



△ Realtime Collaboration on Web



C.C. @ YoMo

<https://github.com/yomorun/yomo>

Nov 25 2021



YoMo

<https://github.com/yomorun/yomo>



map - Figma

https://www.figma.com/file/IFUgjQbUbUhBSDtqyE4MEC/map?node-id=972%3A57319

AllegroPulse Orbit GDrive Cloudcraft DeepL Panelbear 0xCC Excalidraw GA AWS-SDH Lark QUIC AWS-Xile Xile Org Chart Vercel

Geoping / map

Layers Assets 网站v0.9定制...

Search

Used in this file

41%

LOCATION

63%

79%

81%

83%

85%

87%

89%

91%

93%

95%

97%

99%

100%

Design Prototype Inspect

Background

FSF5F5 100%

Export

Asia Total response time 635ms

0 μs 100 ms 200 ms 300 ms 400 ms 500 ms 600 ms


DNS	260 ms	37%
Connection	90 ms	14.1%
SSL	235 ms	41%
Processing	50 ms	7.9%







Navigation bar with icons for headphones, two user avatars, a blue 'Share' button, a play button, and '55%' with a dropdown arrow.

Design Prototype Inspect




1  **Share** ▶ 55% ▾

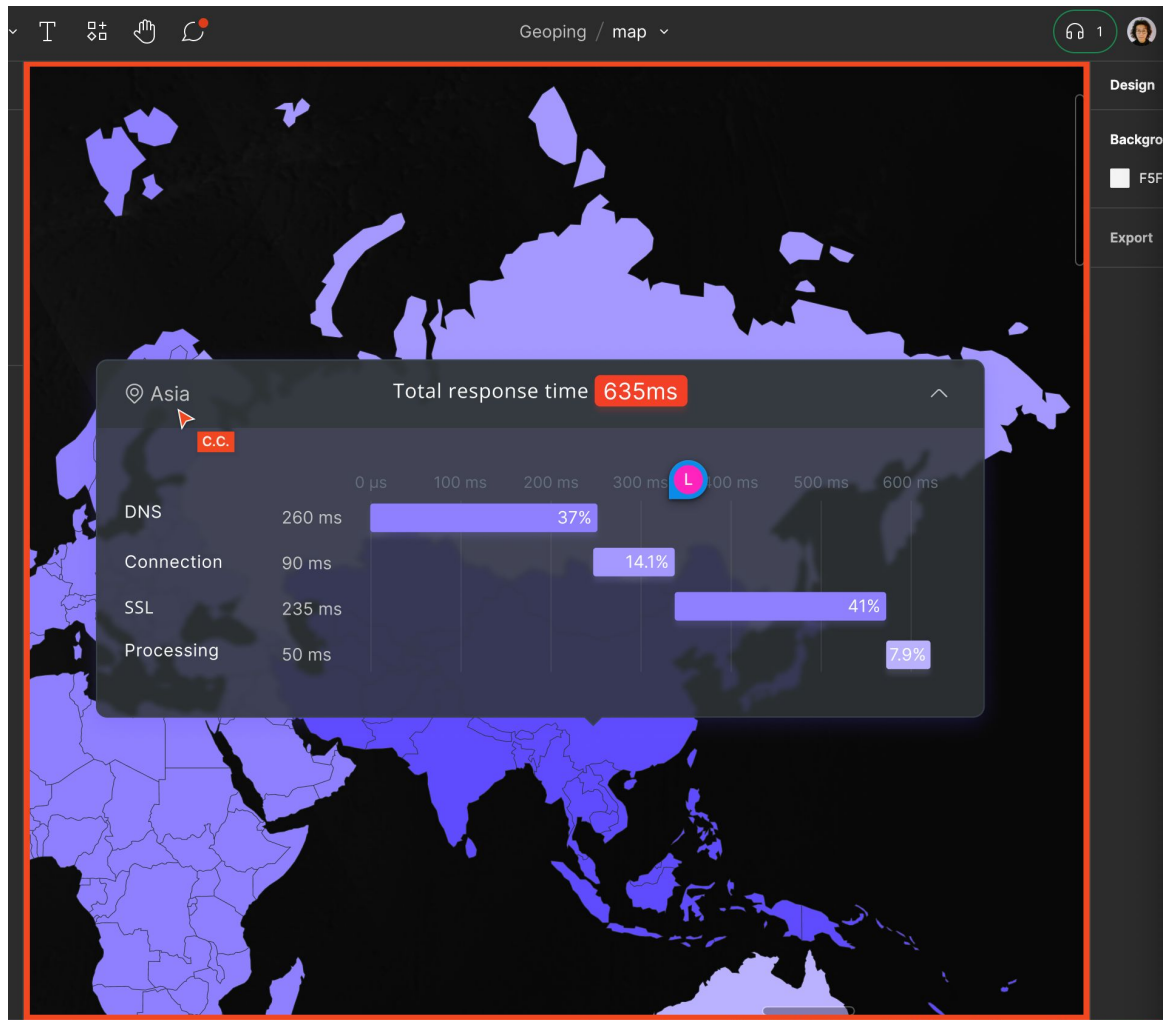
   **Leave** 

Connected

Design Prototype Inspect

Background

<input type="checkbox"/>	F5F5F5	100%	
--------------------------	--------	------	---





share-card

webtransport.day

Yona
like this color

A public webtransport echo service

yep!



× Headers Payload **Messages** Initiator Timing

realtime_v2?release_git_tag=relea...
 livegraph-next
 iFUgjQbUbUhBSDtqyE4MEC?rol...

All

Data	Len...	Time
↓ Binary Message	49 B	00:36:25.465
↑ Binary Message	36 B	00:36:25.466
↓ Binary Message	54 B	00:36:25.519
↓ Binary Message	54 B	00:36:25.574
↓ Binary Message	54 B	00:36:25.575
↓ Binary Message	54 B	00:36:25.629
↓ Binary Message	54 B	00:36:25.684
↑ Binary Message	36 B	00:36:27.188
↑ Binary Message	37 B	00:36:27.229
↓ Binary Message	54 B	00:36:27.360
↓ Binary Message	54 B	00:36:27.415
↓ Binary Message	54 B	00:36:27.470
↓ Binary Message	84 B	00:36:28.985
↓ Binary Message	20 B	00:36:29.047
↑ Binary Message	10 B	00:36:34.983

```
1 00000000: 6364 6265 6464 6067 6460 6a62 6448 6865  cdbedd`gd`jbdHhe
2 00000001: 60d0 61fe cfc0 c000 c68c 10cc 7286 fdf9  `a.....r...
3 00000002: 7e66 a000 00                                ~f...
4
```

3 / 225 requests | 0 B / 234 kB transfe Hex Viewer

Elements Console Sources **Network** Performance Memory Application >> 7 1

Filter Invert Hide data URLs

All Fetch/XHR JS CSS Img Media Font Doc **WS** Wasm Manifest Other Has blocked cookies Blocked Requests

3rd-party requests

5000 ms 10000 ms 15000 ms 20000 ms 25000 ms 30000 ms 35000 ms 40000 ms 45000 ms 50000 ms 55000 ms

Name × **Headers** Payload Messages Initiator Timing

realtime_v2?release_git_tag=relea...
 livegraph-next
 iFUgjQbUbUhBSDtqyE4MEC?rol...

General

Request URL: wss://www.figma.com/api/multiplayer/iFUgjQbUbUhBSDtqyE4MEC?role=edito
r&tracking_session_id=0kGQIqowE3XlsskX&version=62&recentReload=0&scenagraph-queri
es-initial-nodes=972:57319&compression=zstd&user-id=306747767707878005&client_rele
ase=fba0445ff6247e2d9528b5b25c84b6458df5229c

Request Method: GET

Status Code: ● 101 Switching Protocols

Response Headers View source

Alt-Svc: h3=":443"; ma=86400

Connection: upgrade

Date: Wed, 21 Sep 2022 16:27:32 GMT

Sec-WebSocket-Accept: MLKUJLEep0osrDXVZhD8Ys5xtYY=

Strict-Transport-Security: max-age=31536000; includeSubDomains; preload

Upgrade: websocket

Via: 1.1 2ba4fa17a6520457d85279d22c861050.cloudfront.net (CloudFront)

X-Amz-Cf-Id: ZyGpcFbWloBrDxmGm0Qg0sGJuXqzAzfjI_di43Kv0gLouUIpdxCVPW==

X-Amz-Cf-Pop: NRT12-C4

X-Cache: Miss from cloudfront





Geoping - Global Latency checker

https://geoping.gg

AllegroPulse Orbit GDrive Cloudcraft DeepL Panelbear OxCc Excalidraw GA AWS-SDH Lark QUIC AWS-Xile Xile Org Chart

YoMo

wss://www.figma.com/api/multiplaye

Result 7

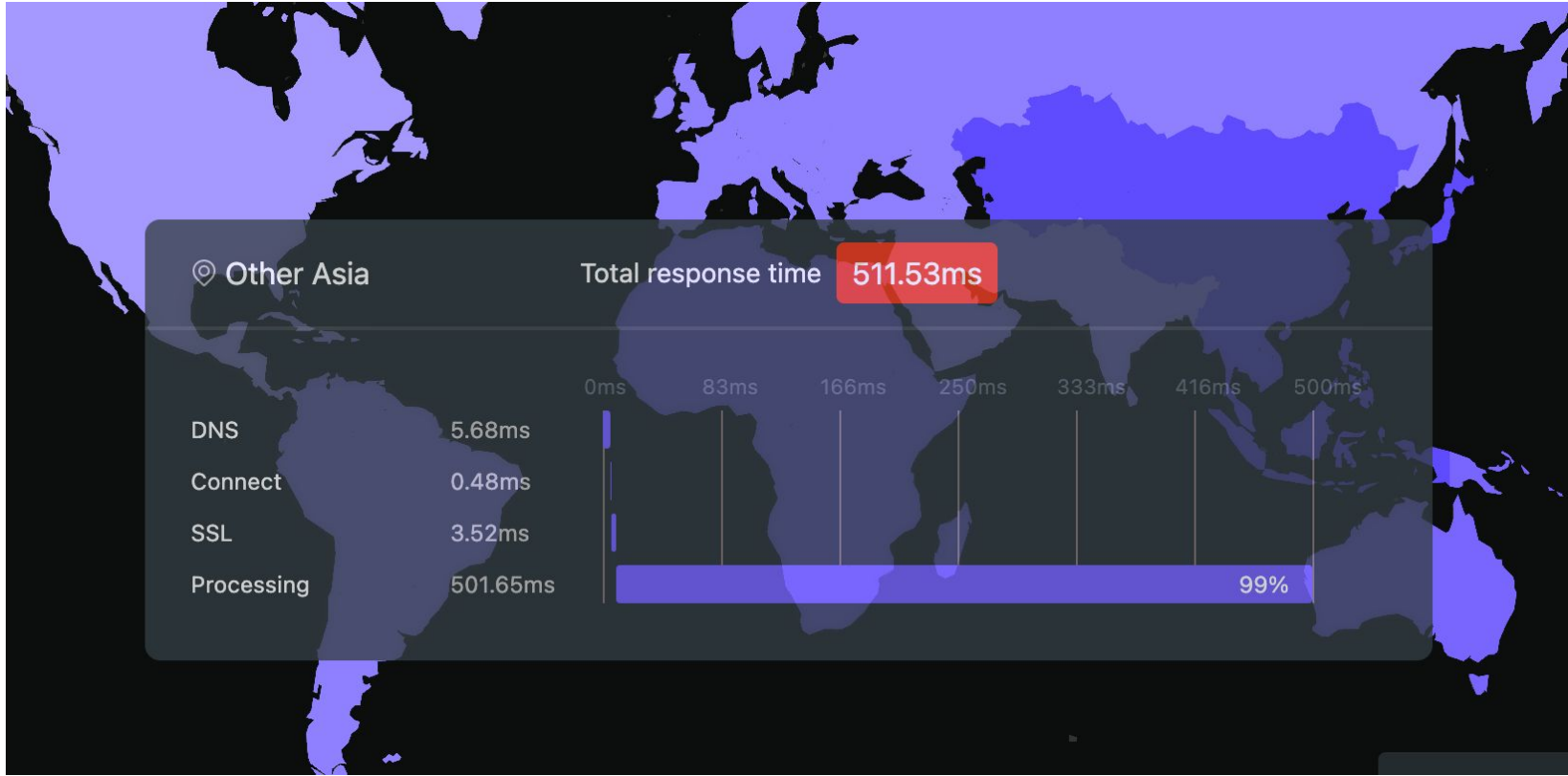
aws

Loc	IP	Conn	Proc	TTFB
Europe	143.204.215.8	1.35ms	153.61ms	173.33ms
North America	65.8.158.72	0.60ms	31.32ms	57.47ms
Other Asia	13.33.88.129	0.48ms	501.65ms	511.53ms
Middle-East	13.227.138.48	1.29ms	222.26ms	246.63ms
Oceania	65.8.33.90	12.20ms	450.85ms	487.44ms
South America	52.84.83.91	2.04ms	201.91ms	220.13ms
Africa	52.85.24.76	0.62ms	280.45ms	484.02ms

Copy Link Share to Twitter

mapbox © Mapbox © OpenStreetMap Improve this map

- 0-50ms
- 50-100ms
- 100-300ms
- >300ms





Loc	IP	Conn	Proc	TTFB
Europe	143.204.215.8	1.35ms	153.61ms	173.33ms
North America	65.8.158.72	0.60ms	31.32ms	57.47ms
Other Asia	13.33.88.129	0.48ms	501.65ms	511.53ms
Middle-East	13.227.138.48	1.29ms	222.26ms	246.63ms
Oceania	65.8.33.90	12.20ms	450.85ms	487.44ms
South America	52.84.83.91	2.04ms	201.91ms	220.13ms
Africa	52.85.24.76	0.62ms	280.45ms	484.02ms



Dynamic wi

Keyboard Controls

Arrow keys (or hjkl) to move a box,
Alt+Up/Down to rotate, and hold
shift to increase the delta.

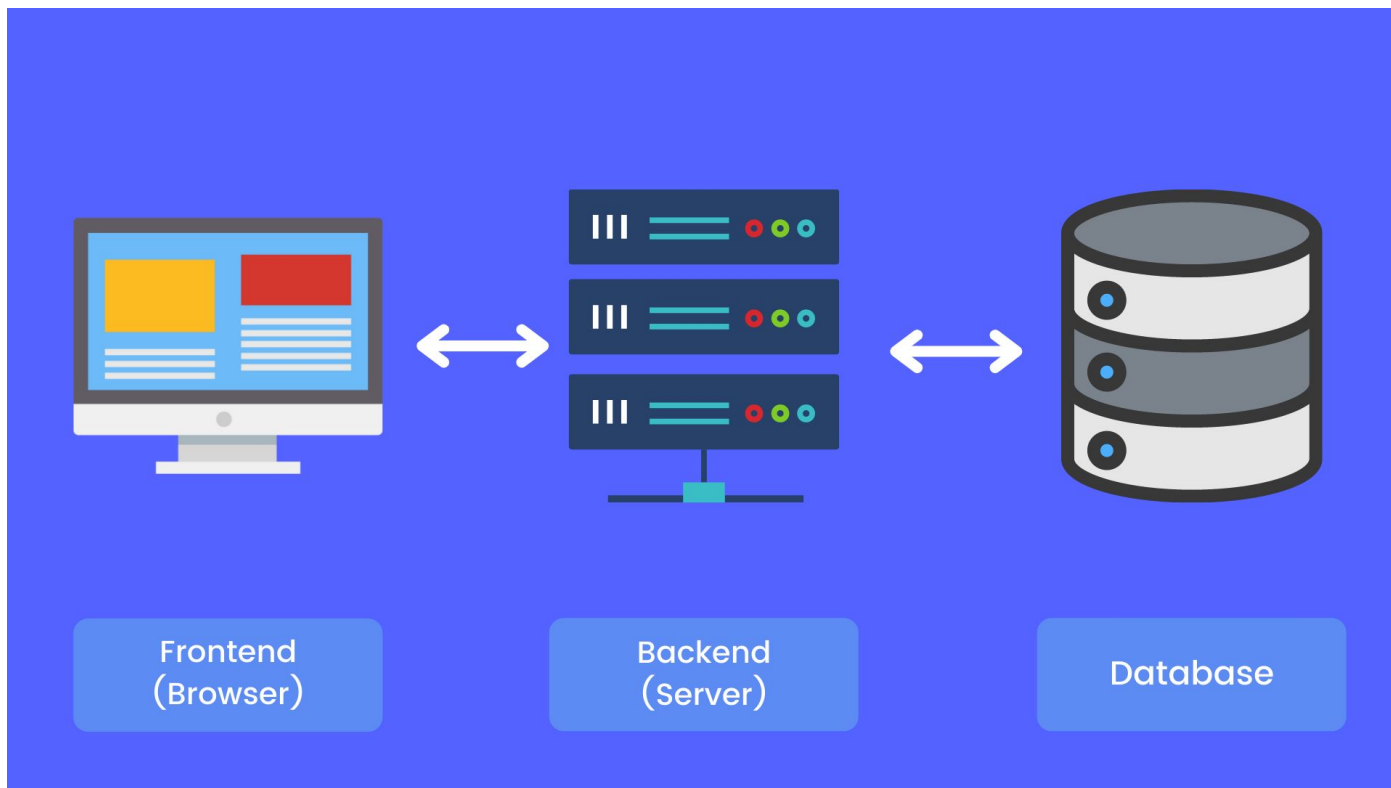
Type / to chat

Without Limits

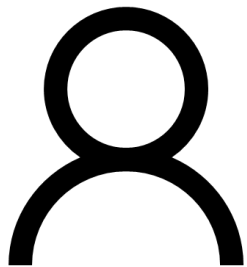
Type / to chat



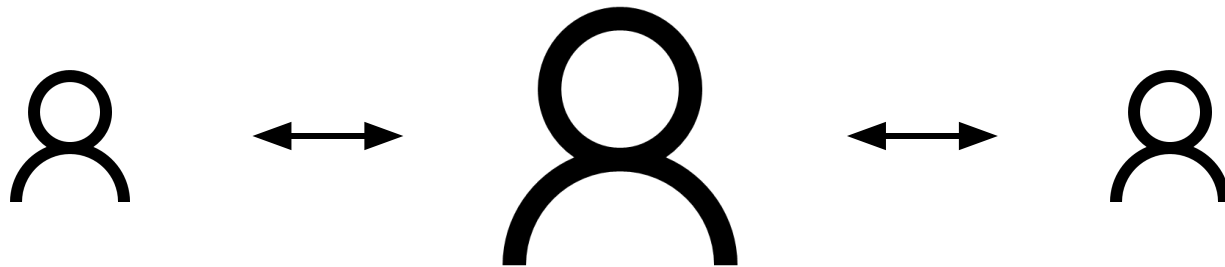
In the past decades



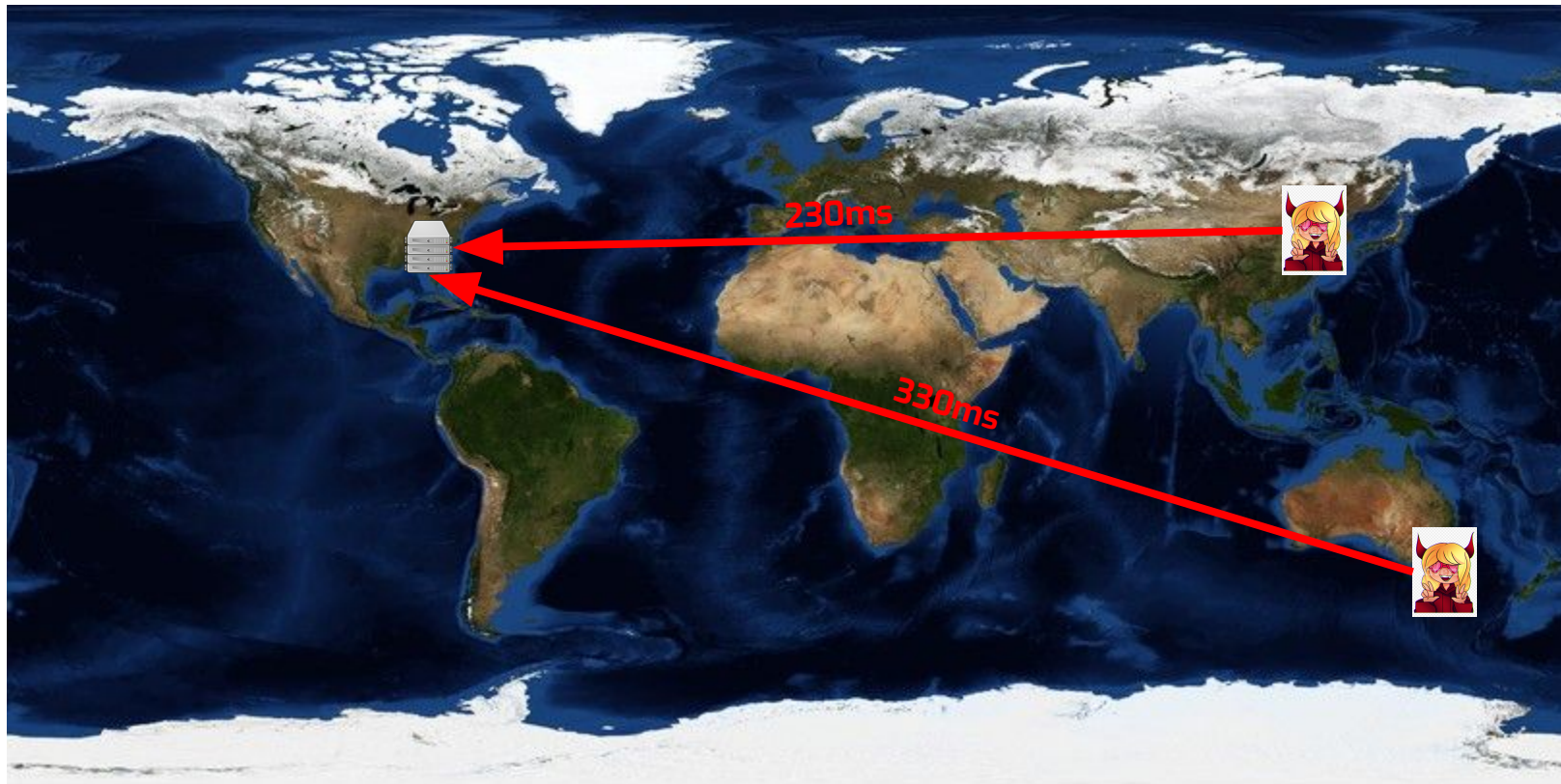
User - Database



User - User



API & Database, Distributed ?





commerce.acme.vercel.live

Search for products...

Next.js Conf SE — T-Shirt

\$50

Love the new logo

Nice detail!

Should we change this color?

Code Chat Draw

```
.. > pages > index.tsx
/
8 export async function getStaticProp
9   preview,
10  locale,
11  locales,
12  }: GetStaticPropsContext) {
13   const config = { locale, locales
14   const { products } = await commerc
15   variables: { first: 12 },
16   config,
17   preview,
18   ...({ featured: true } as any),
19 }
20 const { categories, brands } = awa
21 const { pages } = await commerc.e
22
23 return {
24   props: {
25     products,
26     categories,
27     brands,
28     pages,
29   },
30   revalidate: 14400,
31 }
32 }
33
34 export default function Home({
35   products,
36  }: InferGetStaticPropsType<typeof g
37   return (
38     <Grid>
```

commerce.acme.vercel.live

Search for products...


Next.js Conf SE — T-Shirt

\$50

Love the new logo

Should we change this color

```
.. > pages > index.tsx
/
8 export async function getStaticProp
9   preview,
10  locale,
11  locales,
12 } : GetStaticPropsContext) {
13   const config = { locale, locales
14   const { products } = await commerc
15   variables: { first: 12 },
16   config,
17   preview,
18   ...({ featured: true } as any),
19 })
20 const { categories, brands } = awa
21 const { pages } = await commerc.e
22
23 return {
24   props: {
25     products,
26     categories,
27     brands,
28     pages,
29   },
30   revalidate: 14400,
31 }
32 }
33
34 export default function Home({
35   products,
36 } : InferGetStaticPropsType<typeof g
37   return (
38     <Grid>
```



WORLD OF FORTICE

PRESENTED BY FORT MITCHELL

<> Code | Chat | Draw | +1



 **Web is migrating to the edge**

Hydration: React 18, SSR, Streaming



reactwg / react-18 Public

<> Code

🔗 Pull requests

🗨 Discussions

▶ Actions

🛡 Security

📊 Insights

New Suspense SSR Architecture in React 18 #37

gaearon announced in Deep Dive



gaearon on Jun 5 Maintainer

edited ▾ ⋮

Overview

React 18 will include architectural improvements to React server-side rendering (SSR) performance. These improvements are substantial and are the culmination of several years of work. Most of these improvements are behind-the-scenes, but there are some opt-in mechanisms you'll want to be aware of, especially if you *don't* use a framework.

The primary new API is `renderToPipeableStream`, which you can read about in [Upgrading to React 18 on the Server](#). We plan to write more about it in detail as it's not final and there are things to work out.

The primary existing API is `<Suspense>`.

This page is a high-level overview of the new architecture, its design, and the problems it solves.

SSR Streaming



Server-Side Streaming

Concurrent features in [React 18](#) include built-in support for server-side Suspense and SSR streaming support. This allows you to server-render pages using HTTP streaming. This is an experimental feature in Next.js 12, but **once enabled, SSR will use the same strict runtime as Middleware.**

To enable, use the experimental flag ``concurrentFeatures: true``:

```
// next.config.js
module.exports = {
  experimental: {
    concurrentFeatures: true
  }
}
```




The future of edge computing is here

Now you can have the benefits of dynamic at the speed of static with Vercel Edge Functions.

[Learn about Edge Functions](#)

Say goodbye to cold boots

With Middleware deployed to Vercel, run globally-deployed functions, without any configuration. Even better? Your Next.js application will be optimized for the best performance every time.

De-risk experimentation

From feature flags to A/B testing, Middleware allows your developers to dynamically reconfigure routing at runtime to remove performance impacts and de-risk experimentation.

Make it personalized and fast

Reduce reliance on client-side JavaScript by deploying on the server without any performance tradeoffs. Your team can now serve users the right experience the first time with just code!

unite.shopify.com/...

EXPLORER

- OPEN EDITORS
 - App.server.jsx src
- SNOWDEVIL
 - .github
 - node_modules
 - public
 - src
 - components
 - pages
 - App.server.jsx
 - entry-client.jsx
 - entry-server.jsx
 - favicon.svg
 - index.css
 - .gitignore
 - Dockerfile
 - index.html
 - jsconfig.json
 - package.json

```
src > App.server.jsx
1 > import {
6
7   export default function DefaultRoutes({ pages, serverState,
8     fallback, }): JSX.Element
9     import DefaultRoutes
10
11   return
12     <ShopifyServerProvider>
13       <Switch>
14         @see — https://vitejs.dev/guide/features.html#glob-import
15         <DefaultRoutes
16           pages={pages}
17           serverState={serverState}
18           fallback={<NotFound />}
19         />
20       </Switch>
21     </ShopifyServerProvider>
22   );
23
24
25
26
27
```



implements react router
from hydrogen we get a

Shopify Oxygen



COMING 2022 COMING 2022 COMI

Host directly on Shopify

Stay tuned for the upcoming Shopify-powered, [global hosting solution for Hydrogen storefronts](#). Oxygen is the fastest way to deploy Shopify-backed commerce experiences.



△ **Web is migrating to the edge, but realtime still not**

Region-to-Region

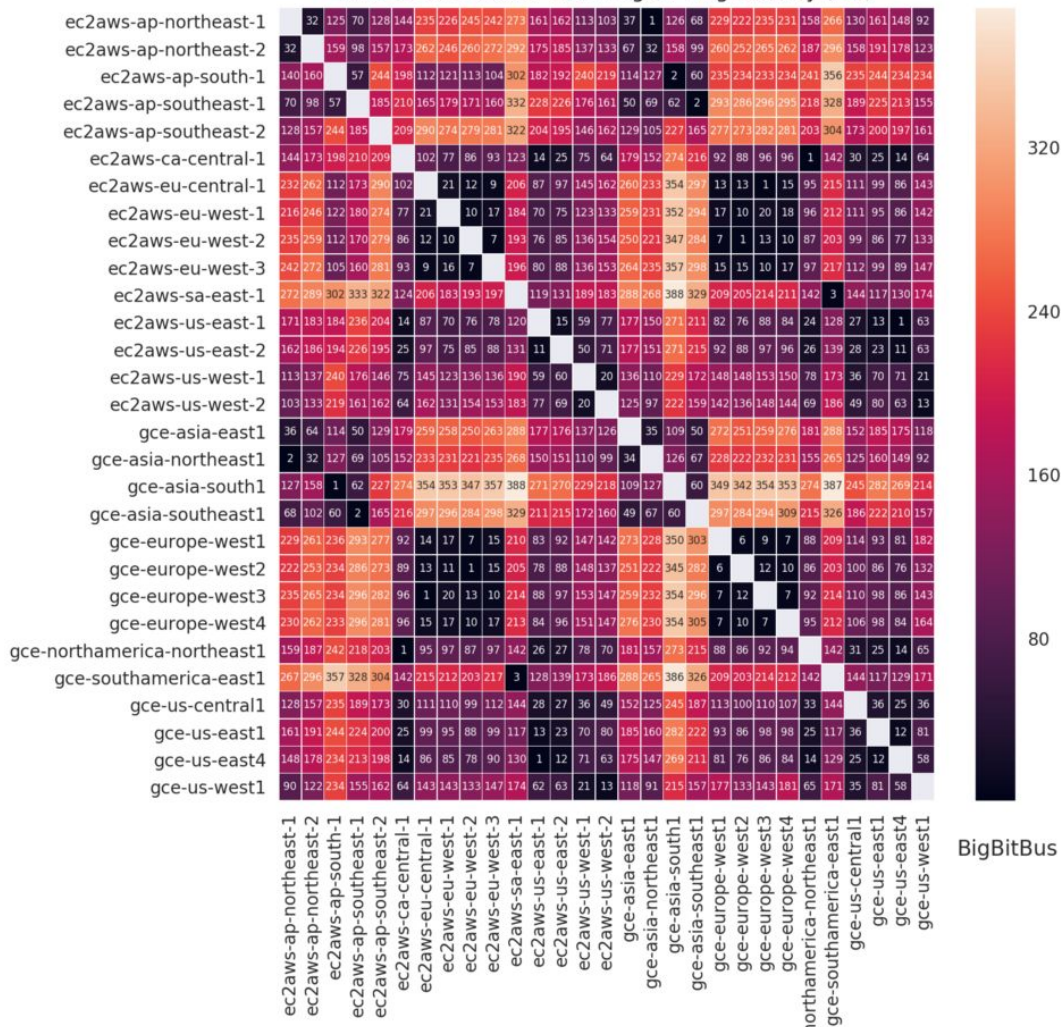


+ Add Source City

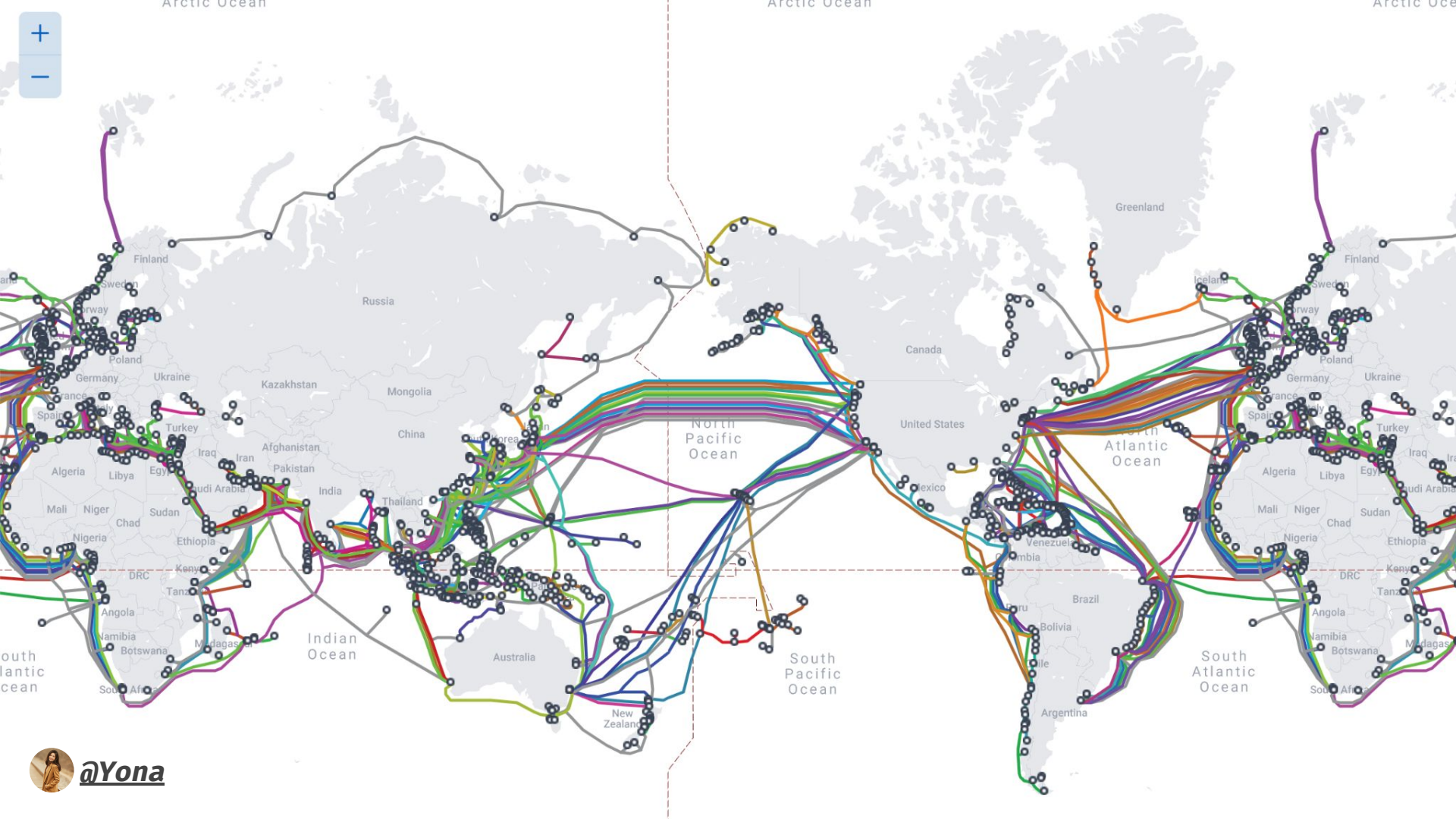
+ Add Destination City

	Barcelona ✖	Hangzhou ✖	Paris ✖	Shanghai ✖	Tokyo ✖	Toronto ✖	Washington ✖
Amsterdam ✖	● 31.509ms	● 209.917ms	● 11.352ms	● 197.382ms	● 231.31ms	● 91.629ms	● 82.043ms
Auckland ✖	● 268.951ms	● 505.796ms	● 303.431ms	● 407.537ms	● 190.376ms	● 259.366ms	● 250.843ms
Copenhagen ✖	● 39.303ms	● 272.211ms	● 23.563ms	● 248.208ms	● 231.074ms	● 111.46ms	● 103.301ms
Dallas ✖	● 127.31ms	● 212.052ms	● 120.314ms	● 200.64ms	● 133.205ms	● 43.383ms	● 36.068ms
Frankfurt ✖	● 35.154ms	● 257.19ms	● 11.317ms	● 247.415ms	● 218.633ms	● 101.32ms	● 94.13ms
Hangzhou ✖	N/A	N/A	● 234.481ms	N/A	● 447.397ms	N/A	N/A
London ✖	● 36.727ms	● 275.612ms	● 8.449ms	● 239.426ms	● 237.192ms	● 92.216ms	● 78.112ms
Los Angeles ✖	● 154.058ms	● 154.344ms	● 159.333ms	● 162.815ms	● 116.154ms	● 70.676ms	● 67.959ms
Moscow ✖	● 83.38ms	● 225.522ms	● 50.12ms	● 203.097ms	● 264.424ms	● 131.561ms	● 126.384ms
New York ✖	● 96.042ms	● 314.683ms	● 73.104ms	● 276.22ms	● 176.782ms	● 20.472ms	● 8.942ms
Paris ✖	● 27.241ms	● 284.159ms	—	● 269.094ms	● 229.661ms	● 93.264ms	● 85.749ms
Shanghai ✖	N/A	● 5.886ms	● 313.506ms	—	N/A	● 240.598ms	N/A
Stockholm ✖	● 51.576ms	● 215.577ms	● 29.546ms	● 246.817ms	● 238.672ms	● 98.411ms	● 100.142ms
Tokyo ✖	● 212.995ms	● 80.695ms	● 228.448ms	● 81.588ms	—	● 163.561ms	● 172.569ms

AWS & GCE Clouds' Inter-region Ping Latency (ms)



BigBitBus

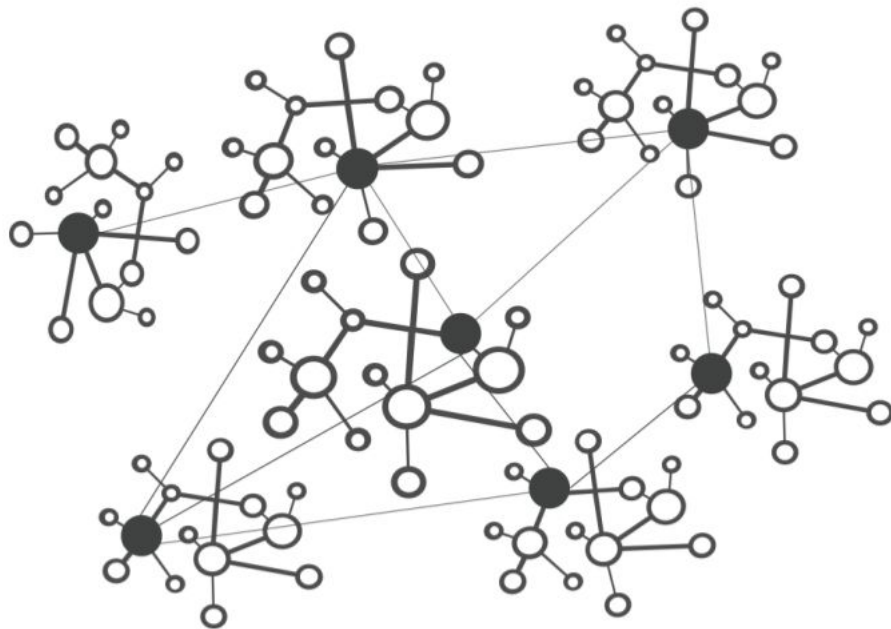


@Yona

YoMo helps organize global workload



The Internet
A Network of Networks



 **New realtime communication protocol**



Using WebTransport

WebTransport is an API offering low-latency, bidirectional, client-server messaging. Learn more about its use cases, and how to give feedback about the future of the implementation.

Jun 8, 2020 — Updated Jul 18, 2022

Available in: [English](#), [Русский](#), [中文](#), [日本語](#), and [한국어](#)



[Jeff Posnick](#)

[Twitter](#)

[GitHub](#)

[Glitch](#)

[Homepage](#)



What's WebTransport?

[WebTransport](#) is a web API that uses the [HTTP/3](#) protocol as a bidirectional transport. It's intended for two-way communications between a web client and an HTTP/3 server. It supports sending data both unreliably via its [datagram APIs](#), and reliably via its [streams APIs](#).

Unreliable Support



[Datagrams](#) are ideal for sending and receiving data that do not need strong delivery guarantees. Individual packets of data are limited in size by the [maximum transmission unit \(MTU\)](#) of the underlying connection, and may or may not be transmitted successfully, and if they are transferred, they may arrive in an arbitrary order. These characteristics make the datagram APIs ideal for low-latency, best-effort data transmission. You can think of datagrams as [user datagram protocol \(UDP\)](#) messages, but encrypted and congestion-controlled.

Streams API



The streams APIs, in contrast, provide [reliable](#), ordered data transfer. They're [well-suited](#) to scenarios where you need to send or receive one or more streams of ordered data. Using multiple WebTransport streams is analogous to establishing multiple [TCP](#) connections, but since HTTP/3 uses the lighter-weight [QUIC](#) protocol under the hood, they can be opened and closed without as much overhead.



Streams API - SendStream

```
// Send two Uint8Arrays to the server.  
const stream = await transport.createSendStream();  
const writer = stream.writable.getWriter();  
const data1 = new Uint8Array([65, 66, 67]);  
const data2 = new Uint8Array([68, 69, 70]);  
writer.write(data1);  
writer.write(data2);  
try {  
  await writer.close();  
  console.log('All data has been sent.');} catch (error) {  
  console.error(`An error occurred: ${error}`);  
}
```

Streams API - ReceiveStream



```
async function readFrom(receiveStream) {
  const reader = receiveStream.readable.getReader();
  while (true) {
    const {done, value} = await reader.read();
    if (done) {
      break;
    }
    // value is a Uint8Array
    console.log(value);
  }
}
```

```
const rs = transport.receiveStreams();
const reader = rs.getReader();
while (true) {
  const {done, value} = await reader.read();
  if (done) {
    break;
  }
  // value is an instance of ReceiveStream
  await readFrom(value);
}
```



Streams API - BidirectionalStream

```
const rs = transport.receiveBidirectionalStreams();
const reader = rs.getReader();
while (true) {
  const {done, value} = await reader.read();
  if (done) {
    break;
  }
  // value is a BidirectionalStream
  // value.readable is a ReadableStream
  // value.writable is a WritableStream
}
```



Try it out

The best way to experiment with WebTransport is to start up a compatible HTTP/3 server. You can then use this page with a [basic JavaScript client](#) to try out client/server communications.

Additionally, a community-maintained echo server is available at webtransport.day.



Public WebTransport Echo Ser x +

https://webtransport.day

AllegroPulse Orbit GDrive Cloudcraft DeepL Panelbear 0xCC Excalidraw GA AWS-SDH Lark QUIC AWS-Xile Xile Org Chart

WebTransport over HTTP/3 client

Establish WebTransport connection

URL:

Send data over WebTransport

Send a datagram
 Open a unidirectional stream
 Open a bidirectional stream

Event log

```
• Initiating connection...
• Connection ready.
• Datagram writer ready.
• Datagram reader ready.
• Sent datagram: hello Akraino
• Datagram received: hello Akraino
• Opened bidirectional stream #1 with data: hi there, nextarch.io
• Received data on stream #1: hi there, nextarch.io
```

This tool can be used to connect to an arbitrary WebTransport server. It has several limitations:

- It can only send an entirety of a stream at once. Once the stream is opened, all of the data is immediately sent, and the write side of the stream is closed.
- This tool does not listen to server-initiated bidirectional streams.
- Stream IDs are different from the one used by QUIC on the wire, as the on-the-wire IDs are not exposed via the Web API.
- The `WebTransport` object can be accessed using the developer console `ViacurrentTransport`.

Learn Resource

- [web.dev - Using WebTransport](#)
- [w3.org WebTransport](#)
- [presence.js](#)
- [W3C WebTransport Working Group Updates - October 2021](#)

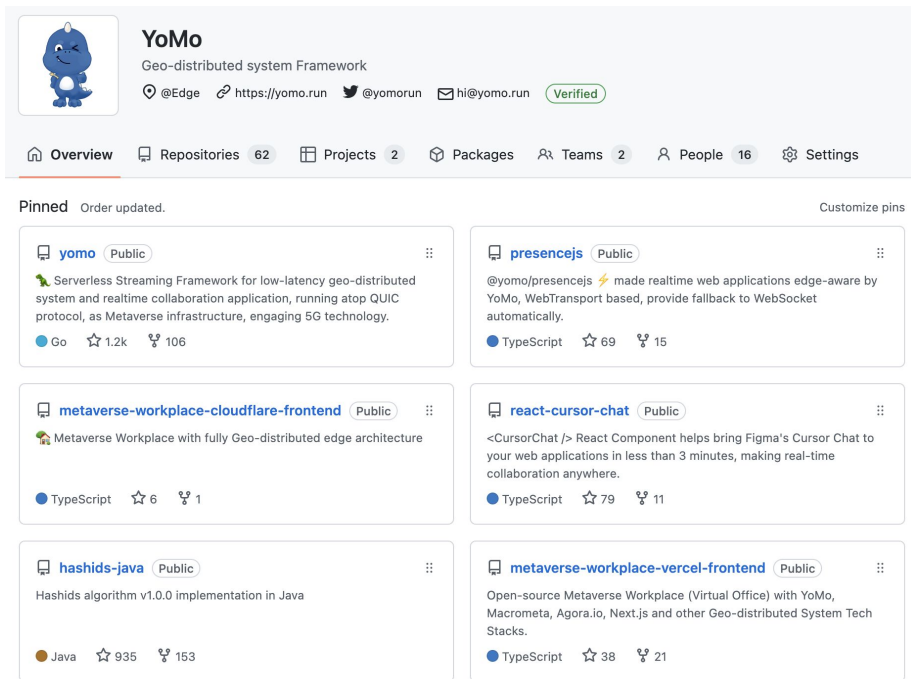
Try it out

- [WebTransport Demo](#)

Serverless

- [Write your own](#)

What we are crafting




The screenshot shows the GitHub profile for YoMo, a verified account. The profile header includes the name 'YoMo', the description 'Geo-distributed system Framework', and contact information: '@Edge', 'https://yomo.run', '@yomorun', and 'hi@yomo.run'. The navigation bar shows 'Overview' as the active tab, along with 'Repositories 62', 'Projects 2', 'Packages', 'Teams 2', 'People 16', and 'Settings'. The 'Pinned' section displays six repositories:

- yomo** (Public): Serverless Streaming Framework for low-latency geo-distributed system and realtime collaboration application, running atop QUIC protocol, as Metaverse infrastructure, engaging 5G technology. 1.2k stars, 106 forks.
- presencejs** (Public): @yomo/presencejs made realtime web applications edge-aware by YoMo, WebTransport based, provide fallback to WebSocket automatically. 69 stars, 15 forks.
- metaverse-workplace-cloudflare-frontend** (Public): Metaverse Workplace with fully Geo-distributed edge architecture. 6 stars, 1 fork.
- react-cursor-chat** (Public): <CursorChat /> React Component helps bring Figma's Cursor Chat to your web applications in less than 3 minutes, making real-time collaboration anywhere. 79 stars, 11 forks.
- hashids-java** (Public): Hashids algorithm v1.0.0 implementation in Java. 935 stars, 153 forks.
- metaverse-workplace-vercel-frontend** (Public): Open-source Metaverse Workplace (Virtual Office) with YoMo, Macrometa, Agora.io, Next.js and other Geo-distributed System Tech Stacks. 38 stars, 21 forks.

- **WebTransport-polyfill** works with web browsers other than Chrome
- Provides developers with an easy-to-use **programming interface** rather than standard Streaming API
- Introduces a **wire protocol**, so every front-end developer doesn't need to build their communication specification.

What we are crafting





YoMo

Geo-distributed system Framework

@Edge <https://yomo.run> @yomorun hi@yomo.run Verified

[Overview](#) [Repositories 62](#) [Projects 2](#) [Packages](#) [Teams 2](#) [People 16](#) [Settings](#)

Pinned [Order updated.](#) [Customize pins](#)

[yomo](#) Public

Serverless Streaming Framework for low-latency geo-distributed system and realtime collaboration application, running atop QUIC protocol, as Metaverse infrastructure, engaging 5G technology.

Go 1.2k 106

[presencejs](#) Public

@yomo/presencejs made realtime web applications edge-aware by YoMo, WebTransport based, provide fallback to WebSocket automatically.

TypeScript 69 15

[metaverse-workplace-cloudflare-frontend](#) Public

Metaverse Workplace with fully Geo-distributed edge architecture

TypeScript 6 1

[react-cursor-chat](#) Public

<CursorChat /> React Component helps bring Figma's Cursor Chat to your web applications in less than 3 minutes, making real-time collaboration anywhere.

TypeScript 79 11

[hashids-java](#) Public

Hashids algorithm v1.0.0 implementation in Java

Java 935 153

[metaverse-workplace-vercel-frontend](#) Public

Open-source Metaverse Workplace (Virtual Office) with YoMo, Macrometa, Agora.io, Next.js and other Geo-distributed System Tech Stacks.

TypeScript 38 21

- **Unified backend** supports both WebTransport and WebSocket.
- Functions can be written in **WebAssembly** and deployed anywhere - browser, edge or cloud.



Thanks

<https://github.com/yomorun/yomo>

2022 Sep