

# THE CODE BEHIND COLLABORATION COMPONENTS

VÍTOR NORTON  
DEV ADVOCATE @ SUPERVIZ

# VITOR NORTON

Brazilian  
10x Microsoft MVP  
Developer Advocate @ SuperViz

# VITOR NORTON

Passionate about connecting people

Highly dependent of my productivity tools

Love the idea of working anywhere in the world

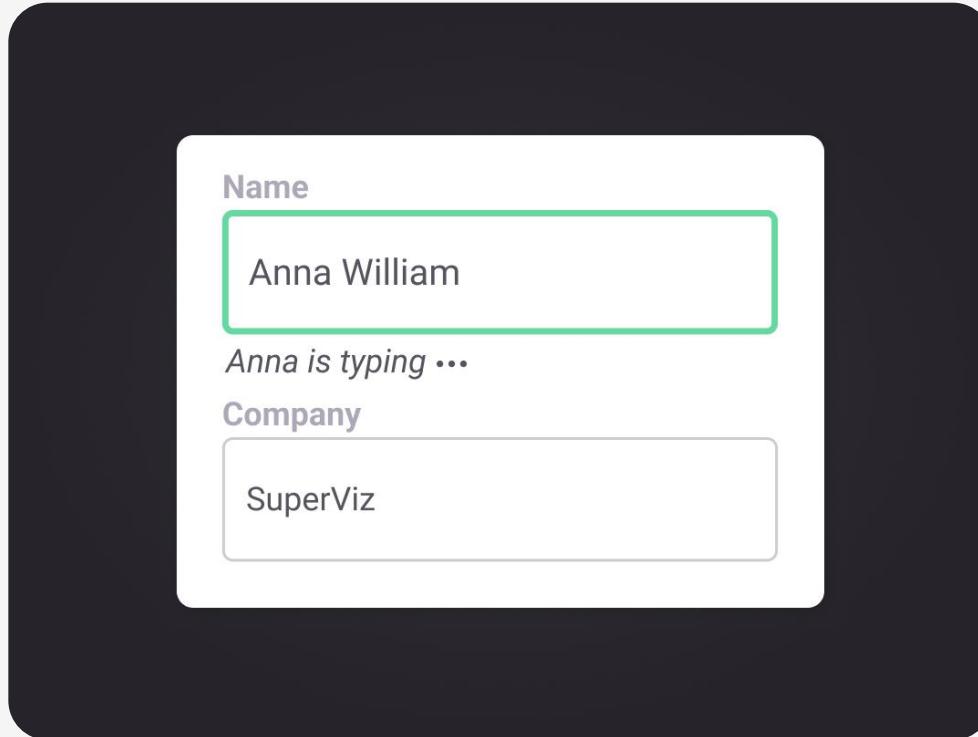
Brazilian

10x Microsoft MVP

Developer Advocate @ SuperViz

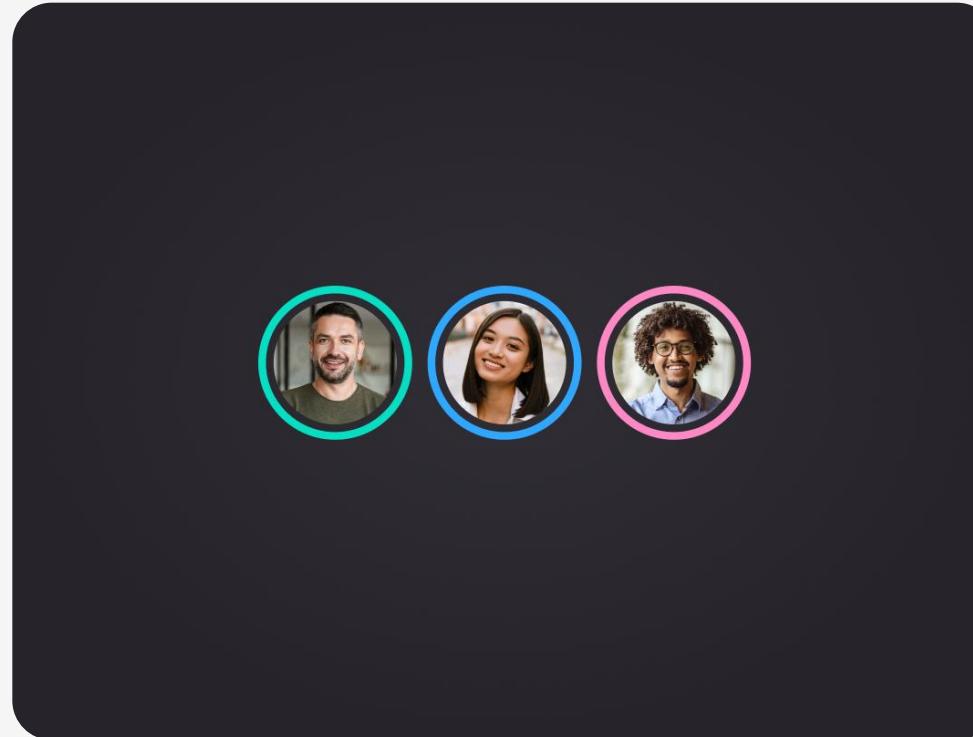
# **COLLABORATION COMPONENTS**

# COLLABORATION COMPONENTS



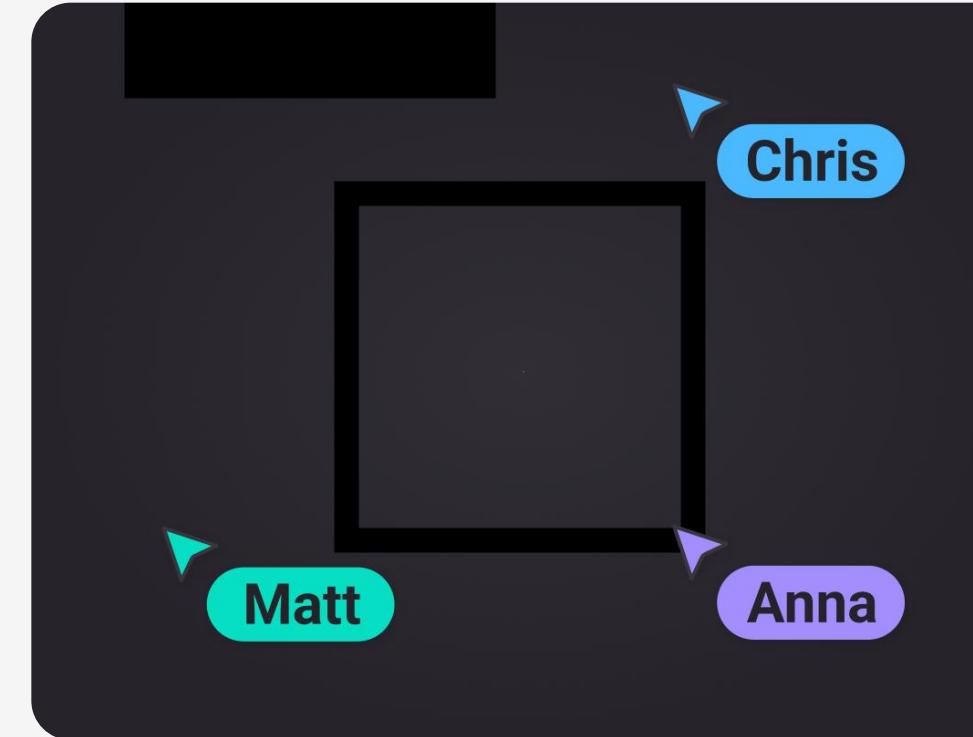
## FORMS ELEMENTS

Enables real-time sync of form elements, such as input fields, checkboxes, among participants in the same room.



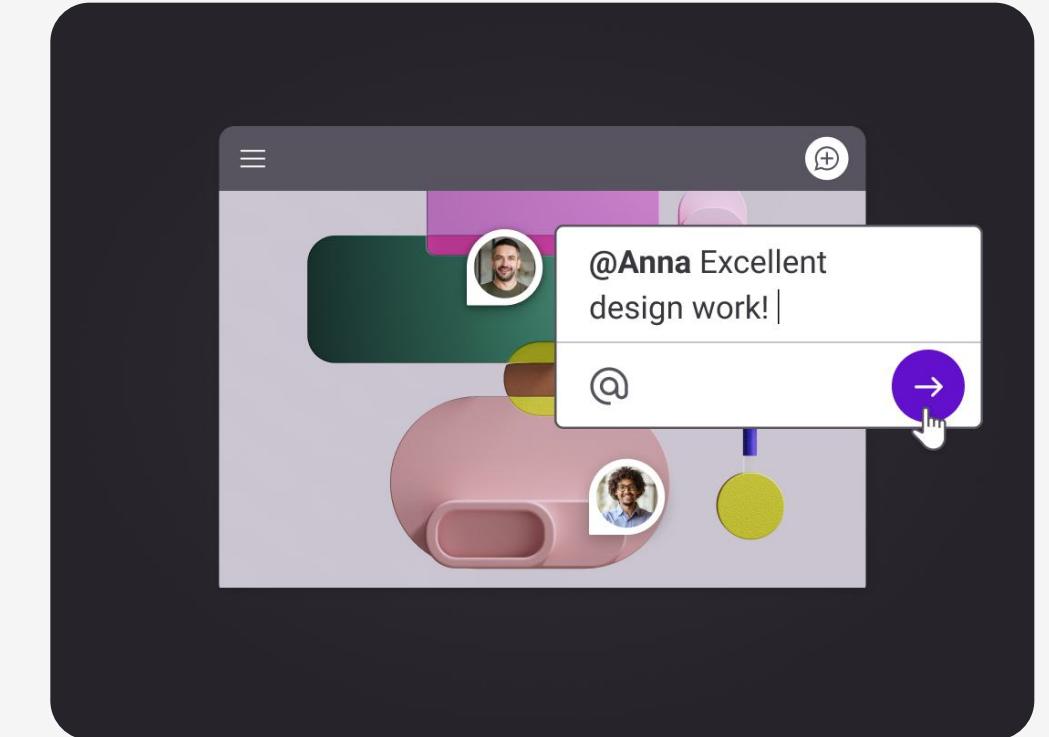
## WHO-IS-ONLINE

The Who-is-On-line component allows you to see all the participants who are connected in a room.



## MOUSE POINTERS

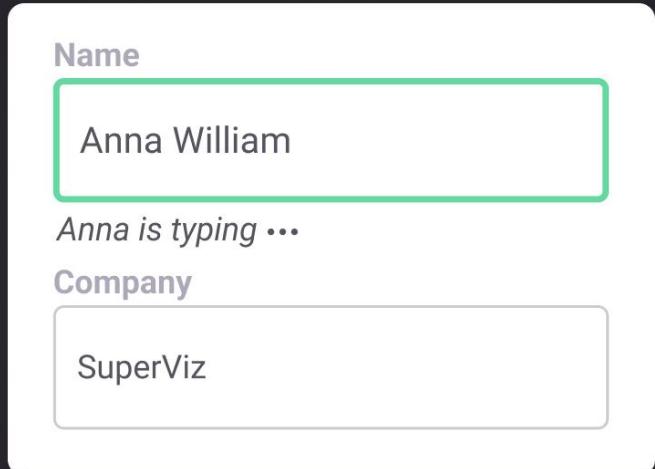
It enables real-time tracking of participants' cursor movements, allowing seamless collaboration within the same room.



## CONTEXTUAL COMMENTS

It embeds a customizable commenting experience into your page to enable people to collaborate.

# COLLABORATION COMPONENTS



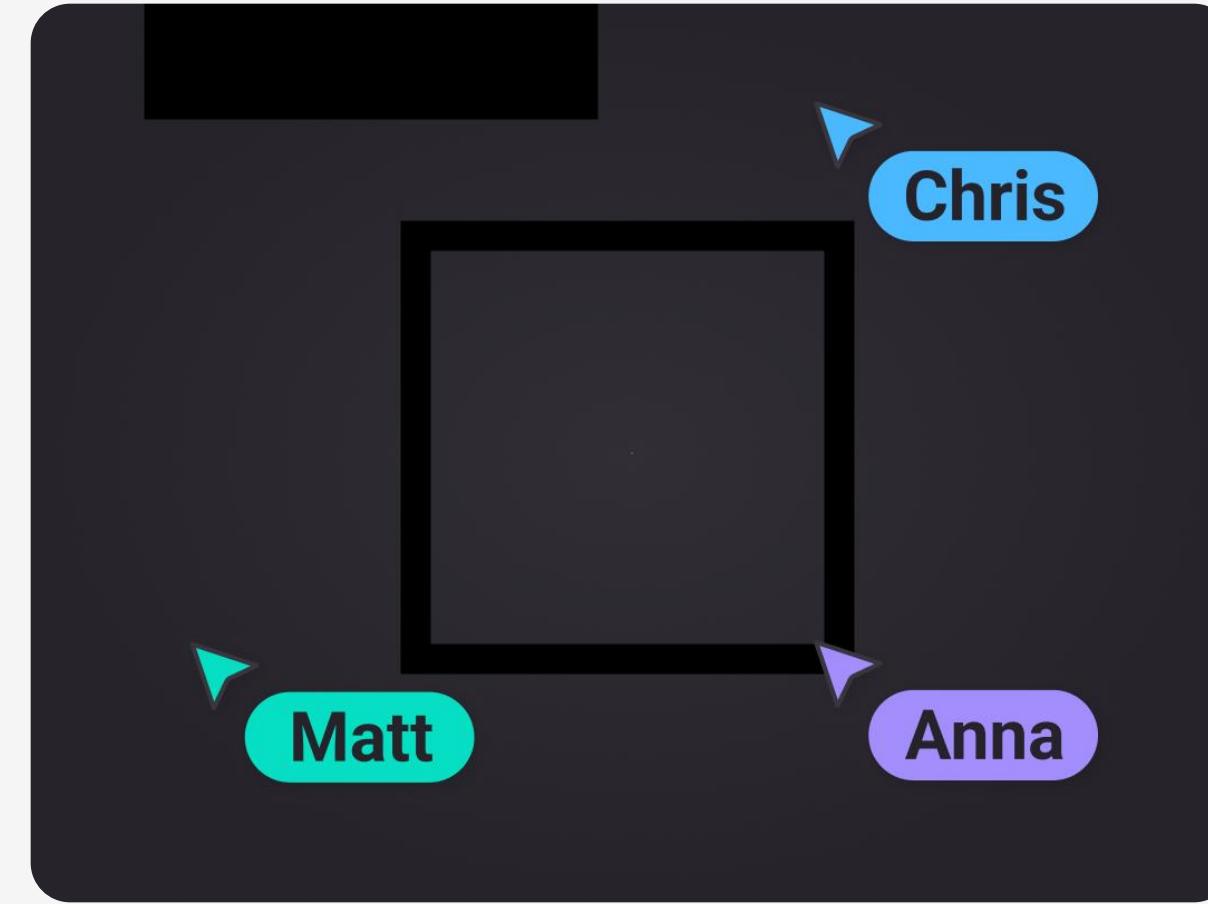
## FORMS ELEMENTS

Enables real-time sync of form elements, such as input fields, checkboxes, among participants in the same room.



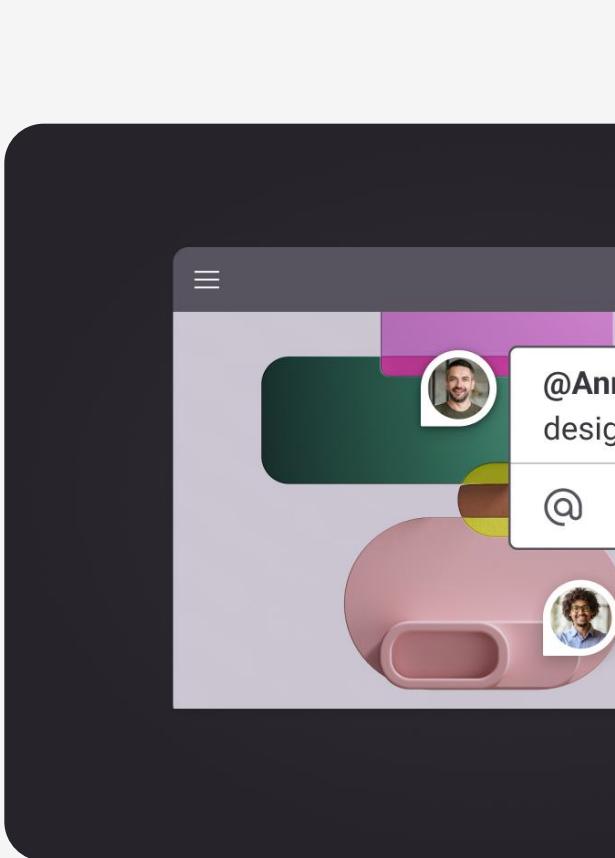
## WHO-IS-ONLINE

The Who-is-On-line component allows you to see all the participants who are connected in a room.



## MOUSE POINTERS

It enables real-time tracking of participants' cursor movements, allowing seamless collaboration within the same room.



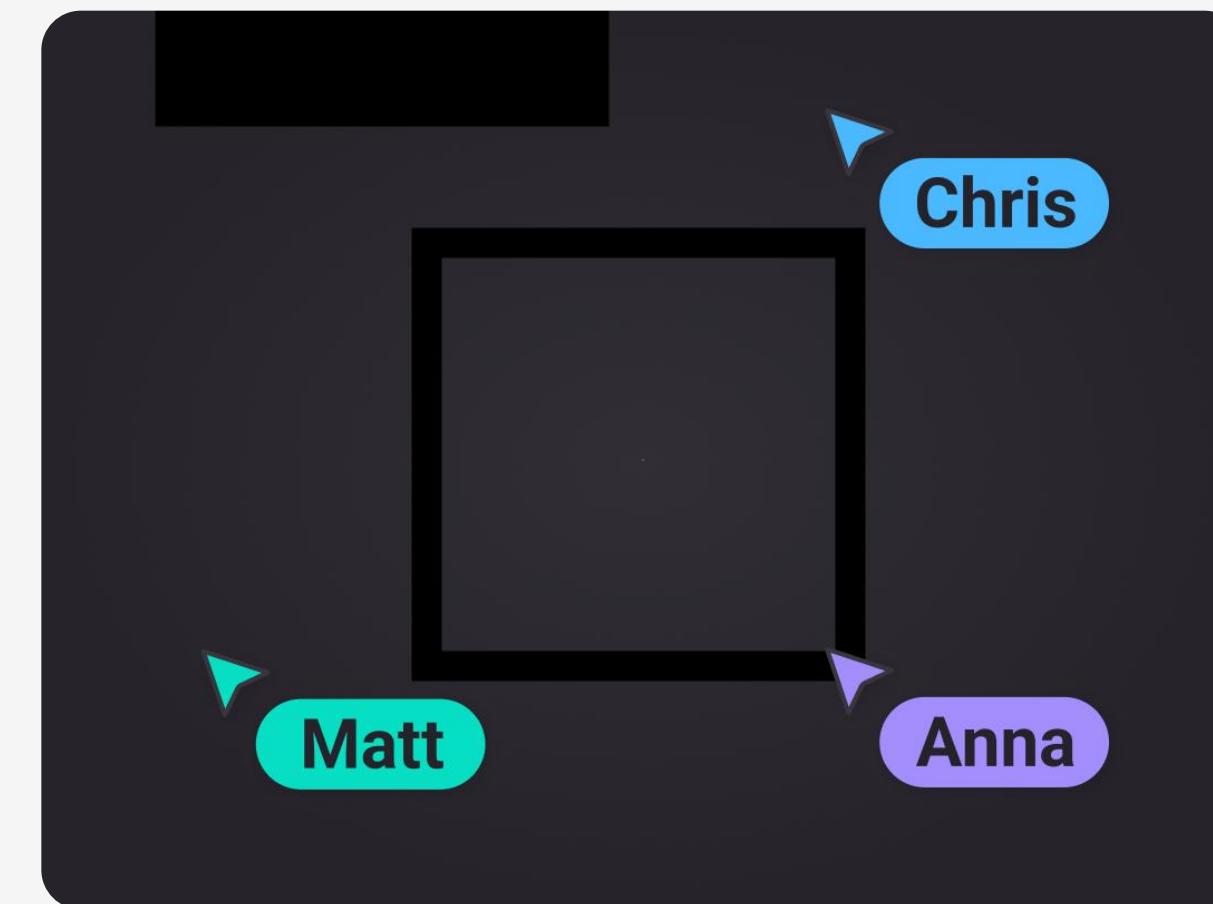
## CONTEXTUAL COMMENTS

It embeds a custom commenting experience directly onto your page to enable real-time collaboration.

# LET'S BREAK IT DOWN

Go to definition

F12



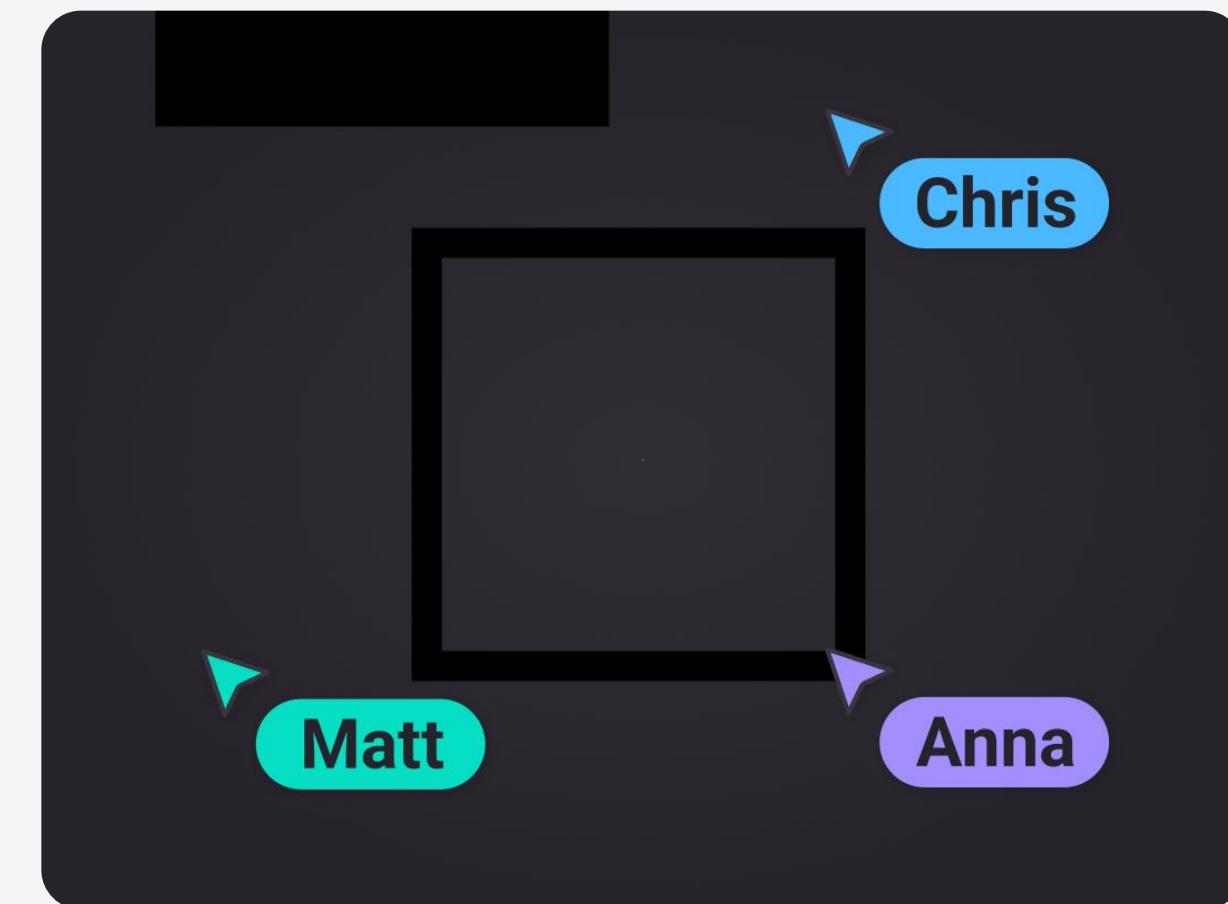
## MOUSE POINTERS

It enables real-time tracking of participants' cursor movements, allowing seamless collaboration within the same room.

# LET'S BREAK IT DOWN

Go to definition

F12



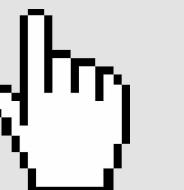
## MOUSE POINTERS

It enables real-time tracking of participants' cursor movements, allowing seamless collaboration within the same room.

# LET'S BREAK IT DOWN

Go to definition

F12



```
1 <MousePointers elementId="element-id" />
2 <div id="element-id" />
```

# LET'S BREAK IT DOWN

Go to definition

F12

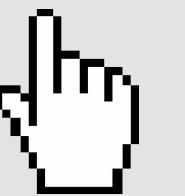


```
1 <MousePointers elementId="element-id" />
2 <div id="element-id" />
```

# LET'S BREAK IT DOWN

Go to definition

F12



```
1 const room = await SuperVizRoom(DEVELOPER_KEY,  
2   {  
3     roomId: groupId,  
4     group: {  
5       id: groupId,  
6       name: groupName,  
7     },  
8     participant: {  
9       id: userId,  
10      name: userName,  
11    },  
12  }  
13 );  
14  
15 const mousePointers = new MousePointers("element-id");  
16  
17 room.addComponent(mousePointers);  
18  
19 return room;
```

# LET'S BREAK IT DOWN

```
1 const room = await SuperVizRoom(DEVELOPER_KEY,  
2   {  
3     roomId: groupId,  
4     group: {  
5       id: groupId,  
6       name: groupName,  
7     },  
8     participant: {  
9       id: userId,  
10      name: userName,  
11    },  
12  }  
13 );  
14  
15 const mousePointers = new MousePointers("element-id");  
16  
17 room.addComponent(mousePointers);  
18  
19 return room;
```



# **NEW MOUSEPINTER()**

See when  
someone moves  
their mouse

Get information about the  
position of other's participants  
mouse pointers.

# NEW MOUSEPINTER()

See when  
someone moves  
their mouse

Get information about the  
position of other's participants  
mouse pointers.

Update your  
position to  
everyone

Let the other participants know  
the position of your pointer.

# NEW MOUSEPINTER()

**See when  
someone moves  
their mouse**

Get information about the position of other's participants mouse pointers.

**Update your  
position to  
everyone**

Let the other participants know the position of your pointer.

**Make it really fast  
and smooth**

Update it in a 30fps (frames per second) so it doesn't seem laggy.

# NEW MOUSEPINTER()

See when  
someone moves  
their mouse

Get information about the  
position of other's participants  
mouse pointers.

```
1 const [mousePosition, setMousePosition] = useState({  
2   x: 0,  
3   y: 0  
4 });  
5  
6 channel.subscribe("mouse.moved", onMouseMove);  
7  
8 function onMouseMove(message: RealtimeMessage) {  
9   setMousePosition({  
10     x: message.data.x,  
11     y: message.data.y,  
12   });  
13 }
```

# NEW MOUSEPINTER()

Update your  
position to  
everyone

Let the other participants know  
the position of your pointer.



```
1 document.addEventListener("mousemove", onMouseMove);
2
3 function onMouseMove(event) {
4   const position = {
5     x: event.clientX,
6     y: event.clientY,
7   };
8
9   channel.publish("mouse.moved", position);
10 }
```

# NEW MOUSEPOINTER()

Make it really fast  
and smooth

Update it in a 30fps (frames per second) so it doesn't seem laggy.

```
● ● ●  
1 let isAnimationFrameRequested = false;  
2 let lastMouseEvent: MouseEvent | null = null;  
3  
4 document.addEventListener("mousemove", (event) => {  
5   lastMouseEvent = event;  
6   if (!isAnimationFrameRequested) {  
7     isAnimationFrameRequested = true;  
8     requestAnimationFrame(onMouseMove);  
9   }  
10});  
11  
12 function onMouseMove() {  
13   if (lastMouseEvent) {  
14     const position = {  
15       x: lastMouseEvent.clientX,  
16       y: lastMouseEvent.clientY,  
17     };  
18     channel.publish("mouse.moved", position);  
19   }  
20 }  
21 isAnimationFrameRequested = false;  
22 }
```

# CONST CHANNEL

## What is this channel you've been using?

I know, it's ridiculous of me not to show what is this `channel` const, but it is actually implementing an event broker, which plays a crucial role in our system architecture.

```
● ● ●  
1 room = await SuperVizRoom(DEVELOPER_KEY, {  
2   roomId: groupId,  
3   participant: {  
4     id: participant,  
5     name: "John " + participant,  
6   },  
7 });  
8  
9 const realtime = new Realtime();  
10  
11 channel = await realtime.connect('mouse.handler');  
12  
13 room.addComponent(realtime);
```

# CONST CHANNEL

## What is this channel you've been using?

I know, it's ridiculous of me not to show what is this `channel` const, but it is actually implementing an event broker, which plays a crucial role in our system architecture.

Go to definition

F12



```
1 export declare class Channel {  
2   static events = {};  
3   subscribe: (  
4     event: string,  
5     callback: (data: any) => void)  
6     => void;  
7   publish: (  
8     event: string,  
9     data?: unknown)  
10    => void;  
11 }
```

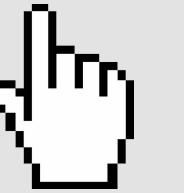
## How does Pub Sub works?

Let's delve into the functionality of the Publisher and Subscriber methods to gain a comprehensive understanding of how they operate.

## The subscribe method

Go to definition

F12

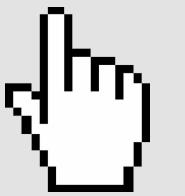


```
● ● ●  
1 function subscribe(eventName, callback) {  
2   // If the event doesn't exist yet,  
3   // initialize it as an empty array  
4   if (!this.events[eventName]) {  
5     this.events[eventName] = [];  
6   }  
7   // Push the callback function into the  
8   // array of callbacks for the given event  
9   this.events[eventName].push(callback);  
10 }  
11 }
```

## The subscribe method

Go to definition

F12



```
● ● ●  
1 function subscribe('mouse.moved', callback) {  
2   // If the event doesn't exist yet,  
3   // initialize it as an empty array  
4   if (!this.events['mouse.moved']) {  
5     this.events['mouse.moved'] = [];  
6   }  
7  
8   // Push the callback function into the  
9   // array of callbacks for the given event  
10  this.events['mouse.moved'].push(callback);  
11 }
```

## The publish method

Go to definition

F12



```
1 // The publish method takes an event name and data
2 function publish(event, data) {
3
4     // For each subscriber of this event,
5     // call the callback function with the provided data
6     this.events[event].forEach(callback => {
7         callback(data);
8     });
9 }
```

## The publish method

Go to definition

F12



```
1 // The publish method takes an event name and data
2 function publish('mouse.moved', data) {
3
4   // For each subscriber of this event,
5   // call the callback function with the provided data
6   this.events['mouse.moved'].forEach((callback) => {
7     callback(data);
8   });
9 }
```

Well, it is this... but

Well, it is this... but

one more thing

one more thing

web sockets

# web sockets

PubSub works locally,  
until you add a  
websocket to it

# web sockets

WebSockets provide a protocol that allows for a continuous, two-way communication channel between a client and server, enabling real-time data exchange over a single, persistent connection without the need for repeated HTTP requests.

# web sockets

WebSockets provide a protocol that allows for a continuous, two-way communication channel between a client and server, enabling real-time data exchange over a single, persistent connection without the need for repeated HTTP requests.

## PERSISTENT CONNECTIONS

Stays open for continuous data exchange

## LOW LATENCY

Ideal for real-time applications like chat, gaming, and live updates

## BIDIRECTIONAL

Both client and server can send messages independently

# web sockets

WebSockets provide a protocol that allows for a continuous, two-way communication channel between a client and server, enabling real-time data exchange over a single, persistent connection without the need for repeated HTTP requests.

## PERSISTENT CONNECTIONS

Stays open for continuous data exchange

## LOW LATENCY

Ideal for real-time applications like chat, gaming, and live updates

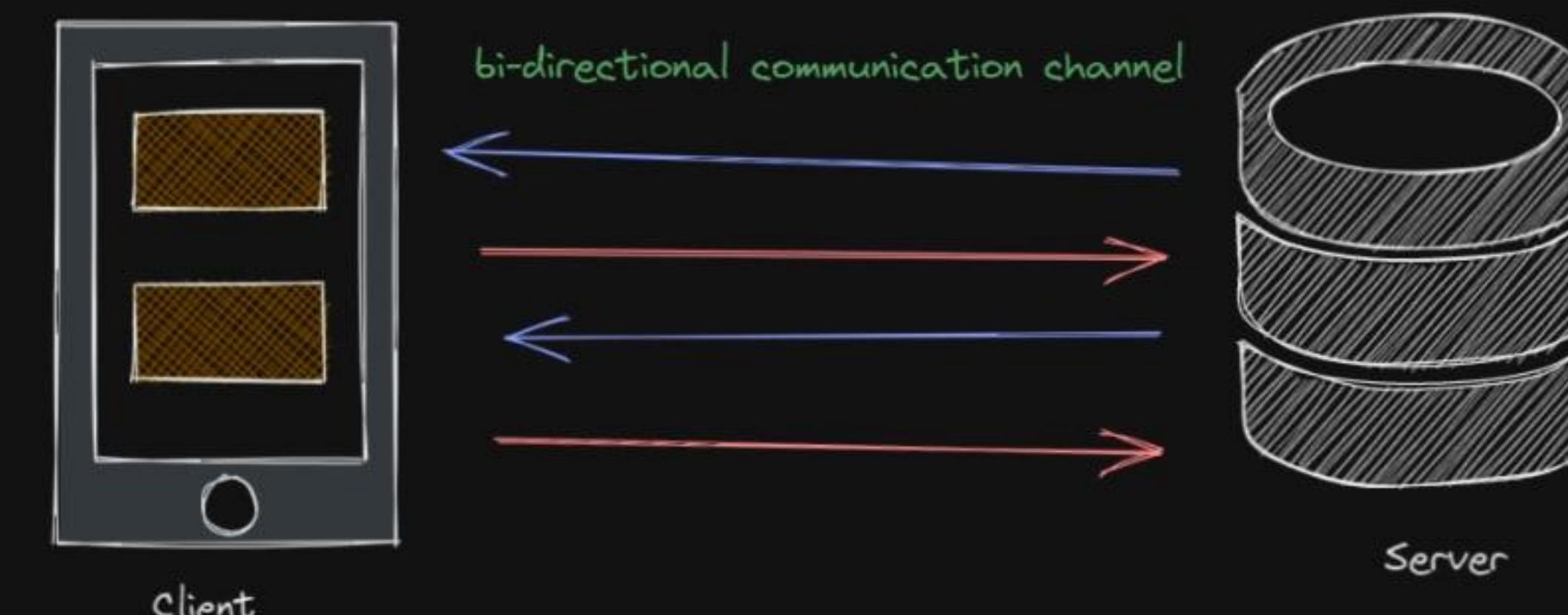
## BIDIRECTIONAL

Both client and server can send messages independently

## HOW IT WORKS

Client initiates a WebSocket handshake with the server.

Once established, messages can be sent in both directions without re-establishing the connection.



# **WHO IS USING THIS?**

Ask ChatGPT

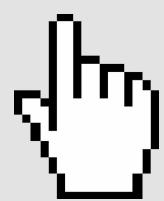
# **WHO IS USING THIS?**

Ask ChatGPT

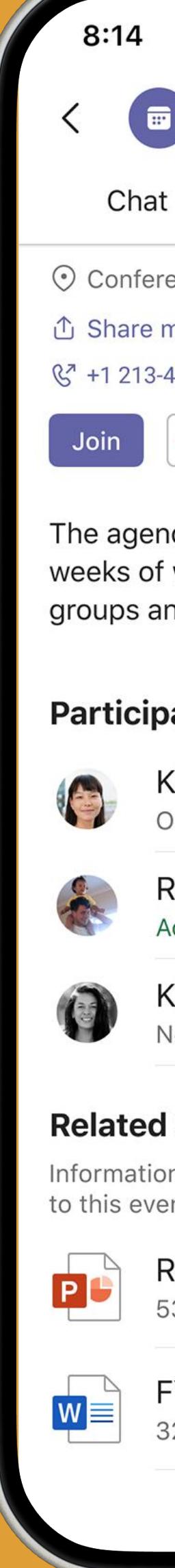
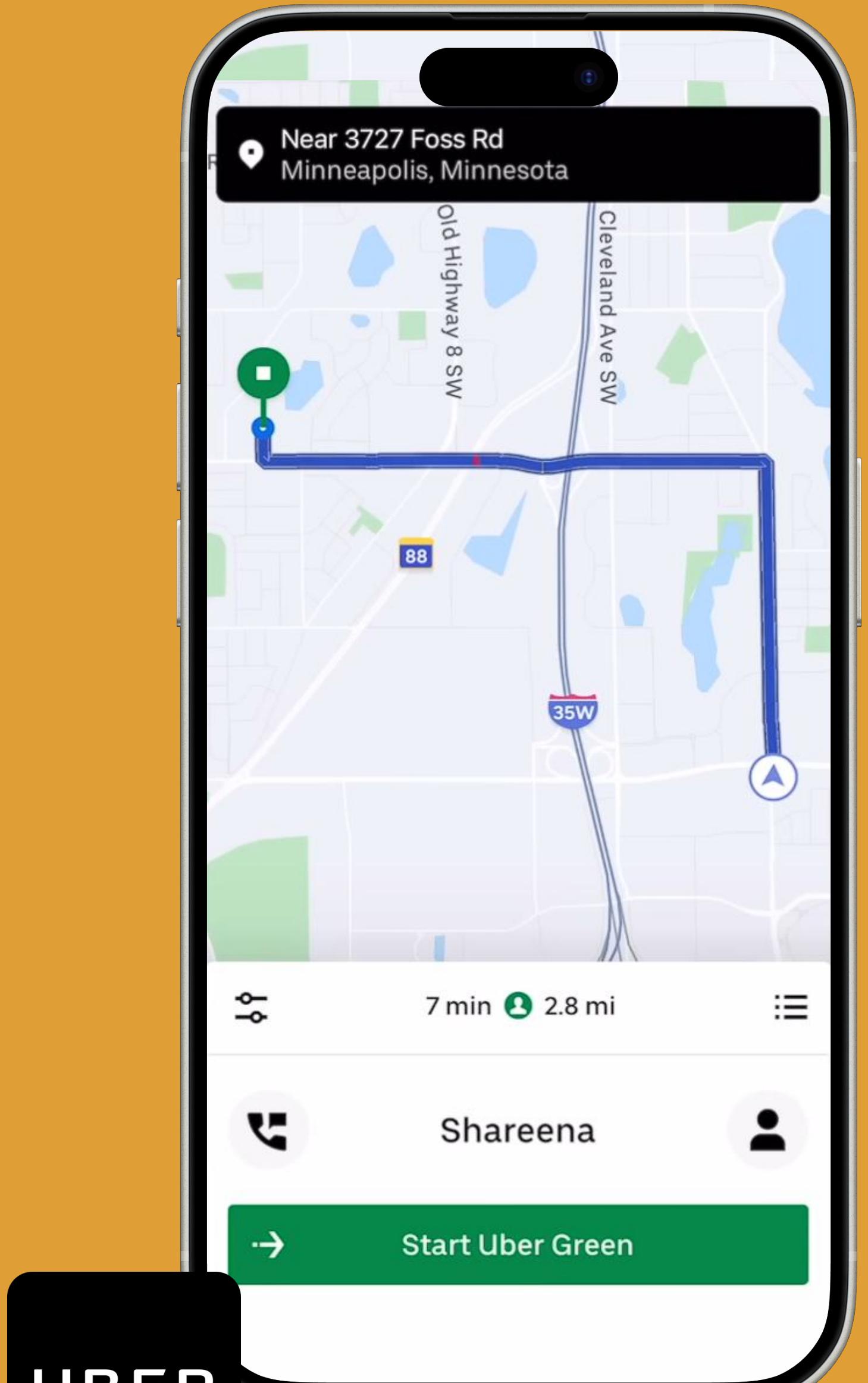


# WHO IS USING THIS?

Ask ChatGPT

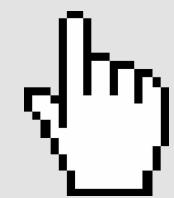


UBER



# WHO IS USING THIS?

Ask ChatGPT



@vt\_norton

The image displays three smartphones side-by-side against a yellow background. The left phone shows a map application with a route highlighted in blue. The middle phone shows a Microsoft Teams meeting invite for 'Research sync' at 'Conference Room 101'. It includes participant details for Kayo Miwa (Organizer), Ray Tanaka (Accepted), and Krystal McKinney (No response). The right phone shows a list of team members on the Microsoft Teams homepage, including Diana, Jim, Michael, Jose, Anton, and Brian.

8:14

Research sync

Chat Details More

Conference Room 101

Share meeting invite

+1 213-436-3272 (Toll) United States

Join Accepted

The agenda of this meeting is to estimate 3 weeks of work ahead, organise them into feature groups and assign owners. I suggest wh See more

Participants (3)

- Kayo Miwa Organizer
- Ray Tanaka Accepted
- Krystal McKinney No response

Related content (4) See all

- Research 8 Proposal.pptx 534 KB Shared by Sarah on 6/18
- FY 19 Research Results.docx 32 MB Shared by Laurence on 9/12

9:41

Diana Go to

Jim H Car

Michael One m

Jose Don't

Anton Do you

Brian Thank

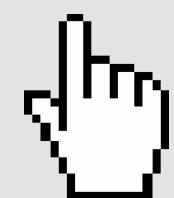
Archives

T

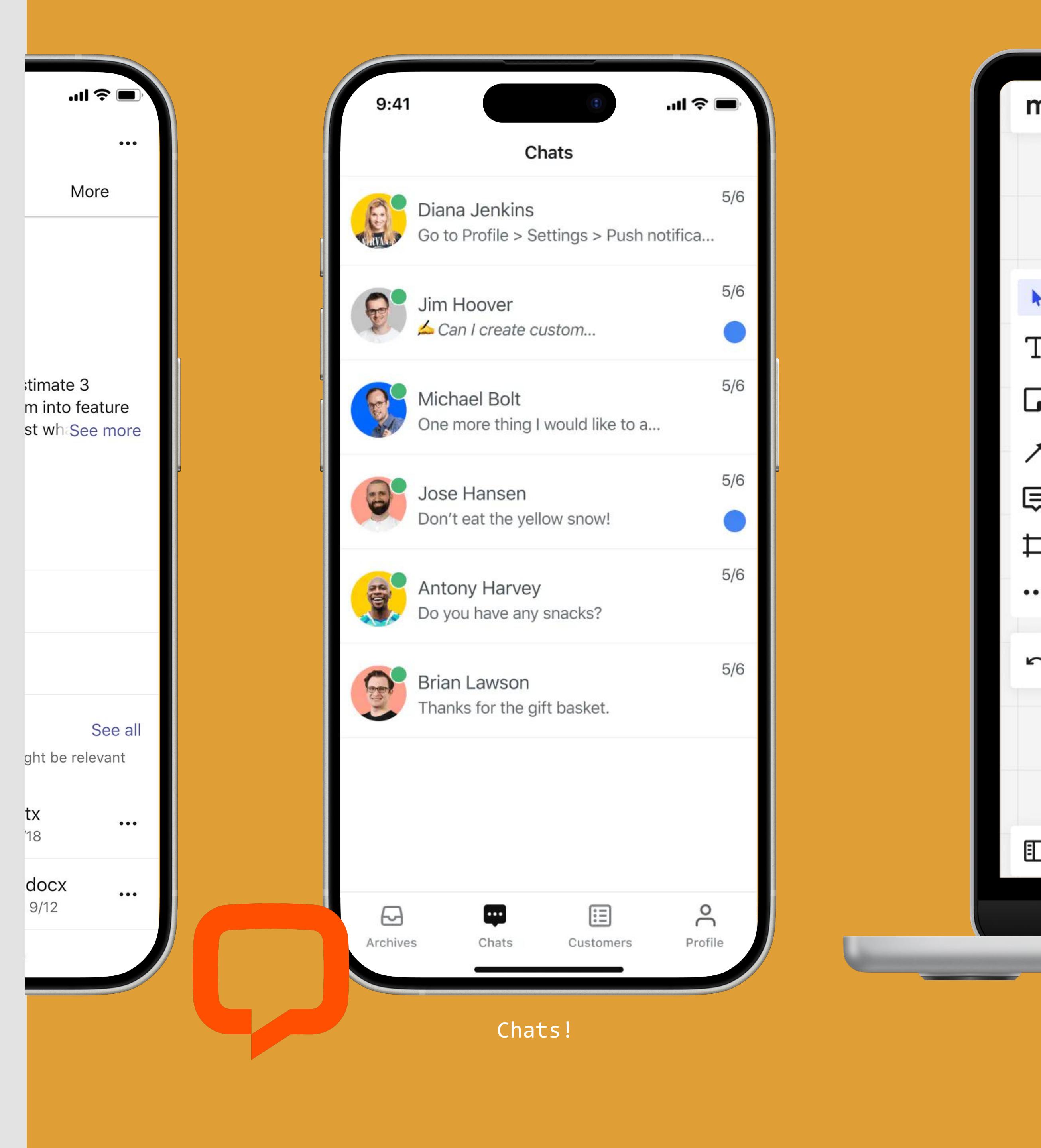
Updated info where it needs real-time

# WHO IS USING THIS?

Ask ChatGPT

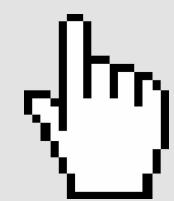


@vt\_norton



# WHO IS USING THIS?

Ask ChatGPT



The image shows a Miro team meeting interface. At the top, it says "miro Team meeting". Below that is a whiteboard titled "Ways to improve performance". The board is organized into several columns of sticky notes. Some notes have small profile pictures next to them. Annotations from users are visible, such as "Hisham" pointing to a note and "Anna" pointing to another. The interface includes a sidebar with various icons and a bottom navigation bar.

## Ways to improve performance

Annotations:

- Hisham (blue callout) points to a yellow sticky note: "Support team needs more people".
- Anna (purple callout) points to a pink sticky note: "Focus on impact".
- Bea (pink callout) points to a green sticky note: "Sales dropped significantly last quarter. We need better monitoring".

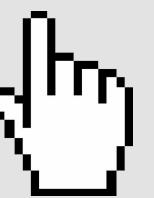
Sticky notes (approximate text):

- Yellow column:
  - "Some teams need to be more transparent"
  - "We need to optimize delivery pipeline"
  - "Last quarter was the best in terms of results"
  - "Break up projects into manageable chunks"
  - "We should revamp our marketing strategy"
- Orange column:
  - "Hiring process is too slow. We're in hypergrowth"
  - "Design team timelines are vague"
  - "We should set clear expectations to customers"
- Blue column:
  - "Kudos to operations team for improved pipeline"
  - "More clarity on what we need to achieve"
  - "We need better KPIs"
- Purple column:
  - "Other teams are not transparent at all"
  - "I don't understand what's expected from me"
  - "Prague office is disconnected from the rest"
- Green column:
  - "Hard to track performance"
  - "Sales dropped significantly last quarter. We need better monitoring"
- Pink column:
  - "Better stakeholder management"
  - "Have more informal team activities"
  - "Change our planning and grooming session"
  - "Focus on impact"

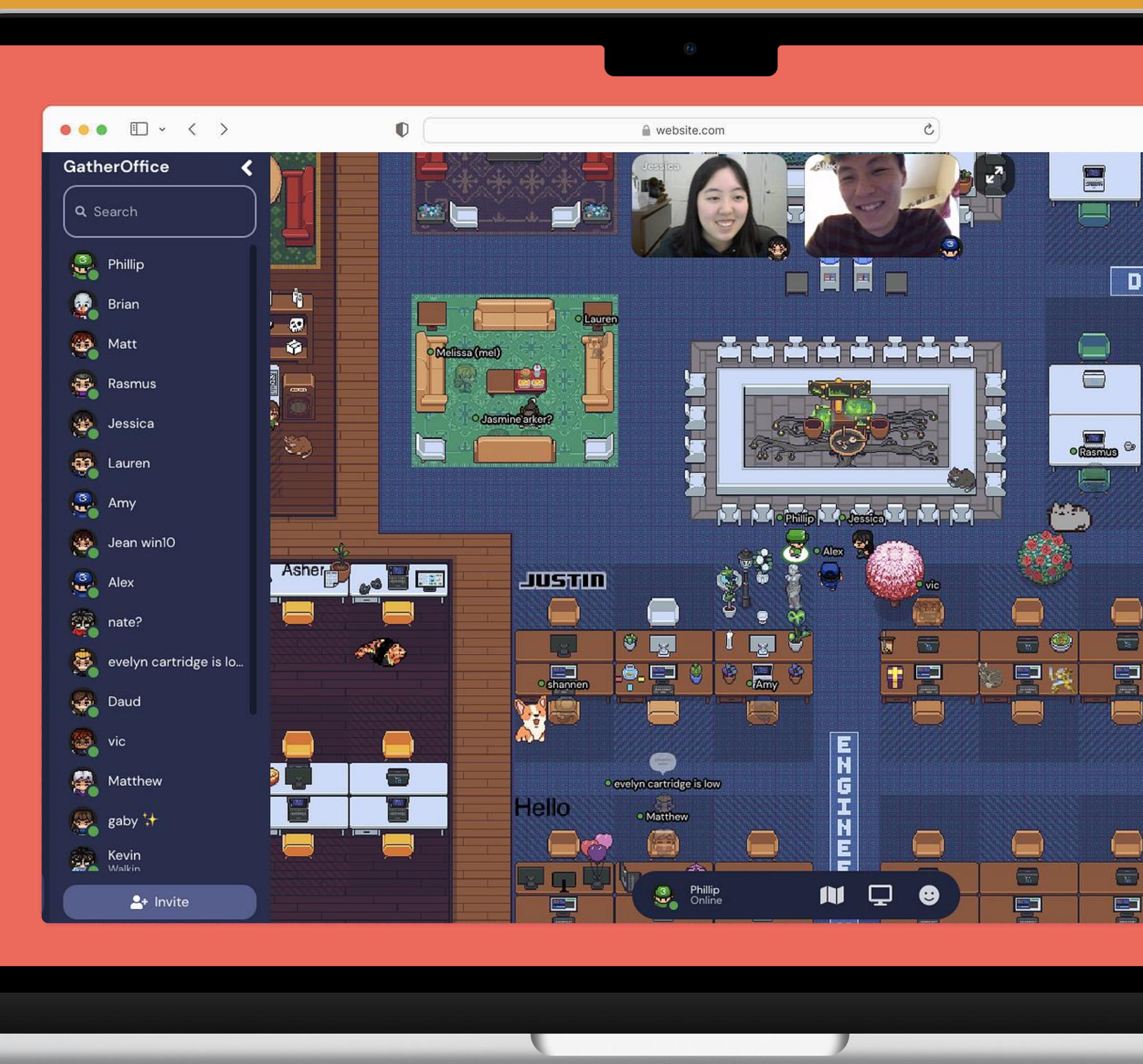
Miro logo and text: Collaborative Whiteboard

# WHO IS USING THIS?

Ask ChatGPT



Interactive work place



# RESOURCES

## Learning JavaScript Design Patterns

Addy Osmani

## How you can use WebSockets with Flutter

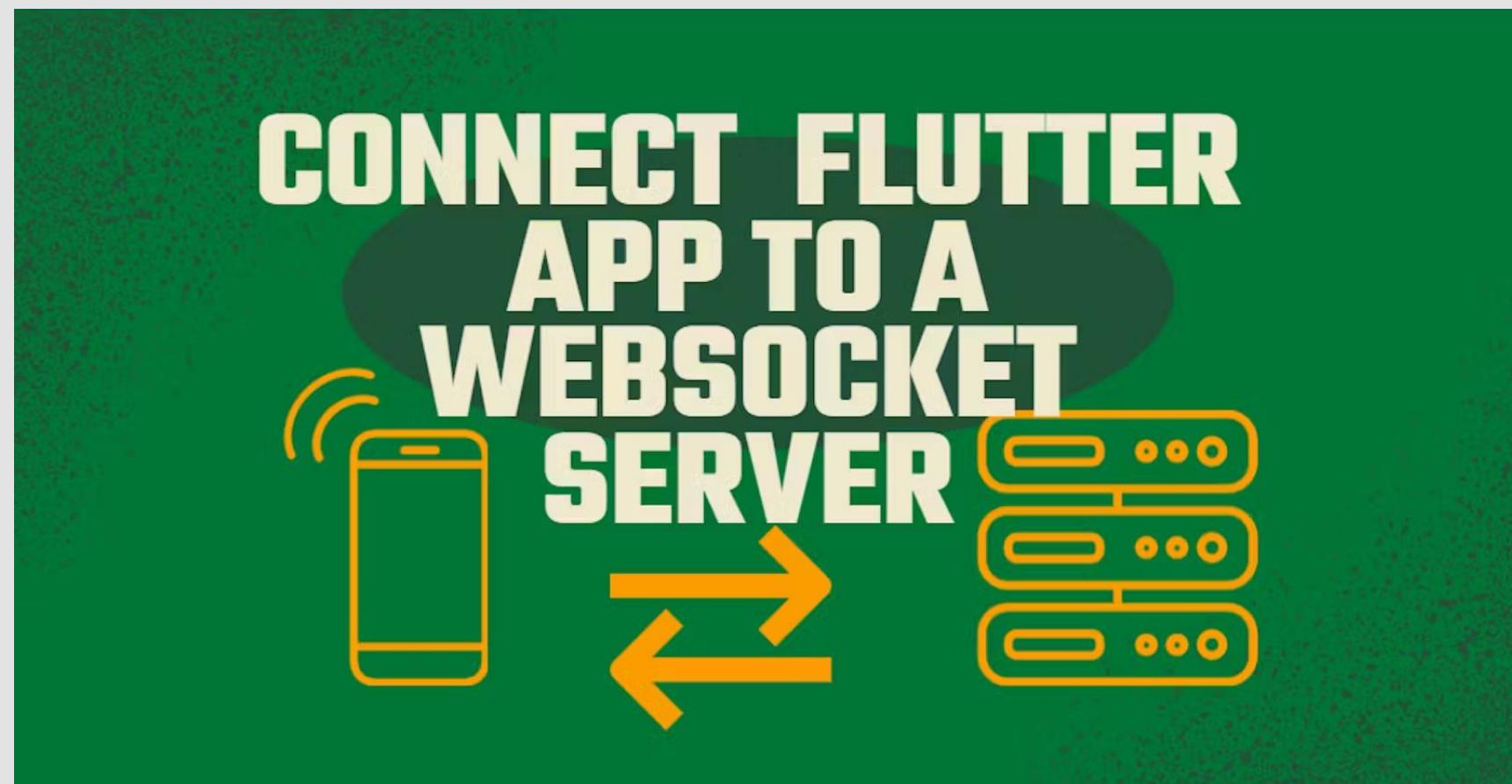
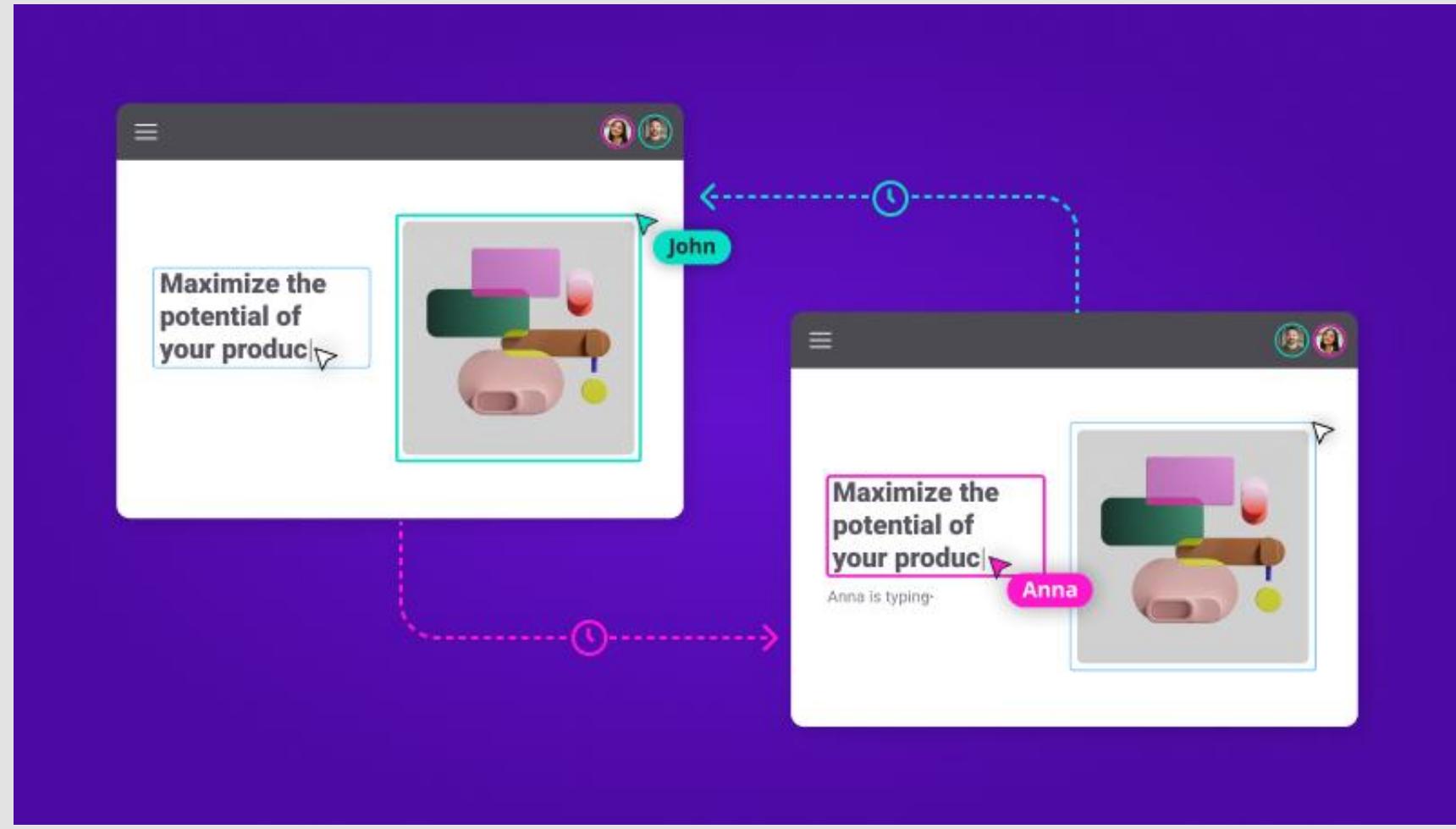
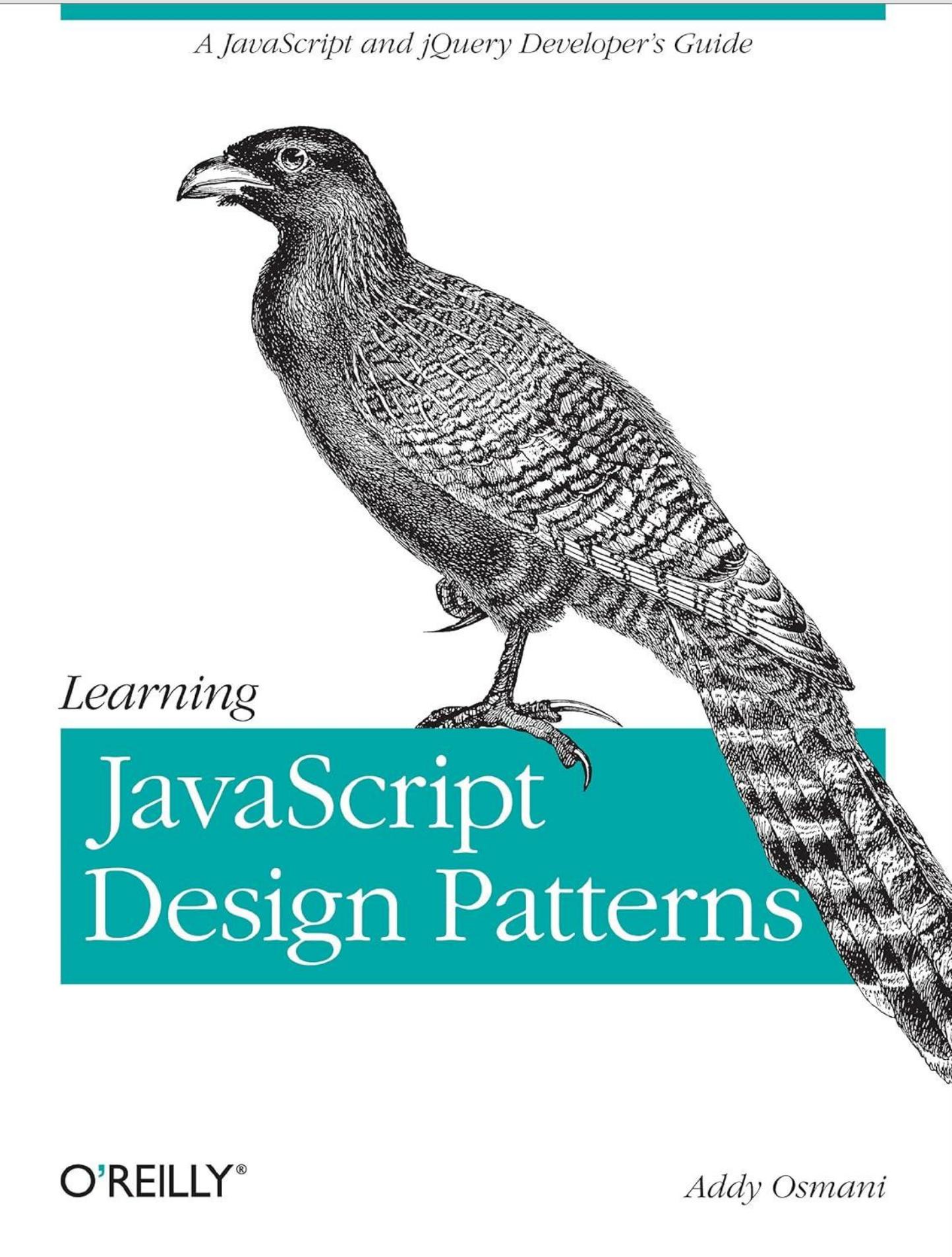
dev.to: Vibali Joshi

<https://dev.to/vibalijoshi/how-you-can-use-websockets-with-flutter-ipn>

## Understanding and implementing Event-Driven Communication in Front-End Development

dev.to: Vitor Norton

<https://dev.to/superviz/understanding-and-implementing-event-driven-communication-in-front-end-development-e75>



# THANKS ;)

LET'S BUILD SOMETHING  
GREAT

Vítor Norton  
@vt\_norton

