EPFL

WordPress at Scale

go.epfl.ch/wpatscale

Dominique Quatravaux
EPFL ISAS-FSD

FOSDEM'2026

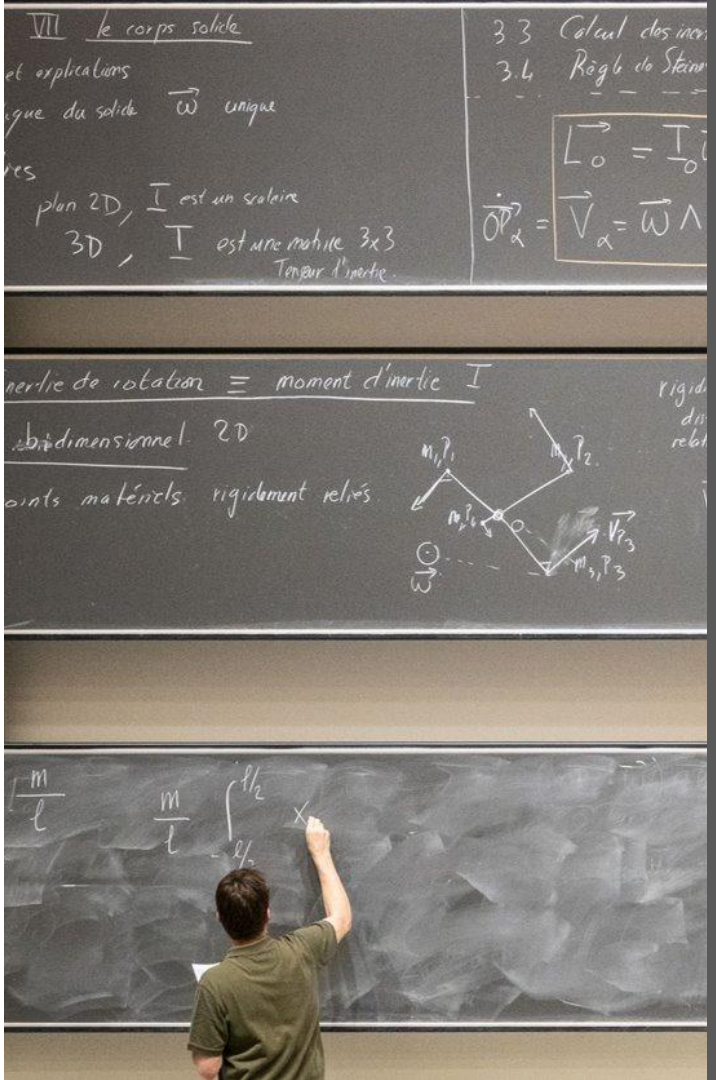
École polytechnique fédérale de Lausanne

EPFL is ... BIG

- 14'012 students
- 372 academic staff
- 6'530 employees
- 130 home countries

... **We have datacenters!**
(... with VMs and NFS)

Our team: 8 employees, 5 interns
... for 26 apps (not just WordPress)



> Serving at Scale

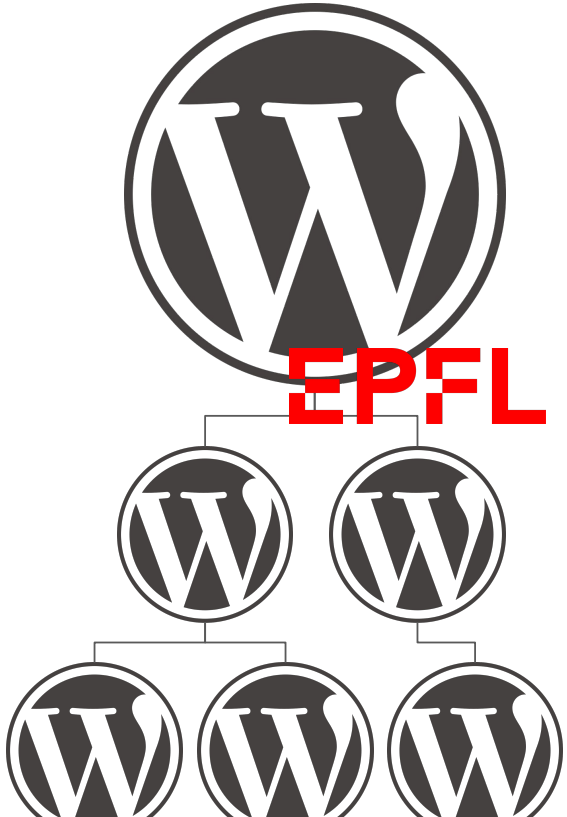
Operating at scale

Developing at scale *

- Kubernetes, containers, pods
- php-fpm, nginx
- Off-premise cache 😇

We'll need… 865 WordPresses.

- … 679 of which live under the same DNS domain, www.epfl.ch
  - Homepage
  - www.epfl.ch/labs/foolab (400×)
  - www.epfl.ch/campus/associations/bar (60×)
  - …
- inside.epfl.ch/department-baz (100×)
- quuxconf2026.epfl.ch
- …



Attempt #1 (~2019) = WordPress of the 2000s

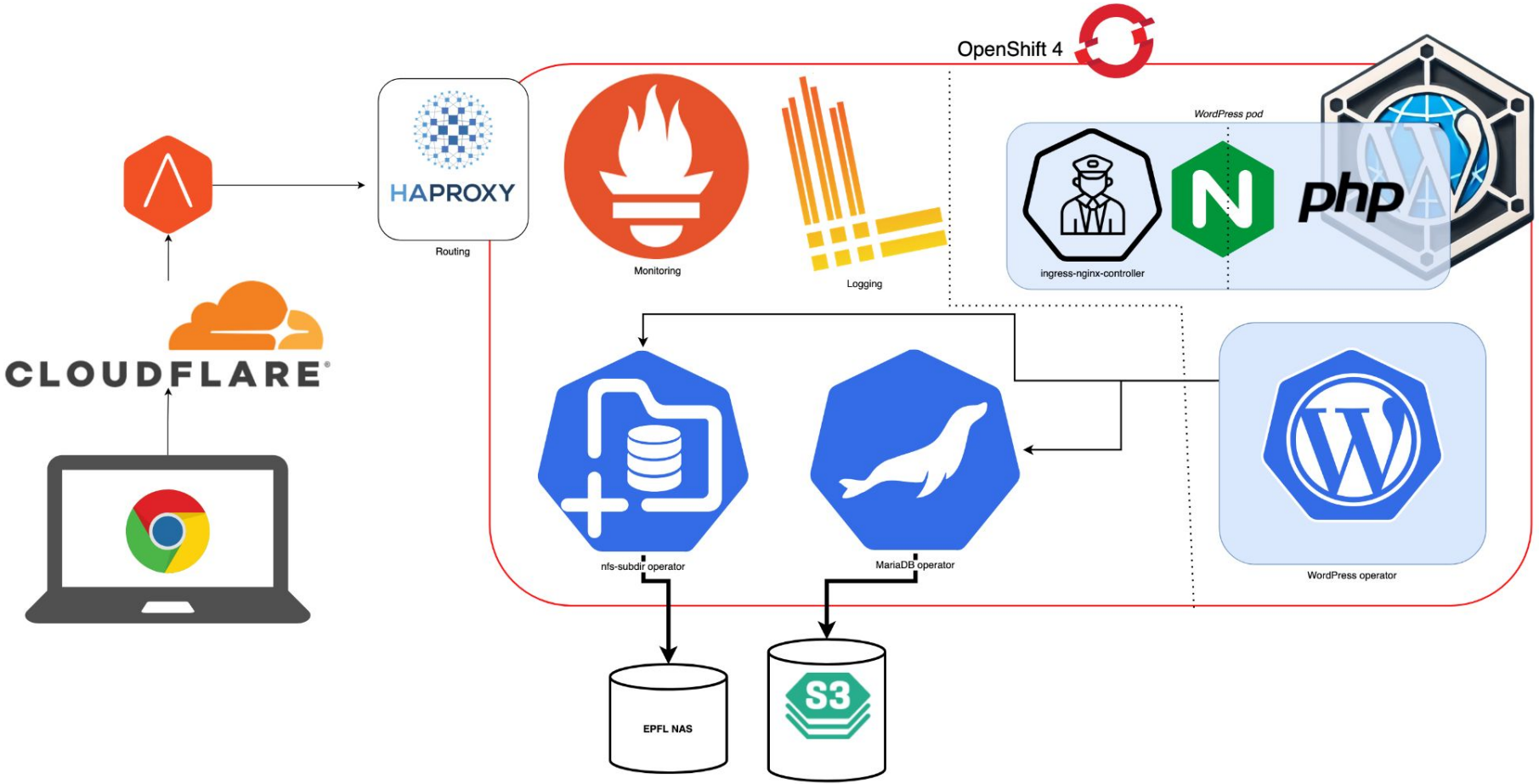
Designed and operated as a general-purpose LAMP hosting

- Tenant uploads
  - WordPress (or whatever)
  - Theme(s)
  - Plug-in(s)
- We kind of didn't know any better back then.

… Not really at scale.

- Apache?
  - 1 request = 1 thread
  - Always costs full price (in RAM, threads)
- PHP over NFS?
  - WordPress includes ~1000 PHP files as part of its hello-world code flow
  - Latency is **at least** 1000× the RTT to your filer
    - Cache?
    - … Doesn't help — revalidates with `stat(2)`

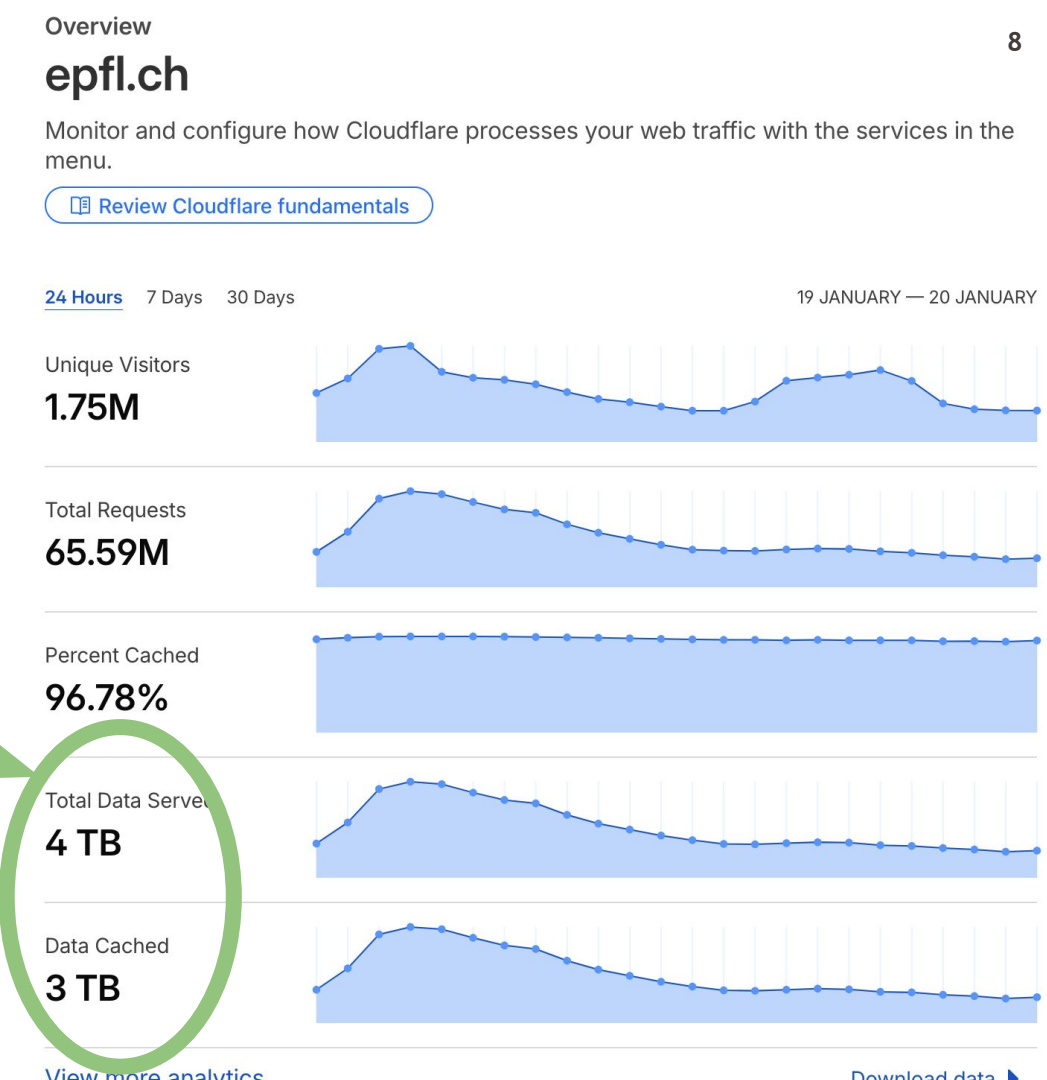
In 2025: do it again, do it right

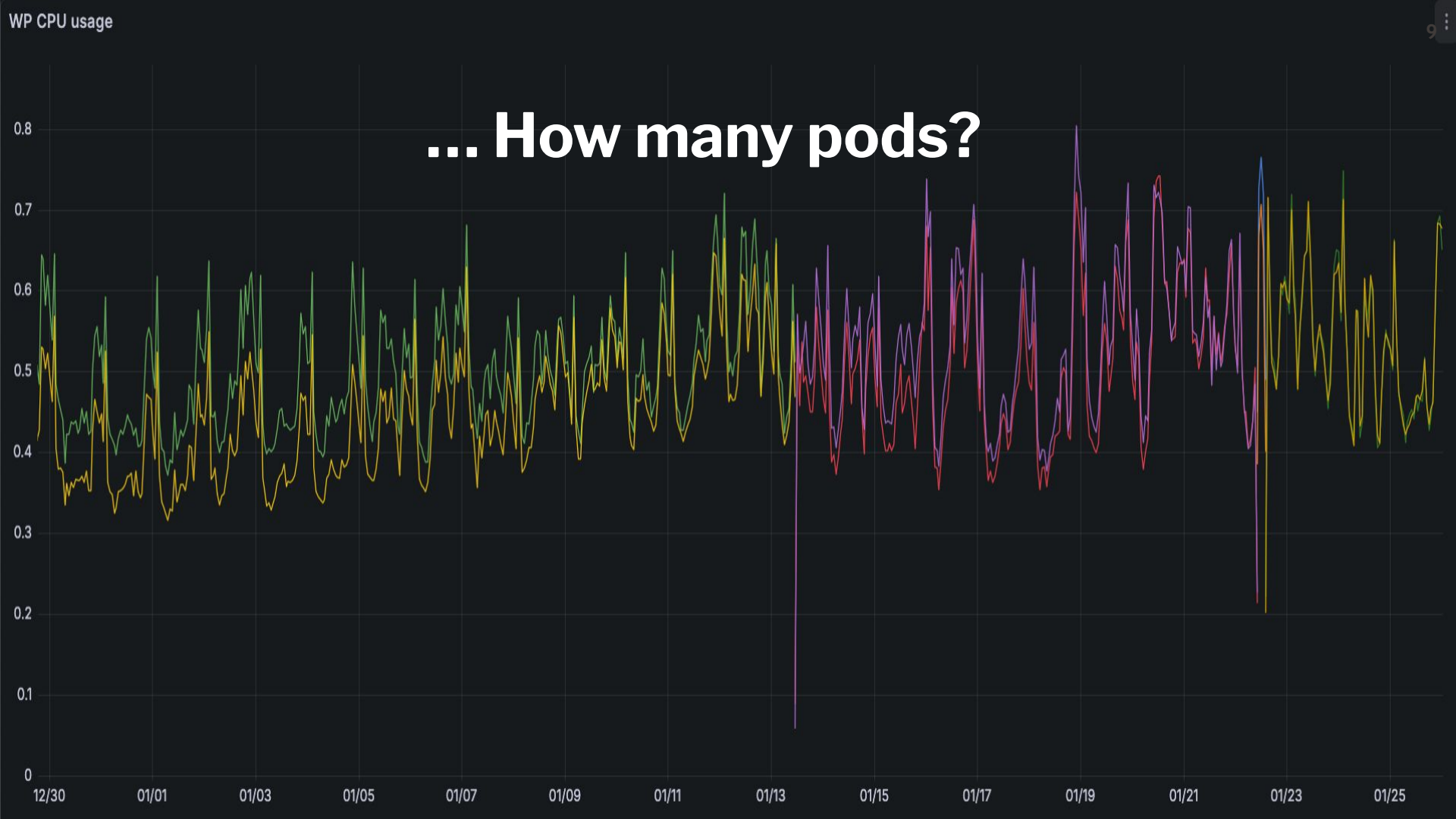


Edge cache

- It protecs!
- It ~~attacs~~ serves captchas!
- It pays for itself in sheer bandwidth savings!

(Disclaimer: not affiliated with Cloudflare in any way, besides being satisfied customers of their service)



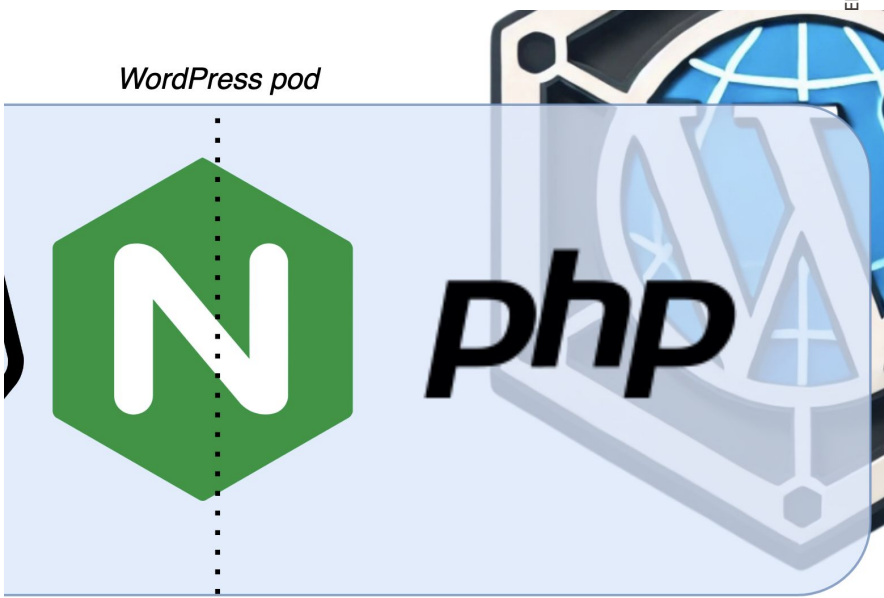

WP CPU usage
... How many pods?
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
12/30
01/01
01/03
01/05
01/07
01/09
01/11
01/13
01/15
01/17
01/19
01/21
01/23
01/25

WordPress in ~~a container~~ two containers

- ~~Apache~~ → nginx: non-blocking Web server
- php-fpm: same pod, separate container
- ships w/ **immutable PHP code**

  👍

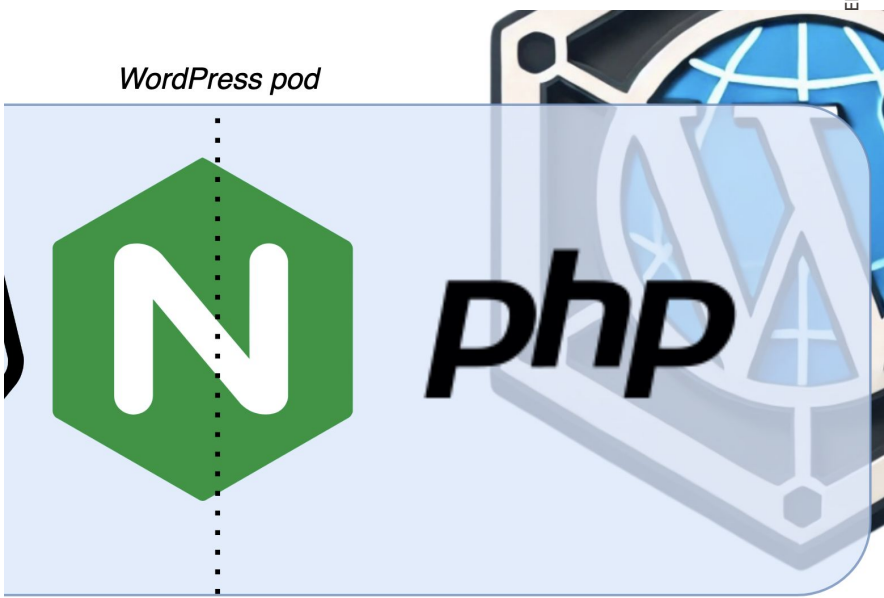
  - Security: hard to backdoor
  - Performance: kernel FS cache
    ≈ µs latency



We used to require another layer of cache on-premise (varnish)**. We no longer do.**

nginx is boss 👑

- **knows** 🧠
  about all sites, all URLs, all filename patterns,
  and all WordPress 12-factored variables
  (i.e. MariaDB credentials)
- **serves** 👩‍🍳
  static files out of
  - The container image
  - or NFS again (just for uploads now 😉)
- **passes** 🏈
  the query to php-fpm for interactively rendered content



Each WordPress request is served independently from the others.

PHP?...

A design oversight turned killer feature:
forget everything, every time.

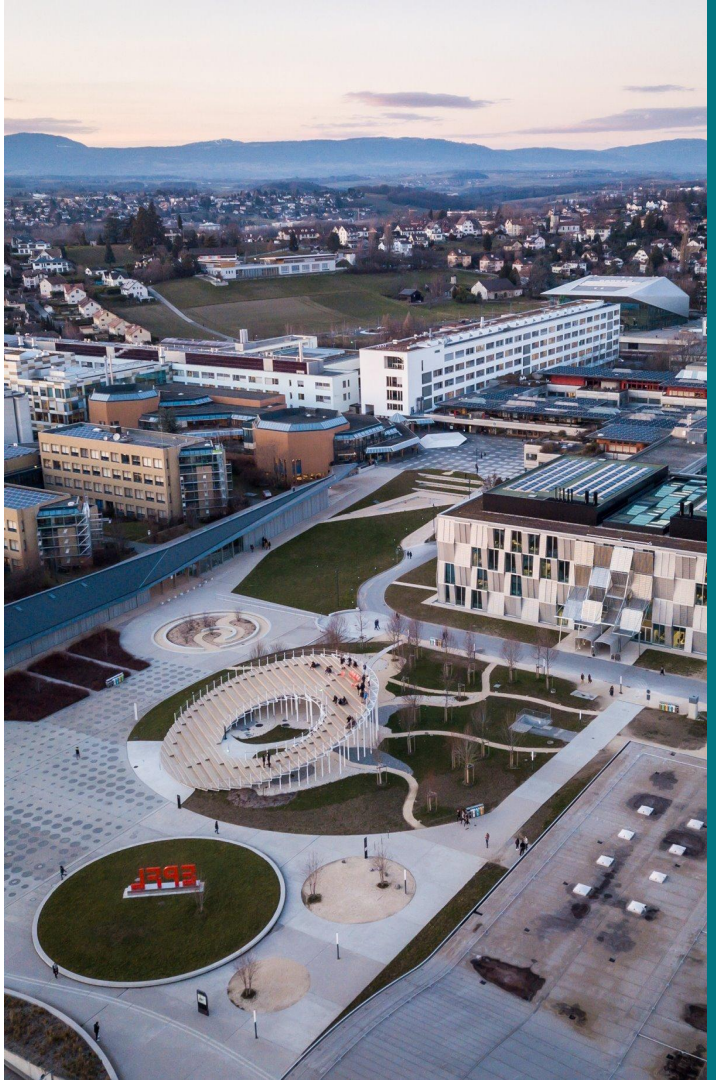
- PHP / WordPress starts over from a blank slate **upon each and every request.**
- Efficient? How?... 🤔
  - “shared memory”
  - “Zend cache”

PHP?...

We are effectively

turning the whole WordPress into a function-as-a-service.

EPFL



Serving at scale
> Operating at Scale
Developing at scale *

- Using **Kubernetes** to the best of its capabilities
- Absorbing **complexity** with the WordPress operator

The operator is a program

- … that operates in your stead.
- In goes: 1× `WordpressSite` Kubernetes object
- Out comes your site… kind of piecemeal:
  - 1× `Ingress` # of a “private” `IngressClass`; more on this later
  - 1× `Route` # or sometimes 0×, e.g. www
  - 1× `Database` # 🏈 to MariaDB operator
  - 1× `mkdir()` on NFS # uploads
  - Homemade PHP code that
    - Initializes the site
    - Initializes or reconciles plug-ins and themes

All of that is made

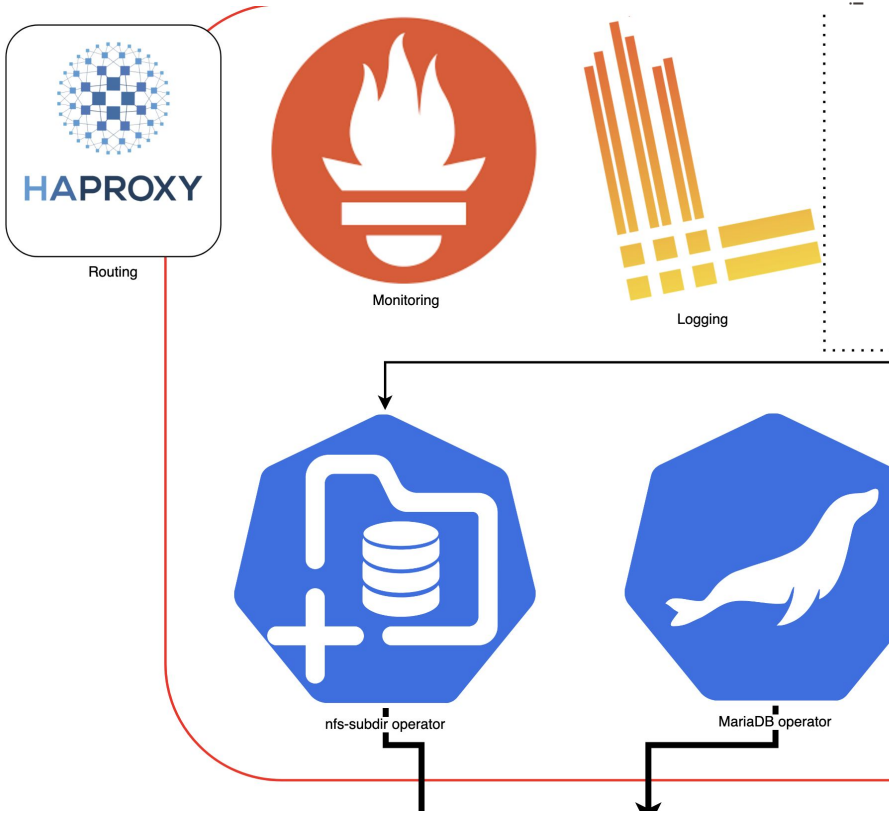
- With Python and Kopf, the Kubernetes Operator Framework
- In a fully EPFL-agnostic way!
  → **#coreplugindesign**


DEMO

What needs doing on a daily basis?

Well… Not a whole lot.

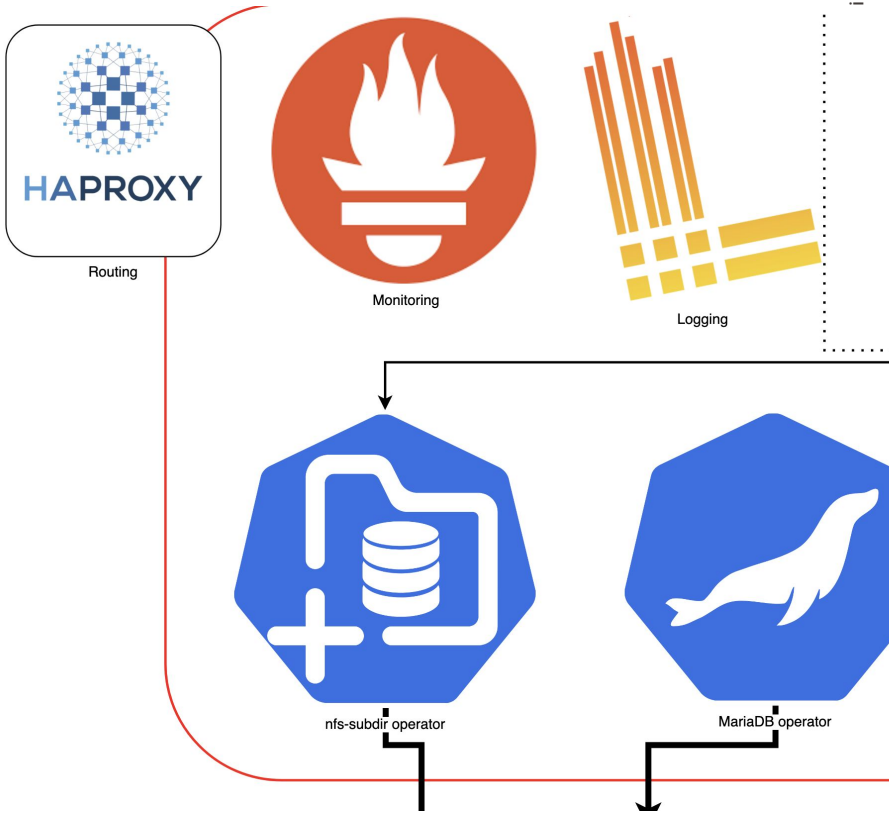
- We wrote wp-veritas (in Next.JS) for our Webgrandmaster to edit `WordpressSite` objects on their own
  - That app captures a lot of EPFL-specific “business logic”…
  - The rest being in our lineup of themes and plugins



What needs doing on a daily basis?

Well… Not a whole lot.

- Backups to our on-premise S3 are being performed by the MariaDB operator
  - Our own WP operator can optionally restore a site from there, under the same URL or a new one



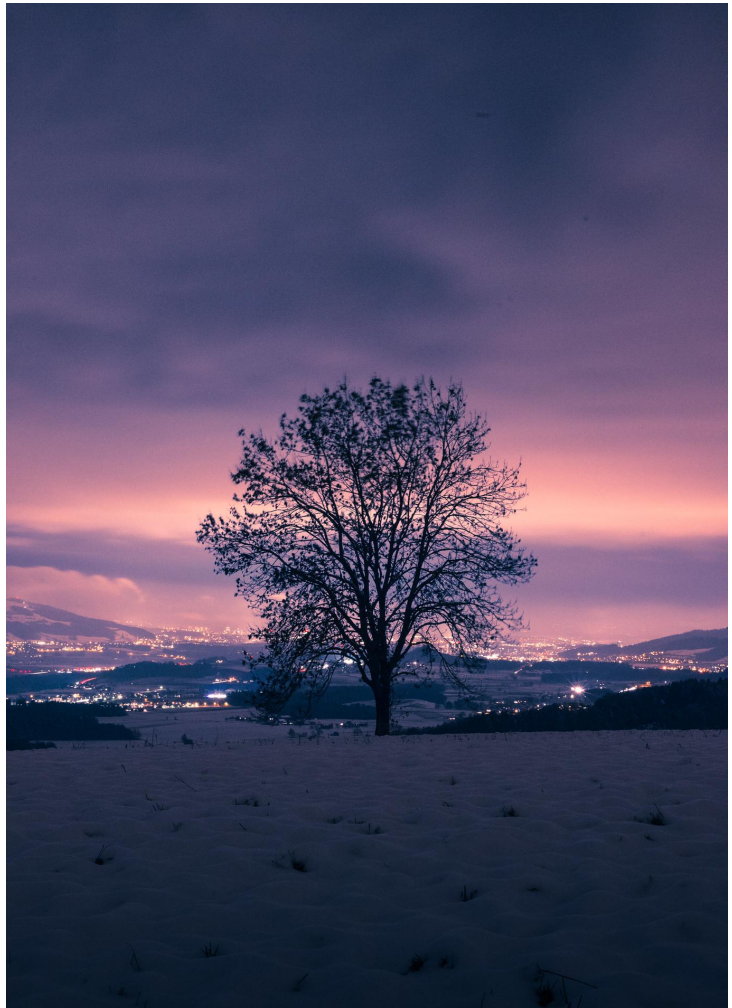
… nightly basis?

We moved wp-cron into its own Kubernetes `CronJob`.

(Solved the 22h22 problem.)

We use that to monitor “slow” metrics into a Prometheus pushgateway:

→ plugin

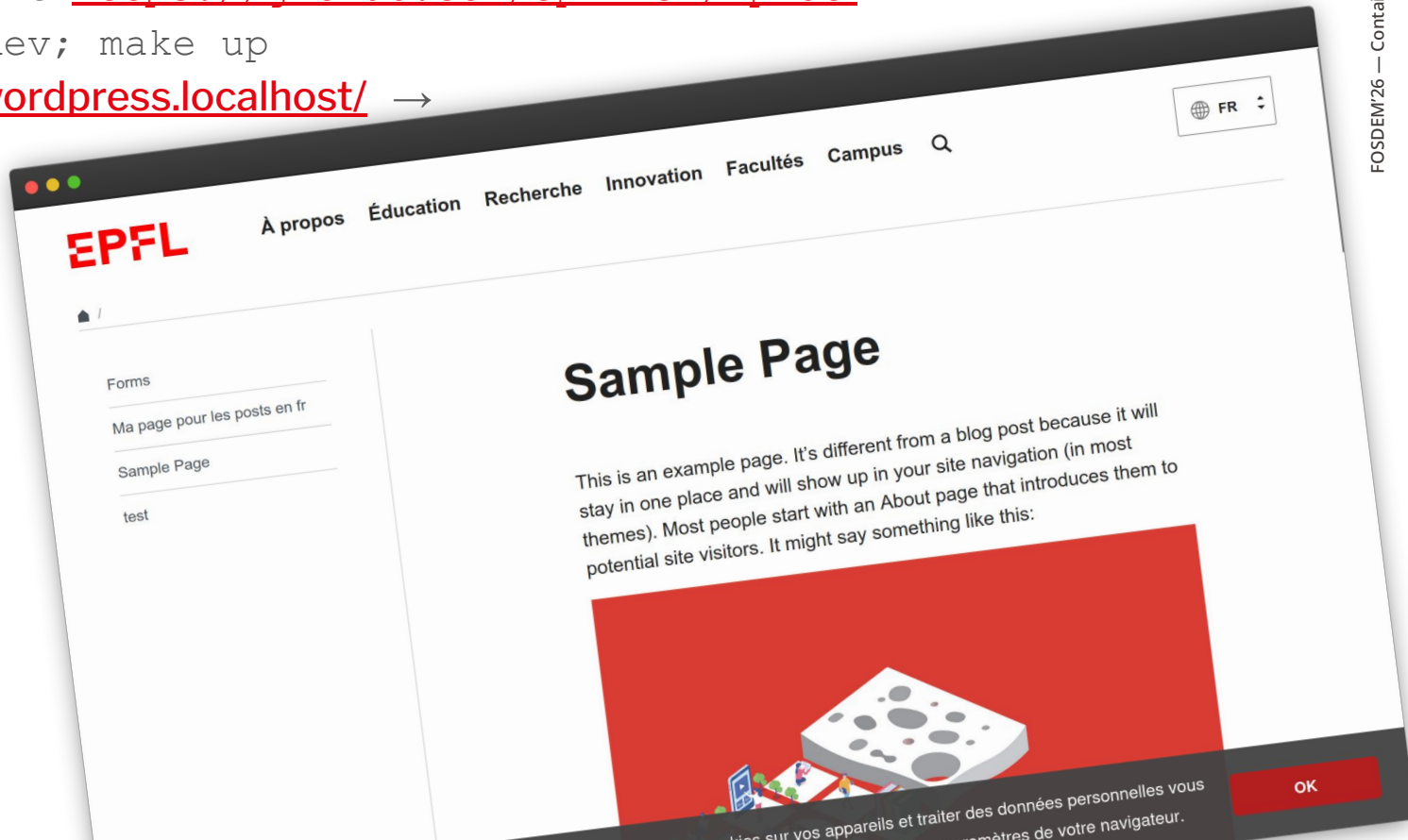


Developing at (some) Scale

There’s 13 of us… But we’d welcome more! 😉

wp-dev: jump right in!

1. `git clone https://github.com/epfl-si/wp-dev`
2. `cd wp-dev; make up`
3. https://wordpress.localhost/ →



Conclusion:
Ready for immediate consumption?

WP-Operator	★★★★★	Flagship contribution
Dockerfiles	★★★★☆	Coupled with Ansible 🤷
EPFL theme	★☆☆☆☆	
Plug-ins: from	★★☆☆☆	e.g. lunch menus
… to	★★★★★	e.g. pushgateway
WP-dev	★★★★☆	
WP-Veritas	★☆☆☆☆	


EPFL
Thanks for attending!
... Please get in touch!
isas-fsd@groupes.epfl.ch
#2026-containers:fosdem.org

Future plans

- Instrumented PHP → OTel
- More documentation

The nginx ingress controller is another program

… that controls nginx, based on `Ingress` objects in Kubernetes.

- In comes: a whole bunch of `Ingress`es, that the WordPress operator created (as seen previously);
- Out comes: the nginx configuration file.

We took this one “as-is” from github.com/kubernetes/ingress-nginx (and substituted our own golang template)

Photo Credits

- "Baby facepalm" by bigpresh is licensed under CC BY 2.0.
- "Blue tang fish close up" and "Realistic blue eye clip art" are marked with CC0 1.0 both.
- "Mystery Box" by mikecogh is licensed under CC BY-SA 2.0.
- "AARP DEMO" by jurvetson is licensed under CC BY 2.0.
- "Nightly Glow / Nikon D600" by Gino Andenmatten | Photography is licensed under CC BY-NC-ND 2.0.
- EPFL branding, logo, slideware, photos and clip art used with permission.
- All product logos belong to their respective owners.
- I'm… honestly not entirely sure about the copyright status of that picture of my younger self; scanned from my own, expired carte d'identité? 🤷‍♂️ I'm going with the #deminimisnoncuratprætor defense