A Rusty Web

Ben Wishovich | @benwis@hachyderm.io

Intro

I've been a web developer for four years now, building high performance ecommerce and bespoke websites.

My first PR to Leptos was 41.5 months ago

Note: Leptos has existed for 42 months

Shoutout to Greg

Creator of Leptos

This man has slowly and paiently helped me learn how to use Leptos, how Leptos works, and continues to be doing a ton of work on it.

Goals

1. **Introduce Leptos**
2. **Tempt You to Try It**

Leptos

is a full-stack web framework that lets you leverage the power of Rust and fine-grained reactivity to deliver interactive, stable, and powerful web applications

Patr

A one stop cloud platform for deploying static sites, web apps, databases, and containers.

Rust Adventure

Chris Biscardi's Rust Video Training Platform

RustyTube

Youtube frontend web and desktop app using Leptos and Tauri

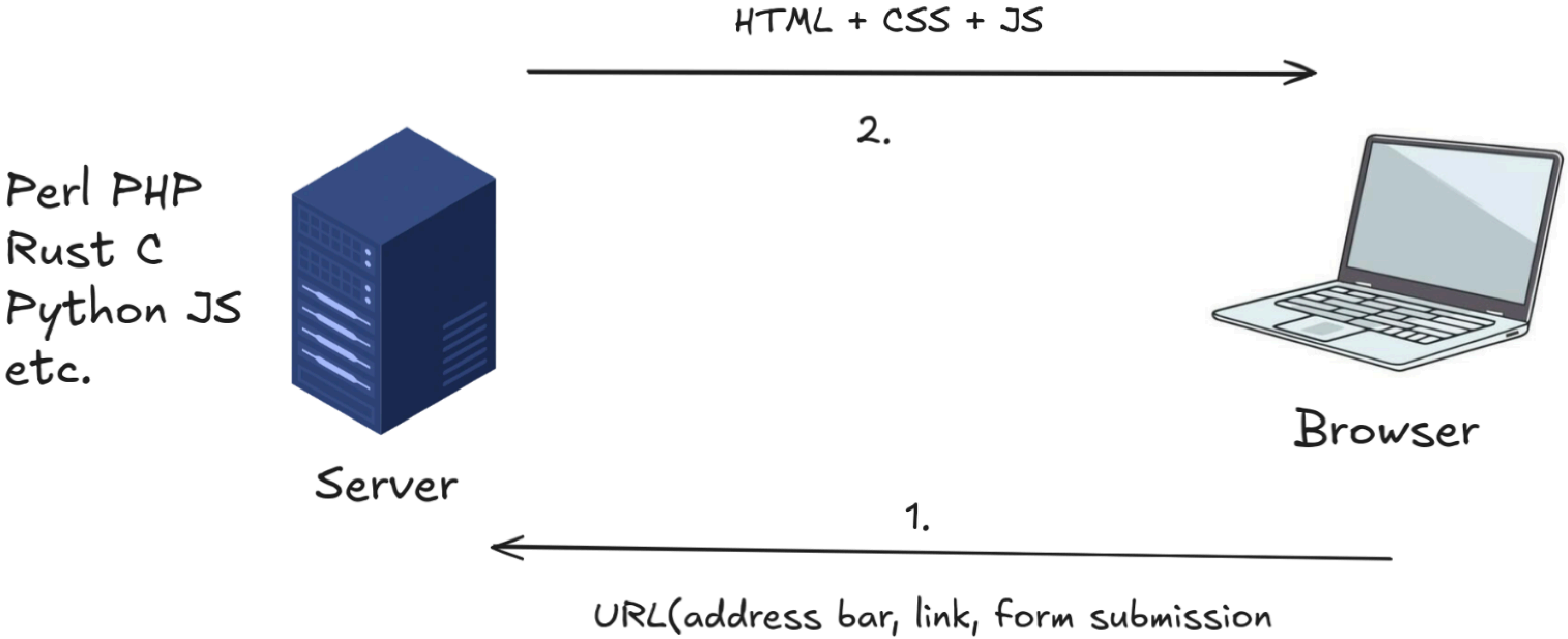
California Beach Volleyball Association

Find and signup for beach volleyball in California

Dream of the Web

- The web doesn't care what language you use to write it, it can use any language ever conceived. Whatever is preferred, from COBOL to JS to Rust and everything in between.
- But how did we get here?

Multi Page Apps



HTML + CSS + JS

2.

Perl PHP
Rust C
Python JS
etc.

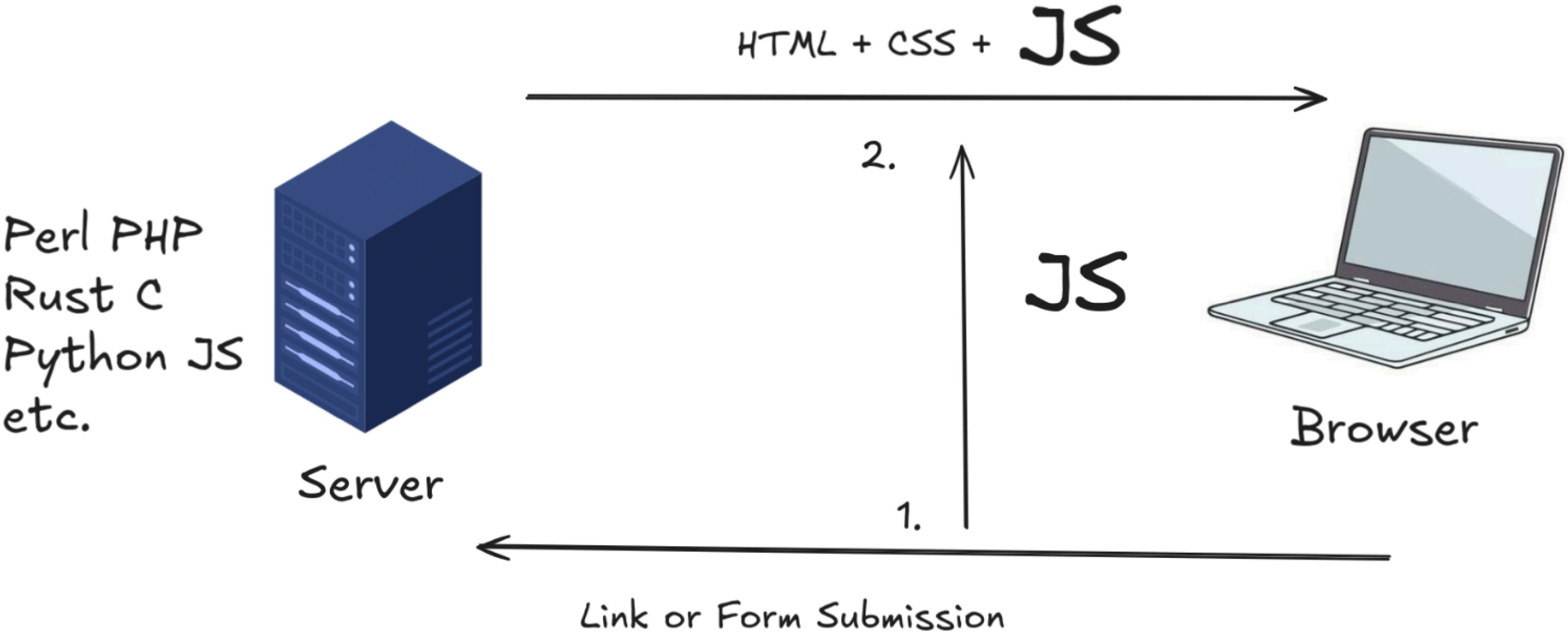
Server

Browser

1.

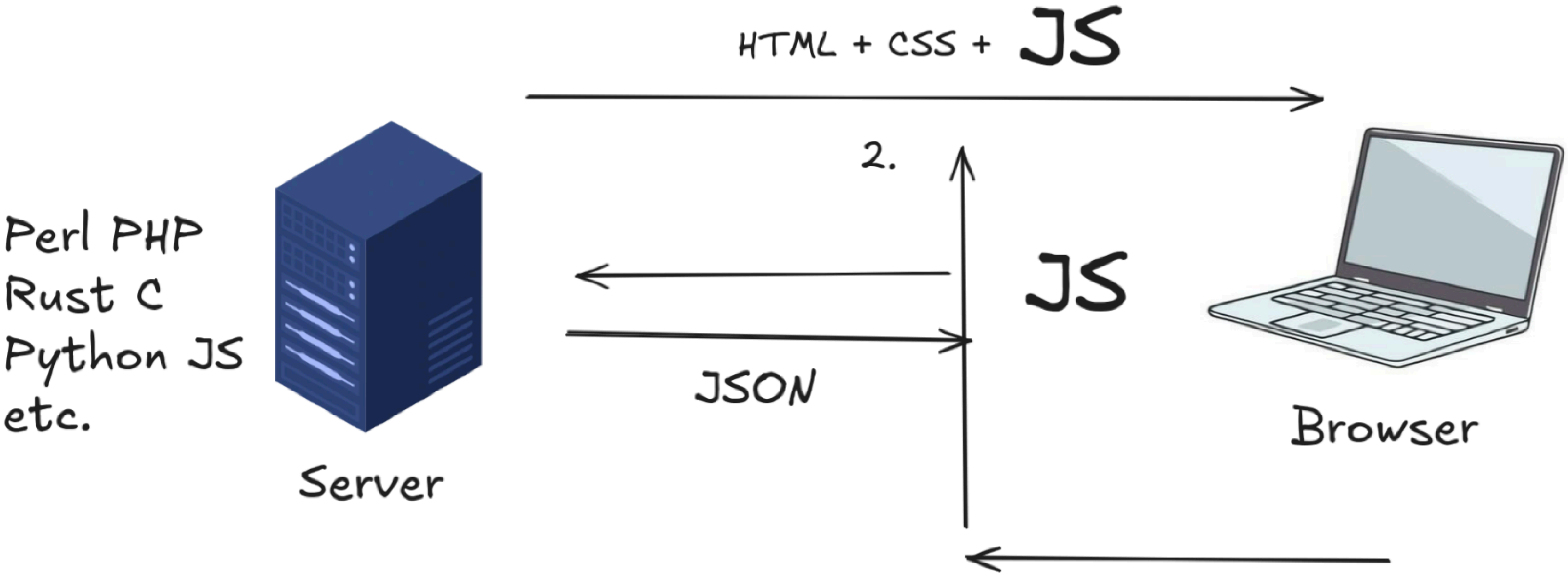
URL(address bar, link, form submission

MPA + JQuery

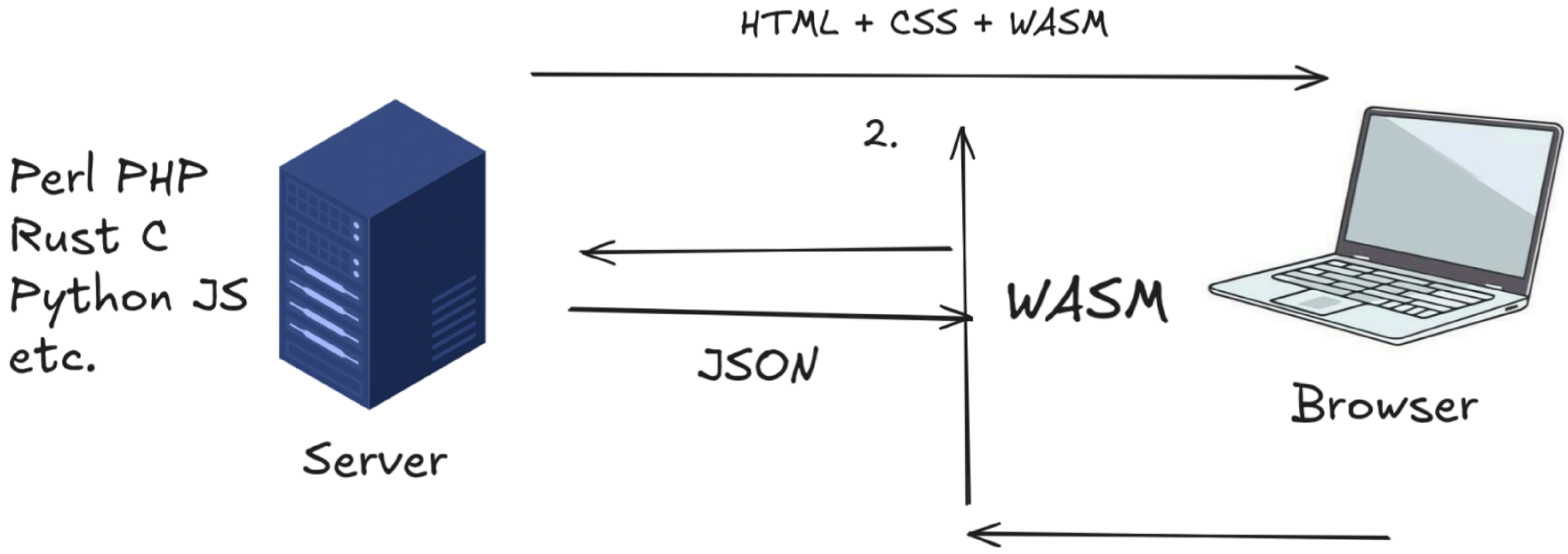


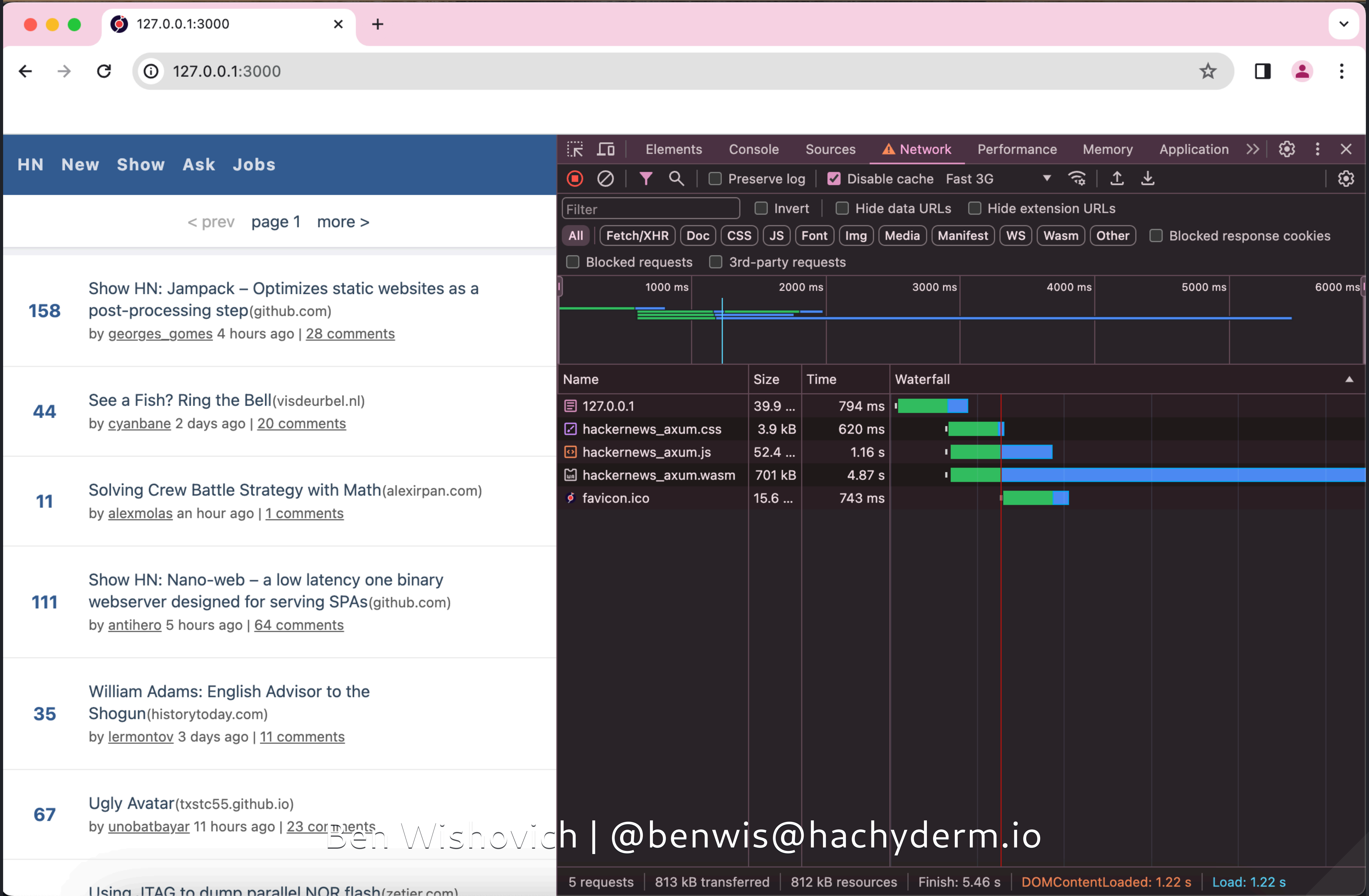
HTML + CSS + JS
2.
JS
Perl PHP
Rust C
Python JS
etc.
Server
Browser
1.
Link or Form Submission

Server Side Rendering With Hydration



Webassembly/Leptos




127.0.0.1:3000
127.0.0.1:3000
HN New Show Ask Jobs
Elements Console Sources Network Performance Memory Application
Filter
All Fetch/XHR Doc CSS JS Font Img Media Manifest WS Wasm Other
1000 ms 2000 ms 3000 ms 4000 ms 5000 ms 6000 ms

< prev page 1 more >

158 Show HN: Jampack – Optimizes static websites as a post-processing step(github.com)
by georges_gomes 4 hours ago | 28 comments

44 See a Fish? Ring the Bell(visdeurbel.nl)
by cyanbane 2 days ago | 20 comments

11 Solving Crew Battle Strategy with Math(alexirpan.com)
by alexmolas an hour ago | 1 comments

111 Show HN: Nano-web – a low latency one binary webserver designed for serving SPAs(github.com)
by antihero 5 hours ago | 64 comments

35 William Adams: English Advisor to the Shogun(historytoday.com)
by lermontov 3 days ago | 11 comments

67 Ugly Avatar(txstc55.github.io)
by unobatbayar 11 hours ago | 23 comments

Using JTAG to dump parallel NOR flash(zetier.com)

□ Preserve log ☑ Disable cache Fast 3G

□ Invert □ Hide data URLs □ Hide extension URLs

□ Blocked response cookies □ Blocked requests □ 3rd-party requests

Name	Size	Time
127.0.0.1	39.9 ...	794 ms
hackernews_axum.css	3.9 kB	620 ms
hackernews_axum.js	52.4 ...	1.16 s
hackernews_axum.wasm	701 kB	4.87 s
favicon.ico	15.6 ...	743 ms

5 requests | 813 kB transferred | 812 kB resources | Finish: 5.46 s | DOMContentLoaded: 1.22 s | Load: 1.22 s

Ben Wishovich | @benwis@hachyderm.io

Components

```
#[component]
pub fn BasicComponent() -> impl IntoView{
  // Run regular rust code here


  view!{
    // JSX but Rust, I dub thee RXML
    <h1 class="header">My Leptos Component</h1>
    <p>It's the best</p>
  }
}
```

Server Functions

- Call a function on the server as if it was on the client
- Creates a publicly accessible API endpoint
- server macro automatically generates code to serialize and deserialize the values. Fully type checked
- Users select desired encoding and http method

```
/// Get a post from storage
#[server]
pub async fn get_count() -> Result<i64, ServerFnError> {
    let count = fetch(count);
    Ok(count)
}
```

Resources

```
#[component]
pub fn CountPage() -> impl IntoView{
  let count = create_resource(|| (), || get_count())
  // Keep track of an async Rust function that we want to run at page load time
  view!{
    <Suspense fallback= view!{<p>Loading...</p>}>
    {
      move || {
        count.get().map(|p| match p {
          Ok(p) => view!{<content>"Count: "{count}</content>}.into_view(),
          Err(e) => view!{<span>Error: {e.to_string()}</span>}.into_view(),
        })
      }
    }
    </Suspense>
  }
}
```

Server Function #2

```
#[server]
pub async fn update_count(count: i64, increment_by: i64) -> Result<i64, ServerFnError> {
    let new_count = count + increment_by;
    store(new_count);
    Ok(new_count)
}
```

Actions and ActionForm

```
#[component]
pub fn CountPage() -> impl IntoView{
 let count = create_resource(|| (), || get_count())
 let update_count = create_server_action::<UpdateCount>();
 //Keep track of an async Rust function that we want to run on some condition, like a button press or a keystroke -->
 view!{
   <Suspense fallback= view!{<p>Loading...</p>}>
   {
     move || {
       count.get().map(|p| match p {
         Ok(p) => view!{<content>"Count: "{count}</content>}.into_view(),
         Err(e) => view!{<span>Error: {e.to_string()}</span>}.into_view(),
       })
     }
   }
   </Suspense>
     <ActionForm action=update_count>
     <input type="number" name="increase_by">
```

Routing

```
<Router >
   <main>
       <Routes>
           <Route path="" view=CountPage/>
           <Route path="/about/us" view=CountPage/>
           <Route path="/all/*this" view=All/>
           <Route path="inbox/:user_id" view=Inbox>
       </Routes>
   </main>
 </Router>
```

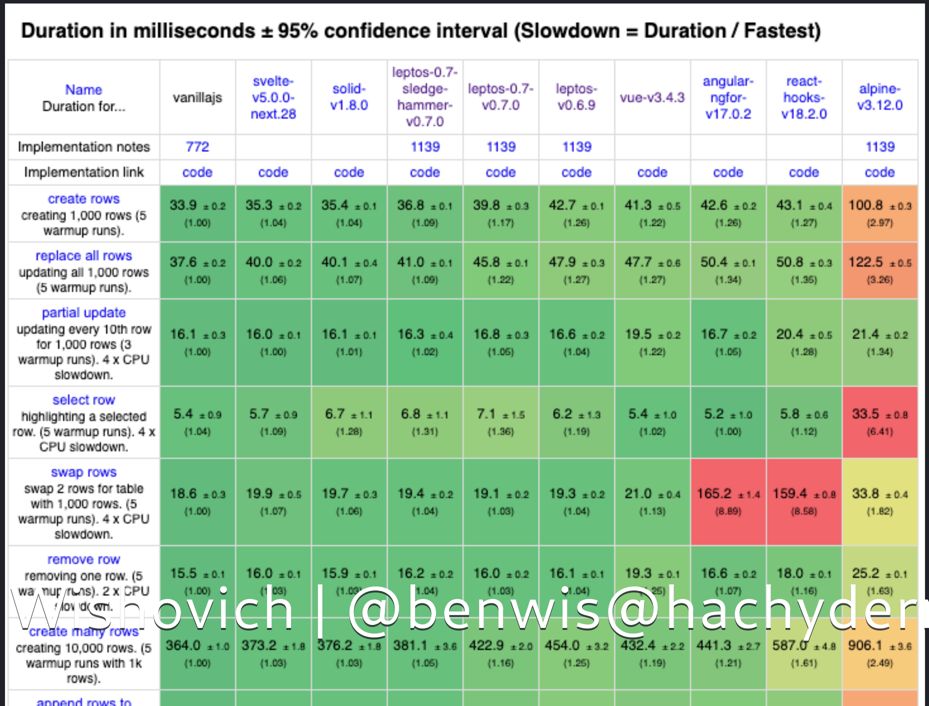
Tempting You

Myths of Leptos and Webassembly

1. The bundle size is too big
2. The startup time is too slow
3. It's limited by a lack of direct DOM access for Webassembly
4. Compiling takes too long
5. The community is too small

Performance

JS Framework Bench



Duration in milliseconds ± 95% confidence interval (Slowdown = Duration / Fastest)

Name Duration for...	vanillajs	svelte-v5.0.0-next.28	solid-v1.8.0	leptos-0.7-sledge-hammer-v0.7.0	leptos-0.7-v0.7.0	leptos-v0.6.9	vue-v3.4.3	angular-ngfor-v17.0.2	react-hooks-v18.2.0	alpine-v3.12.0
Implementation notes	772			1139	1139	1139				1139
Implementation link	code	code	code	code	code	code	code	code	code	code
create rows creating 1,000 rows (5 warmup runs).	33.9 ± 0.2 (1.00)	35.3 ± 0.2 (1.04)	35.4 ± 0.1 (1.04)	36.8 ± 0.1 (1.09)	39.8 ± 0.3 (1.17)	42.7 ± 0.1 (1.26)	41.3 ± 0.5 (1.22)	42.6 ± 0.2 (1.26)	43.1 ± 0.4 (1.27)	100.8 ± 0.3 (2.97)
replace all rows updating all 1,000 rows (5 warmup runs).	37.6 ± 0.2 (1.00)	40.0 ± 0.2 (1.06)	40.1 ± 0.4 (1.07)	41.0 ± 0.1 (1.09)	45.8 ± 0.1 (1.22)	47.9 ± 0.3 (1.27)	47.7 ± 0.6 (1.27)	50.4 ± 0.1 (1.34)	50.8 ± 0.3 (1.35)	122.5 ± 0.5 (3.26)
partial update updating every 10th row for 1,000 rows (3 warmup runs). 4 x CPU slowdown.	16.1 ± 0.3 (1.00)	16.0 ± 0.1 (1.00)	16.1 ± 0.1 (1.01)	16.3 ± 0.4 (1.02)	16.8 ± 0.3 (1.05)	16.6 ± 0.2 (1.04)	19.5 ± 0.2 (1.22)	16.7 ± 0.2 (1.05)	20.4 ± 0.5 (1.28)	21.4 ± 0.2 (1.34)
select row highlighting a selected row. (5 warmup runs). 4 x CPU slowdown.	5.4 ± 0.9 (1.04)	5.7 ± 0.9 (1.09)	6.7 ± 1.1 (1.28)	6.8 ± 1.1 (1.31)	7.1 ± 1.5 (1.36)	6.2 ± 1.3 (1.19)	5.4 ± 1.0 (1.02)	5.2 ± 1.0 (1.00)	5.8 ± 0.6 (1.12)	33.5 ± 0.8 (6.41)
swap rows swap 2 rows for table with 1,000 rows. (5 warmup runs). 4 x CPU slowdown.	18.6 ± 0.3 (1.00)	19.9 ± 0.5 (1.07)	19.7 ± 0.3 (1.06)	19.4 ± 0.2 (1.04)	19.1 ± 0.2 (1.03)	19.3 ± 0.2 (1.04)	21.0 ± 0.4 (1.13)	165.2 ± 1.4 (8.89)	159.4 ± 0.8 (8.58)	33.8 ± 0.4 (1.82)
remove row removing one row. (5 warmup runs). 2 x CPU slowdown.	15.5 ± 0.1 (1.00)	16.0 ± 0.1 (1.03)	15.9 ± 0.1 (1.03)	16.2 ± 0.2 (1.04)	16.0 ± 0.2 (1.03)	16.1 ± 0.1 (1.04)	19.3 ± 0.1 (1.25)	16.6 ± 0.2 (1.07)	18.0 ± 0.1 (1.16)	25.2 ± 0.1 (1.63)
create many rows creating 10,000 rows. (5 warmup runs with 1k rows).	364.0 ± 1.0 (1.00)	373.2 ± 1.8 (1.03)	376.2 ± 1.8 (1.03)	381.1 ± 3.6 (1.05)	422.9 ± 2.0 (1.16)	454.0 ± 3.2 (1.25)	432.4 ± 2.2 (1.19)	441.3 ± 2.7 (1.21)	587.0 ± 4.8 (1.61)	906.1 ± 3.6 (2.49)
append rows to										

Compile Times

Patr: 5s

CBVA: 5s

Rust Adventure: 2s

- Incremental build times

Hot Reload

```
cargo leptos watch --hot-reload
```

Send an HTML/CSS patch to the browser to instantly update the view(if only it changed) while waiting for a compile

Ben's Blog Performance Test

- Wrote my blog in two different web frameworks as similar as possible, Remix and Leptos
- Measure how long it takes to serve the home page using each framework, under differing levels of load

Details

- Home page fetches 3 most recent posts from a sqlite database, displays post metadata
- Html/CSS/Logic as functionally similar as possible

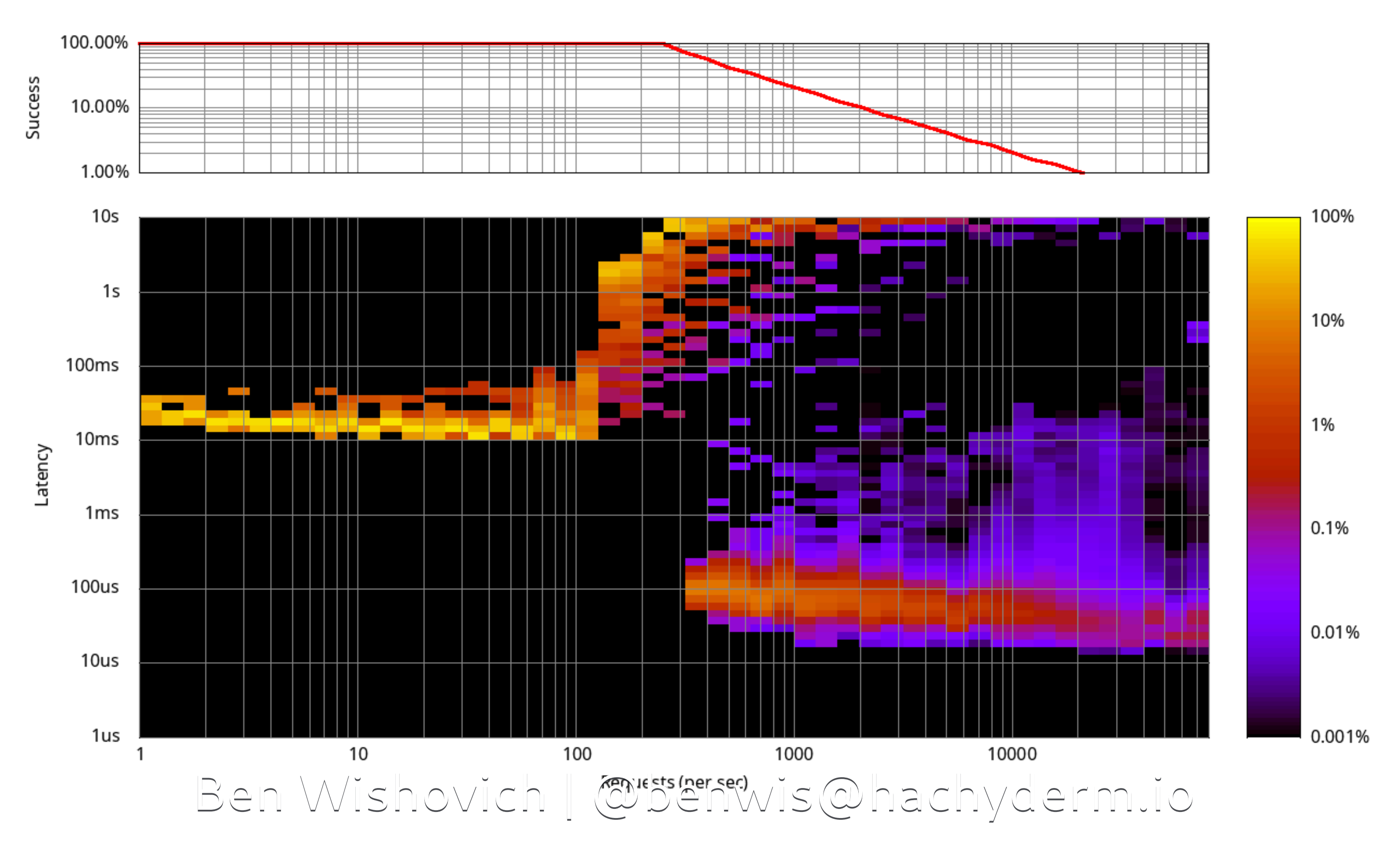
Test Hardware

Web apps run on a Digital Ocean VM with:

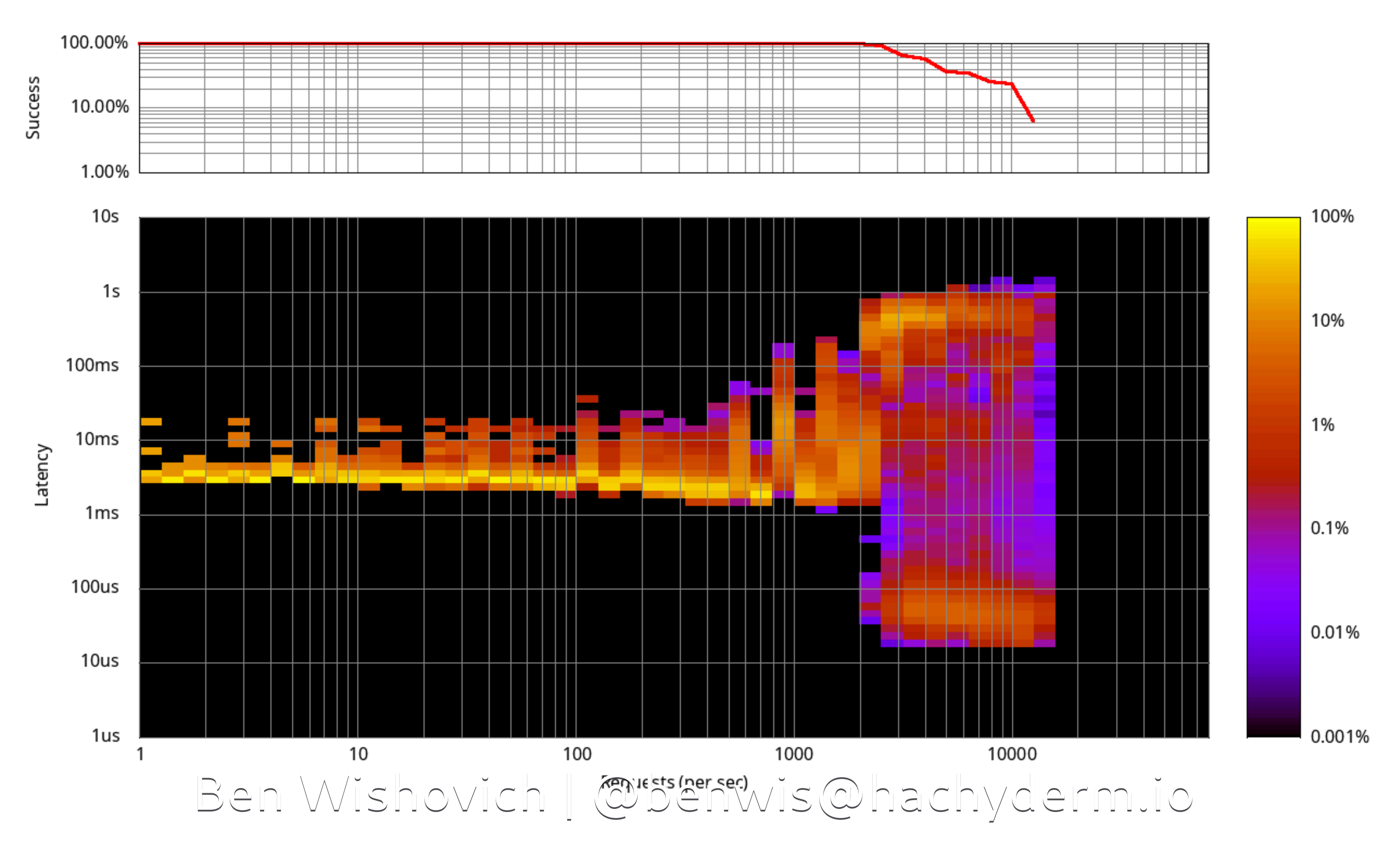
- Two dedicated "vcpus"
- 8GB RAM
- 2Gbps bandwidth

Load tester run on equivalent VM, using vegeta

Remix + Express.js



Leptos v0.6 + Axum



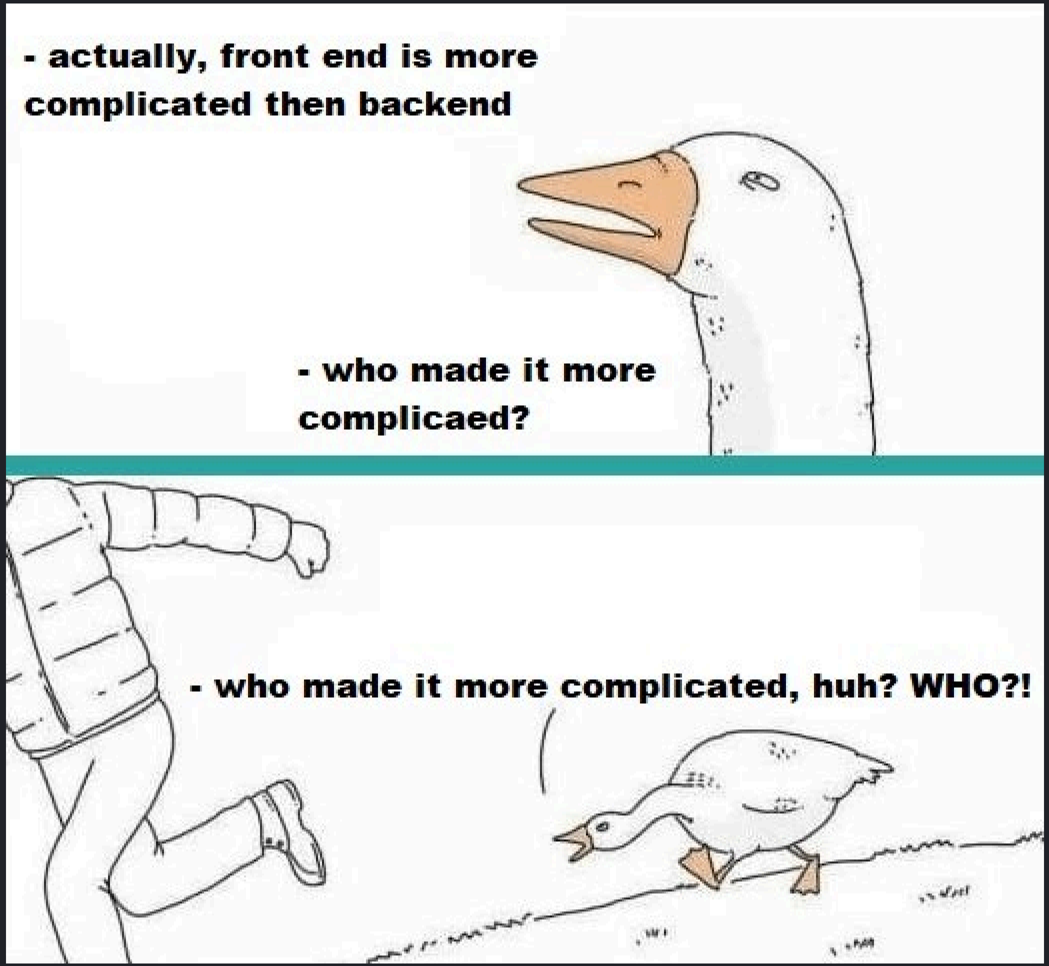
Rust's Type System and Tooling

Types across the boundary between Client and Server

Types at compile time and run time

Cargo, cargo-leptos, and rustfmt vs Eslint/Prettier/Npm/tsc/vite/etc.

Ben Wishovich | @benwis@hachyderm.io


- actually, front end is more complicated then backend
- who made it more complicaed?
- who made it more complicated, huh? WHO?!

Developer time

- The more work the tooling does, the less the programmer needs to keep things in their head
- Simplifies building and maintaing your codebase

> The rust part of it + reactivity brings amazing benefits to making sure that we spend (a little bit) more time building our application and almost no time debugging the version that's already running. So most of our time is spent on building new features and focusing on the product / user experience rather than fixing bugs and pushing patches. - BlackjackHack, Patr

> Having a language built with a type system from the beginning combined with a framework that is competitive with modern JS frameworks and all of the use cases that implies means that I can build comparable sites to what I've done my entire career with far less cognitive overhead. - Chris Biscardi, Rust Adventure

> Leptos is essentially taking all the benefits of Rust and marrying them to all the benefits of Signals & SSR... I have done truly nothing to optimize yet and I already have top notch time to paint and time to reactive. Even on poor LTE beaches. - Alex, CBVA

Ecosystem and Community

Shoutout to the Leptos Discord

Awesome Leptos

Tempted Yet?

1. Server functions
2. Performance
3. Infra costs
4. Rust's type system, error messages, and tooling
5. Reduced developer time vs app complexity

Join Us

Leptos Website: https://leptos.dev

Discord

Github

The End

Questions?