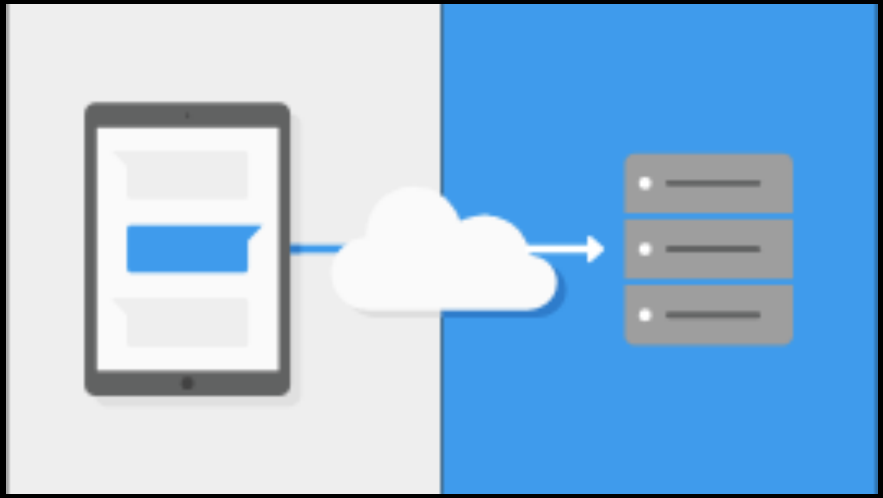
DANIEL GRAHAM PHD

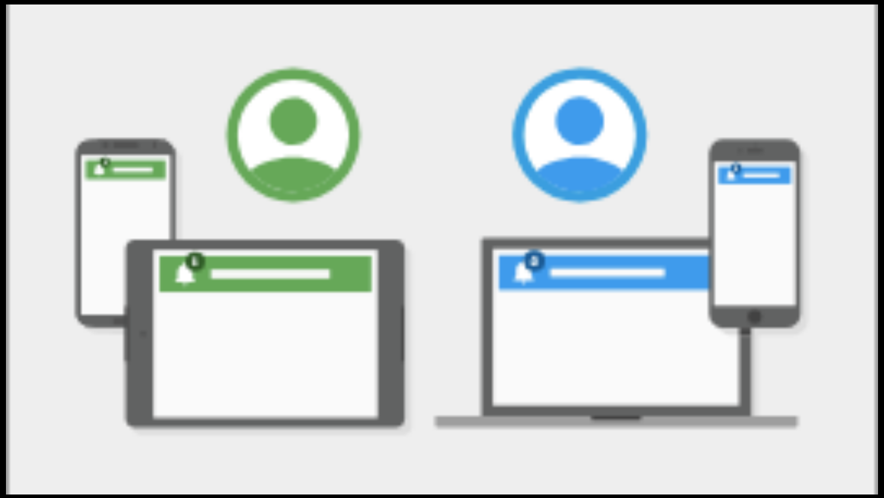
NOTIFICATIONS ARCHITECTURES

NOTIFICATION/ MESSAGING GOALS

PUSH NOTIFICATIONS TO A DEVICE (DOWN STREAM)



PUSH NOTIFICATIONS FROM DEVICE TO SERVER (UPSTREAM)



SEND TARGET MESSAGES

CHALLENGE DOWNSTREAM MESSAGING

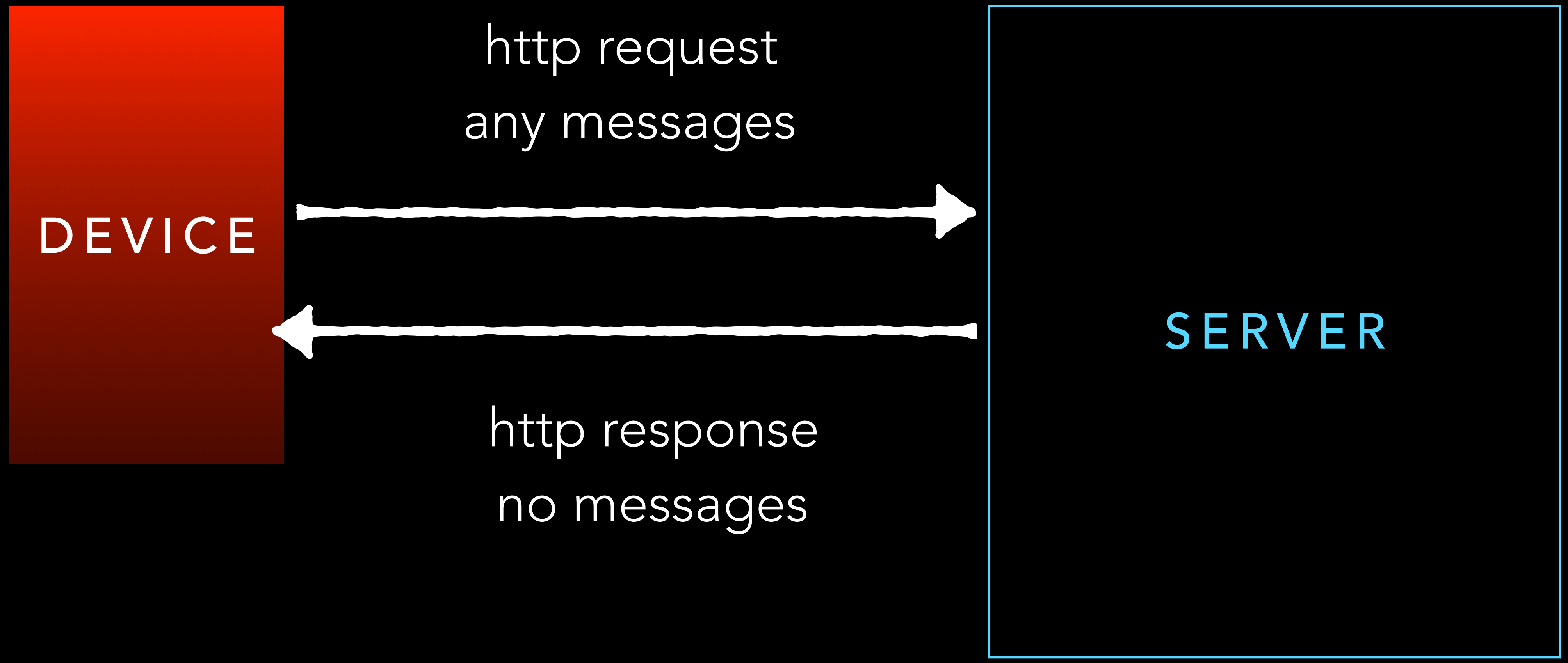
- How ensure that the message a delivered with low latency without polling? (Hundreds on milli seconds)
- How do we deliver target messages?
- How do we deliver message across platforms: Targeting users on both Android and IOS devices.

LET CONSIDER THE CASE OF DOWNSTREAM MESSAGES

HOW DO WE CHECK IF THE SERVER HAS NEED MESSAGES?

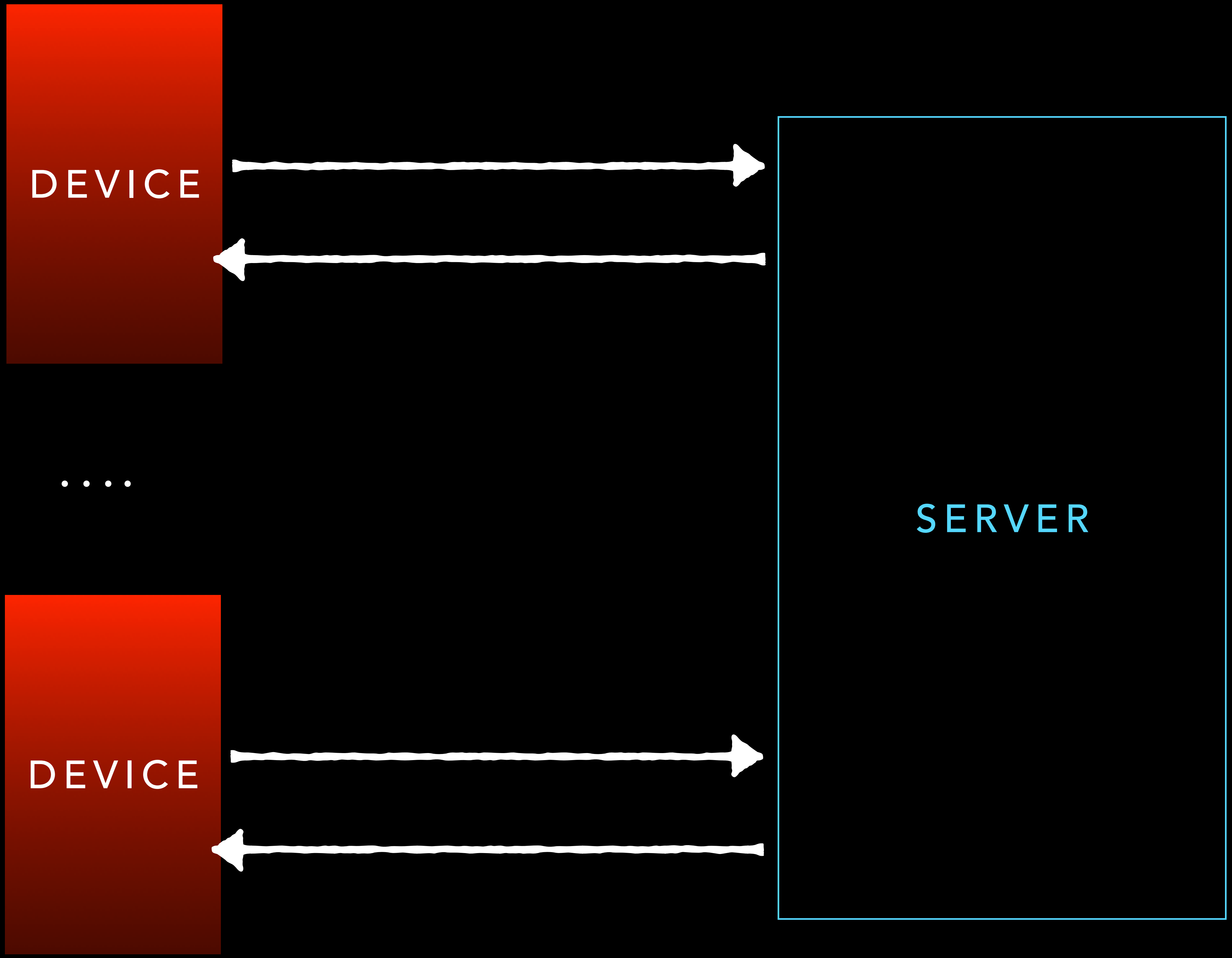
BAD CASE

THINK OF THE FETCH REQUEST IN JAVASCRIPT



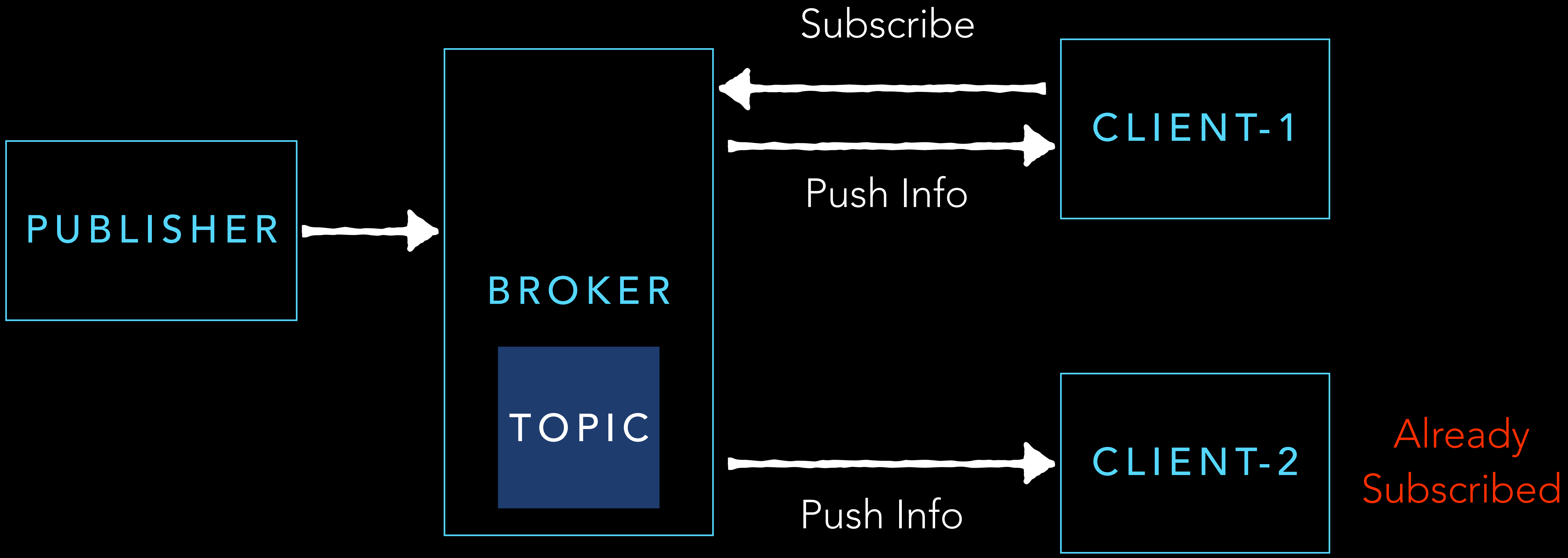
HOW OFTEN DO WE POLL THE SERVER?
EVERY 200 MILLISECONDS

BAD CASE DOES NOT SCALE (DOS)



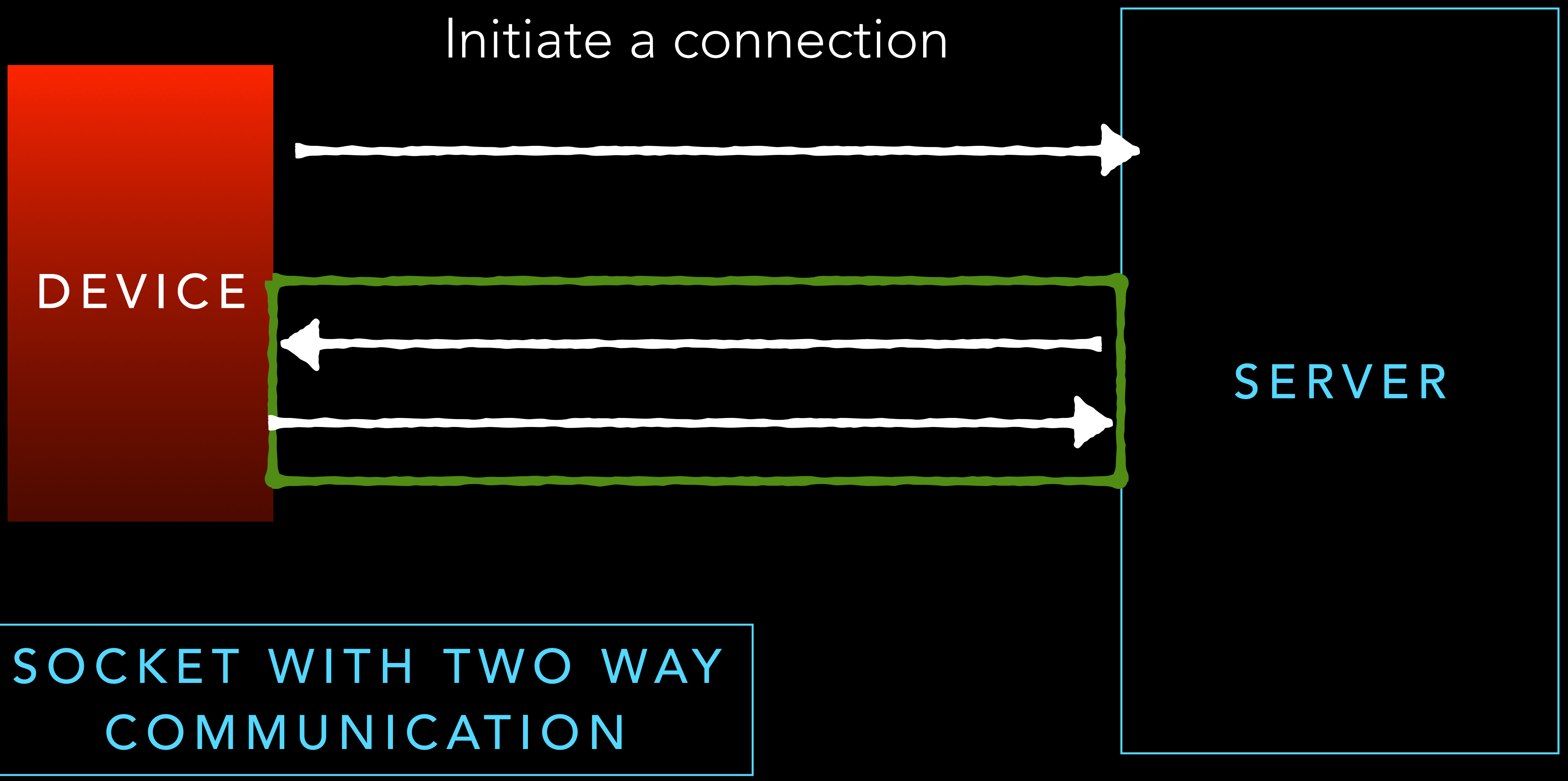
ARCHITECTURE SOLUTION

- Publish/ Subscribe architecture
  - Need way for the server to send messages to client without the client requesting



SOCKET ALLOW FOR TWO WAY COMMUNICATION

- Need way for the server to send messages to client without the client requesting



BUT SOCKETS ARE ONLY AVAILABLE AND OS LEVEL

THERE ARE WEB SOCKETS

ADD DEEP DIVE INTO SOCKETS:

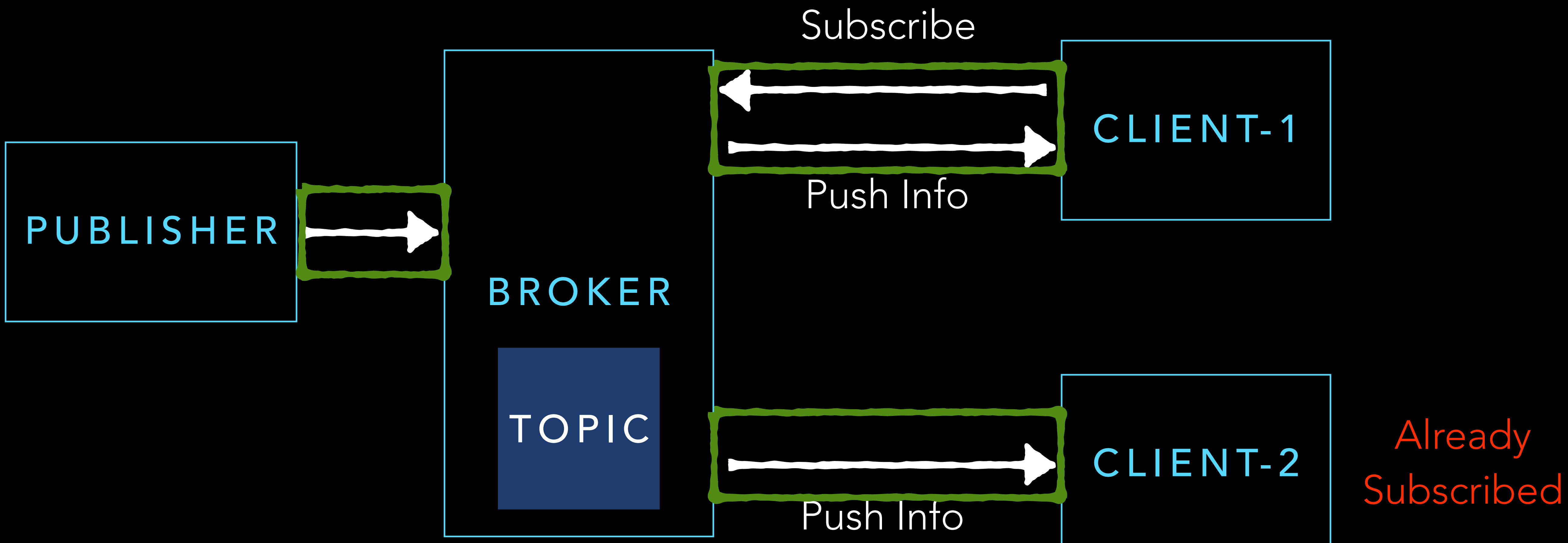
WEB SOCKETS

NO HTTP
BUT WS

why?

```
var ws = new WebSocket('ws://host.com/path');

ws.onopen = () => {
  // connection opened
  ws.send('something'); // send a message
};

ws.onmessage = (e) => {
  // a message was received
  console.log(e.data);
};

ws.onerror = (e) => {
  // an error occurred
  console.log(e.message);
};

ws.onclose = (e) => {
  // connection closed
  console.log(e.code, e.reason);
};
```

PUBLISHER SUBSCRIBER WITH SOCKETS



MAIN MESSAGING/ NOTIFICATION SERVICES

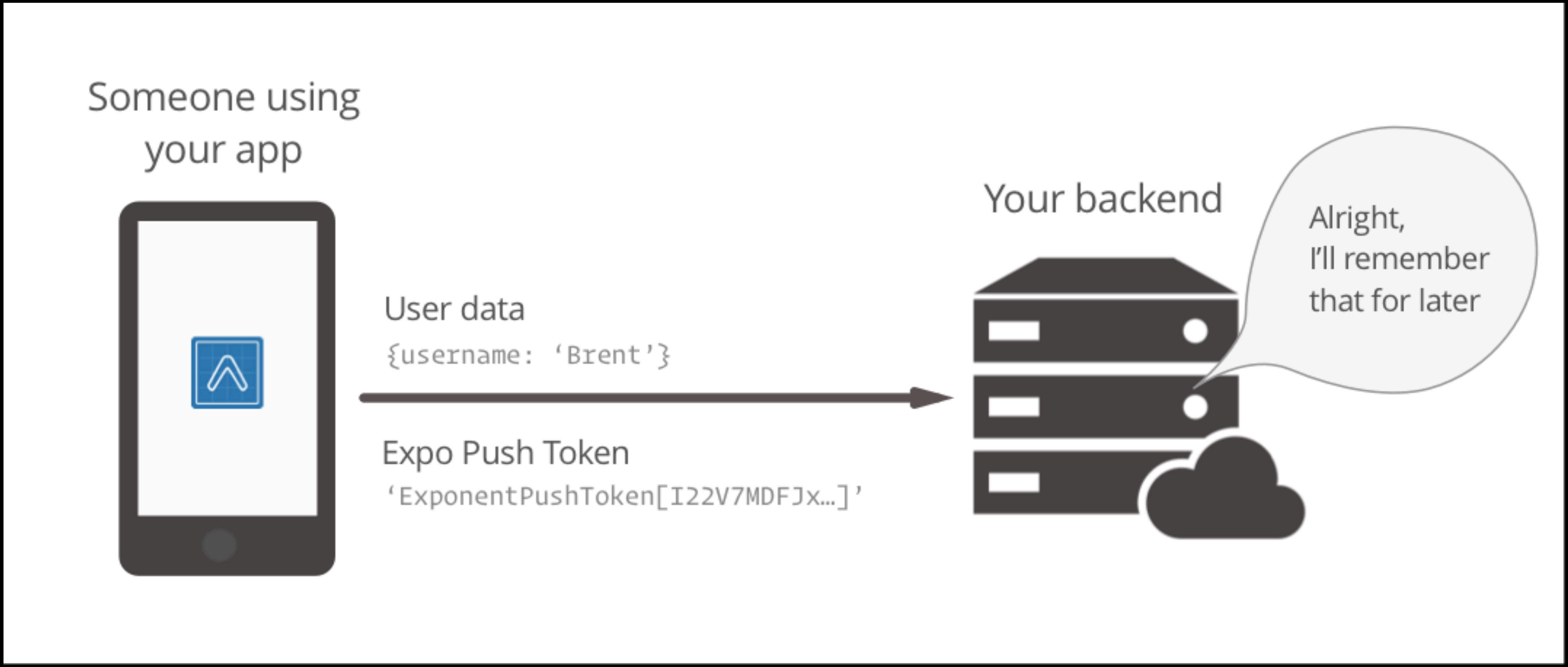
Google Cloud Messaging Service (Depreciated)

Firebase Cloud Messaging Service

Apple Push Notification Service

Expo Push Notification Service ←

DEEP DIVE EXPO NOTIFICATION ARCHITECTURE



SAVING USERS PUSH TOKEN

```
import { Permissions, Notifications } from 'expo';

export default async function registerForPushNotificationsAsync() {
  const { status: existingStatus } = await Permissions.getAsync(
    Permissions.NOTIFICATIONS
  );
  let finalStatus = existingStatus;
  if (existingStatus !== 'granted') {
    const { status } = await Permissions.askAsync(Permissions.NOTIFICATIONS);
    finalStatus = status;
  }

  if (finalStatus !== 'granted') {
    return;
  }

  let token = await Notifications.getExpoPushTokenAsync();
  return token
  }
```

STILL NOT GRANTED RETURN

ADD CODE FOR HANDLING NOTIFICATION

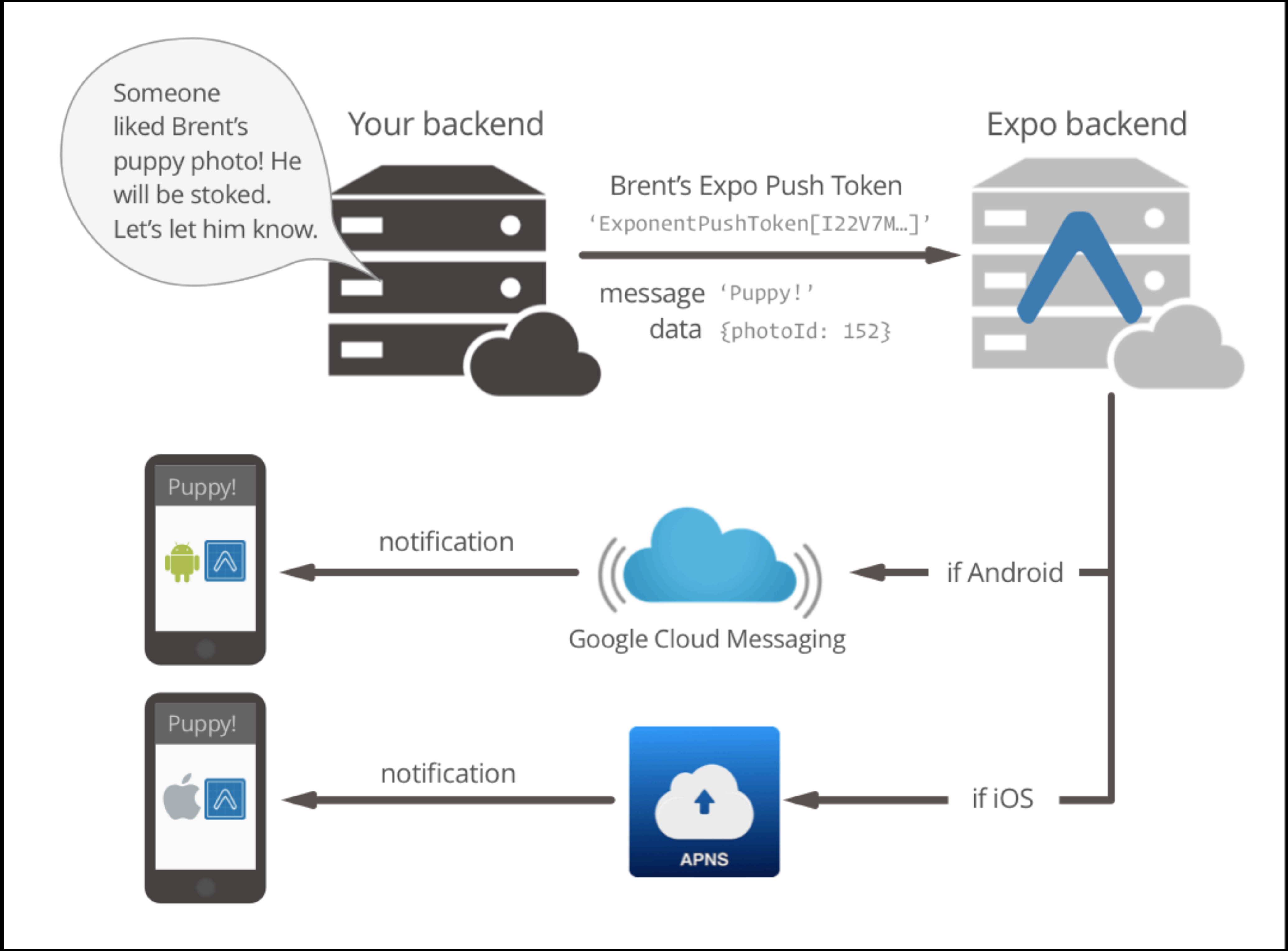
- Handling Notification went the application is open

```
    componentDidMount(){
    this._notificationSubscription = Notifications.addListener(this._handleNotification);
    }

    _handleNotification = (notification) => {
      this.setState({notification: notification})
      console.log("got Notification")

    };
```

Modify the home screen to look

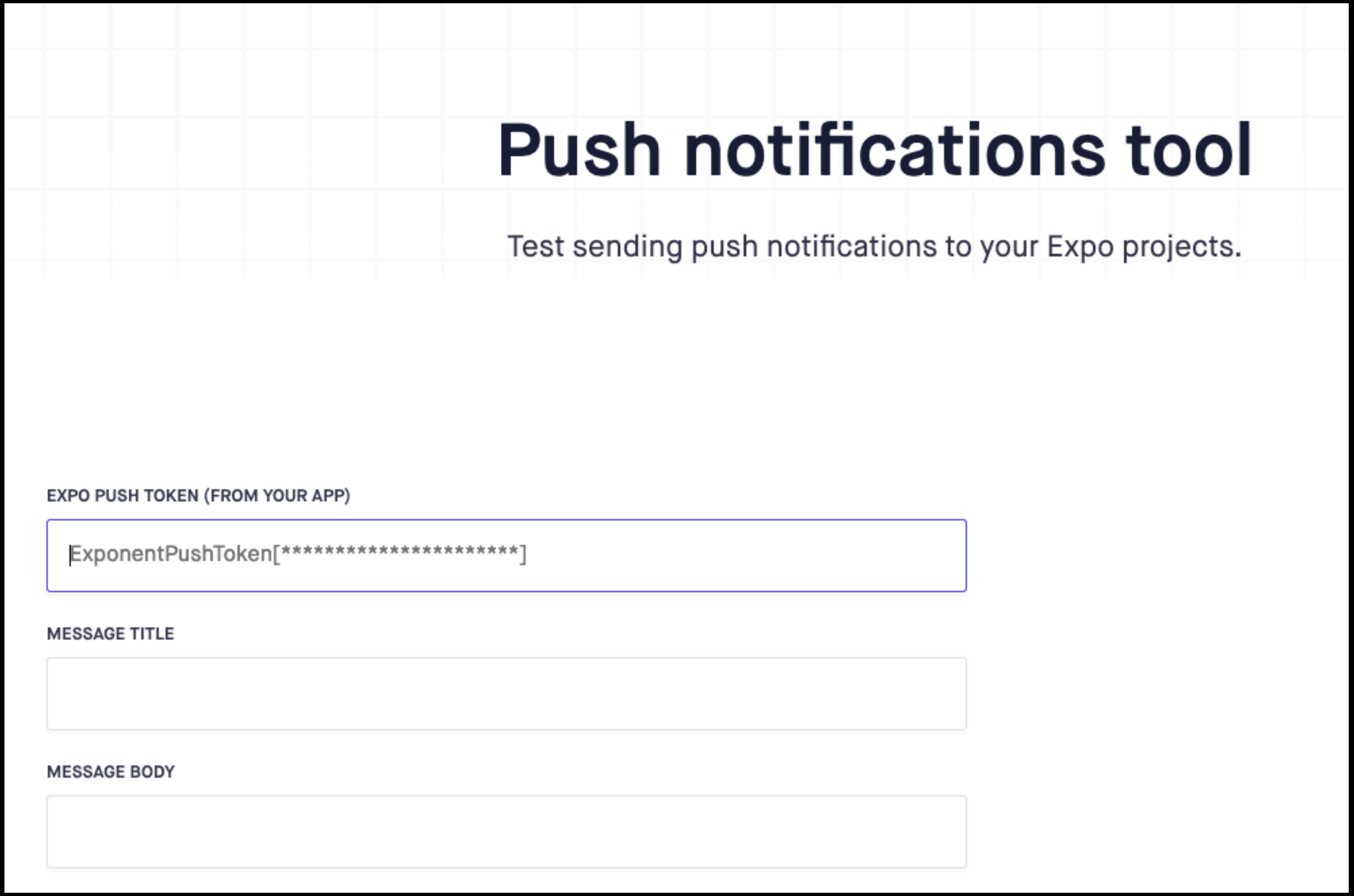
Handling notifications while app is open

DEEP DIVE EXPO NOTIFICATION ARCHITECTURE



DEMO TIME

https://expo.io/dashboard/notifications



Push notifications tool

Test sending push notifications to your Expo projects.

EXPO PUSH TOKEN (FROM YOUR APP)

ExponentPushToken[**********************]

MESSAGE TITLE

MESSAGE BODY

REFERENCES

- https://docs.expo.io/versions/latest/guides/push-notifications/