EVENTS, EVENT EMITTERS, HTTP & LONG POLLING

Building real-time software

```
class EventEmitter {









}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }











}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }

  on (eventName, callback) {


  }

  emit (eventName, ...payload) {



  }
}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }

  on (eventName, callback) {
    this.callbacks[eventName] = this.callbacks[eventName] || []

  }

  emit (eventName, ...payload) {



  }
}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }

  on (eventName, callback) {
    this.callbacks[eventName] = this.callbacks[eventName] || []
    this.callbacks[eventName].push(callback)
  }

  emit (eventName, ...payload) {



  }
}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }

  on (eventName, callback) {
    this.callbacks[eventName] = this.callbacks[eventName] || []
    this.callbacks[eventName].push(callback)
  }

  emit (eventName, ...payload) {
    this.callbacks[eventName].forEach(callback => {

    })
  }
}
```

```
class EventEmitter {
  constructor () {
    this.callbacks = {}
  }

  on (eventName, callback) {
    this.callbacks[eventName] = this.callbacks[eventName] || []
    this.callbacks[eventName].push(callback)
  }

  emit (eventName, ...payload) {
    this.callbacks[eventName].forEach(callback => {
      callback(...payload)
    })
  }
}
```

```
const ee = new EventEmitter()
const ee = new EventEmitter()
```

```
constructor () {
  this.callbacks = {}
}
```

```
const ee = new EventEmitter()
```

```
{
  callbacks: {}
}
```

```
constructor () {
  this.callbacks = {}
}
```

```
const ee = new EventEmitter()


ee.on('tweet', (m) => console.log(m))
```

```
{
  callbacks: {}
}
```

```
const ee = new EventEmitter()
```

```
ee.on('tweet', (m) => console.log(m))
```

```
{
  callbacks: {}
}
```

```
on (eventName, callback) {
  this.callbacks[eventName] = this.callbacks[eventName] || []
  this.callbacks[eventName].push(callback)
}
```

```
const ee = new EventEmitter()
```

```
ee.on('tweet', (m) => console.log(m))
```

```
{
  callbacks: {
    'tweet': [(m) => console.log(m)]
  }
}
```

```
on (eventName, callback) {
  this.callbacks[eventName] = this.callbacks[eventName] || []
  this.callbacks[eventName].push(callback)
}
```

```
const ee = new EventEmitter()

ee.on('tweet', (m) => console.log(m))

ee.emit('tweet', {message: 'Hello'})
```

```
{
  callbacks: {
    'tweet': [(m) => console.log(m)]
  }
}
```

```
const ee = new EventEmitter()

ee.on('tweet', (m) => console.log(m))

ee.emit('tweet', {message: 'Hello'})
```

```
{
  callbacks: {
    'tweet': [(m) => console.log(m)]
  }
}
```

```
emit (eventName, ...payload) {
  this.callbacks[eventName].forEach(callback => {
    callback(...payload)
  })
}
```

```
const ee = new EventEmitter()


ee.on('tweet', (m) => console.log(m))

ee.emit('tweet', {message: 'Hello'})
```

```
{
  callbacks: {
    'tweet': [(m) => console.log(m)]
  }
}
```

```
emit (eventName, ...payload) {
  this.callbacks[eventName].forEach(callback => {
    callback(...payload)
  })
}
```

SOME TERMINOLOGY YOU MAY HAVE HEARD…THAT I HATE

- **"Event Emitter"**
  - Sounds like events are somehow shooting out of it…they're not
  - Better name: function storage
- **"Event Listener"**
  - Sounds like there is something *actively* listening…waiting…watching…
  - There's not - functions are just sitting in the array, completely passive

EVENT EMITTERS

- **Objects that store callbacks associated with a certain label (“event”)**
- **Invokes all callbacks with the specified label when asked**
- **An instance of the “observer/observable” a.k.a “pub/sub” pattern**
- **Feels at-home in an *event*-driven environment**
    - Clicks, changes, submits in the DOM
    - Receiving requests
    - “Lifecycles” (in both Sequelize and React)
    - Basically…**everything we do**

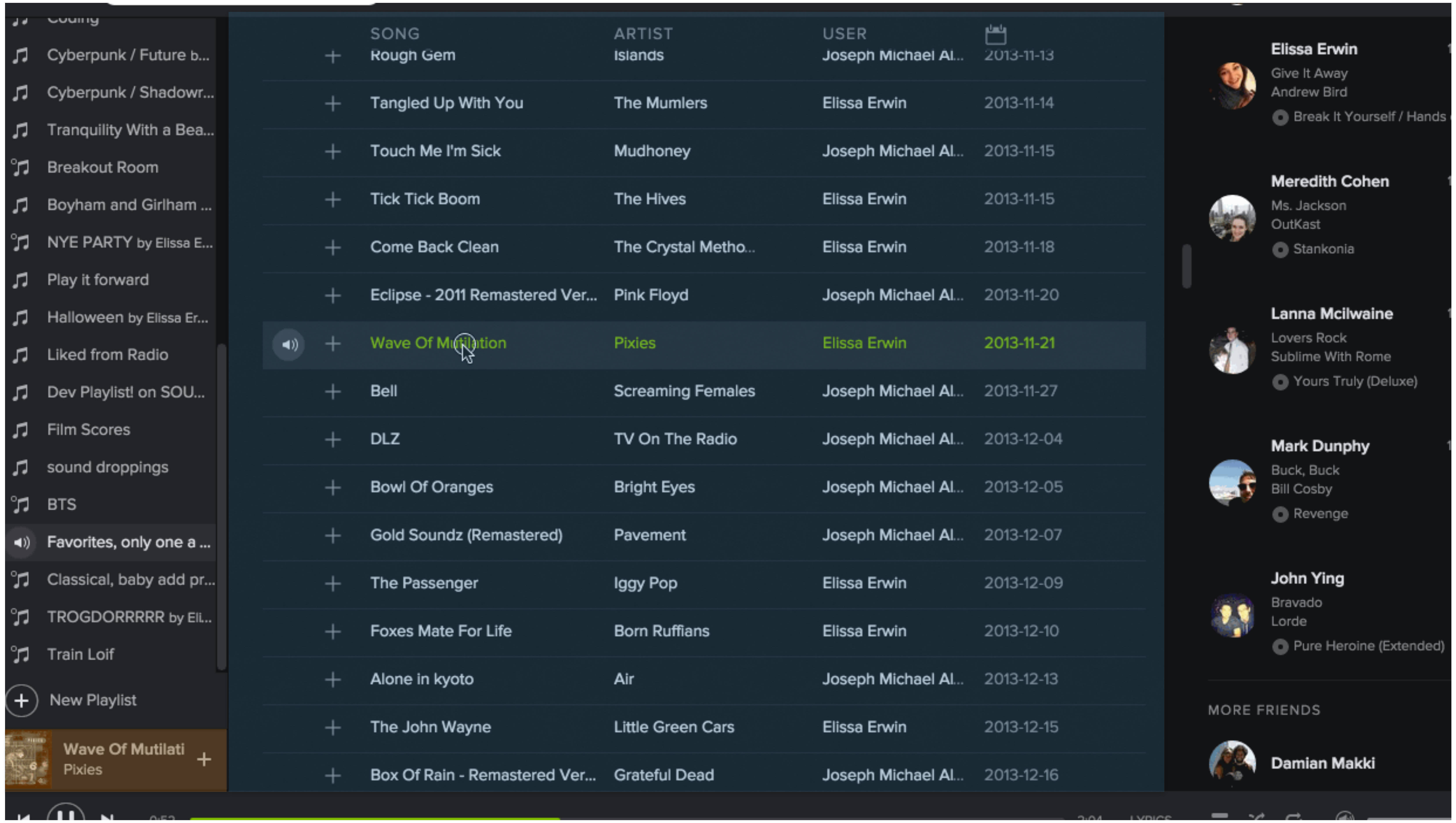
PRACTICAL USES

- Connect two decoupled parts of an application

```
const currentTrack = new EventEmitter()
```

```
currentTrack.emit(‘changeTrack', newTrack)
```

```
currentTrack.on('changeTrack', (newTrack) => {
  displayNewTrack(newTrack)
})
```



- Cyberpunk / Future b...
- Cyberpunk / Shadowr...
- Tranquility With a Bea...
- Breakout Room
- Boyham and Girlham ...
- NYE PARTY by Elissa E...
- Play it forward
- Halloween by Elissa Er...
- Liked from Radio
- Dev Playlist! on SOU...
- Film Scores
- sound droppings
- BTS
- Favorites, only one a ...
- Classical, baby add pr...
- TROGDORRRRR by Eli...
- Train Loif
- New Playlist

Wave Of Mutilati
Pixies

SONG	ARTIST	USER	
Rough Gem	Islands	Joseph Michael Al...	2013-11-13
Tangled Up With You	The Mumlers	Elissa Erwin	2013-11-14
Touch Me I'm Sick	Mudhoney	Joseph Michael Al...	2013-11-15
Tick Tick Boom	The Hives	Elissa Erwin	2013-11-15
Come Back Clean	The Crystal Metho...	Elissa Erwin	2013-11-18
Eclipse - 2011 Remastered Ver...	Pink Floyd	Joseph Michael Al...	2013-11-20
Wave Of Mutilation	Pixies	Elissa Erwin	2013-11-21
Bell	Screaming Females	Joseph Michael Al...	2013-11-27
DLZ	TV On The Radio	Joseph Michael Al...	2013-12-04
Bowl Of Oranges	Bright Eyes	Joseph Michael Al...	2013-12-05
Gold Soundz (Remastered)	Pavement	Joseph Michael Al...	2013-12-07
The Passenger	Iggy Pop	Elissa Erwin	2013-12-09
Foxes Mate For Life	Born Ruffians	Elissa Erwin	2013-12-10
Alone in kyoto	Air	Joseph Michael Al...	2013-12-13
The John Wayne	Little Green Cars	Elissa Erwin	2013-12-15
Box Of Rain - Remastered Ver...	Grateful Dead	Joseph Michael Al...	2013-12-16

Elissa Erwin
Give It Away
Andrew Bird
Break It Yourself / Hands

Meredith Cohen
Ms. Jackson
OutKast
Stankonia

Lanna Mcilwaine
Lovers Rock
Sublime With Rome
Yours Truly (Deluxe)

Mark Dunphy
Buck, Buck
Bill Cosby
Revenge

John Ying
Bravado
Lorde
Pure Heroine (Extended)

MORE FRIENDS

Damian Makki

PRACTICAL USES

- **Represent multiple asynchronous events on a single entity.**

```
const upload = uploadFile()

upload.on('error', (err) => {
  handleError(err)
})

upload.on('progress', (percentage) => {
  updateProgressBar(percentage)
})

upload.on('complete', () => {
  tellUserThatUploadCompleted()
})
```

ALL OVER NODE

- server.on(‘request’)
- request.on(‘data’) / request.on(‘end’)
- process.stdin.on(‘data’)
- db.on(‘connection’)
- Streams

HTTP, PART 2

Sequels are always worse than the original

WHAT WE KNOW ABOUT HTTP

- **A client makes a “request” to a server**
- **Server receives this “request” and generates a “response”**
- **One request, one response: them’s the rules**
- **Requests can include a body (payload)**
- **Responses can include a body (payload)**

The New York Times

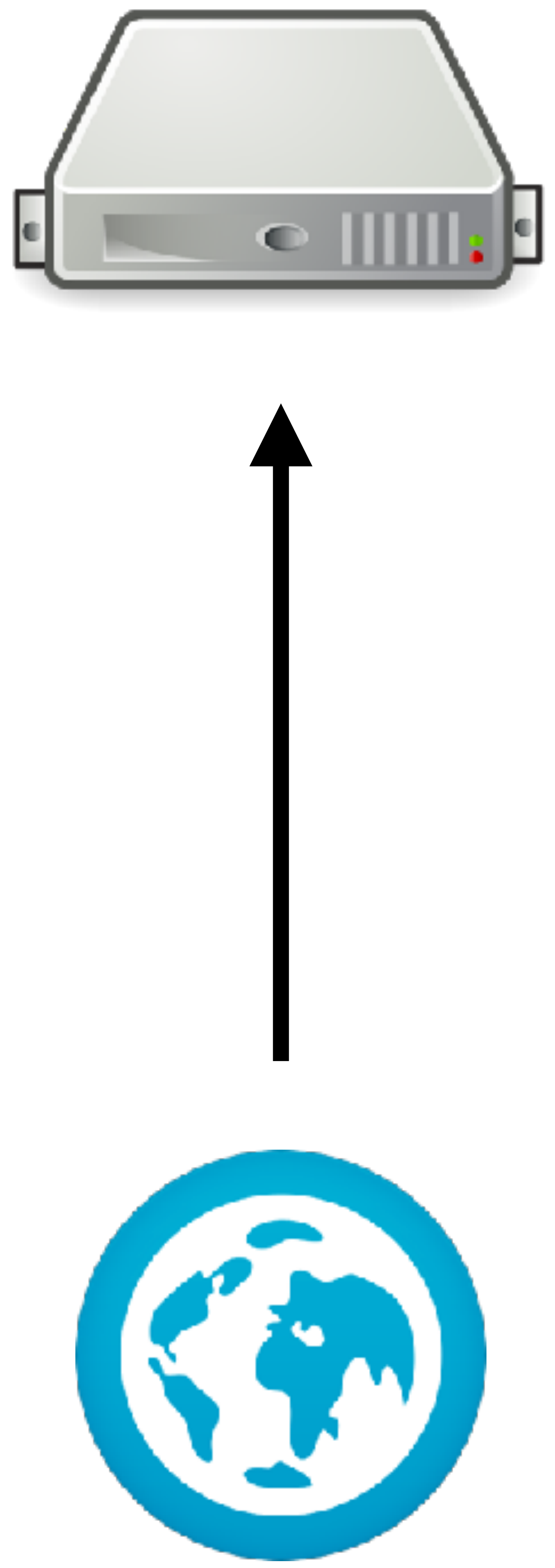
2014

FIFA WORLD CUP
Brasil

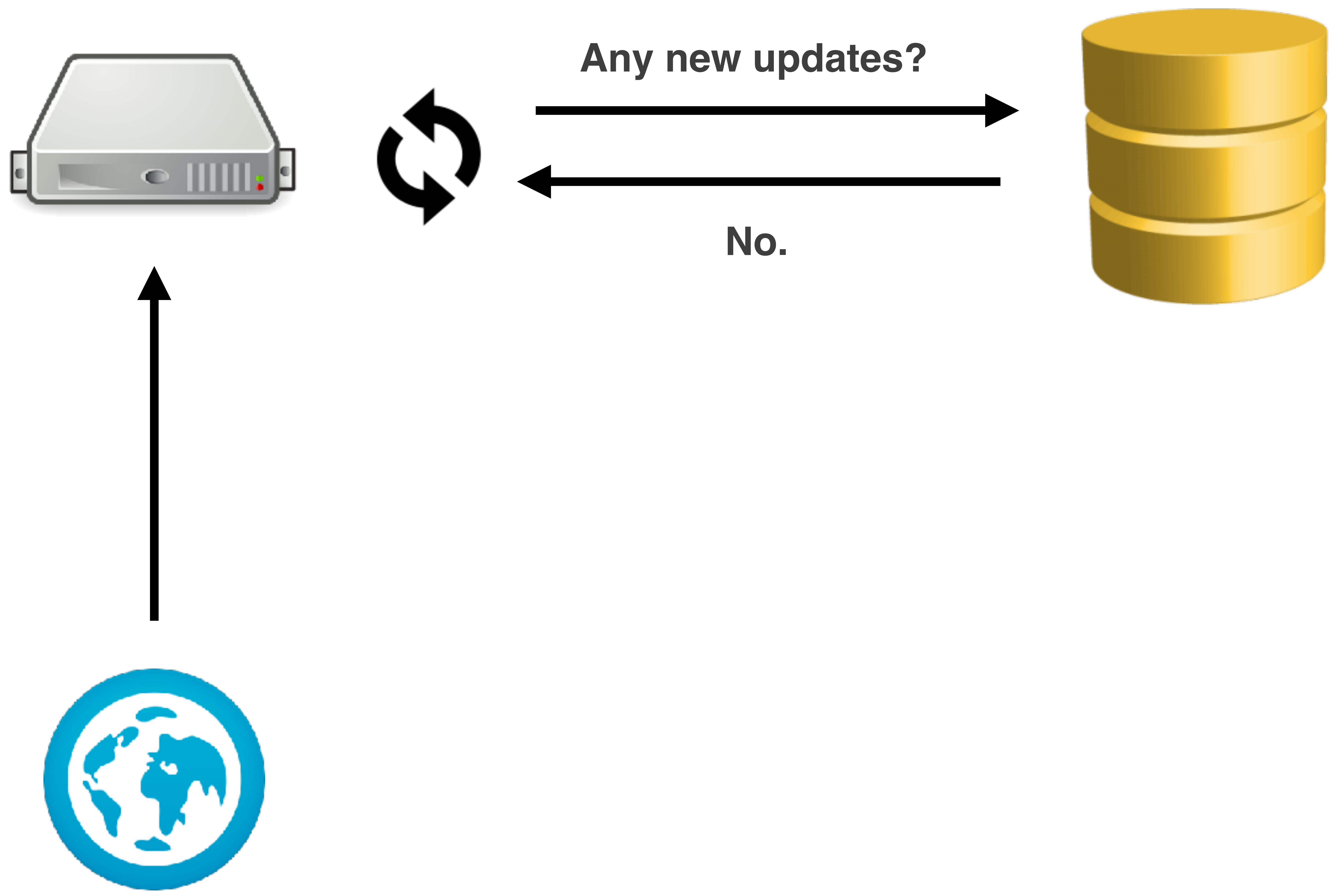
LIVE WORLD CUP COVERAGE

- A user visits a web page
- This web page has a live updating list of game coverage (“events”) provided by New York Times commentator (“Brazil receives yellow card”/“Germany scores goal”)
- When the event line is submitted by the commentator, it should immediately display to the user

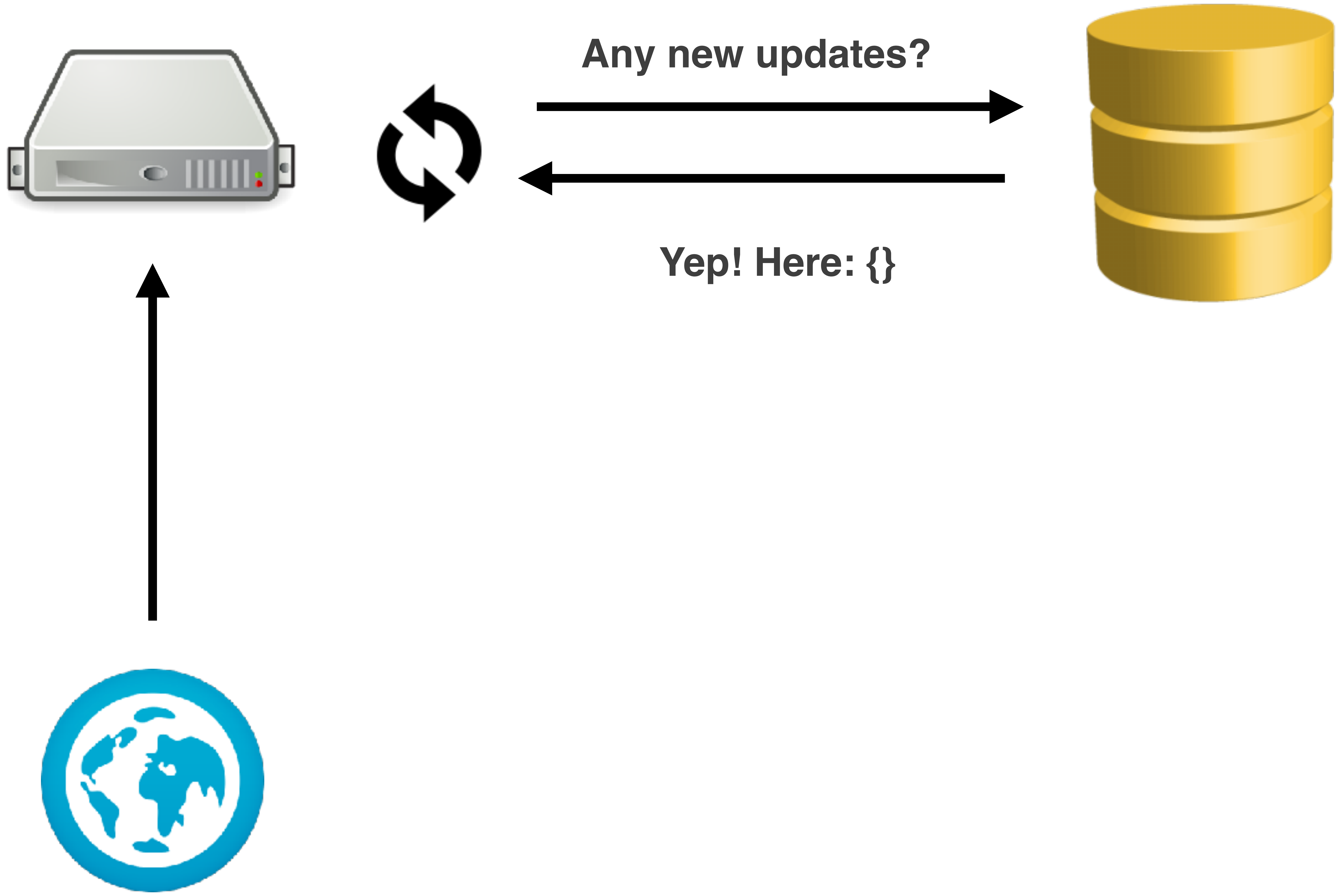
HTTP LONG POLLING



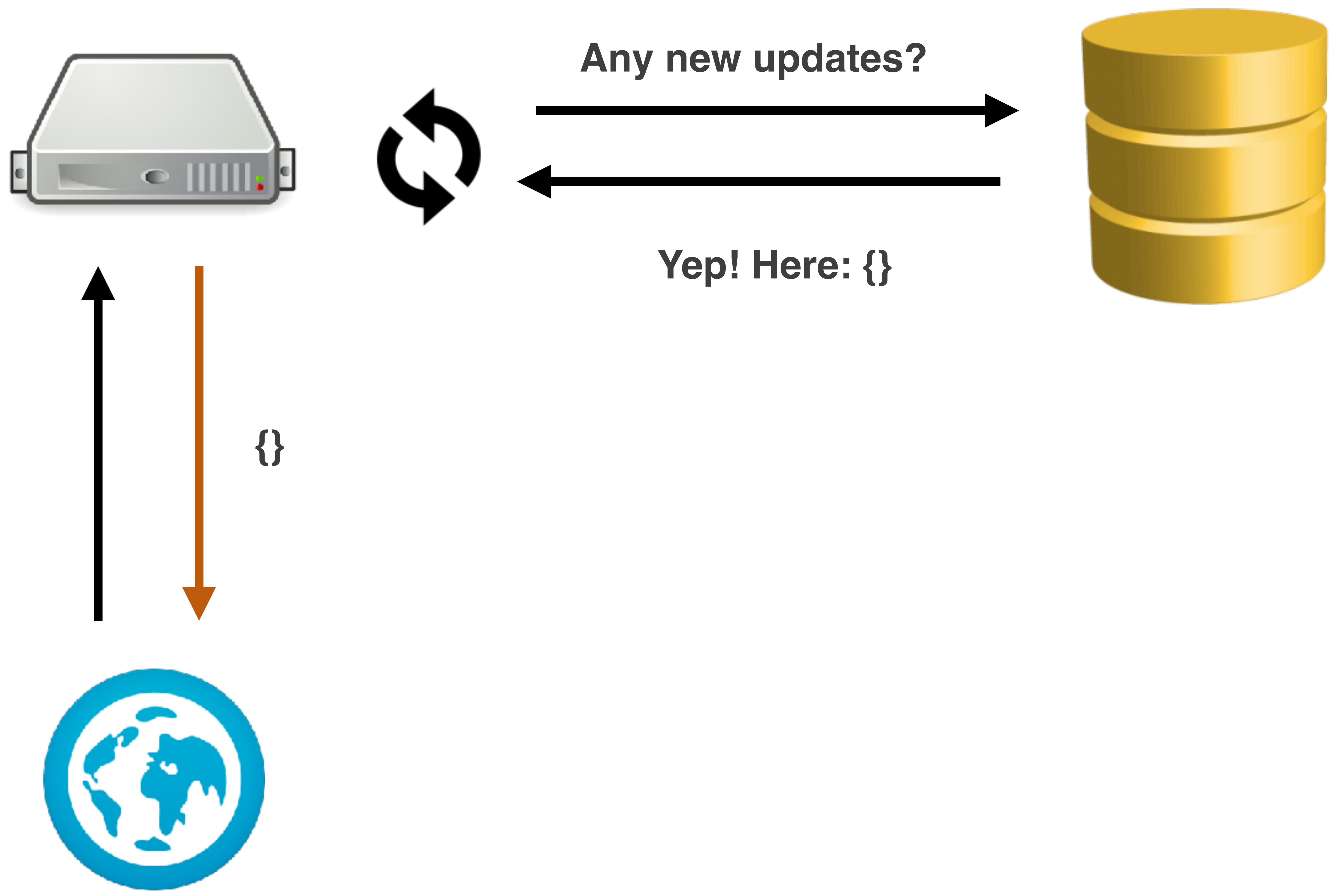
HTTP LONG POLLING



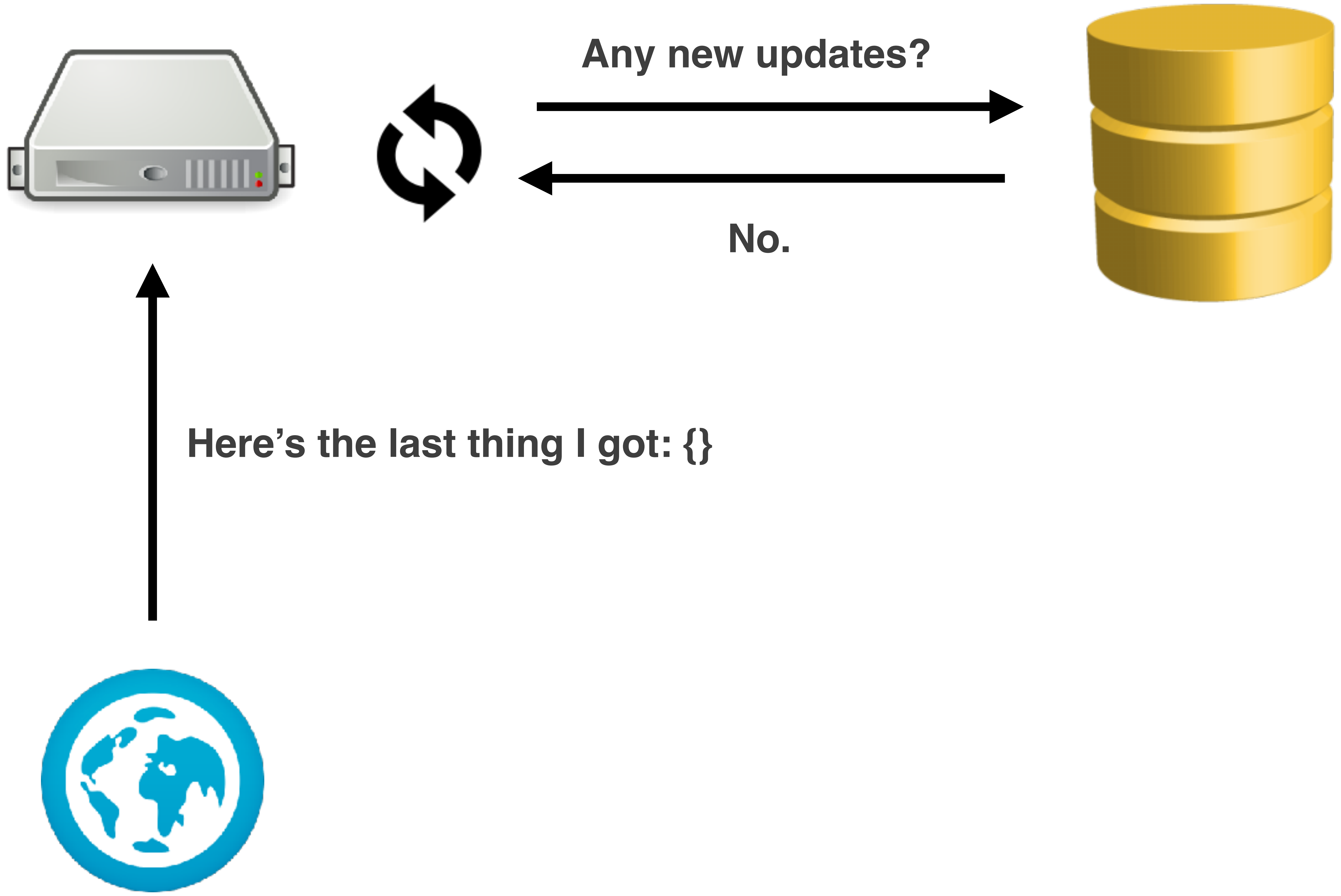
HTTP LONG POLLING



HTTP LONG POLLING



HTTP LONG POLLING



HTTP IS A REQUEST/RESPONSE PROTOCOL

- Clients must send a *request* before the server can issue a *response*
- There is no way for the server to *push* data to the client without an outstanding request
- No live updates without long polling 😢