

Running a GraphQL Trello Clone without Servers

SKILup 2021

Agenda

1. My 5 Lessons Learned
2. Why Running a Trello Clone
3. How the journey started and important steps
4. Learnings in detail: What went well or was unexpected

Lessons Learned

A blurred background image of a classroom. A male teacher in a plaid shirt stands at the front, facing a class of students. The students are seated at desks, and the teacher is pointing towards a whiteboard or screen. The classroom has green walls and various educational posters.

1. A solid CI/CD pipeline makes live much easier.
2. Test all important parts, because even simple things can break so easily
3. There will be bottlenecks you did not yet notice
4. Embrace change and be ready to throw away or rewrite parts
5. You can get more insights with monitoring, logging and tracing together

I ♥ Trello

The screenshot shows a Trello board interface with a blue header and background. The board is titled "Graphql and React - Planning" and is set to "Personal" and "Private". It features three columns: "Doing", "Done", and a third column with a blue header "Add a list...".

Boards [Search] **Trello** [Add] [Refresh] [Notifications] [Profile]

Graphql and React - Planning [Favorites] [Personal] [Private] [Show Menu]

Doing

- Product 1: Build a Trello like React web
 - Section 1: Graphql basics and starting on graph.cool
 - Building a Trello like React web application, cloud based backend
 - Section 2: Building the UI client and connect per Apollo with server
 - Section 3: Client driven Mutations
 - Section 4: Update the client after changes on the server
 - Section 5: Add User Authentication
 - Section 6: Summary and outlook

Done

- Refine outline
- Build Demo App
- Audio sample
- Buy Audio Equipment
- Create Outline
- Coarse Outline
- Collect material

Add a list...

Home > All Products > All Videos > Web-development > Hands-on Application Building with GraphQL [Video]

Robert Hostlowsky

Hands-on Application Building with GraphQL

Build a Trello-like web application with GraphQL and React



Packt

Hands-on Application Building with GraphQL [Video]

By Robert Hostlowsky

- 1 Getting Started with GraphQL
- 2 Creating Your Own GraphQL Server
- 3 Building the UI Client with a Server Connection
- 4 Working with Client-Driven Mutations
- 5 Subscriptions: Updating the Board on Changes
- 6 Adding User Authentication
- 7 Troubleshooting, Error Handling, and Tuning

About this video

GraphQL is a data-fetching API developed by Facebook, which has been using it for five years; it powers millions of devices and most components of the Facebook and Instagram website. In this course, you will get an introduction into GraphQL as a bridge for React client application to communicate with servers as the missing data-fetching or query language.

In this course, you will learn how to build your own Trello-like web application using GraphQL. The course starts by teaching you GraphQL basics and comparing it with REST; you will then learn to run queries and specify types in its schema system. The course then shows you how to build a GraphQL server and a client UI and connect this Apollo-based client to the server. You will then learn to add features to your board such

Publication date:

July 2018

Publisher

Packt

Duration

6 hours 46 minutes

ISBN

9781788991865



Hands-on Application Building with GraphQL

Section 1

Video 1.1 Intro

Video 1.2 GraphQL/Rest

Video 1.3 ...

Video 1.4

Video 1.5

+ Add a card

Section 2

Video 2.1

Video 2.2

Video 2.3

Video 2.4

Video 2.5

+ Add a card

Section 3

Video 3.1

Video 3.2

Video 3.3

Video 3.4

Video 3.5

+ Add a card

Video 4.1

Video 4.2

Video 4.3

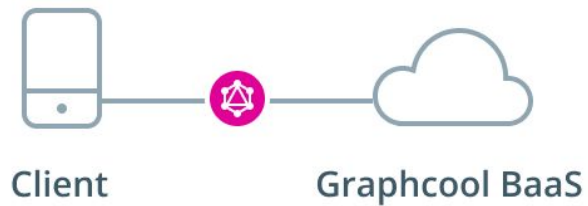
Video 4.4

Video 4.5

<https://www.coolboard.fun>

How it all started...

2017 - Single Page App with a serverless GraphQL Database



MAY 2017

2019

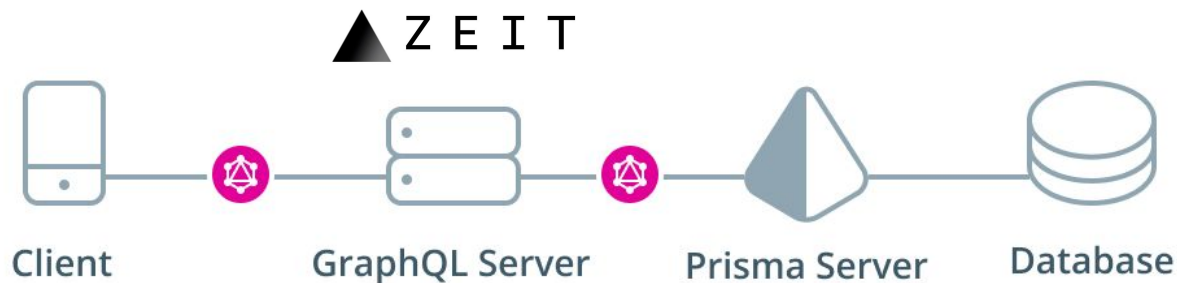


Graphcool



2018: GraphQL Server

µService for handling authorisation + customer-facing API



MAY 2017



Graphcool

JAN 2018



Prisma 1.0 + Prisma Bindings

AUG 2018

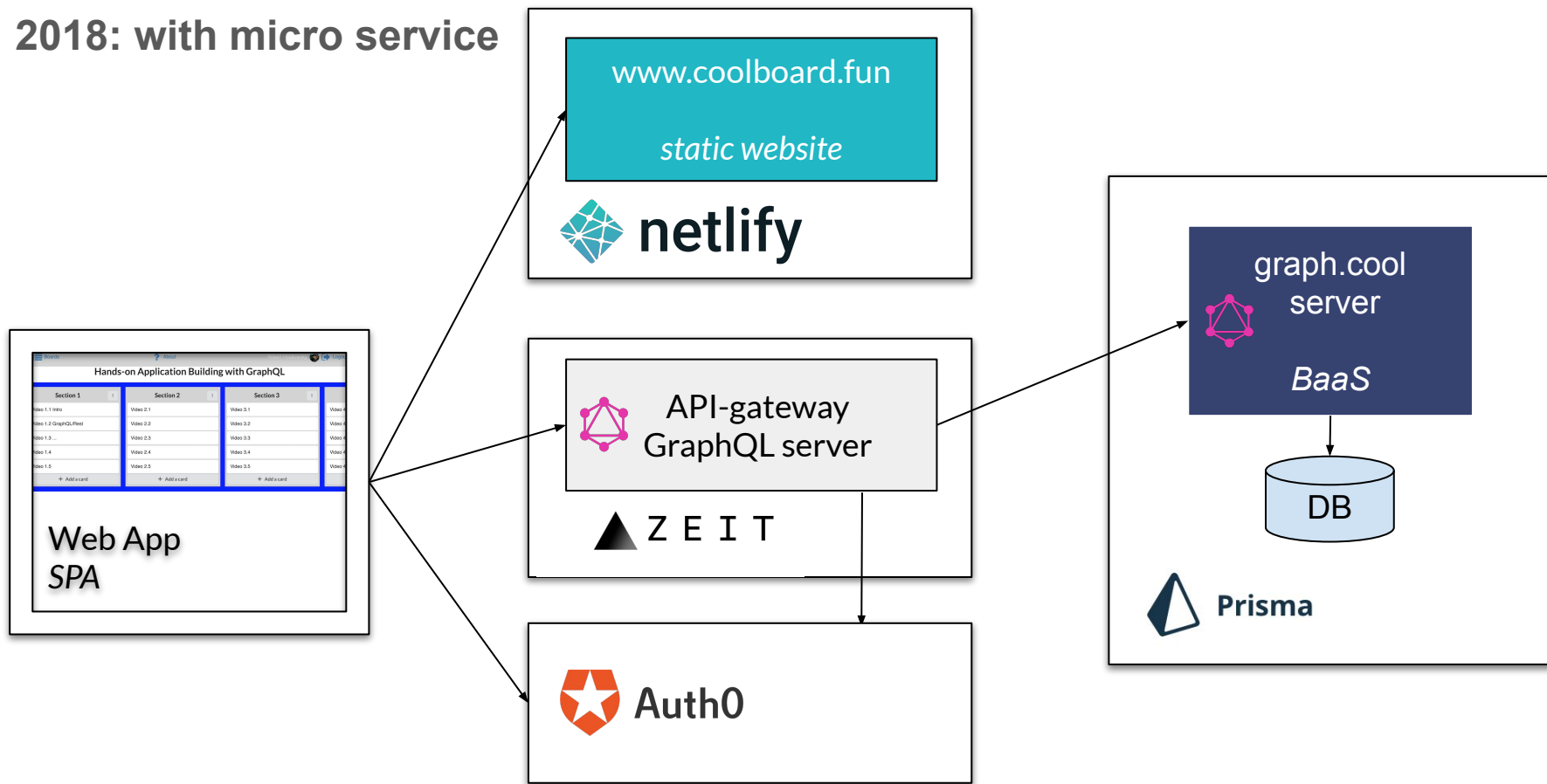


Prisma Client

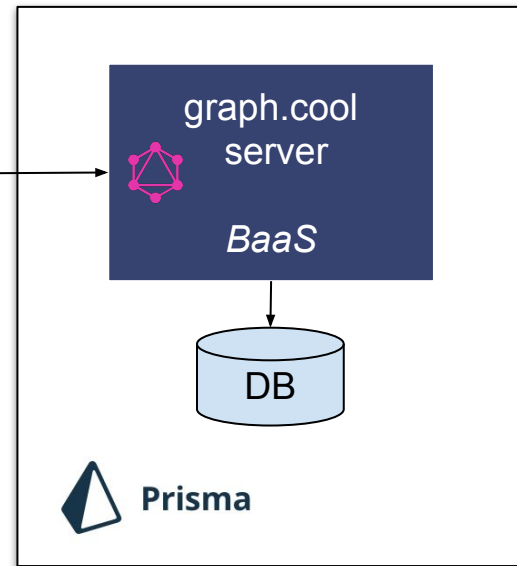
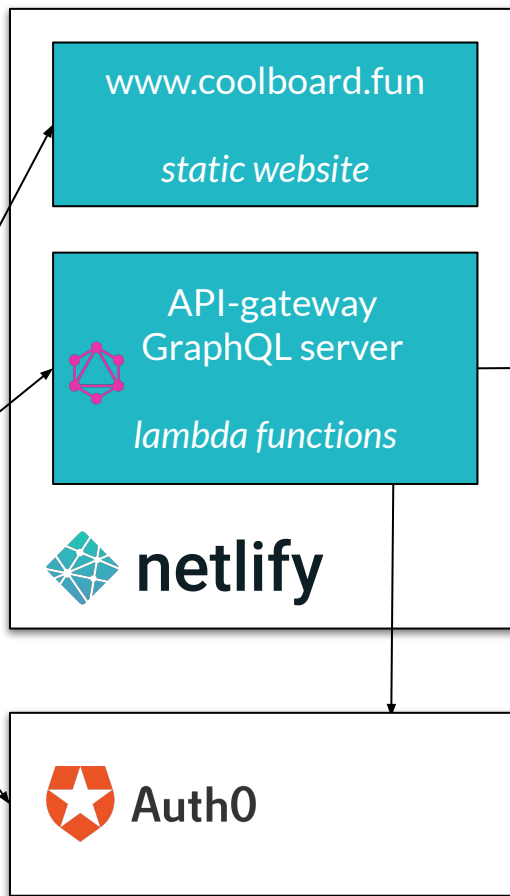
2019



2018: with micro service



2019: Fully Serverless



Learnings:

- **A solid CI/CD pipeline makes live much easier.**
- Test all important parts, because even simple things can break easily!
- There will be bottlenecks, you didn't notice yet
- You can get more insights with monitoring, logging and tracing
- Embrace change and be ready to throw away or rewrite parts

Solid CI/CD pipeline

Deploy each feature branch separately





netlify bot commented 20 days ago • edited ▼



✓ Deploy preview for *coolboard* ready!

🔨 Explore the source changes: [a9b1bbf](#)

🔍 Inspect the deploy logs: <https://app.netlify.com/sites/coolboard/deloys/6005d14df504c400072e5943>

😎 Browse the preview: <https://deploy-preview-633--coolboard.netlify.app>

Filters

is:pr is:open

Labels 12

Milestones 0

New pull request

☐ 9 Open ✓ 637 Closed Author Label Projects Milestones Reviews Assignee Sort

☐ **fix(deps): update dependency @apollo/client to v3.3.8** ✓ 1
#647 opened 2 days ago by renovate bot 0 of 1

☐ **fix(deps): update dependency react-scripts to v4.0.2** ✓ 2
#646 opened 4 days ago by renovate bot 0 of 1

☐ **chore(deps): update babel monorepo to v7.12.13** ✓ 1
#645 opened 5 days ago by renovate bot 0 of 1

☐ **chore(deps): update dependency @types/react to v17.0.1** ✓ 1
#644 opened 5 days ago by renovate bot 0 of 1

☐ **fix(deps): update dependency cypress to v6.4.0** ✓ 2
#643 opened 6 days ago by renovate bot 0 of 1

☐ **fix(deps): update dependency aws-sdk to v2.839.0** ✓ 2
#642 opened 9 days ago by renovate bot 0 of 1

☐ **chore(deps): update dependency core-js to v3.8.3** ✓ 3
#634 opened 20 days ago by renovate bot 0 of 1

Learnings:

- A Solid CI/CD pipeline makes live much easier.
- **Test all important parts**
- There will be bottlenecks, you don't notice yet
- You can get more insights with monitoring, logging and tracing
- Embrace change and be ready to throw away or rewrite parts

Smoke tests for each deployment

Automated browser based tests via



A screenshot of the Cypress dashboard sidebar. At the top, there is a user profile section with a circular placeholder for a profile picture, the text 'Your personal orga...' followed by 'lowsky' on the next line, and a downward arrow icon. Below this is a section with a green checkmark icon and the text 'coolboard-smoketest', with a link 'View all projects' underneath. The next section is titled 'Latest runs' with a blue checkmark icon. Below that is the 'Analytics' section, indicated by a bar chart icon, which contains a list of items: 'Run status', 'Run duration', 'Test suite size', 'Top failures', and 'Slowest tests'. Further down is the 'Project settings' section, indicated by a list icon. At the bottom of the sidebar are two links: 'Support' with a question mark icon and 'Documentation' with a document icon.

Latest runs

fix(deps): update dependency cypress to v5.1.0	000400
fix(deps): update dependency react-apollo-network-status to v5	000400
fix(deps): update dependency styled-components to v5.2.0	000400
fix(deps): update dependency fomantic-ui-css to v2.8.7	000400
fix(deps): update dependency cypress to v5.1.0	000400
fix(deps): update dependency apollo-server-lambda to v2.17.0	000400
chore(deps): update dependency @babel/preset-env to v7.11.5	000400
chore(deps): update dependency graphql-tools to v6.2.2	000400
chore(deps): update dependency @babel/preset-env to v7.11.5	000400
chore(deps): update dependency graphql-tools to v6.2.2	000400
Update dependency @instana/collector to v1.105.1	000400

Learnings:

- A solid CI/CD pipeline makes live much easier.
- Test all important parts, because even simple things break easily
- **There are invisible bottlenecks, you just didn't notice - yet**
- You can get more insights with monitoring, logging and tracing
- Embrace change and be ready to throw away or rewrite parts

There are bottlenecks...

I ran into performance issues, and did performance analysis ...

Blog posts:

[Performance Analysis of a GraphQL application with Instana](#) - analysis

[Performance optimization of a GraphQL app with Instana](#) - fix

Conclusion:

- Distributed systems in the Cloud env behave differently: - Do load testing, Start monitoring early - choose the appropriate tool

Learnings:

- A solid CI/CD pipeline makes live much easier.
- Test all important parts, because even simple things can break so easily
- There are bottlenecks, you didn't notice yet
- **You can get more insights with monitoring, logging and tracing**
- Embrace change and be ready to throw away or rewrite parts



robert-netlify-lambda



Sep 20
Last 10 minutes



✓ No Issues

Analyze Page Loads

Filters Browser OS Country Subdivision Meta Window Width

Summary Speed Resources HTTP Requests JS Errors Geography Custom Events Pages Alerts Configuration

Page Loads

25

Page Transitions

31

onLoad Time (mean)

883ms

onLoad Time (90th)

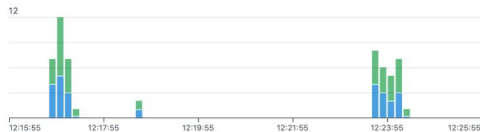
1.77s

onLoad Time (95th)

2.07s

Page Views

Page Loads Page Transitions



Releases
Alerts

JS Errors

JS Errors



Releases
Alerts

onLoad Time

onLoad Time



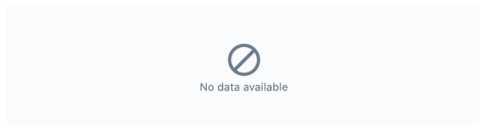
Releases
Alerts

Geography



Top JS Errors

Occurrences Affected Users



Top Pages

Page Views onLoad Time Errors

Home	26
boards	20
login	2
logout	2
root	2

View all pages

ADD ALERT



Result 10 Page Loads



Path

/boards	2020-09-20 12:17:19	426ms
/boards	2020-09-20 12:17:07	686ms
/boards	2020-09-20 12:17:03	463ms
/boards	2020-09-20 12:17:02	287ms
/boards	2020-09-20 12:16:55	977ms
/boards	2020-09-20 12:14:18	427ms
/boards	2020-09-20 12:14:06	397ms
/boards	2020-09-20 12:14:02	208ms
/boards	2020-09-20 12:14:00	981ms
/boards	2020-09-20 12:13:52	1,452ms



boards on https://deploy-preview-522--coolboard.netlify.app



User Information



No user data defined [Learn how to add user data](#)

Browser Electron 8
Window Dimensions 1000x660
Preferred Languages en-US
IP Address 3.90.110.0
Effective Connection Ty... 4g
Session ID 33b261342a3ab6ba

User Location

Ashburn
Virginia, United States, North America



Meta

No meta data defined. Meta data can be used to transport information about the deployment or settings. Meta data is available as filter and grouping within the analyze area.

[Learn how to add meta data](#)

Activity

Pages

All

Search



Types

All XHR JS CSS Img Font Doc Tra Err Cus Other



boards



Request
POST /.netlify/functions/graphql

Start Time
707ms

Retrieval Time
642ms

[View Backend Trace](#)



Request
POST /.netlify/functions/graphql

Start Time
755ms

Retrieval Time
910ms

[View Backend Trace](#)

board ckfay558k445y0a356rv2wx5u



Request
query board (POST /.netlify/functions/graphql)

Start Time
2.16s

Retrieval Time
919ms

[Show more](#)

[View Backend Trace](#)



Request
mutation deleteListsOfBoard (POST /.netlify/function...

Start Time
3.20s

Retrieval Time
930ms

[View Backend Trace](#)



Request
POST /.netlify/functions/graphql

Start Time
3.38s

Retrieval Time
926ms

[View Backend Trace](#)



Request
...

Start Time
...

Retrieval Time
...

[View Backend Trace](#)



Result 15 Traces



Trace Timestamp Latency

05de03ba676a65856bcd25c60d...	2020-09-20 12:23:55	5ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:23:54	718ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:23:45	3ms
162ed33d5d8b038e9e76227f5c...	2020-09-20 12:17:11	561ms
162ed33d5d8b038e9e76227f5c...	2020-09-20 12:17:02	254ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	676ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	758ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	728ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	757ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	776ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:52	726ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:51	584ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:50	578ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:47	577ms
05de03ba676a65856bcd25c60d...	2020-09-20 12:15:46	820ms

Unspecified Trace ID: dccd1e192050b662

Download

The root call of the trace is missing or has not yet arrived in the processing pipeline.

Parent span is missing for some of the entry spans in this trace during the processing.

Sub Calls

2

Erroneous Calls

0

Error Logs

0

Warn Logs

0

Latency

N/A

Corresponding Website Activity

This trace is caused by activity on the robert-netlify-lambda website.

Hide Website Information

View Website Activity

User Information



No user data defined

Learn how to add user data

Browser Electron 8

Window Dimensions 1000x660

Preferred Languages en-US

IP Address 3.90.110.0

Effective Connection Ty... 4g

Session ID 33b261342a3ab6ba

User Location

Ashburn

Virginia, United States, North America



Meta

No meta data defined. Meta data can be used to transport information about the deployment or settings. Meta data is available as filter and grouping within the analyze area.

Learn how to add meta data

Timeline

Started: 2020-09-20 12:17:10

Root call not yet received

162ed33d5d8b038e9e76227f5c7dcd4fc5b2d1a7de6d1b7a37be1ef605bad26b:SLATEST

POST /lowsky-dcfacd/coolboardsecure/prod

HTTP POST /lowsky-dcfacd/coolboardsec...

Self

Network

Waiting

276ms (100%)

Total

276ms

Technology

POST /lowsky-dcfacd/coolboardsecure/prod HTTP

SOURCE

162ed33d5d8b038e9e76227f5c7dcd4fc5b2d1a7de6d1b7a37be1ef

Details & Stack Trace

Type HTTP Call

Category http

Host eu1.prisma.sh

Request Path /lowsky-dcfacd/coolboardsecure/prod

URL https://eu1.prisma.sh/lowsky-dcfacd/coolboardsecure/prod

Method POST

Status Code 200 - OK

StackTrace

```
<anonymous> in /var/task/src/node_modules/http-link-data-loader/node_modules/node-fetch/lib/index.js
new Promise in /var/task/src/node_modules/http-link-data-loader/node_modules/node-fetch/lib/index.js
Object.fetch in /var/task/src/node_modules/http-link-data-loader/node_modules/cross-fetch/dist/index.js
Object.fetch in /var/task/src/node_modules/http-link-data-loader/dist/src/BatchedGraphQLClient.js:40
Object.next in /var/task/src/node_modules/http-link-data-loader/dist/src/BatchedGraphQLClient.js
<anonymous> in /var/task/src/node_modules/http-link-data-loader/dist/src/BatchedGraphQLClient.js
new Promise in /var/task/src/node_modules/http-link-data-loader/dist/src/BatchedGraphQLClient.js:11
awaiter in /var/task/src/node_modules/http-link-data-loader/dist/src/BatchedGraphQLClient.js:11
Please note: Source code can only be retrieved for processes that are still under monitoring by Instana.
```

Infrastructure Unknown

DESTINATION POST /lowsky-dcfacd of eu1.prisma.sh

Learnings:

- A solid CI/CD pipeline makes live much easier.
- Test all important parts, because even simple things can break so easily
- There will be bottlenecks, you didn't notice yet
- You can get more insights with monitoring, logging and tracing
- **Embrace change and be ready to throw away or rewrite parts**


Be prepared to throw away and rewrite parts


MAY 2017

JAN 2018

AUG 2018

2019



 ZEIT

Hello,

The time has come for us to **sunset ZEIT Now 1.0** and focus our efforts on ZEIT Now 2.0.

Upgrade to Prisma 2

want to

Takeaways and Recommendations

1. CI/CD pipeline with smoke tests for each deployment
2. Find bottlenecks early - run load tests
3. Observability can be live saving. Do it properly with the right tool
4. Be prepared to throw away and rewrite code to react quickly
5. GraphQL is a great technology, get used to it.

Thanks for listening 🙏

Looking forward to your
questions!

[#rhosts](#)

robert.hostlowsky@instana.com

github.com/lowsky



Thanks to

- Class room Photo by [NeONBRAND](#) on [Unsplash](#)

<https://unsplash.com/photos/zFSo6bnZJTw>

- Logo bricks:

<http://pngimg.com/imgs/miscellaneous/lego/>