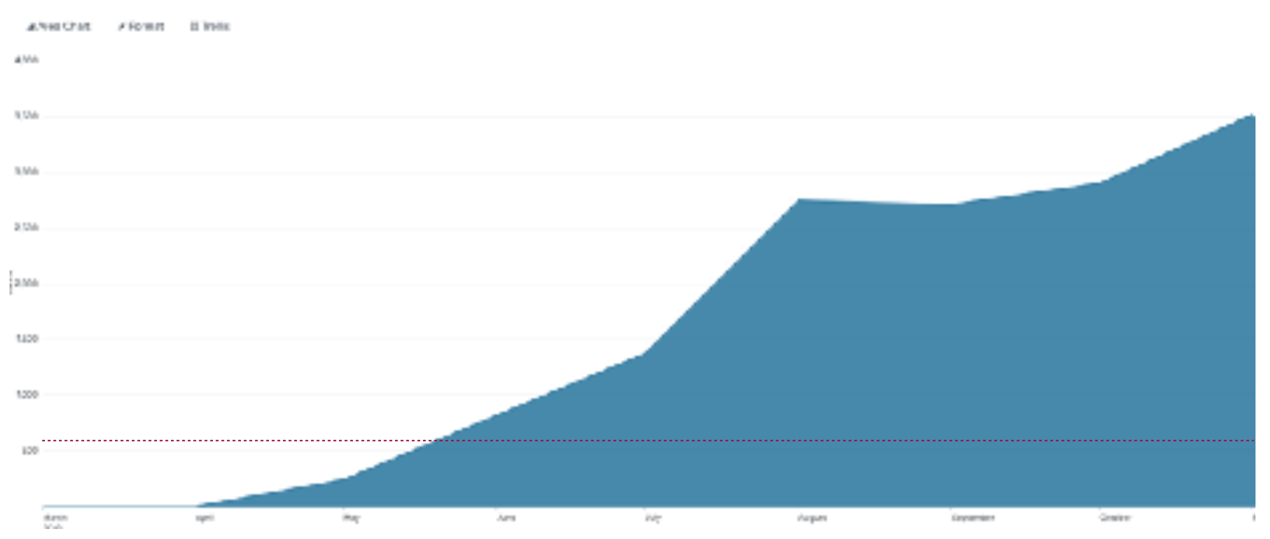
Heimdall, our monitoring platform

FT

We also integrated Change API with our monitoring platform.

This screenshot illustrates a banner on a service's monitoring page if it was released within an hour.

Now, we could immediately co-relate a service monitoring alerting with its recent release.



@NikitaLohia

Change API adoption gradually increasing

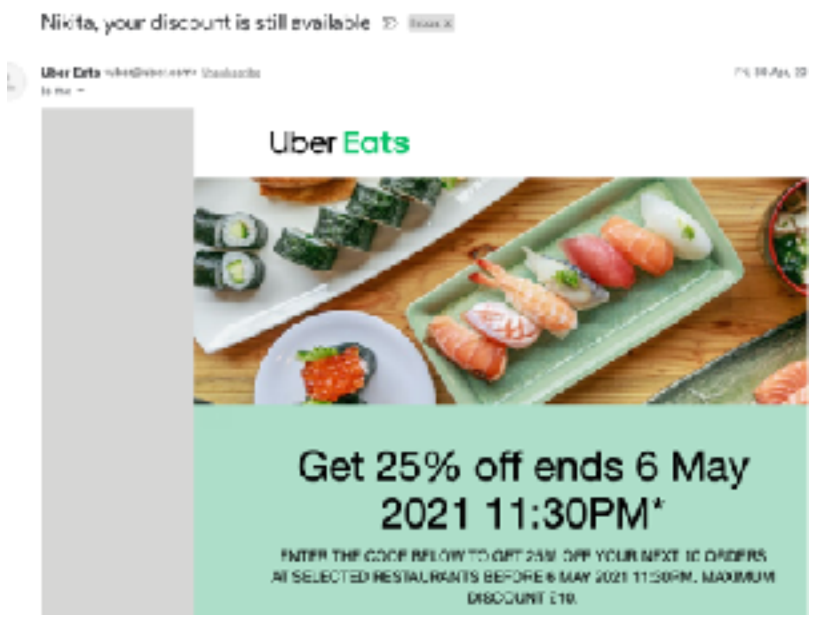
Here is a graph showing the number of Change API events gradually increasing over time. The dotted line here shows legacy CR API usage. We decommissioned the legacy API soon after rolling out Change API fully.

Dare I say it? we surfed our way to the top :)

Use behavioural science to drive change

All through the development of Change API - we were using behavioural science to nudge people towards it..

Use behavioural sciences - Nudges



Nikita, your discount is still available

Uber Eats

Uber Eats

Get 25% off ends 6 May 2021 11:30PM*

ENTER THE CODE BELOW TO GET 25% OFF YOUR NEXT 10 ORDERS AT SELECTED RESTAURANTS BEFORE 6 MAY 2021 11:30PM. MAXIMUM DISCOUNT £10.

@NikitaLohia

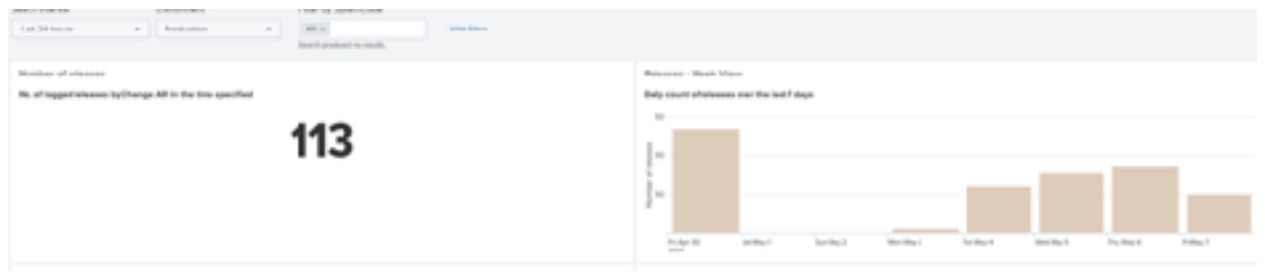
FT

Nudges, once you are aware of them , are everywhere. It is a subtle hint that encourages you to make a certain decision without forcing you.

Take this UberEats email for example, its not asking me to order something from the app directly.

Rather, its encouraging or “nudging” me by trying to remind me I still have a discount available and I should use it …

Use behavioural science- Nudges



@NikitaLohia

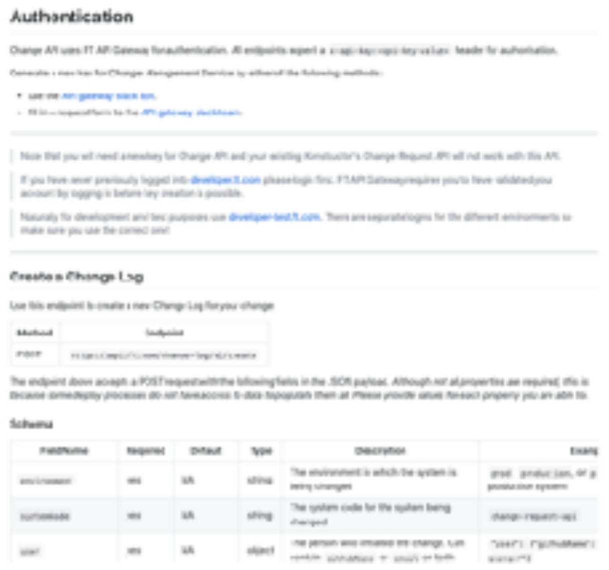
Change API dashboard

A few nudges we used to encourage more teams to use Change API was to start a little bit of a healthy competition. We all love a competition, don't we !

We told teams, did you know that this other team made 50 code releases in the past week? Would you like to find out how many your team did?

We created a dashboard to get people talking about Change API. At the FT, we clock anywhere between a 100 ~ 150 daily releases !

Use behavioural sciences - Make it default



Authentication

Change API uses FT API Gateway for authentication. All endpoints expect a `[illegible]` header for authorisation.

[illegible]

- use the [illegible]
- [illegible] API gateway [illegible]

> Note that you will need a new key for Change API and your existing [illegible] Change Request API will not work with this API.

> If you have never previously logged into developer.ft.com please login first. FT API Gateway requires you to have validated your account by logging in before key creation is possible.

> Naturally for development and test purposes use developer-test.ft.com. There are separate logins for the different environments so make sure you use the correct one!

Create a Change Log

Use this endpoint to create a new Change Log for your change

Method	Endpoint
POST	[illegible]

The endpoint above accepts a POST request with the following fields in the JSON payload. Although not all properties are required, this is because some deploy processes do not have access to data to populate them all. Please provide values for each property you are able to.

Schema

FieldName	Required	Default	Type	Description	Example
environment	yes	NA	string	The environment in which the system is being changed	`prod`, `production`, or [illegible] production system
systemCode	yes	NA	string	The system code for the system being changed	`change-request-api`
user	yes	NA	object	The person who initiated the change. [illegible] `githubName` or `email` or both	[illegible]

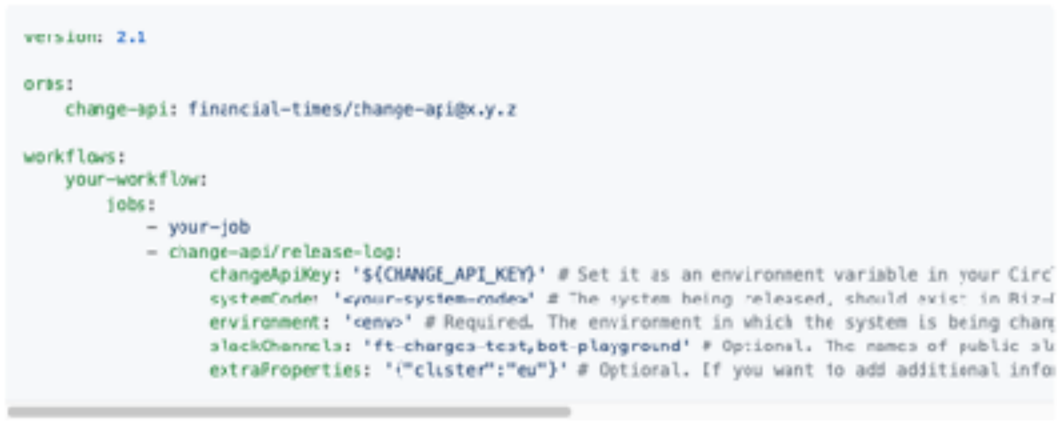
@NikitaLohia

A default nudge is the most powerful nudge. As humans, we tend to be inherently averse to making decisions, even if we know that they are good for us.

We tried to remove all blockers to Change API adoption -

We wrote verbose documentation with step by step instructions on how to integrate Change API with different deployment pipelines.

Use behavioural sciences - Make it default



```
version: 2.1

orbs:
    change-api: financial-times/change-api@x.y.z

workflows:
    your-workflow:
        jobs:
            - your-job
            - change-api/release-log:
                  changeApiKey: '${CHANGE_API_KEY}' # Set it as an environment variable in your Circ
                  systemCode: '<your-system-code>' # The system being released, should exist in Biz-
                  environment: '<env>' # Required. The environment in which the system is being chan
                  slackChannels: 'ft-changes-test,bot-playground' # Optional. The names of public sl
                  extraProperties: '{"cluster":"eu"}' # Optional. If you want to add additional info
```

https://circleci.com/orbs/

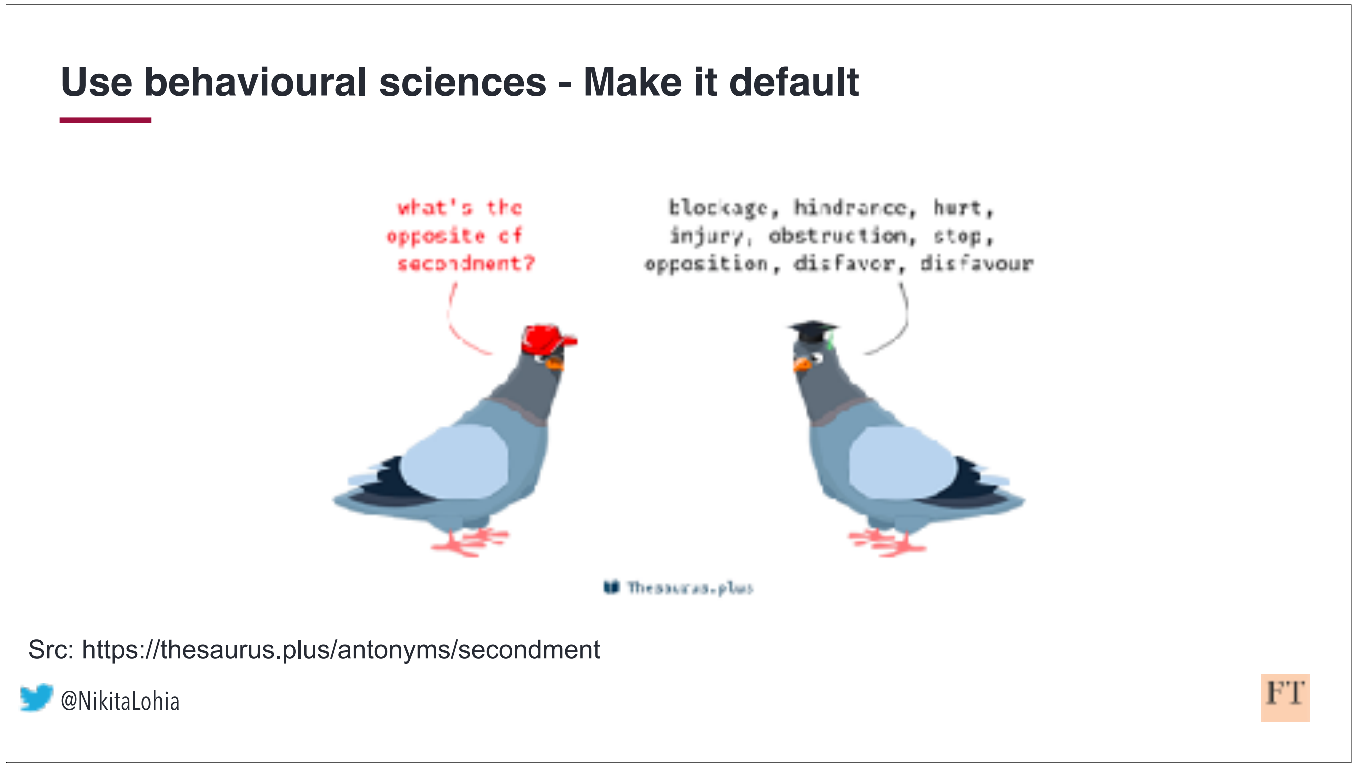
@NikitaLohia

FT

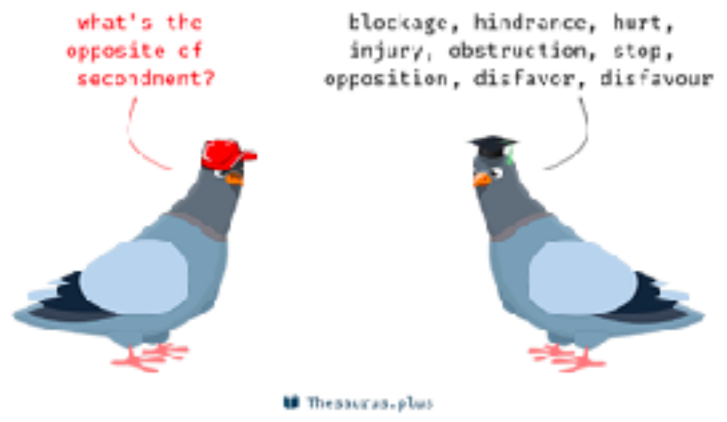
We wrote some tooling.

Since a majority of our teams favour CircleCI we created a Change API Orb which would allow engineering teams to use YAML instead of bash and curl commands in their config files to integrate with Change API. This is what an orb config looks like.

Orbs, for people who don't know, are an easy way to create reusable yaml configs in CircleCI.



Use behavioural sciences - Make it default


what's the opposite of secondment?
blockage, hindrance, hurt, injury, obstruction, stop, opposition, disfavor, disfavour
Thesaurus.plus

Src: https://thesaurus.plus/antonyms/secondment

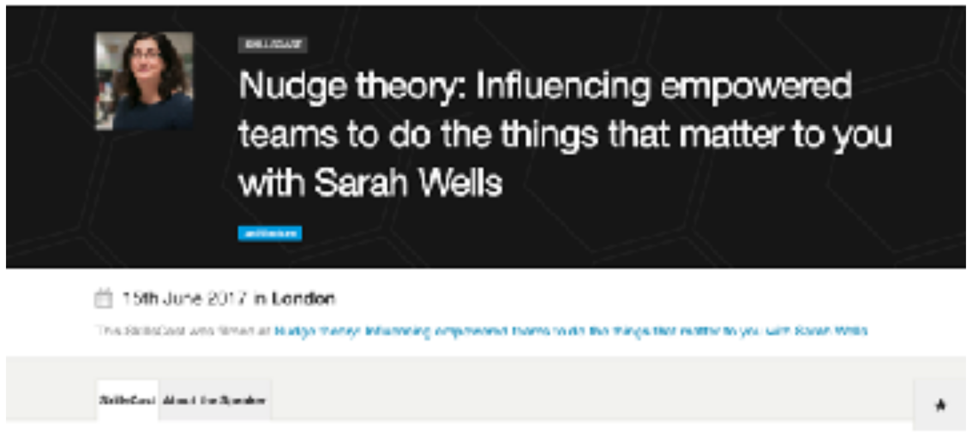
@NikitaLohia

FT

We encourage secondments at the FT - i.e. people temporarily take a break from their home teams and work with other teams in the department to either learn a new skill or help other teams up-skill on a particular tooling or technology.

We used secondments to integrate Change API into hundreds of repos !

Use behavioural sciences



Nudge theory: Influencing empowered teams to do the things that matter to you with Sarah Wells

15th June 2017 in London

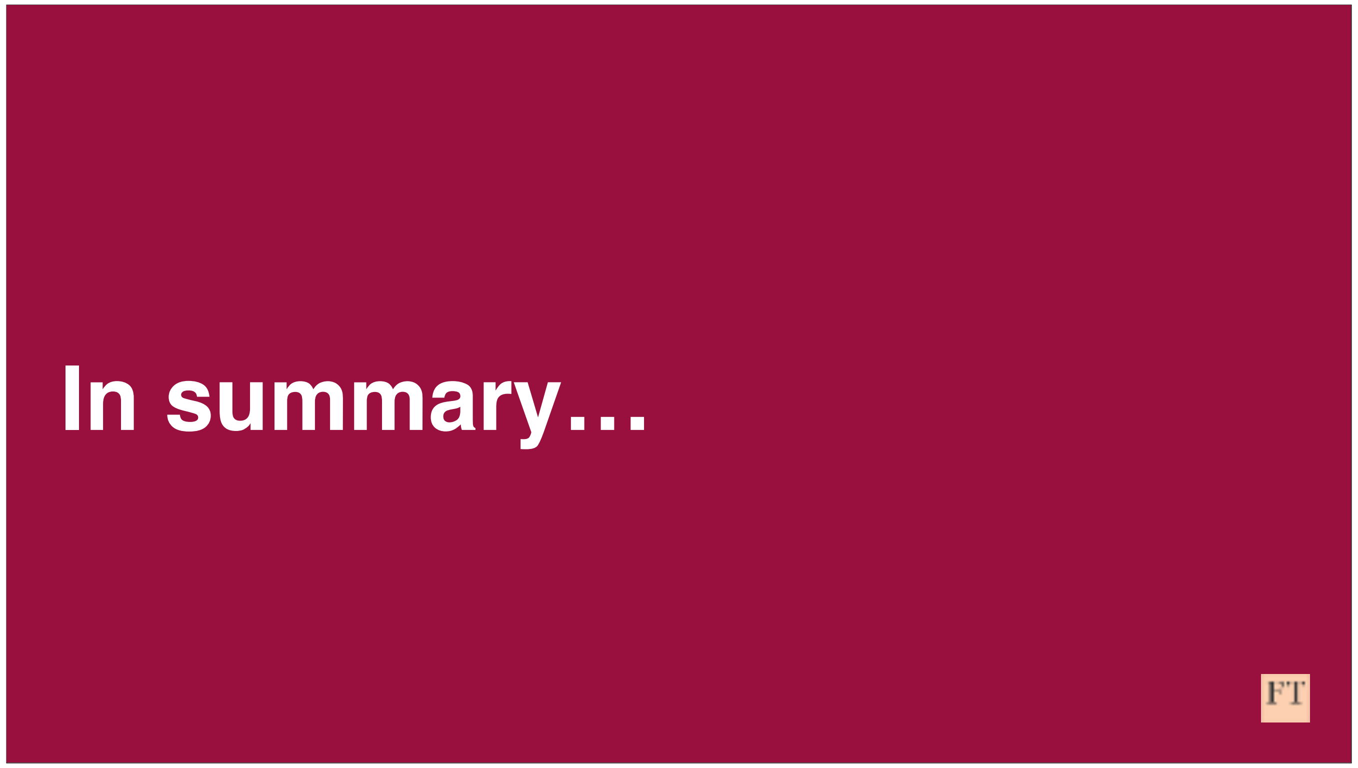
https://skillsmatter.com/skillscasts/9858-nudge-theory-influencing-empowered-teams-to-do-the-things-that-matter-to-you-sarah-wells

@NikitaLohia

FT

There is an excellent talk by my colleague Sarah Wells who talks about Nudge Theory in greater detail.

She also talks about the EAST framework which can be a really good guide to influence behaviour.

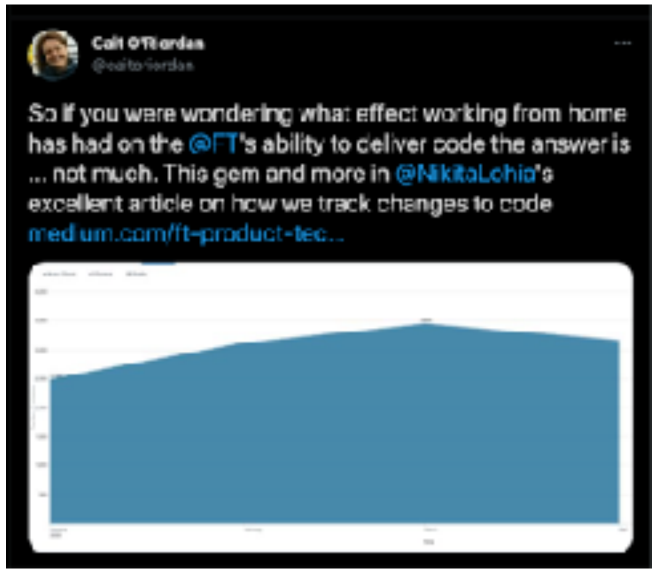


In summary…

So, to summarise -

Hopefully I have managed to convince you that we solve really interesting problems at the FT , and in a really fun way !

We ended up with some really interesting data..

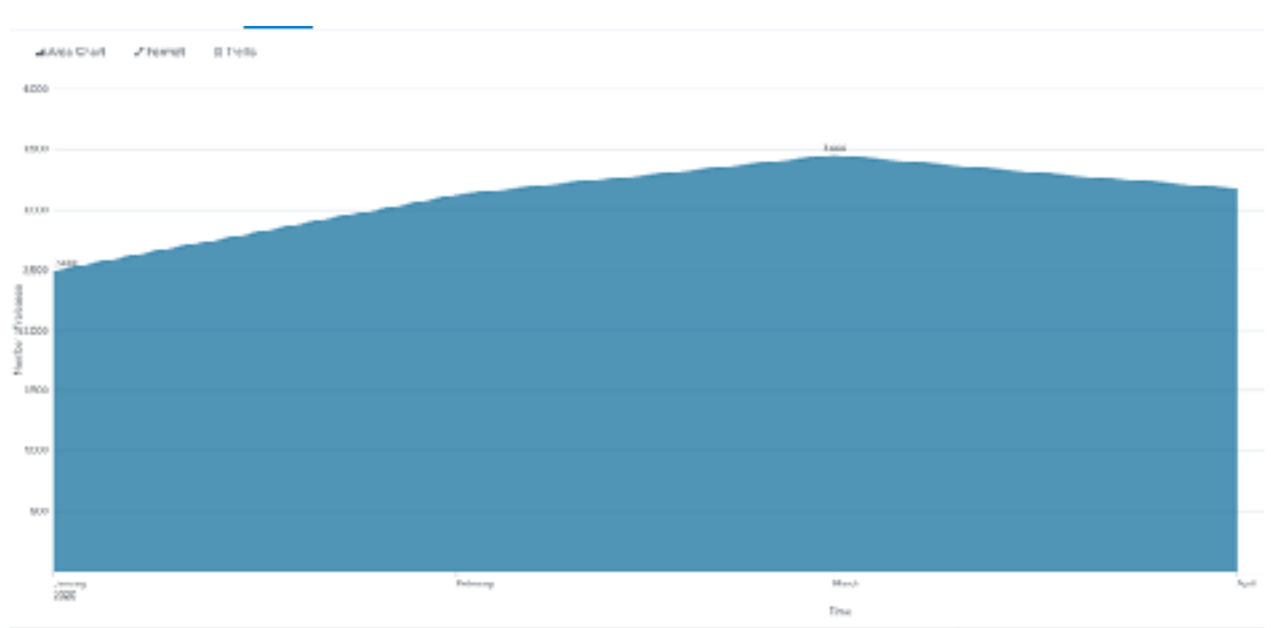

Cait O'Riordan
@caitoriordan

So if you were wondering what effect working from home has had on the @FT's ability to deliver code the answer is ... not much. This gem and more in @NikitaLohia's excellent article on how we track changes to code medium.com/ft-product-tec...

@NikitaLohia
https://medium.com/ft-product-technology/the-advent-of-change-api-8dae0f95245e

FT

Along with the capability to consistently and confidently be aware of what changed in our microservice estate , we were able to answer some really interesting questions like “How has the pandemic affected our code releases?”, and “Do our developers prefer a specific time or day to push changes?”

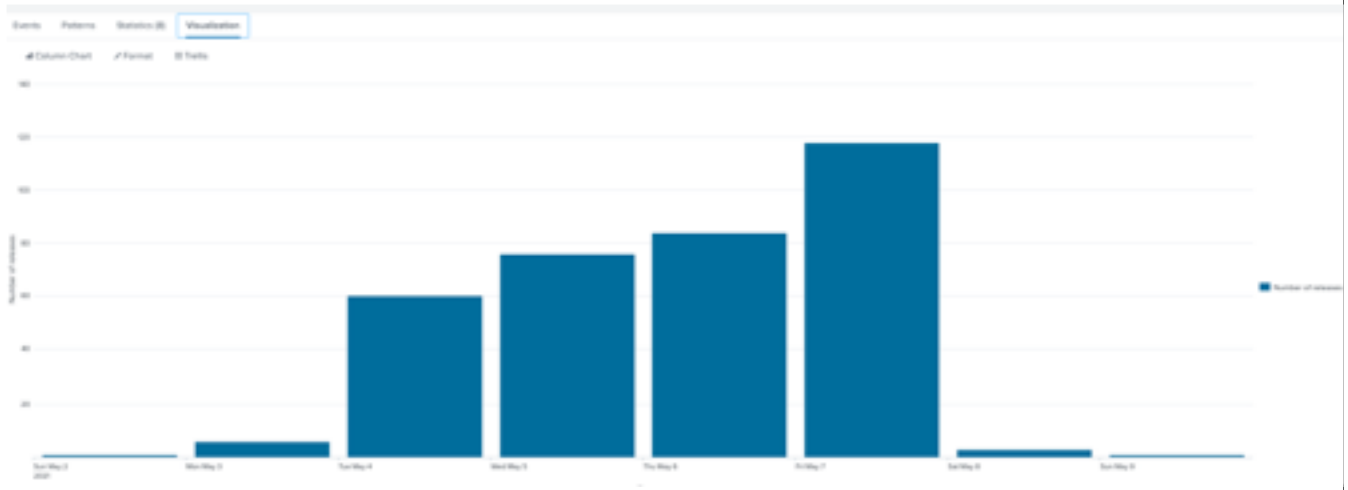


@NikitaLohia

Monthly code releases

FT

Even during the first peak of the pandemic back in 2020, we were pushing out code, business as usual..



@NikitaLohia

Daily code releases

And yes… we also definitely do releases on a Friday ! :)

Thank you!

- https://medium.com/ft-product-technology

@NikitaLohia

FT