



THE DEVELOPER'S CONFERENCE

Trilha – Python

Bruno Gomes França e Yuri Piratello
Chapter Lead e Dev Senior na Stoodi



Quem somos?

Bruno França

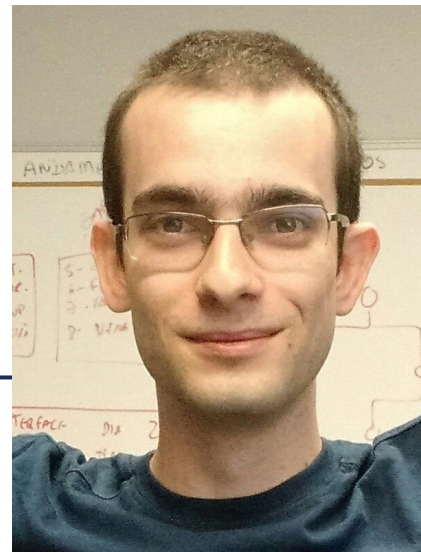
Chapter Lead no
Stoodi



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

Yuri Piratello

Dev Sênior no
Stoodi



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



Otimizando sistemas em Python com Kibana, Elasticsearch e APM

Julho 2019



Nossa Stack



PostgreSQL



elasticsearch



redis

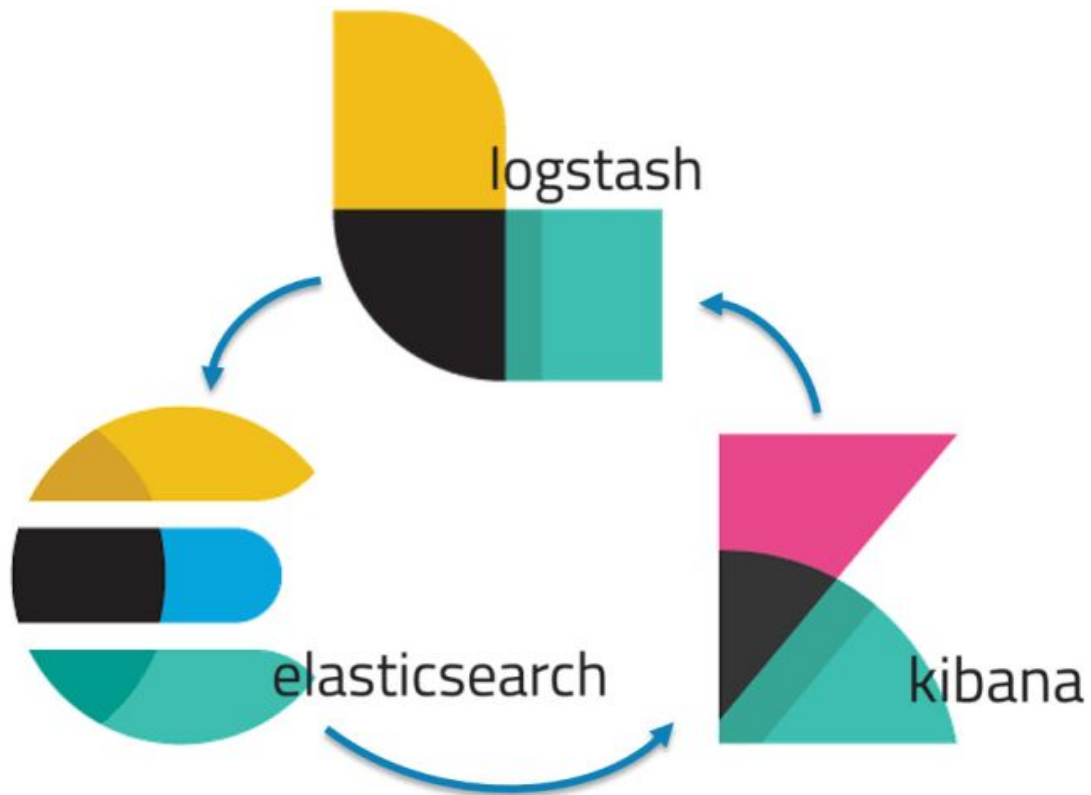


DynamoDB

E mais...



Monitoria, quase um...





Stack de Monitoria

Elasticsearch - Engine de pesquisa e análise.

Kibana - Interface visual para acesso aos dados do Elasticsearch.

APM - Gestão de desempenho de aplicações (Application Performance Management).



E agora?

Ok, mas e como isso funciona?



Kibana

- kibana
- Discover
- Visualize
- Dashboard
- Timelion
- Canvas
- Machine Learning
- Infrastructure
- Logs
- APM
- Dev Tools
- Monitoring
- Management
- Default

APM / stodi-prod / Transactions / request

APM feedback

Auto-refresh



June 13th 2019, 15:48:28.342 to June 13th 2019, 17:48:28.342



stodi-prod

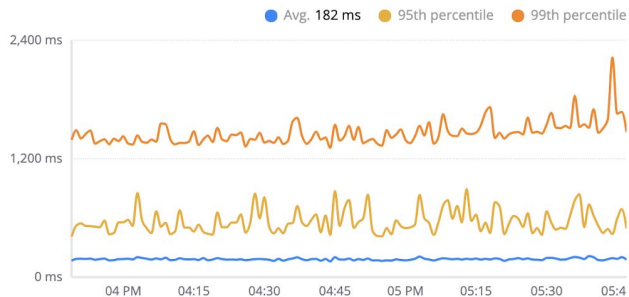
Integrations

Transaction type:

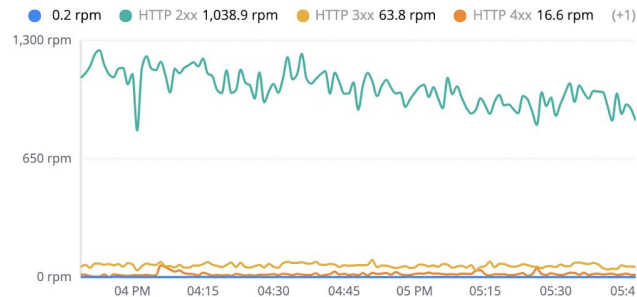
Request

Errors

Response times



Requests per minute



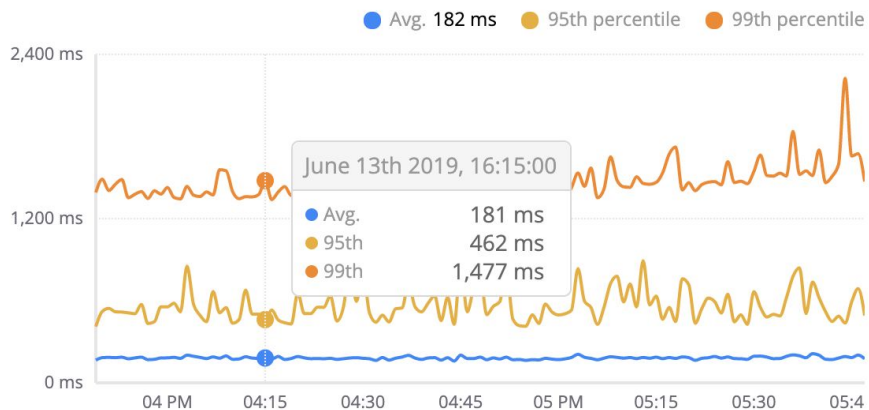
Request

Name	Avg. resp. time	95th percentile	Req. per minute	Impact
GET <code>exercises.views.site_view.subject_exercise</code>	152 ms	232 ms	160.5 rpm	
GET <code>area.views.mainViews.lesson</code>	288 ms	925 ms	65.8 rpm	

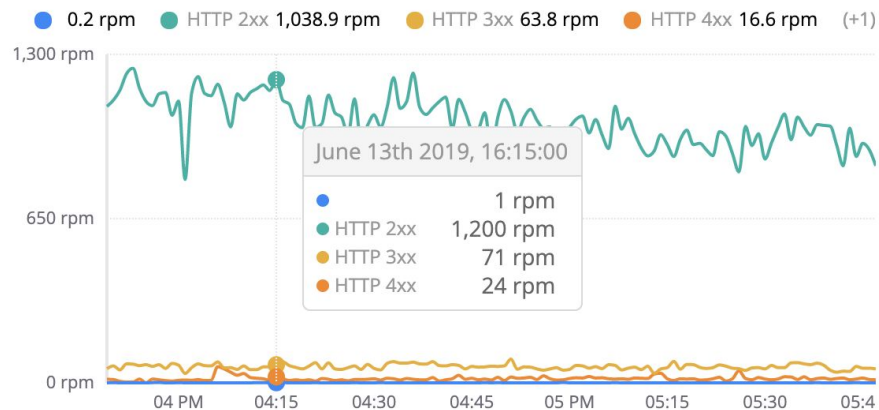


Kibana - Gráficos

Response times



Requests per minute





Kibana - Lista de Endpoints

Request

Name	Avg. resp. time	95th percentile	Req. per minute	Impact ↓
GET exercises.views.site_view.subject_exercise	152 ms	232 ms	160.5 rpm	
GET area.views.mainViews.lesson	288 ms	925 ms	65.8 rpm	
GET exercises.views.site_view.video_modal	1,100 ms	2,301 ms	15.9 rpm	
POST area.views.mainViews.can_watch_this_video	266 ms	1,392 ms	59.3 rpm	
GET video.views.siteView.video	676 ms	1,799 ms	19.3 rpm	
GET main_page.views.generalViews.index	173 ms	294 ms	52.0 rpm	
GET exercises.views.site_view.subject_exercise_list	177 ms	272 ms	50.3 rpm	
GET subjectPanel.views.areas	233 ms	492 ms	31.7 rpm	
GET area.views.mainViews.lesson_exercise	187 ms	299 ms	37.5 rpm	
GET exercises.views.site_view.exercise	116 ms	171 ms	59.5 rpm	
GET subjectPanel.views.area	252 ms	399 ms	24.9 rpm	

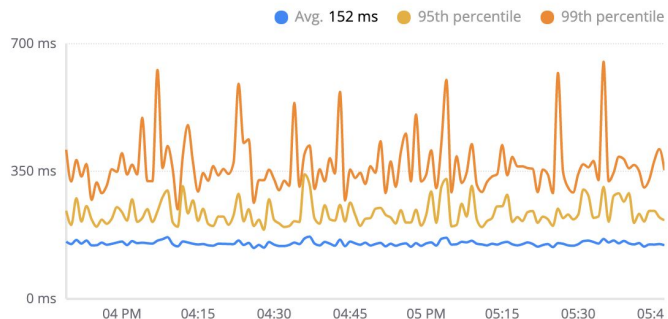


Kibana - Detalhes Endpoint

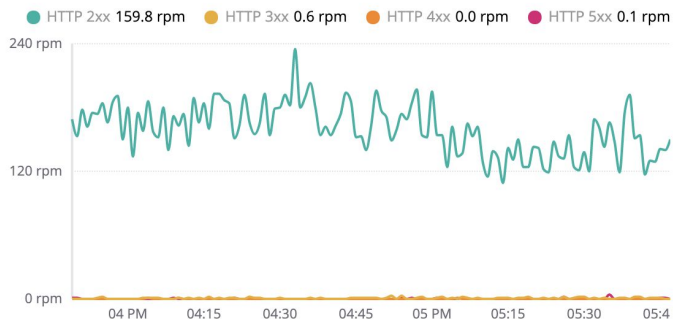
GET exercises.views.site_view.subject_exercise

Search transactions and errors... (E.g. transaction.duration.us > 300000 AND context.response.status_code >= 400)

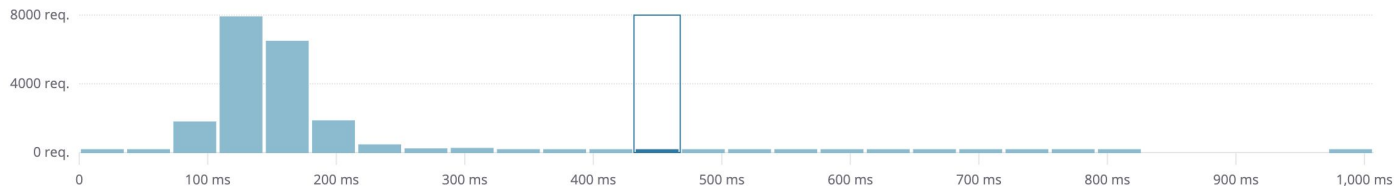
Response times



Requests per minute



Response time distribution



Transaction sample

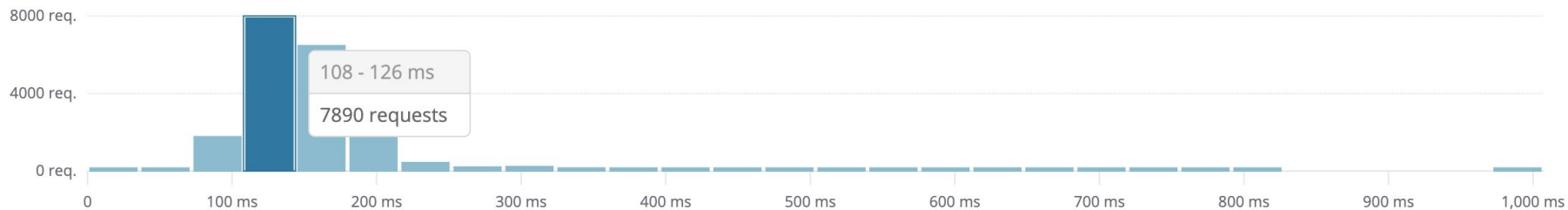
[View transaction in Discover](#)

[View full trace](#)

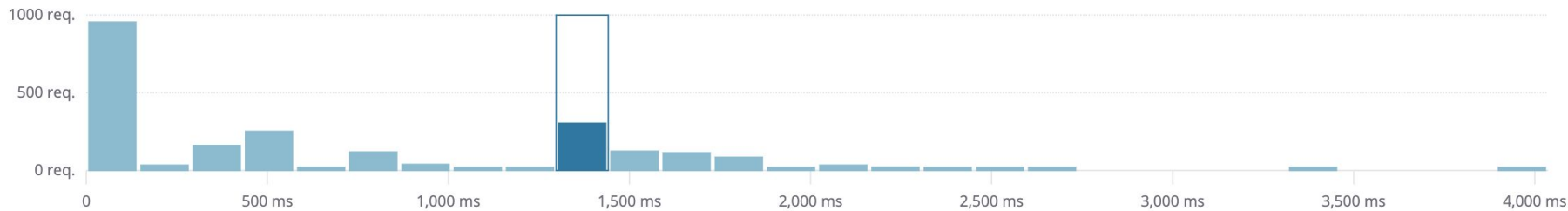


Kibana - Response time distribution

Response time distribution ?



Response time distribution ?





Kibana - Transaction Sample

Transaction sample

[View transaction in Discover](#)

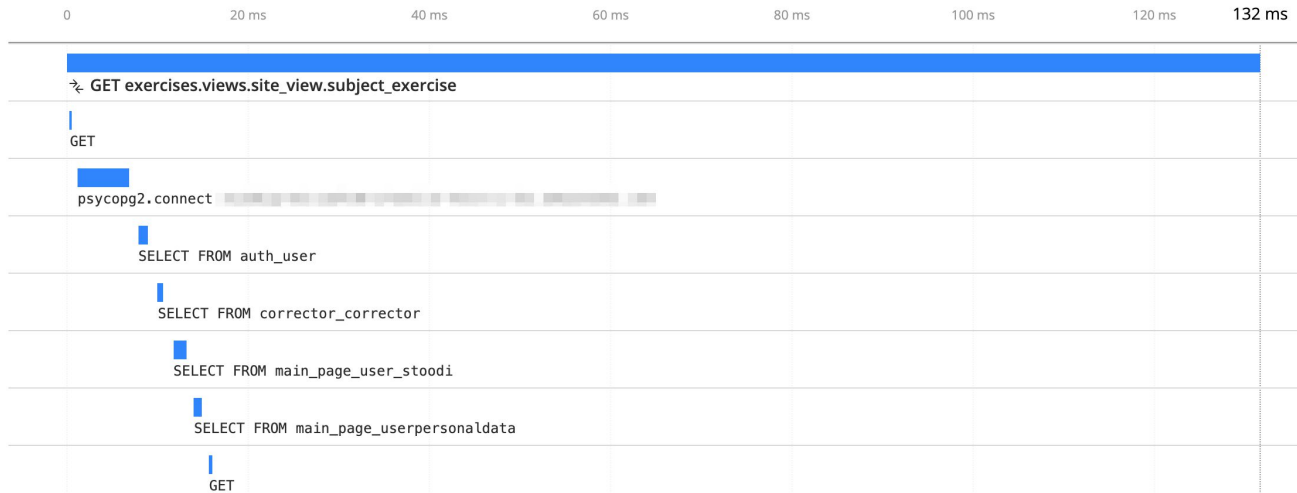
[View full trace](#)

Timestamp	URL		
a month ago (June 13th 2019, 16:02:53.889)	https://www.stoodi.com.br/exercicios/historia/ditadura-militar/uern-2015-observ...		
Duration	% of trace	Result	User ID
132 ms	100.00%	HTTP 2xx	2440783

[Timeline](#) [Request](#) [Response](#) [System](#) [Service](#) [Process](#) [User](#) [Tags](#) [Custom](#)

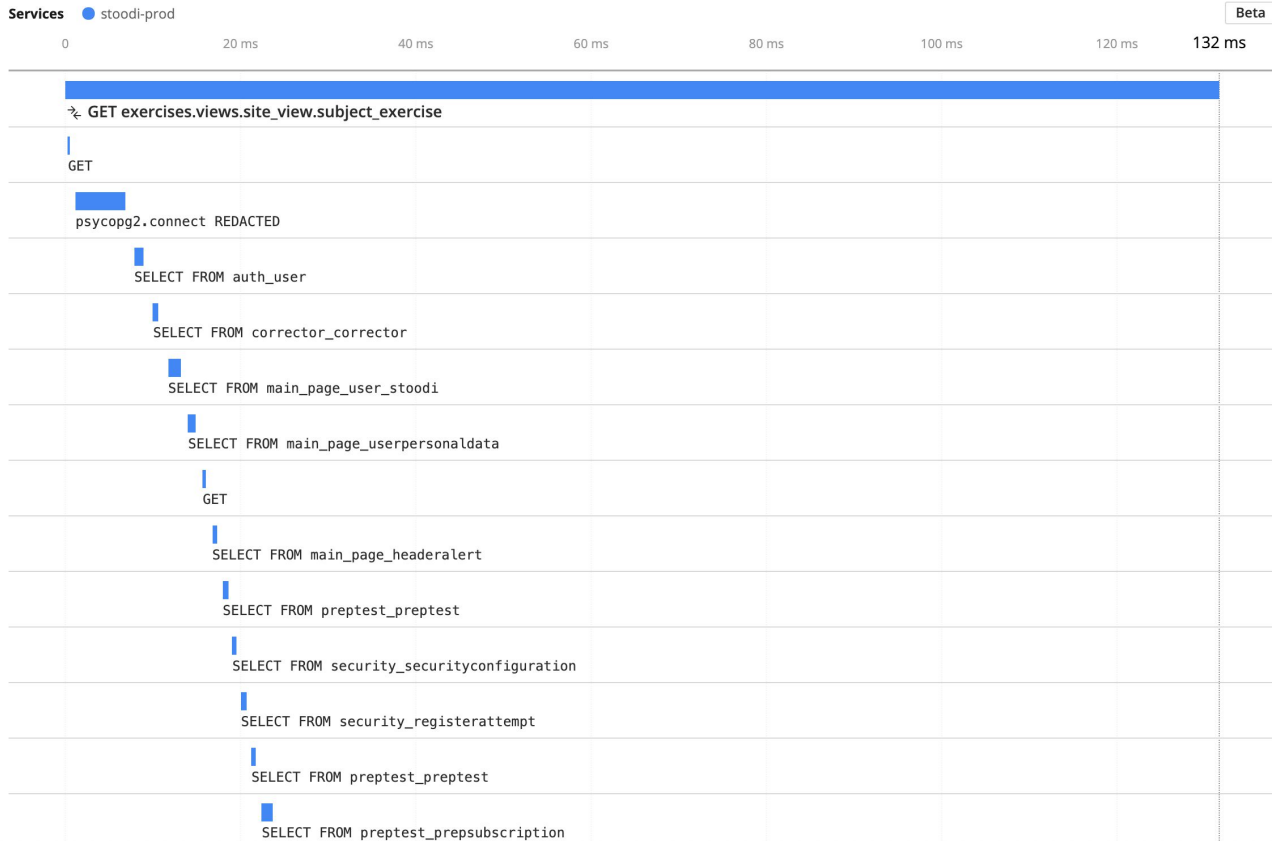
Services ● stoodi-prod

Beta



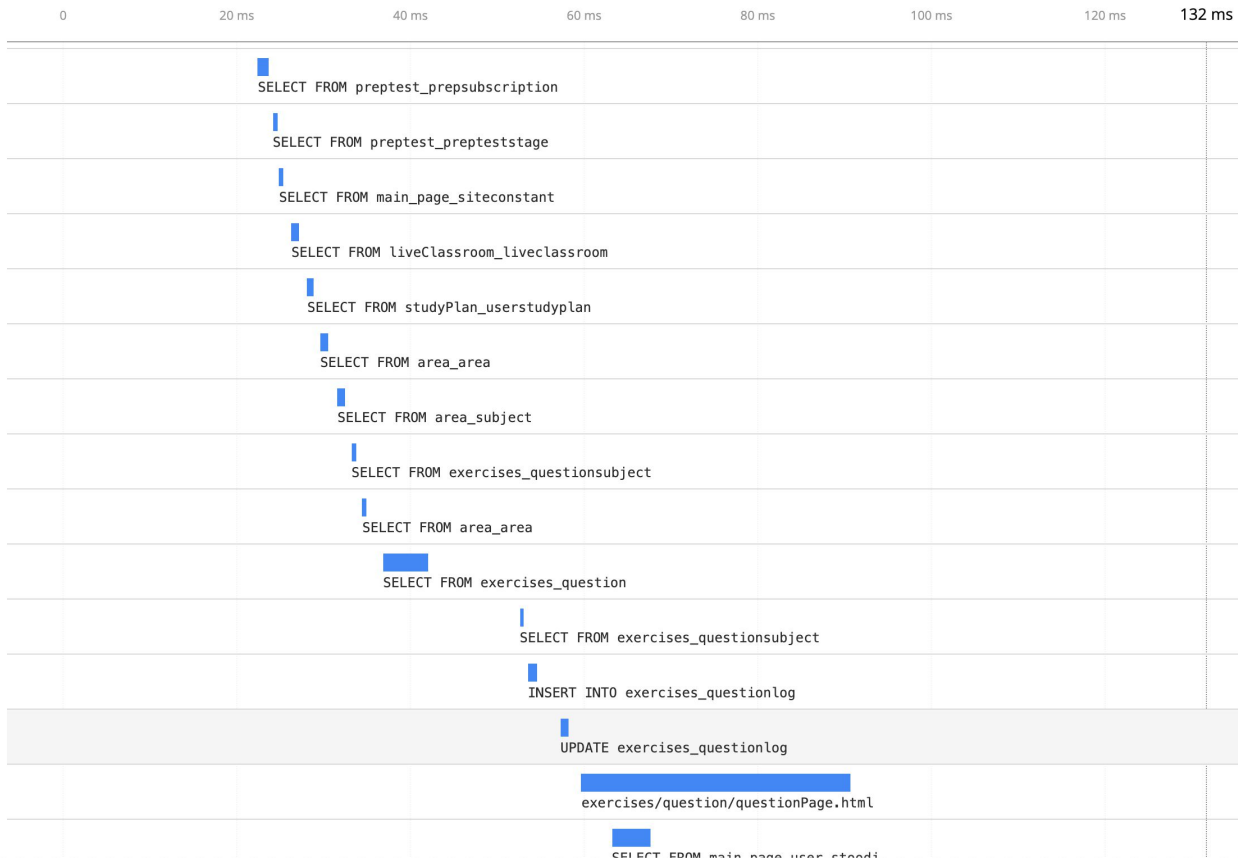


Kibana - Timeline



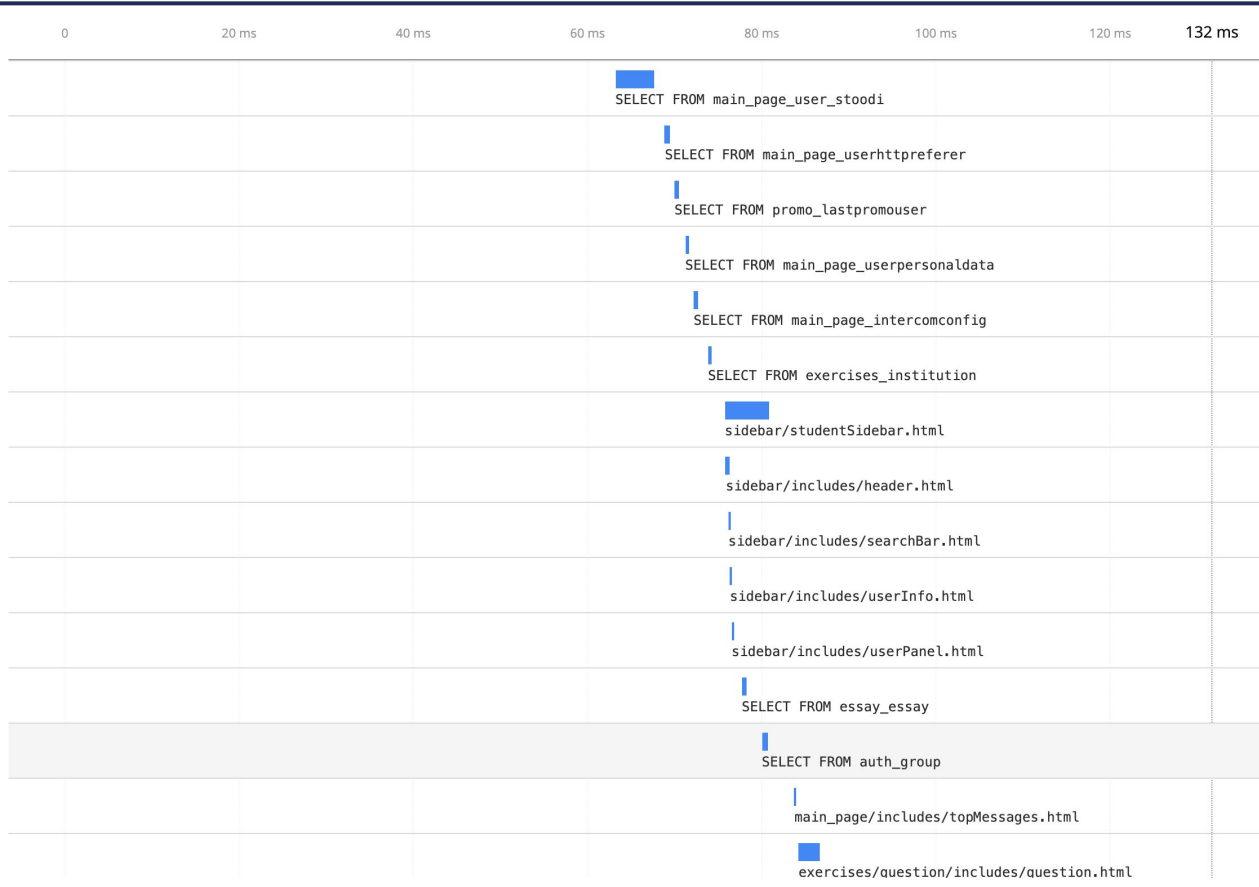


Kibana - Timeline





Kibana - Timeline





Kibana - Span Details

Span details [View span in Discover](#) ✕

Service	Transaction
stoodi-prod	GET exercises.views.site_view.subject_exercise

Name	Type
exercises/question/questionPage.html	template

Duration	% of transaction
40 ms	27.85%

Stack traces
3 library frames

```
exercises/views/site_view.py in exercise_core at line 263
```

```
exercises/views/site_view.py in subject_exercise at line 209
```

5 library frames



Kibana - Span Details

Span details [View span in Discover](#)

Service	Transaction
stoodi-prod	GET exercises.views.site_view.subject_exercise
Name	Type
SELECT FROM area_subject	DB
Duration	% of transaction
1 ms	0.90%

Database statement

```
SELECT "area_subject"."id", "area_subject"."area_id",
"area_subject"."sub_area_id", "area_subject"."order",
"area_subject"."name", "area_subject"."created_at",
"area_subject"."updated_at", "area_subject"."has_document",
"area_subject"."url", "area_subject"."description",
"area_subject"."has_summary", "area_subject"."has_exam",
"area_subject"."meta_description", "area_subject"."thumb_url",
"area_subject"."is_introduction", "area_subject"."order_group",
"area_subject"."enem_exam_id", "area_subject"."average_exercise_time",
"area_subject"."exercise_success_rate",
"area_subject"."time_factor_video", "area_subject"."active" FROM
"area_subject" WHERE ("area_subject"."active" = %s AND
"area_subject"."url" = %s AND "area_subject"."area_id" = %s)
```



Exemplos

The screenshot displays the Stoodi website interface. On the left is a dark blue sidebar menu with the Stoodi logo and a search bar. The main content area has a dark teal header with the text 'MATÉRIAS' and a grid of subject cards. Each card features an icon, the subject name, and statistics for video lessons and exercises.

Stoodi | MATÉRIAS

Buscar

Bruno França

- MATÉRIAS
- PLANO DE ESTUDOS
- CORREÇÃO DE REDAÇÃO
- AULAS AO VIVO
- SIMULADOS

EXPLORAR

- Provas
- Vídeoaulas
- Banco de Exercícios
- Resumos Teóricos
- Downloads

BLOG DO STOODI

5537 vídeoaulas | 30200 exercícios

Subject	Icon	Video Lessons	Exercises
FÍSICA	Atom	385	2798
MATEMÁTICA	\sqrt{x}	798	3983
QUÍMICA	Flask	593	1452
BIOLOGIA	DNA	795	5571
LITERATURA	Leaf	125	1723
GRAMÁTICA	A	486	1715
REDAÇÃO	Pen	278	3
HISTÓRIA	Column	615	4545
GEOGRAFIA	Globe	466	4507
ATUALIDADES	Open Book	50	10
FILOSOFIA	Brain	178	1618
SOCIOLOGIA	People	395	786
ARTE	Paintbrush	138	673
INGLÊS	In	305	1201
ESPAANHOL	Es	218	1098



Exemplos

Antes

GET <code>exercises.views.site_view.subject_exercise_pdf_list</code>	3,826 ms	11,359 ms	3.6 rpm	
GET <code>summary.views.siteView.DownloadSummaryPDF</code>	6,992 ms	43,559 ms	2.7 rpm	

Depois

GET <code>exercises.views.site_view.subject_exercise_pdf_list</code>	1,098 ms	8,028 ms	1.8 rpm	
GET <code>summary.views.siteView.DownloadSummaryPDF</code>	139 ms	199 ms	1.4 rpm	



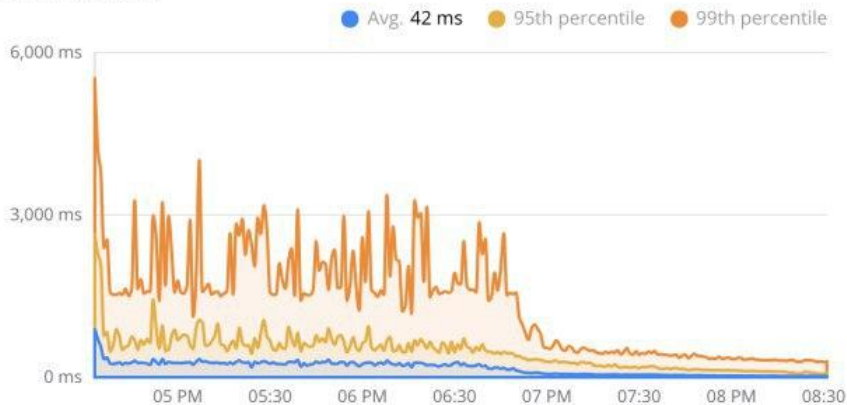
Exemplos

stodi-prod

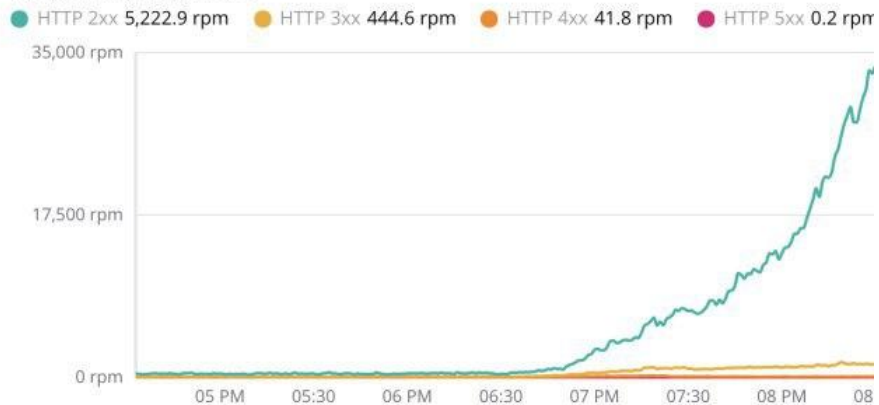
Request

Errors

Response times



Requests per minute



Request



O que preciso?

Elasticsearch

<https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started-install.html>

Kibana

<https://www.elastic.co/guide/en/kibana/4.6/setup.html>

APM

<https://www.elastic.co/guide/en/apm/get-started/6.5/install-and-run.html>



Como faço no meu projeto Django?

```
$ pip install elastic-apm
```

```
INSTALLED_APPS = (  
    # ...  
    'elasticapm.contrib.django',  
)  
  
ELASTIC_APM = {  
    'SERVICE_NAME': '<SERVICE-NAME>',  
    'SECRET_TOKEN': '<SECRET-TOKEN>',  
}  
  
MIDDLEWARE = (  
    'elasticapm.contrib.django.middleware.TracingMiddleware',  
    # ...  
)
```



E no lambda?

<https://github.com/UnitedIncome/serverless-python-requirements>

```
$ pip install elastic-apm
```




E no lambda?

```
import functools
import elasticapm

name = 'serverless_apm'

def get_apm_client():
    elasticapm.instrument()
    return elasticapm.Client()
```

```
def apm_report(func):
    @functools.wraps(func)
    def execute(event, context):
        apm = get_apm_client()
        apm.begin_transaction('Request')
        elasticapm.set_custom_context({'event': event})

        try:
            result = func(event, context)
        except Exception:
            apm.capture_exception()
            raise
        finally:
            apm.end_transaction(func.__name__)
            apm.close()
            elasticapm.uninstrument()

        return result

    return execute
```



E no lambda?

```
from serverless_apm import apm_report

@apm_report
def index(event, context):
    body = {
        'message': 'Olá mundo!'
    }

    return {
        'body': json.dumps(body),
        'statusCode': status_code,
        'headers': default_headers,
    }
```



Fique Atento!

Auto Scale - Escalamos nossas instâncias de APM e Elasticsearch por quantidade de requests

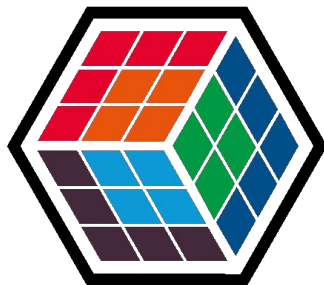
Rotação de logs - Fazemos a nossa a cada ~ 4 meses

Quantidade de dados gerados - 60 Gb de dados por dia



É isso...

Obrigado!



THE DEVELOPER'S CONFERENCE