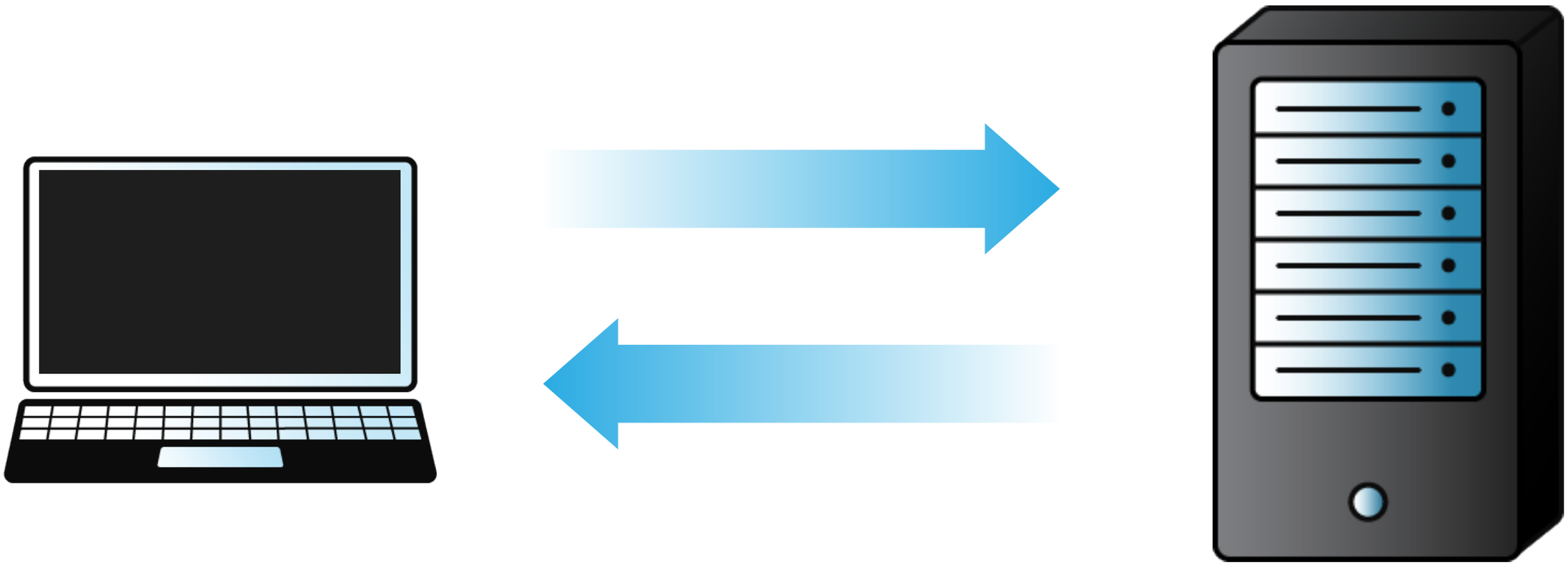
PASSING DATA OVER THE INTERNET

HTTP(S)

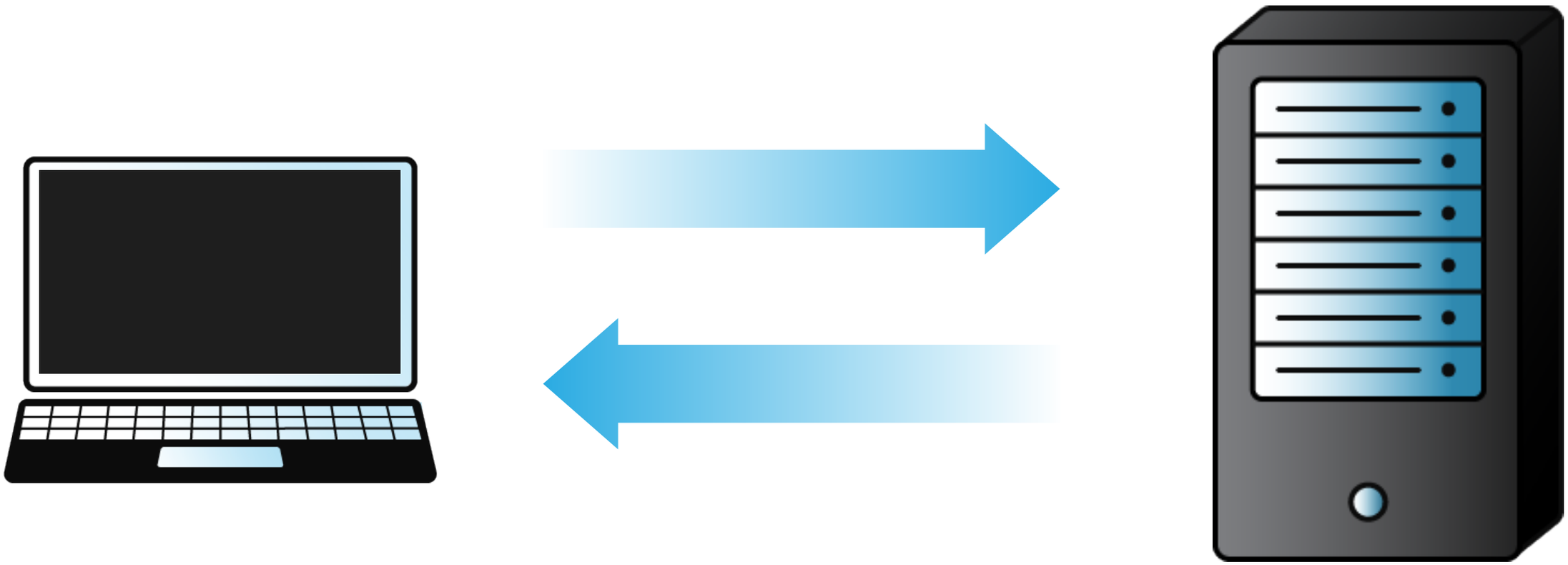
HYPER TEXT TRANSFER PROTOCOL

(SECURE)



Client

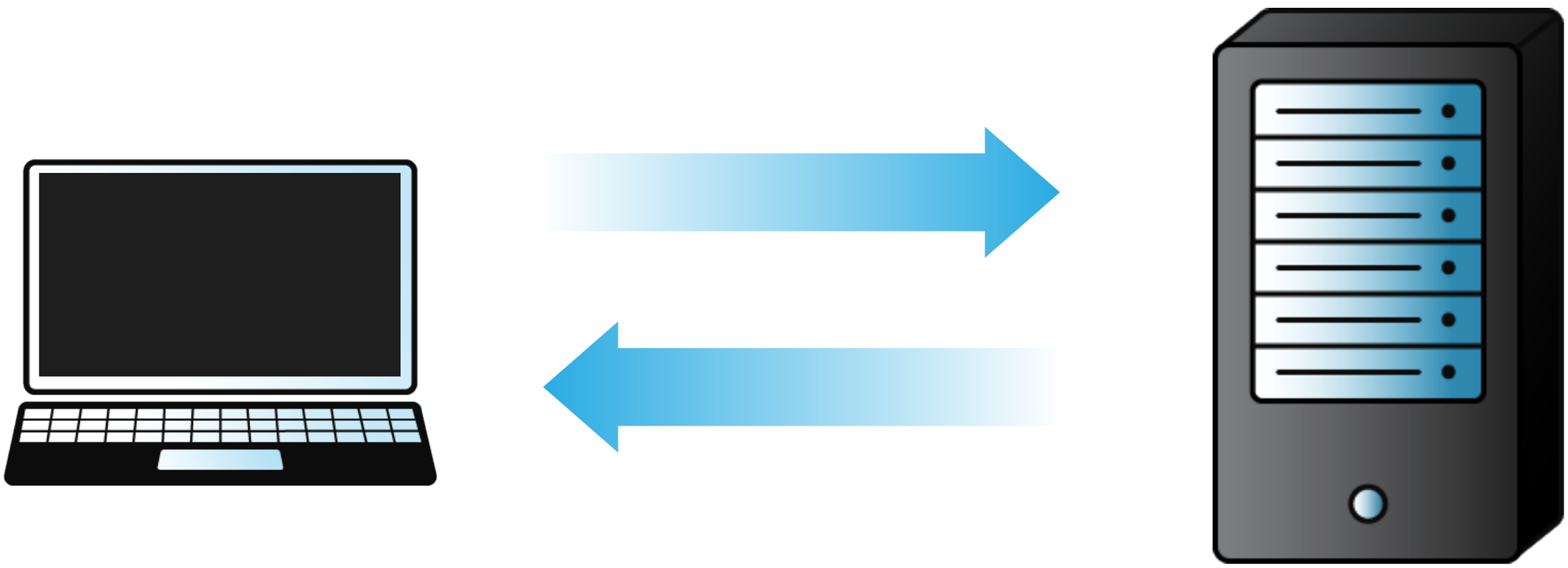
Server



Client

Server

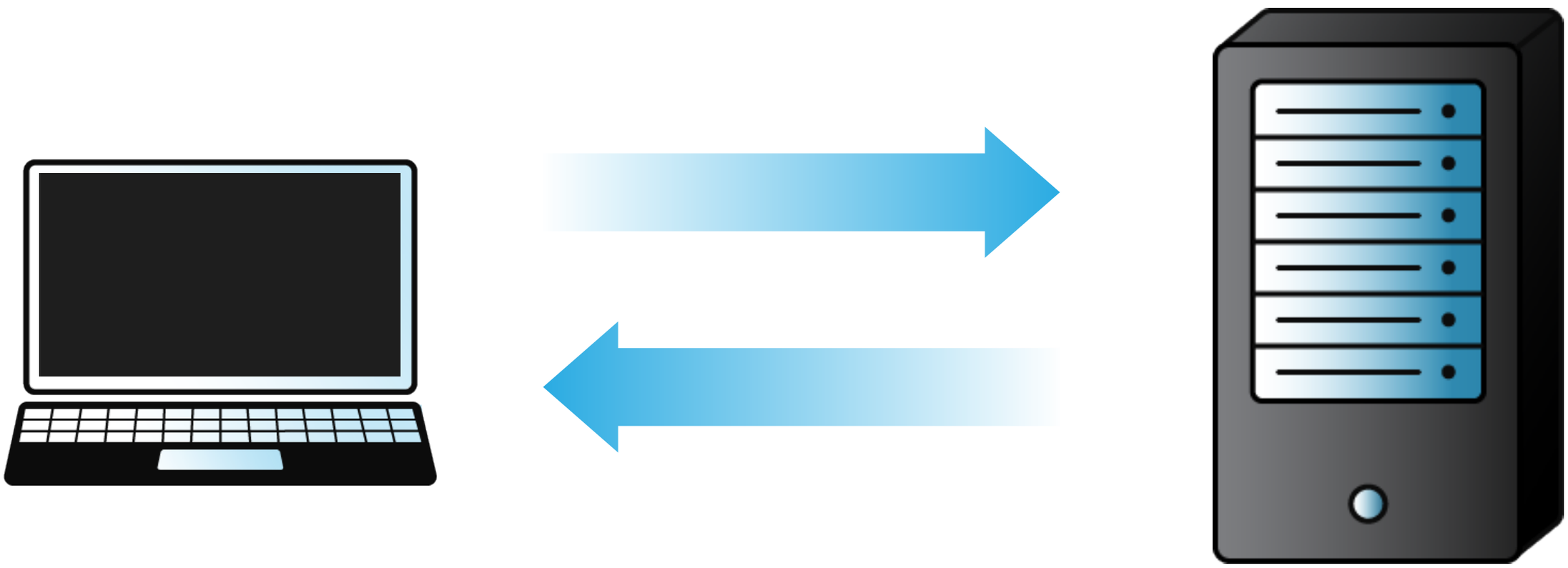
URL



URL

Uniform Resource Locator

```
https://devmountain.com
```

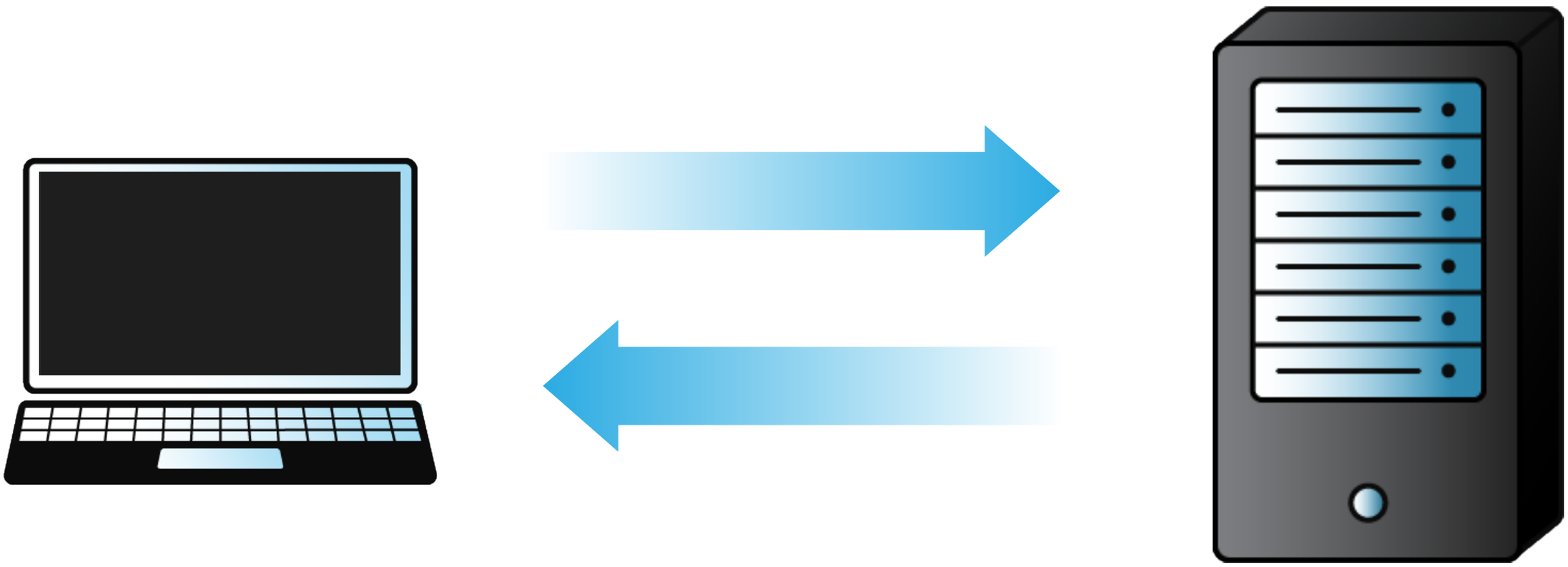


URL

Uniform Resource Locator

```
https://devmountain.com
   |
```

The protocol



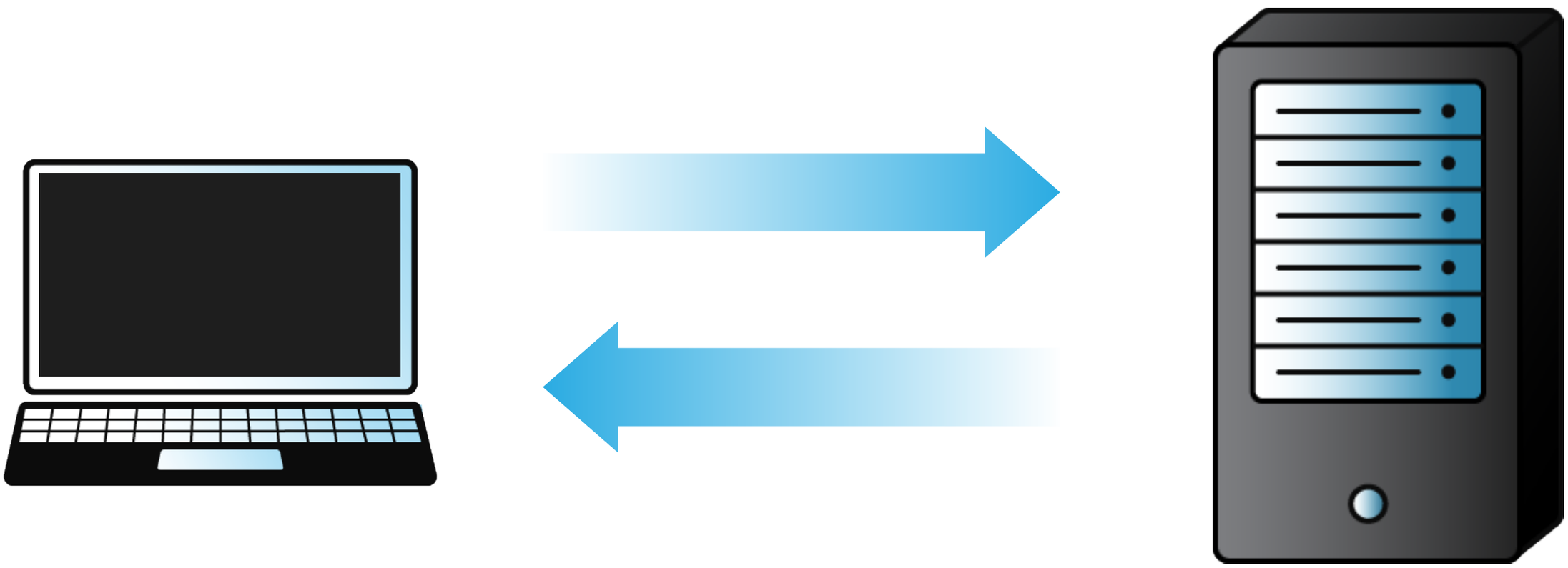
URL

Uniform Resource Locator

```
https://devmountain.com
```

The protocol

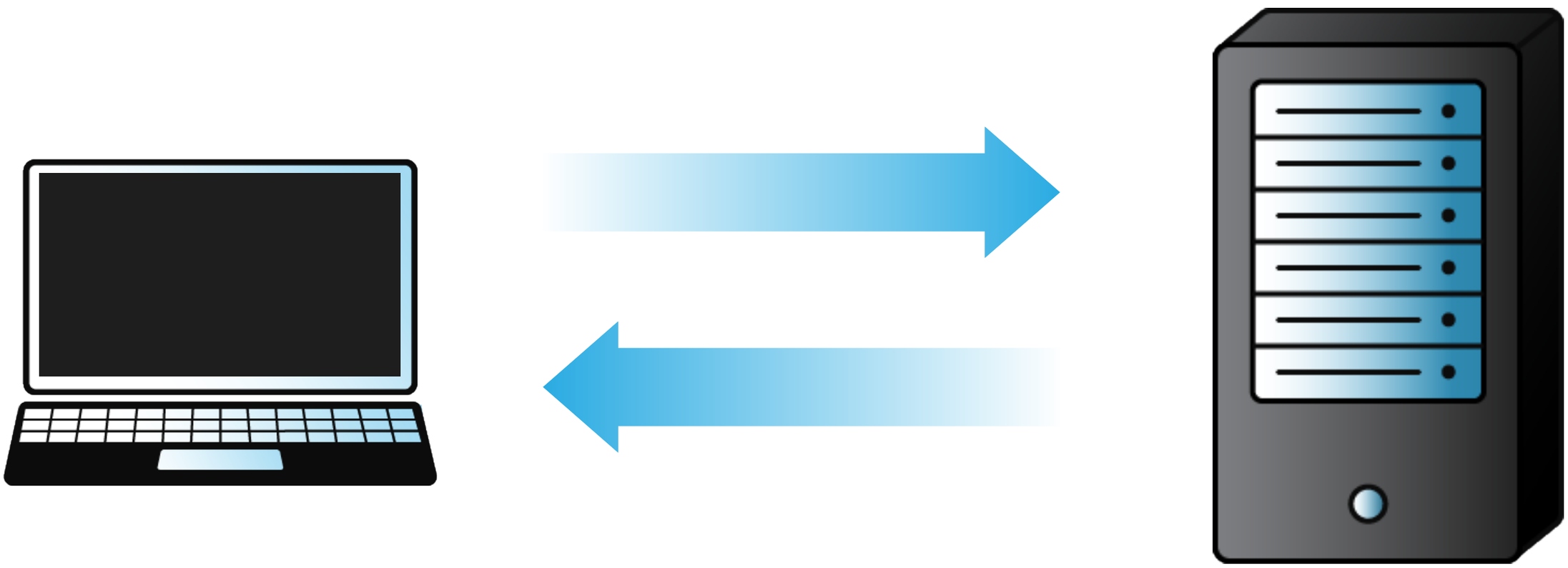
Domain. Or what will be interpreted as an IP address



URL

Uniform Resource Locator

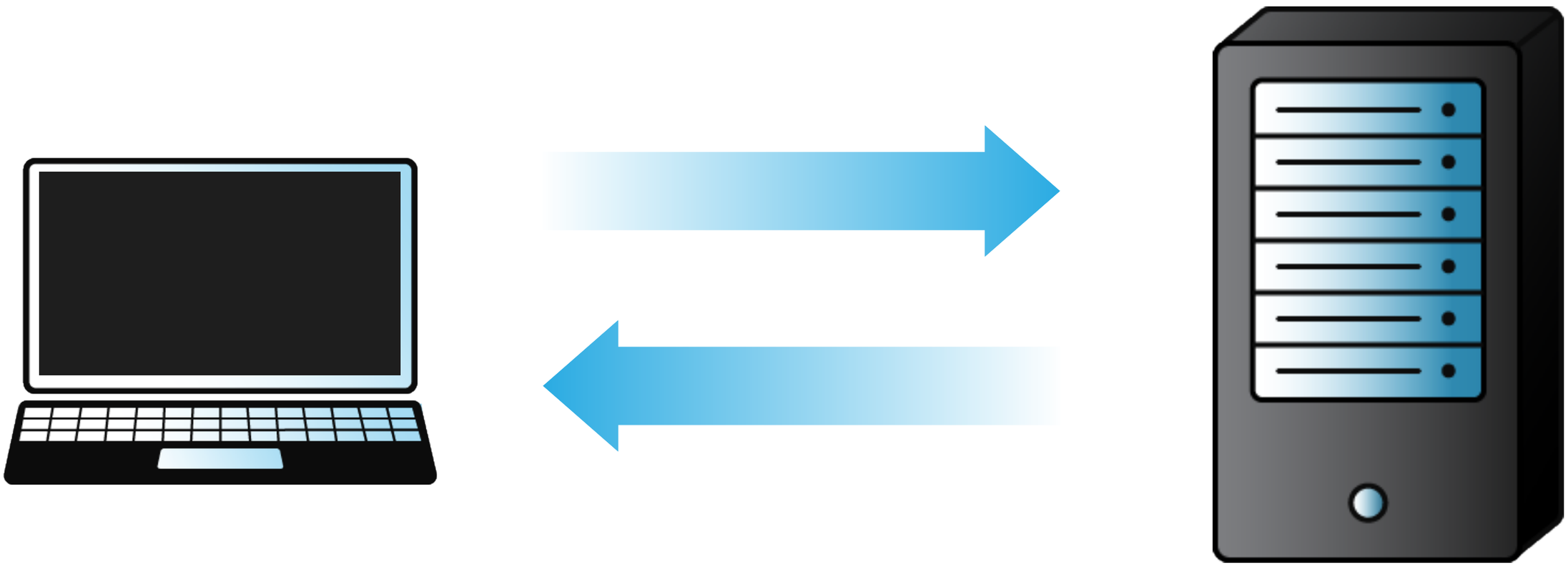
```
https://devmountain.com/about.html
```



URL

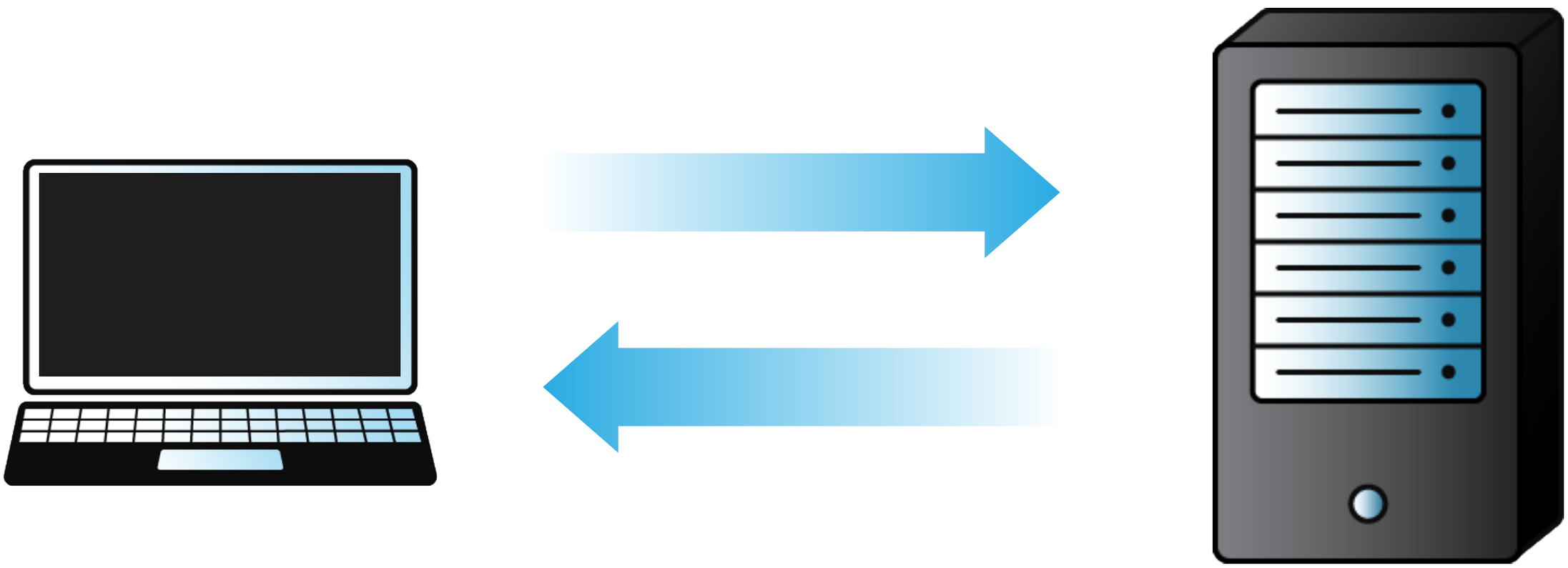
Uniform Resource Locator

```
https://blog.devmountain.com/?s=tech
```



Header

Contains information about the request or response.

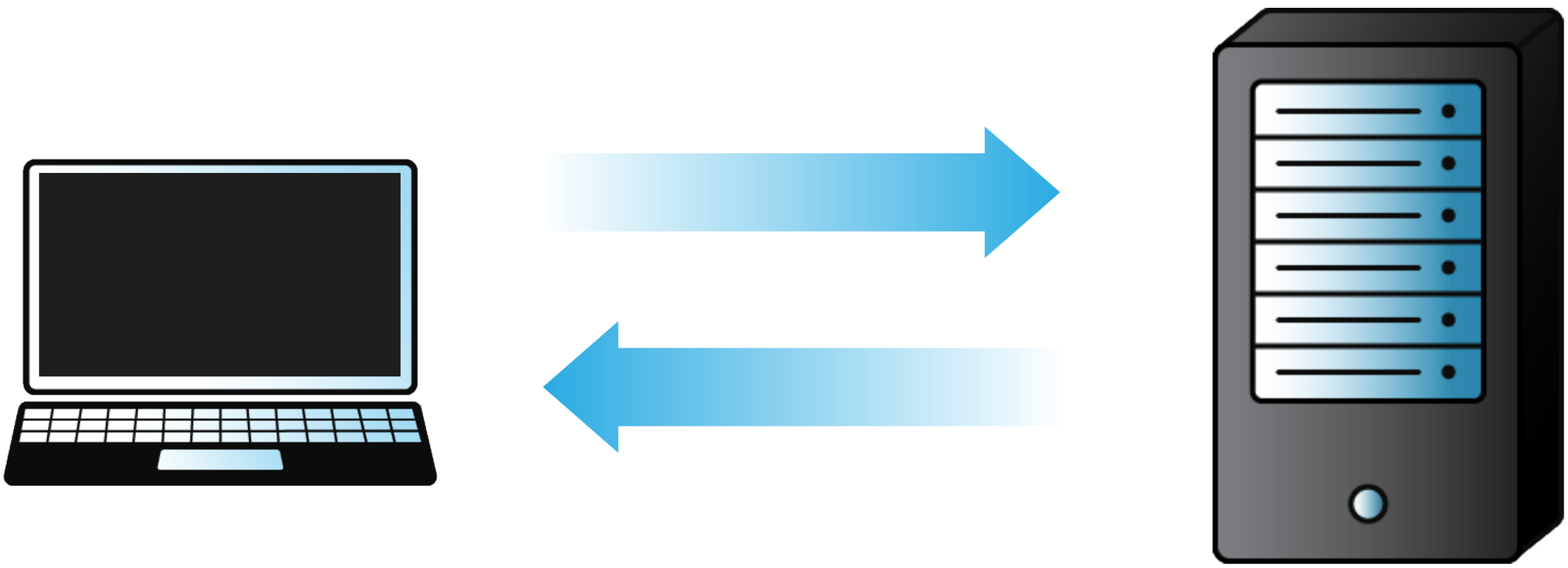


Header

Contains information about the request or response.

Body

Data being sent with the request or response. This can be empty depending on the request/response.



JSON

JAVASCRIPT OBJECT NOTATION

FORMAT FOR STRUCTURING DATA

```
var superHero  =  {
                    realName: "Steve Rogers",
                    heroName: "Captain America",
                    origin: "Scientific Experiment",
                    age: 95
                    }
```

```
{
"superHero": {
              “realName": "Steve Rogers",
              “heroName": "Captain America",
              “origin": "ScientificExperiment",
              “age": 95
             }
}
```

```
{
"superHero": {
              “realName": "Steve Rogers",
              “heroName": "Captain America",
              “origin": "ScientificExperiment",
              “age": 95
             }
}
```

Property names are in quotes.

```
{
"superHero": {
              “realName": "Steve Rogers",
              “heroName": "Captain America",
              “origin": "ScientificExperiment",
              “age": 95
             }
}
```

Property names are in quotes.

All quotes are double quotes.

```
{
"superHero": {
              “realName": "Steve Rogers",
              “heroName": "Captain America",
              “origin": "ScientificExperiment",
              “age": 95
             }
}
```

Property names are in quotes.

All quotes are double quotes.

The values are only strings and numbers.

```
{
"superHero": {
              “realName": "Steve Rogers",
              “heroName": "Captain America",
              “origin": "ScientificExperiment",
              “age": 95
             }
}
```

Property names are in quotes.

All quotes are double quotes.

The values are only strings and numbers.

No trailing commas.

```
{
   "superHeroes":[
       {
           "realName": "Steve Rogers",
           "heroName": "Captain America",
           "origin": "ScientificExperiment",
           "age": 95
       },
       {
           "realName": "Barry Allen",
           "heroName": "The Flash",
           "origin": "Freak Particle Reactor Explosion",
           "age": 33
       },
       {
           "realName": "Hal Jordan",
           "heroName": "Green Lantern",
           "origin": "Cosmic Power Ring",
           "age": 44
       }
   ]
}
```

```
{
   "superHeroes":[
       {
           "realName": "Steve Rogers",
           "heroName": "Captain America",
           "origin": "ScientificExperiment",
           "age": 95
       },
       {
           "realName": "Barry Allen",
           "heroName": "The Flash",
           "origin": "Freak Particle Reactor Explosion",
           "age": 33
       },
       {
           "realName": "Hal Jordan",
           "heroName": "Green Lantern",
           "origin": "Cosmic Power Ring",
           "age": 44
       }
   ]
}
```

The first property's value is an array of objects.

```
{
   "Avengers League" : {
      "boss": "Fury Man",
      "hq": "123 supers lane",
      "superHeroes":[
          {
              "realName": "Steve Rogers",
              "heroName": "Captain America",
              "origin": "ScientificExperiment",
              "age": 95
          },
         {
              "realName": "Barry Allen",
              "heroName": "The Flash",
              "origin": "Freak Particle Reactor Explosion",
              "age": 33
         },
         {
              "realName": "Hal Jordan",
              "heroName": "Green Lantern",
              "origin": "Cosmic Power Ring",
              "age": 44
         }
      ]
   }
}
```

```
{
   "Avengers League" : {
      "boss": "Fury Man",
      "hq": "123 supers lane",
      "superHeroes":[
          {
              "realName": "Steve Rogers",
              "heroName": "Captain America",
              "origin": "ScientificExperiment",
              "age": 95
          },
         {
              "realName": "Barry Allen",
              "heroName": "The Flash",
              "origin": "Freak Particle Reactor Explosion",
              "age": 33
         },
         {
              "realName": "Hal Jordan",
              "heroName": "Green Lantern",
              "origin": "Cosmic Power Ring",
              "age": 44
         }
      ]
   }
}
```

The first property's value is an object.

```
{
   "Avengers League" : {
      "boss": "Fury Man",
      "hq": "123 supers lane",
      "superHeroes":[
          {
              "realName": "Steve Rogers",
              "heroName": "Captain America",
              "origin": "ScientificExperiment",
              "age": 95
          },
         {
              "realName": "Barry Allen",
              "heroName": "The Flash",
              "origin": "Freak Particle Reactor Explosion",
              "age": 33
         },
         {
              "realName": "Hal Jordan",
              "heroName": "Green Lantern",
              "origin": "Cosmic Power Ring",
              "age": 44
         }
      ]
   }
}
```

The first property's value is an object.
One of the object's properties is an array of objects.

REST

REPRESENTATIONAL STATE TRANSFER

AN ARCHITECTURE OR DESIGN CONCEPT FOR TRANSFERRING DATA

POST

POST
GET

POST

GET

PUT

POST

GET

PUT

DELETE

CRUD

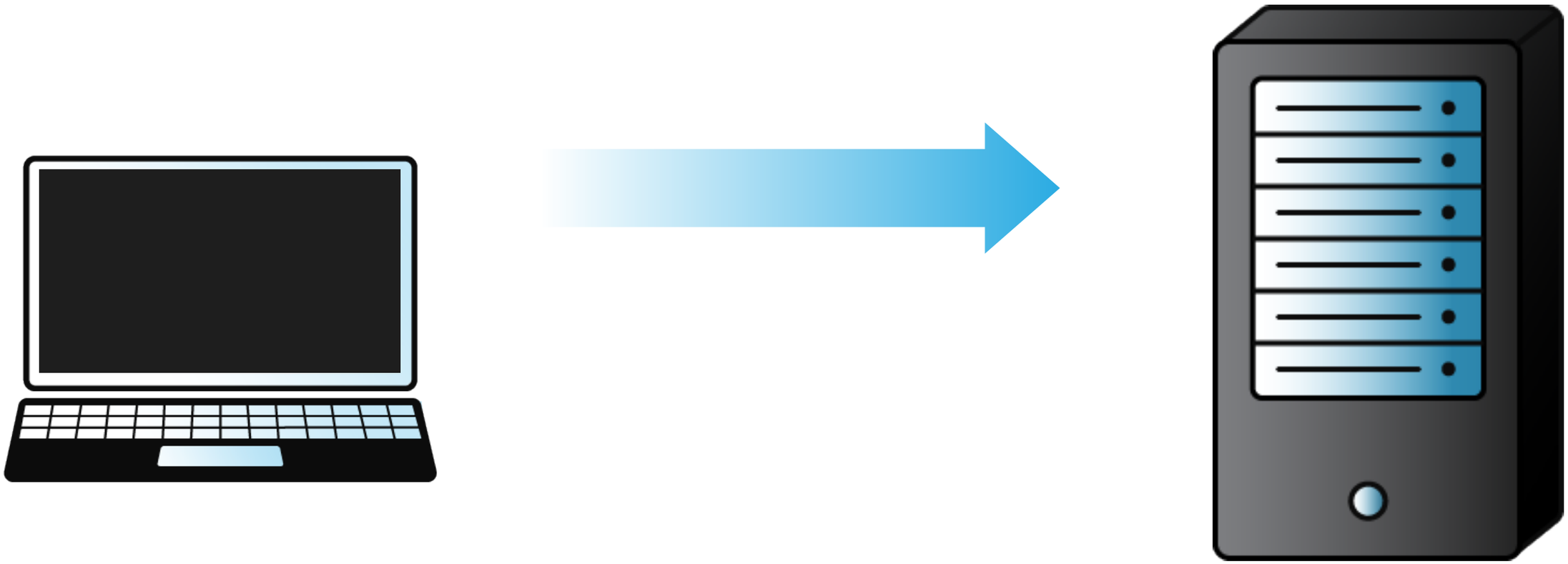
BASIC OPERATIONS TO MANIPULATE AND MANAGE DATA

POST – CREATE

GET – READ

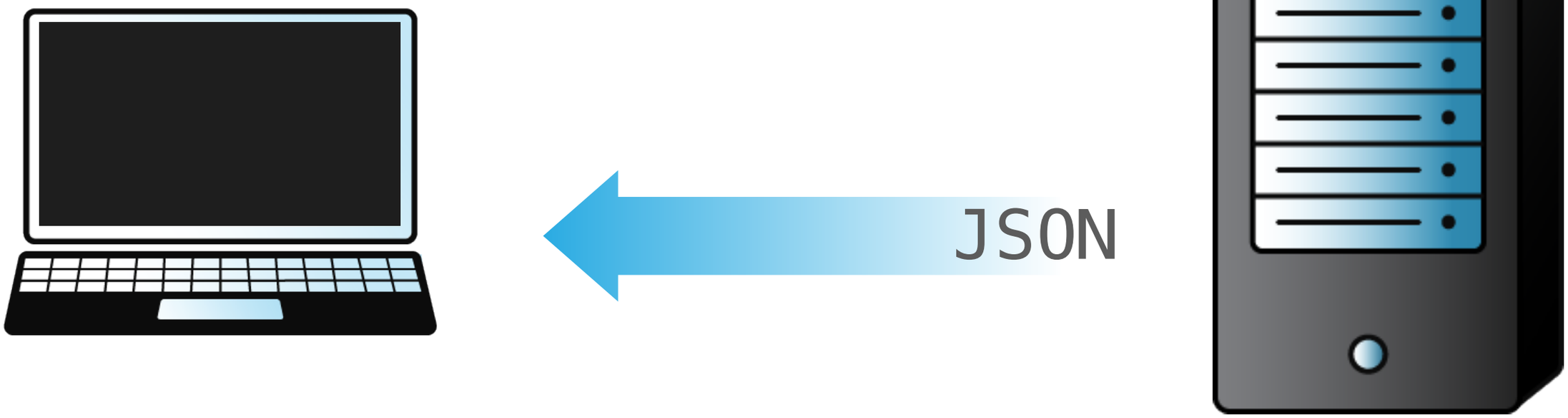
PUT – UPDATE

DELETE – DELETE



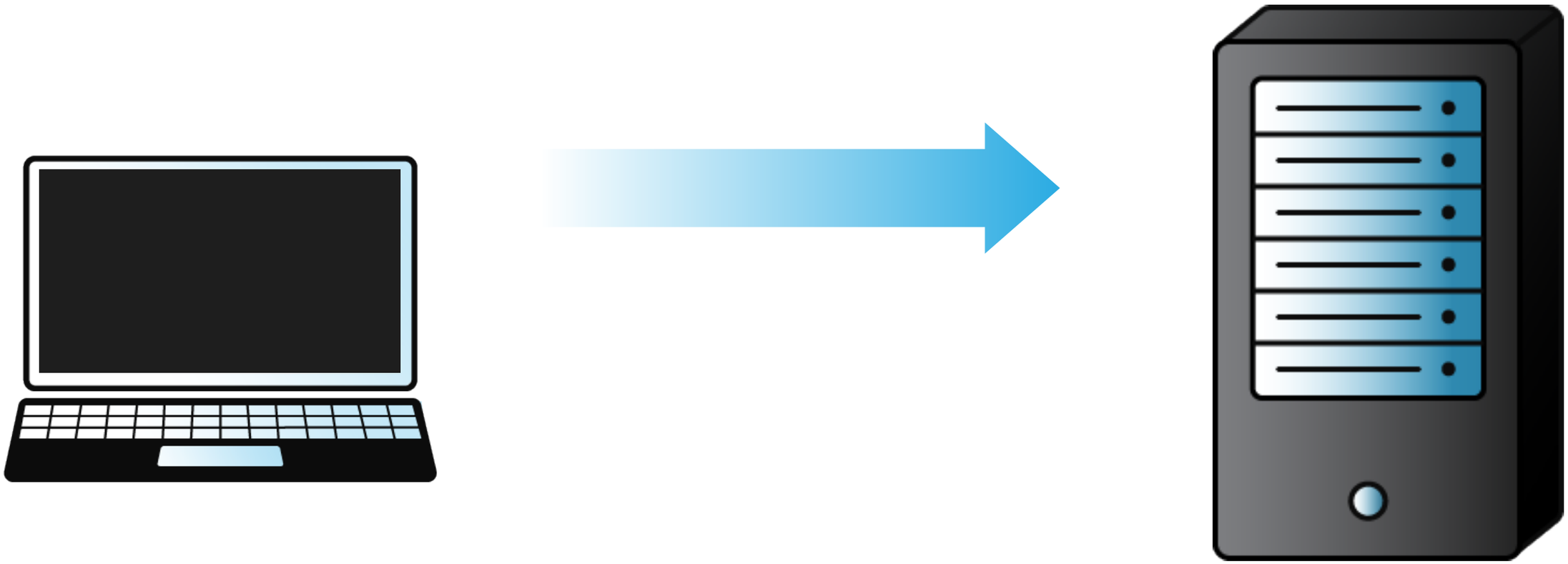
GET

http://www.devmounta.in/api/students


JSON

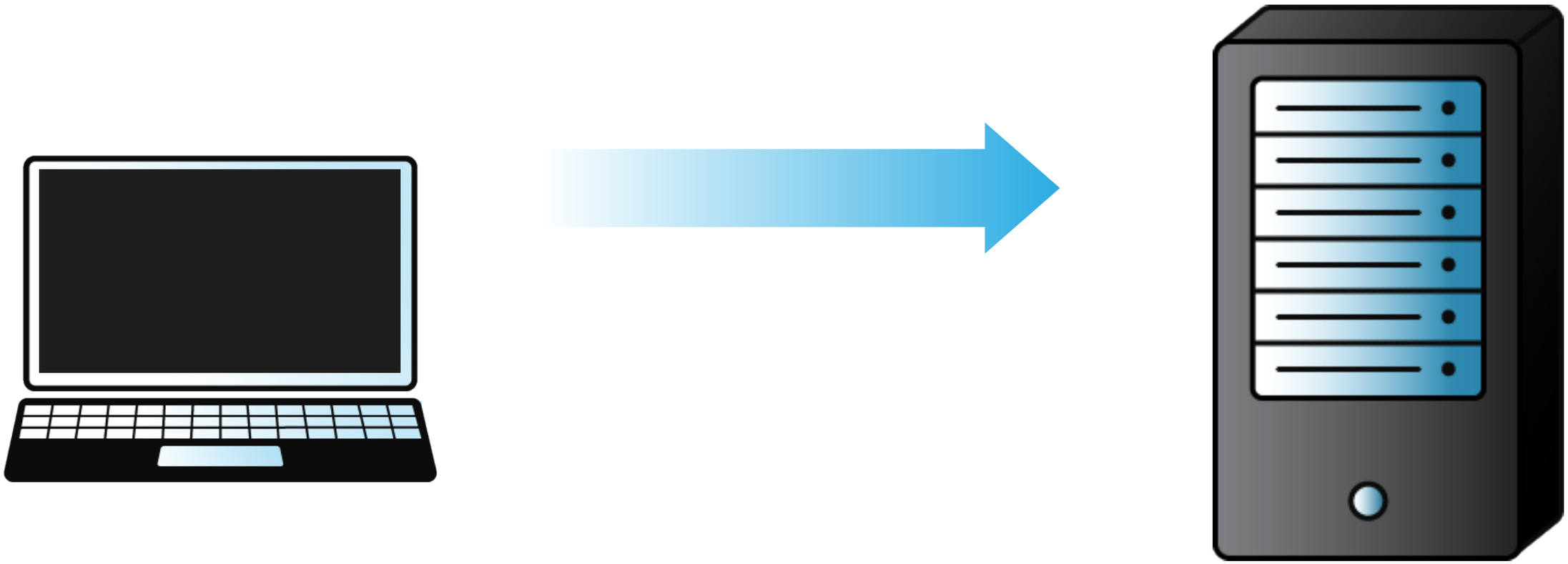
GET

http://www.devmounta.in/api/students/9435



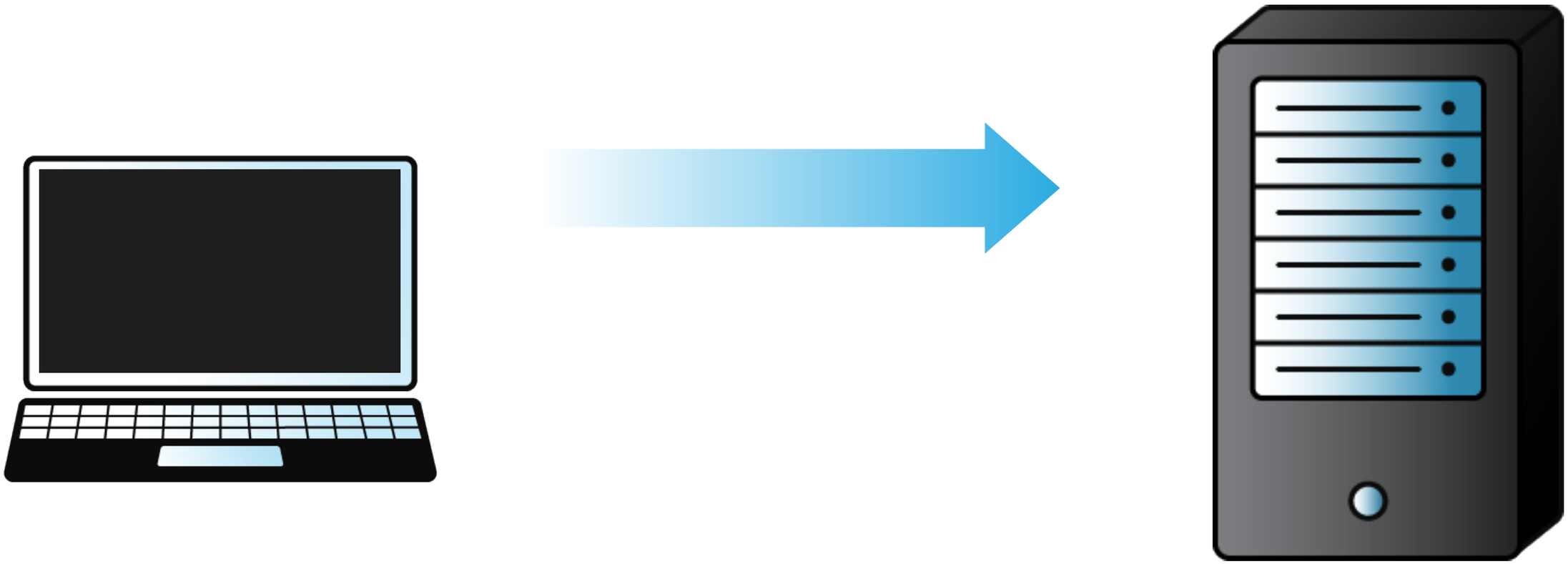
PUT

`http://www.devmounta.in/api/students/9435/?name='Jenny'`



POST

http://www.devmounta.in/api/student, BODY: {data}



DELETE

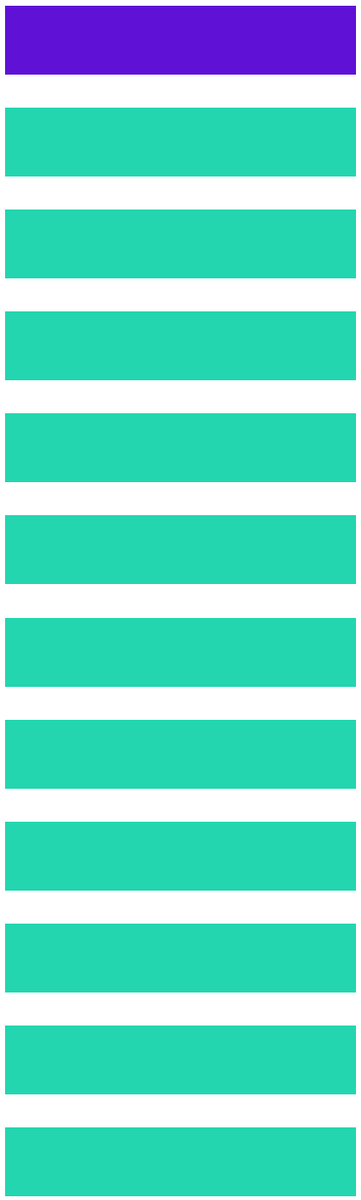
http://www.devmounta.in/student/9998

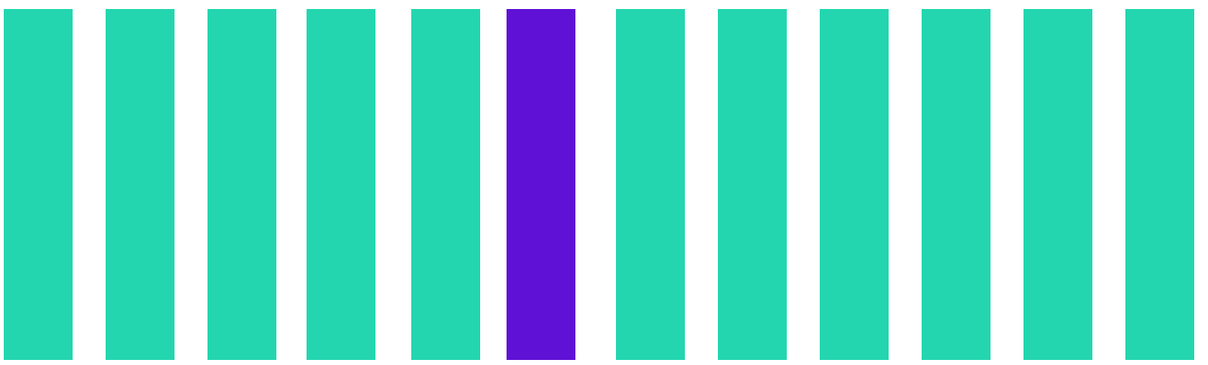
HOW DOES THIS WORK WITH JAVASCRIPT?

JAVASCRIPT IS

SYNCHRONOUS

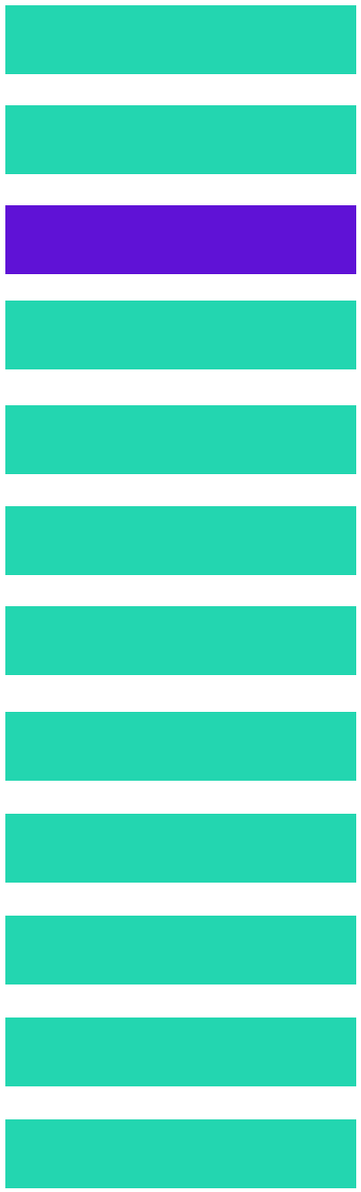
"ONE THING AT A TIME"



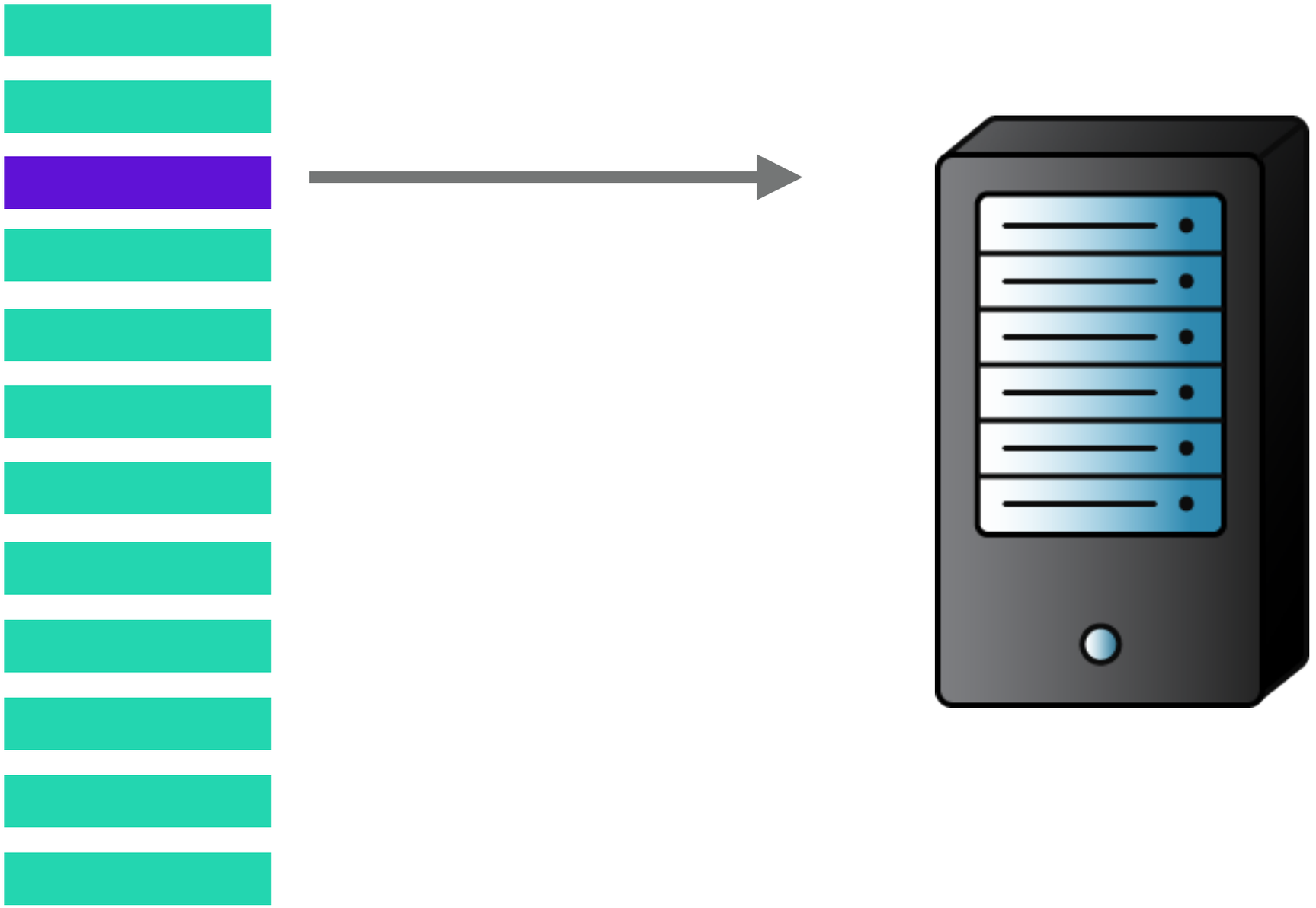


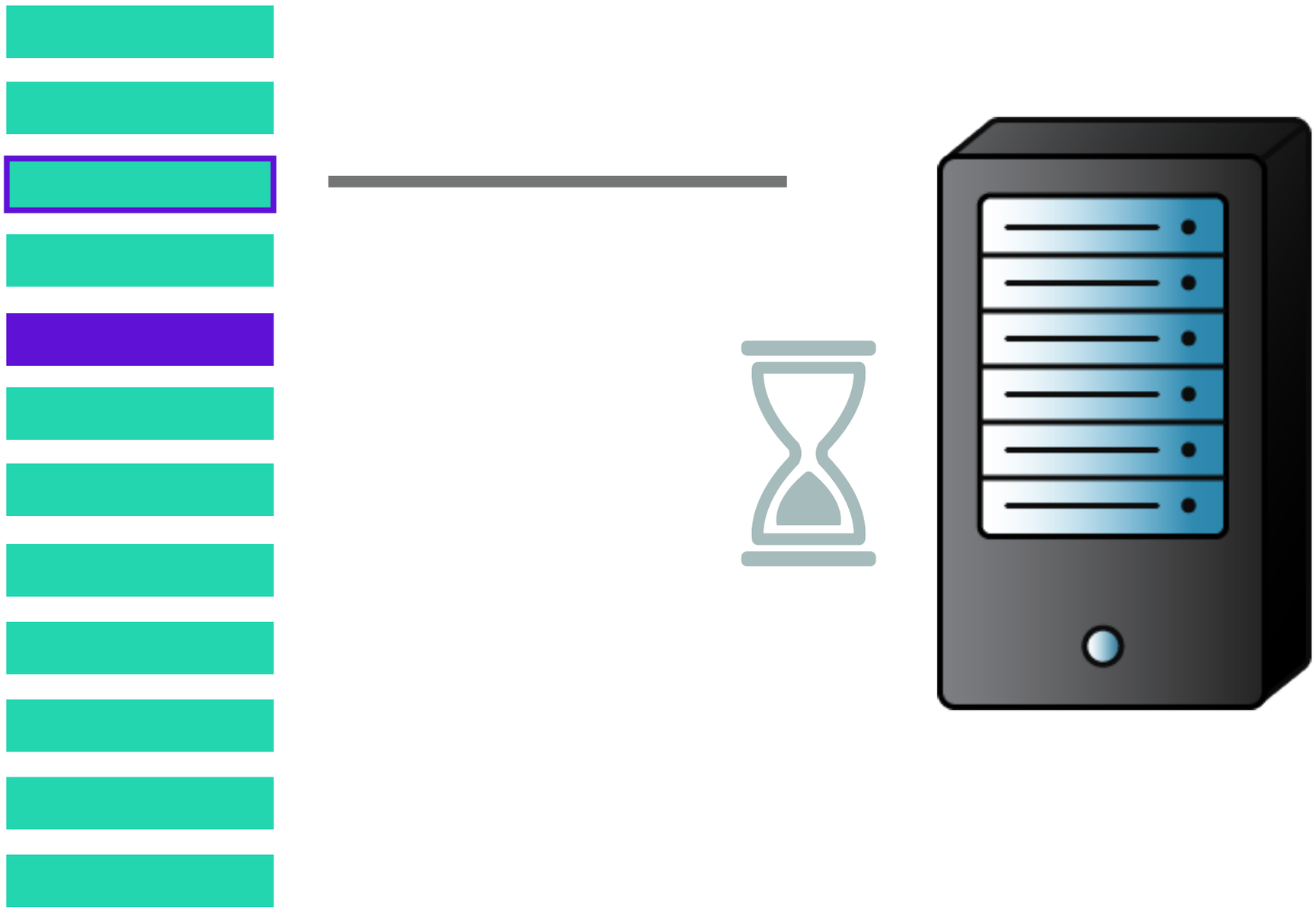
WHAT HAPPENS WHEN YOU MAKE AN HTTP REQUEST?

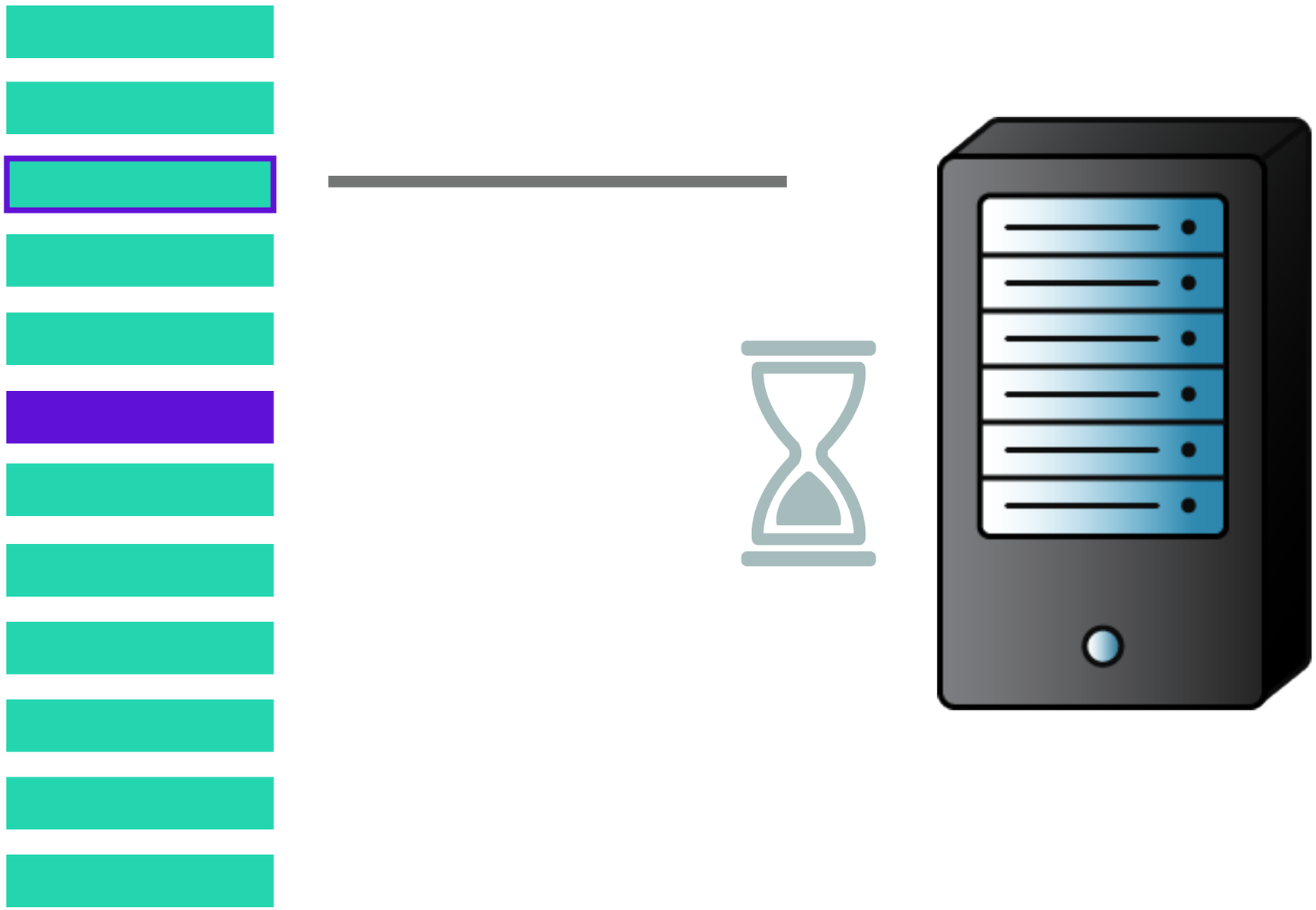
WE NEED THINGS TO BE ASYNCHRONOUS

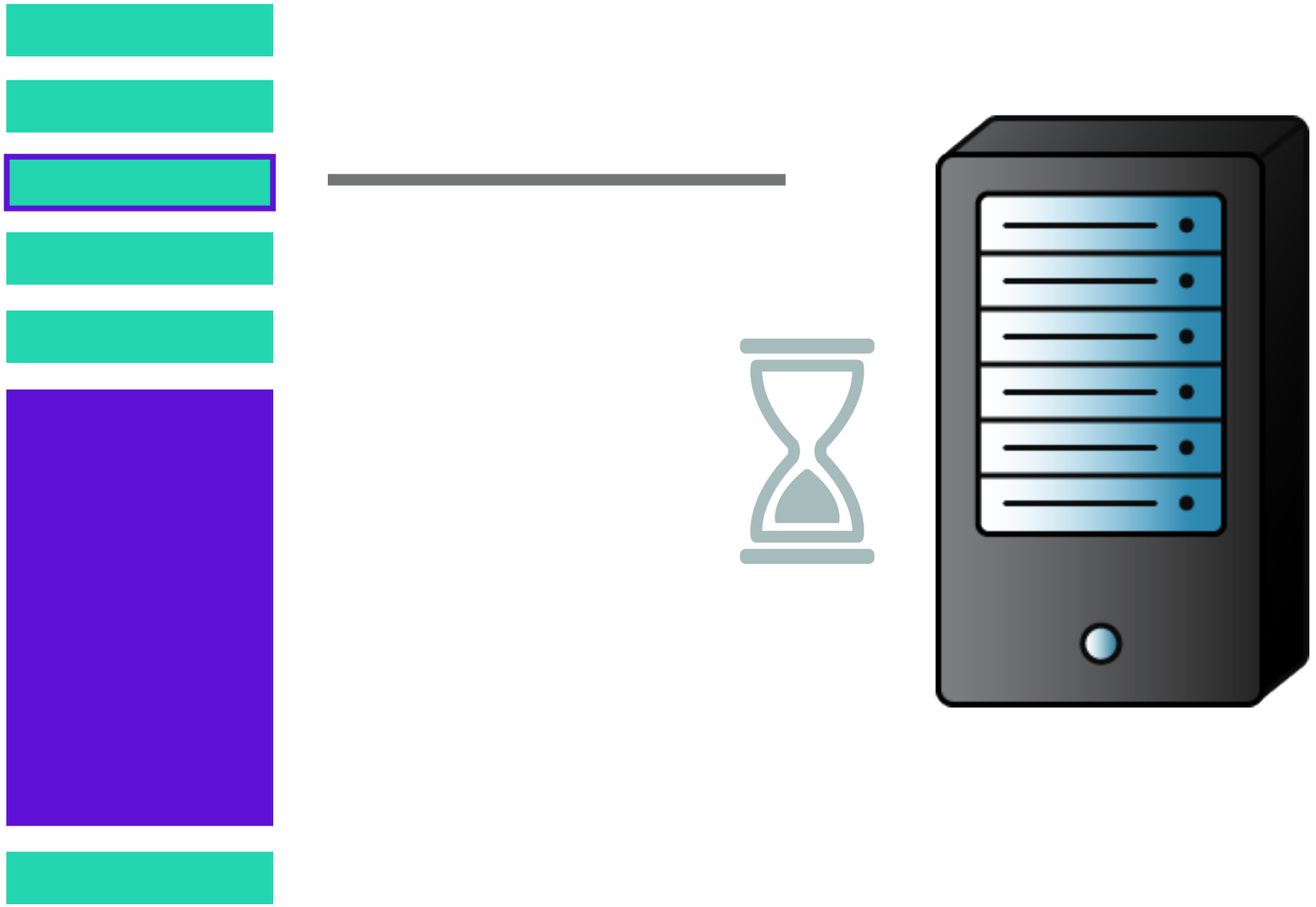


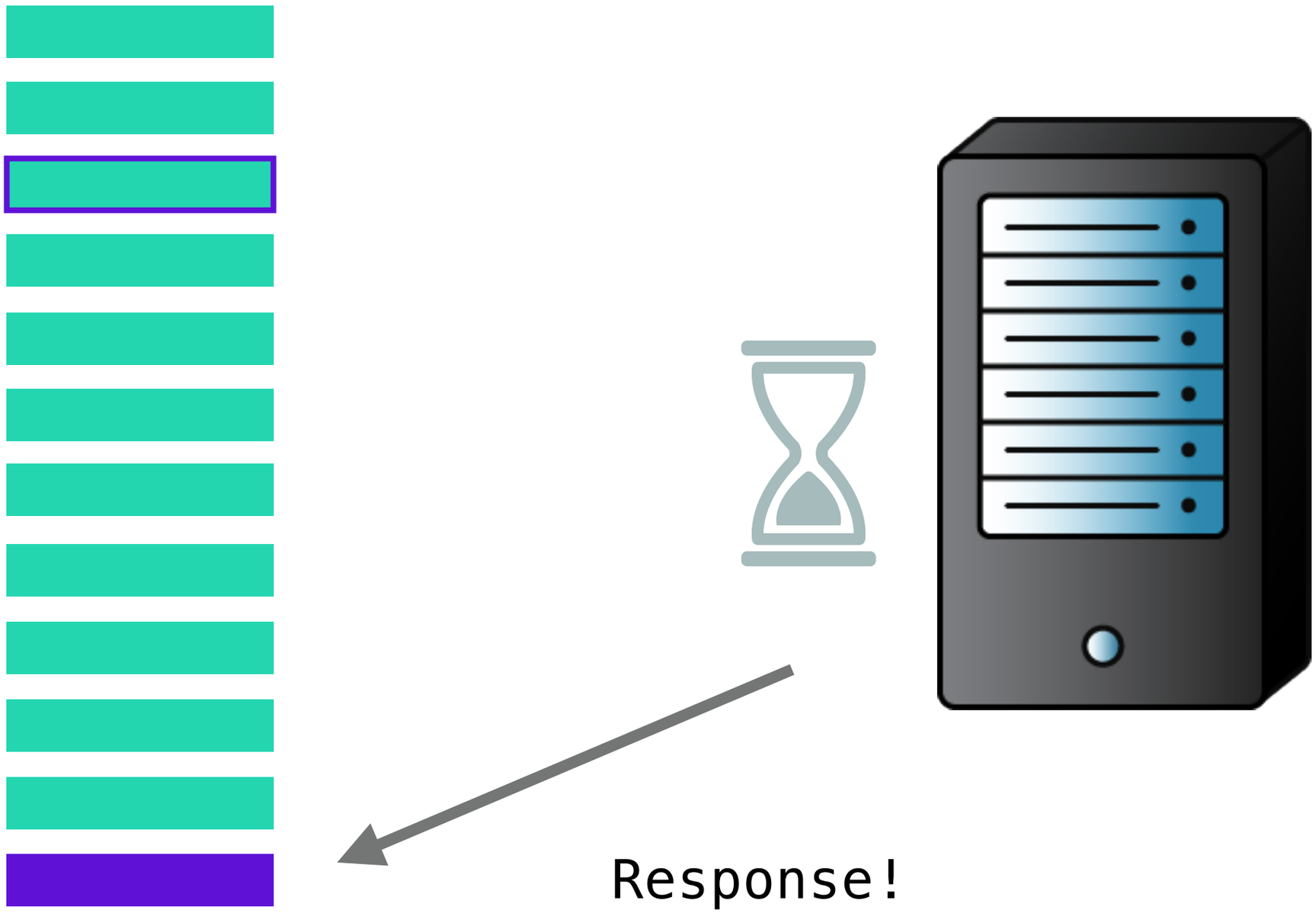
GET request!









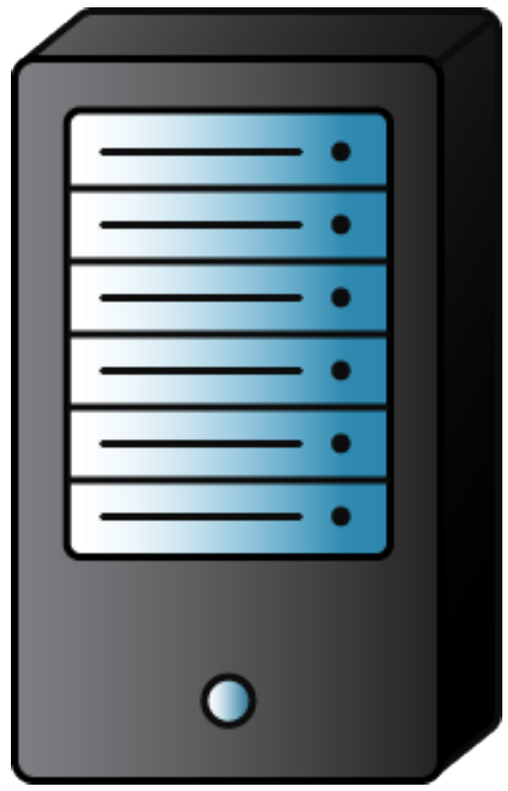

Response!

AXIOS!

(but first a component lifecycle review)

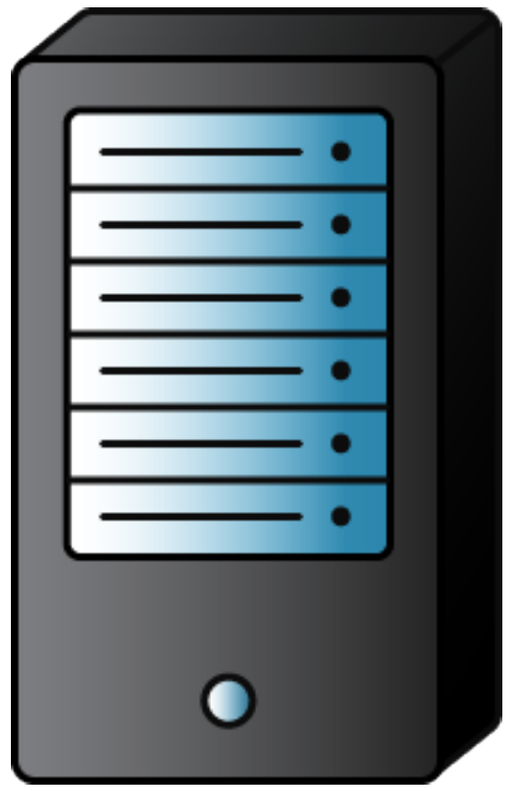
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



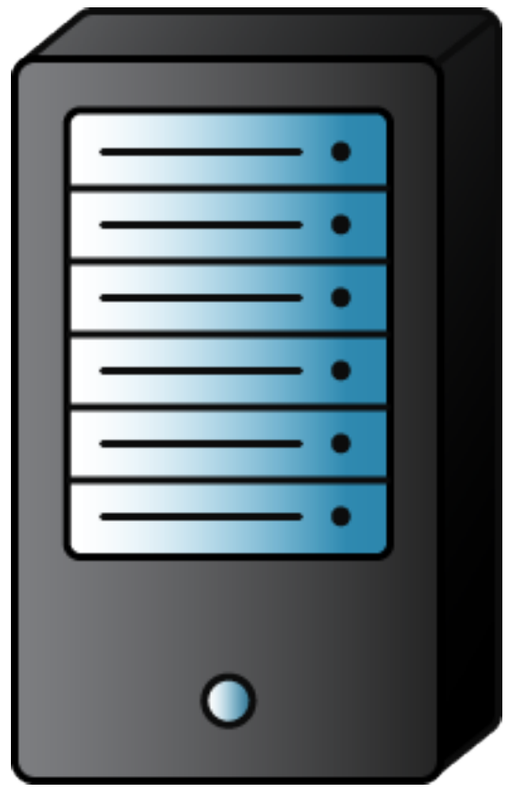
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



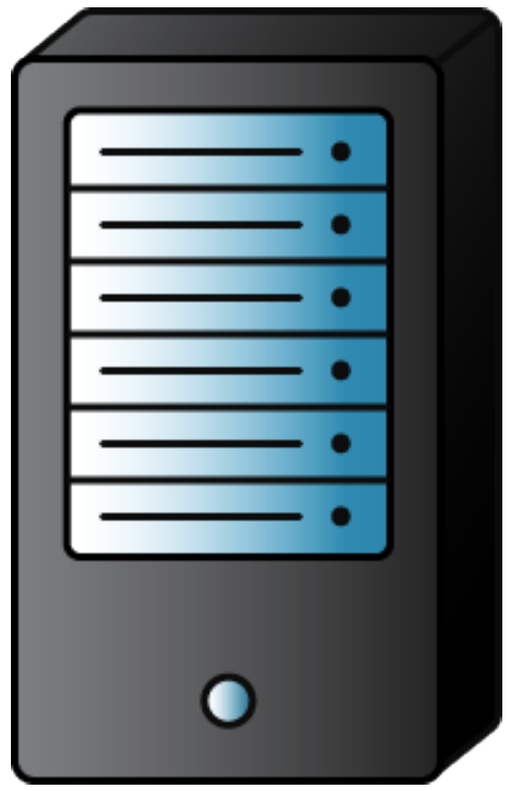
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



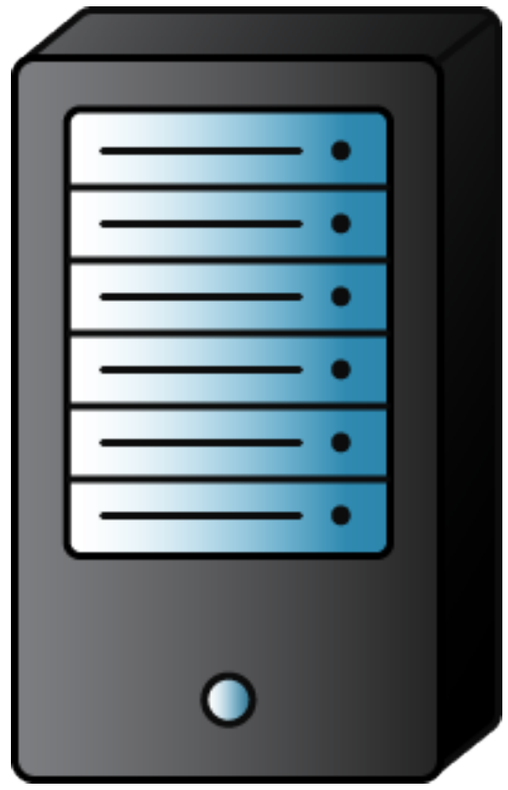
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



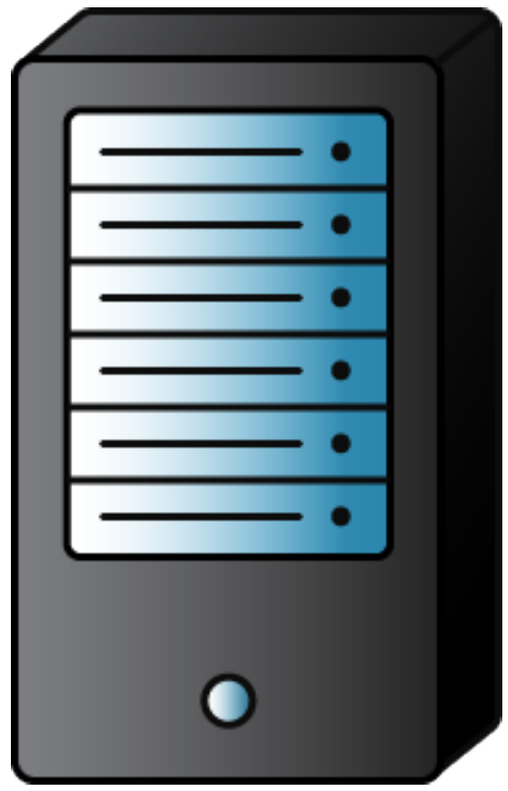
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



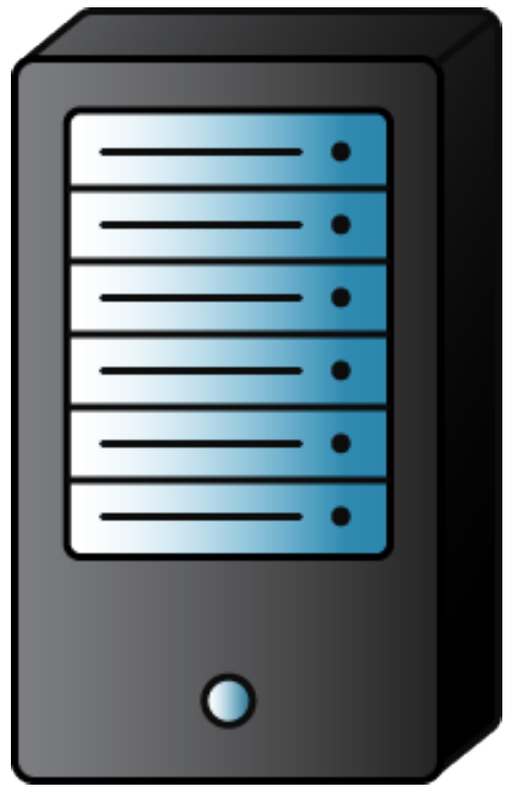
```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```



```jsx
import React, { Component } from 'react';
import axios from 'axios';

export default class HTTPDemo extends Component {
  constructor() {
    super()
    this.state = {
      people: []
    }
  }
  componentDidMount() {
    let promise = axios.get('https://swapi.co/api/people/')
    promise.then(res => {
      this.setState({
        people: res.data.results
      })
    })
  }
  render() {
    const people = this.state.people.map((e, i) => {
      return (
        <h3 key={i}>{e.name}</h3>
      )
    })
    return (
      <div>
        <h1>STAR WARS!</h1>
        { people }
      </div>
    )
  }
}
```