A Practical Example

Scenario:

- A large batch of operations that will each take time to complete
- We want to run them concurrently

- BUT -

- There are too many to run all at once (API throttling, server overload)

How do we achieve this?

The Setup:

```
const LETTERS = [
  "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m",
  "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"
]

const model = {
  findAll({ offset, limit }) {
    return new Promise((resolve, reject) => {
      setTimeout(() => resolve(LETTERS.slice(offset, offset + limit)),
                 200)
    });
  }
}
```

The Setup:

```
function doLongRequestForBatch(batch) {
  return new Promise((resolve, reject) => {
    setTimeout(
      () => {
        const upper = batch.map(letter => letter.toUpperCase());
        console.log("Processed batch: ", upper);
        resolve(200);
      },
      Math.floor(Math.random() * 1000 + 400)
    )
  });
}
```

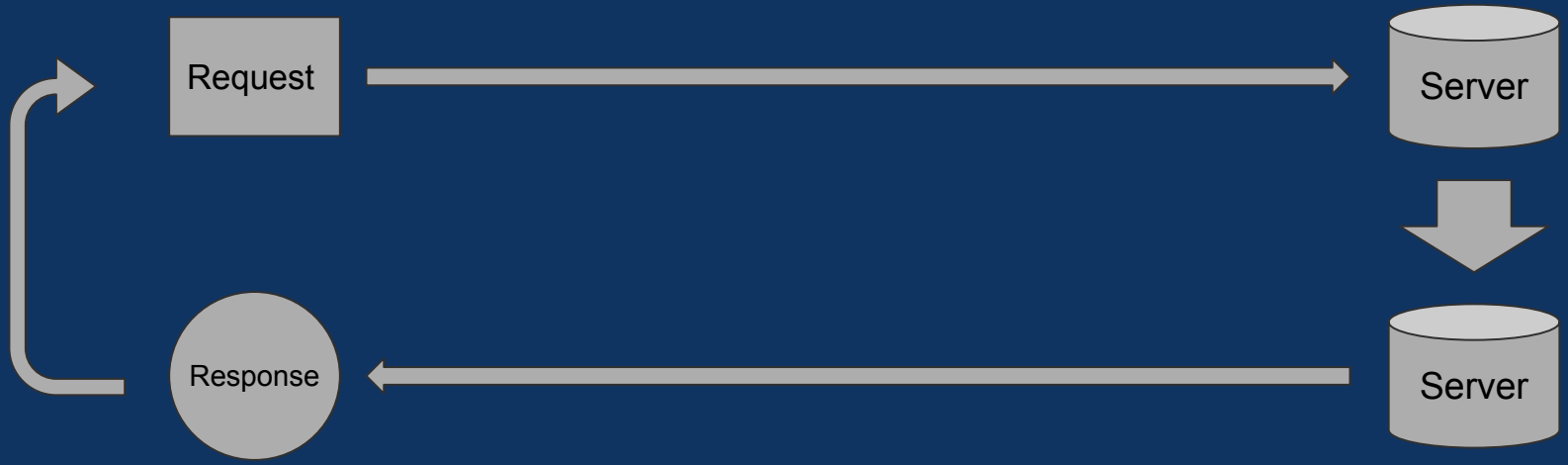
A Naive Approach

```
SET query (this has options for the query)
SET batch by making DB query

WHILE batch is not empty
```

- increment offset property of query
- make request using query
- await the result
- SET batch by making DB query

A Naive Approach



Let’s see some code...

What are the tradeoffs with this approach?

Discuss...

- Have to wait for each batch to finish before starting the next
- Can solve this by making batch bigger, but then failure could ruin a large batch
- Others?

A Better Approach

```
SET query
SET batch by making a DB request
SET pendingPromises = to empty array

WHILE batch is not empty
 ● increment offset property of query
 ● SET promise to the return of making request
 ● ADD promise to pendingPromises
 ● IF length of pendingPromises >= concurrency limit
       await resolution of all promises in pendingPromises
 ● make request to DB for next batch
```

A Better Approach

Can you spot the bottleneck?

What can we do to make this even better?

```
SET query
SET batch by making a DB request
SET pendingPromises = to empty array

WHILE batch is not empty
 ● increment offset property of query
 ● SET promise to the return of making request
 ● ADD promise to pendingPromises
 ● IF length of pendingPromises >= concurrency limit
       await resolution of all promises in pendingPromises
 ● make request to DB for next batch
```

Even Better

```
SET query
SET batch by making a DB request
SET pendingPromises = to empty array

WHILE batch is not empty
 ● increment offset property of query
 ● SET promise to the return of making request
 ● ADD promise to pendingPromises
 ● IF length of pendingPromises >= concurrency limit
       await resolution of one promise in pendingPromises
 ● make request to DB for next batch
Await the rest of the promises in pendingPromises
```

A Even Better Approach