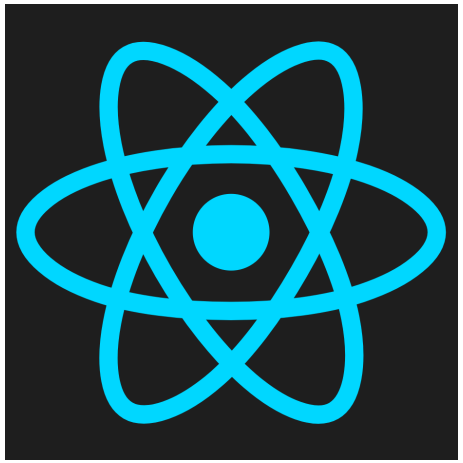


SERVERLESS SERVER-SIDE RENDERING

Hello 👋

I work with…

- React
- Typescript
- Cloud things on AWS
- Node.js
- Terraform

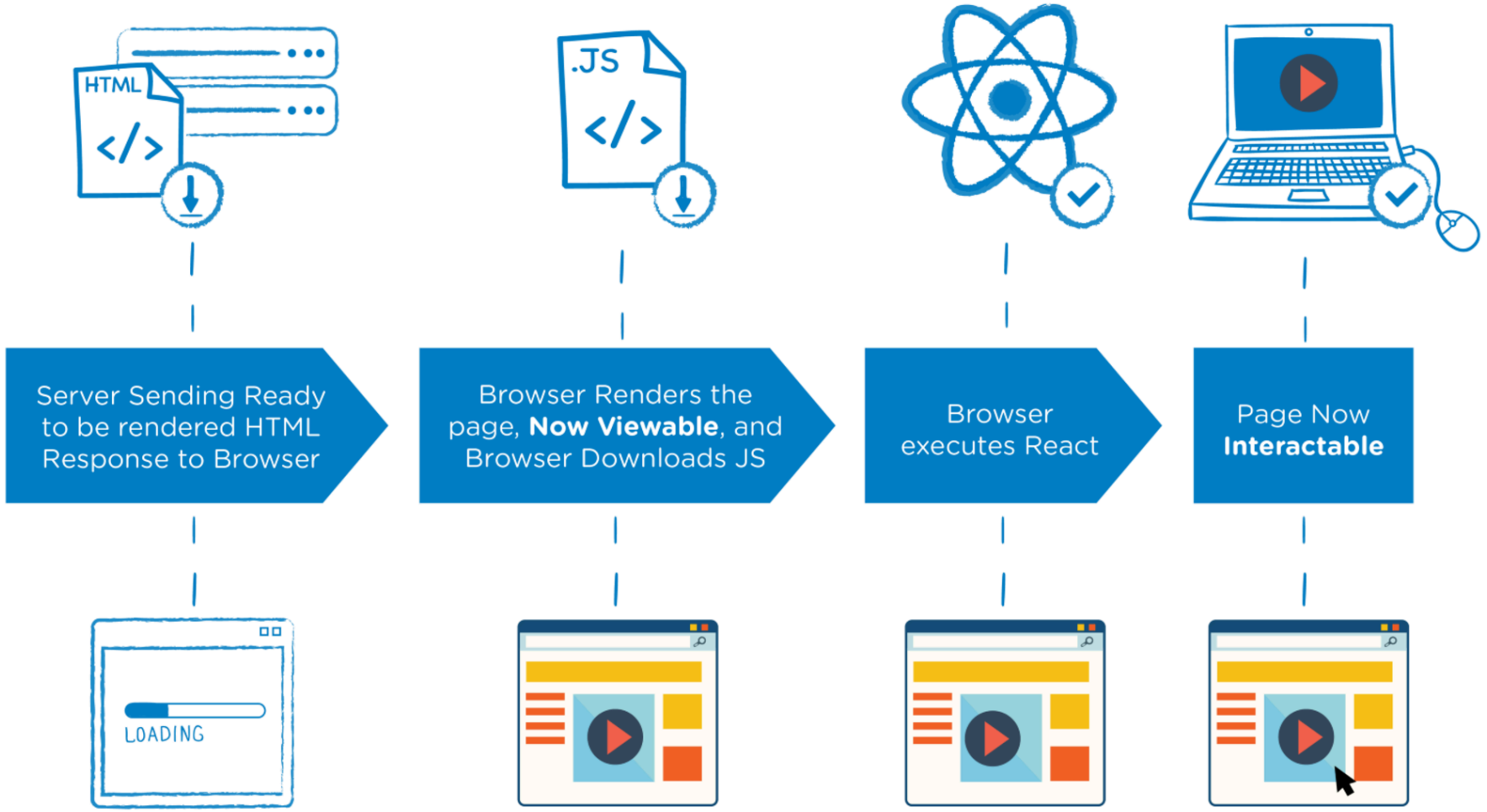

THIS IS ME WAITING
FOR YOUR SLOW WEBSITE TO LOAD

PERFORMANCE ⚡

USER EXPERIENCE ✨

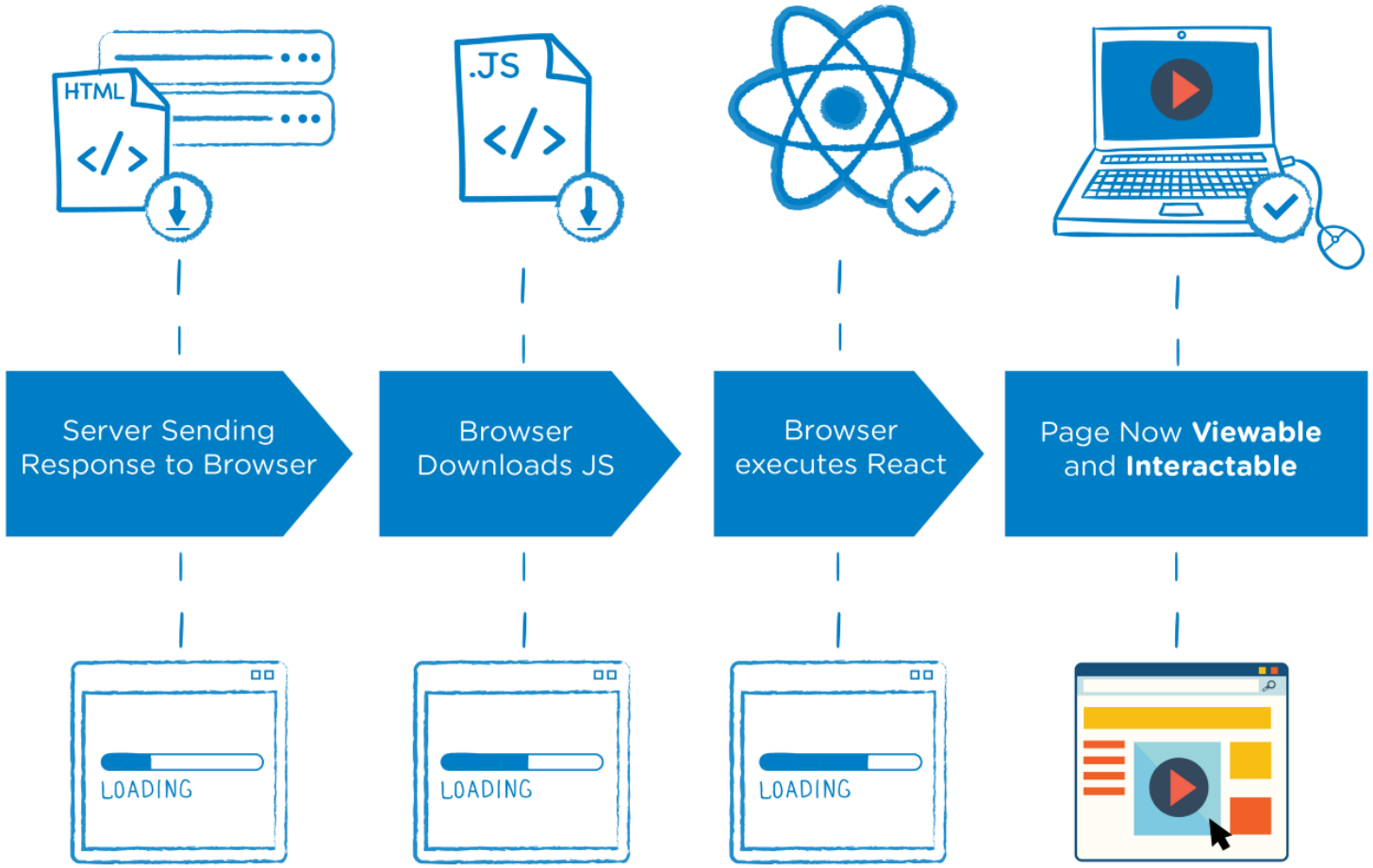
SEO 

SSR



From https://medium.com/walmartlabs/the-benefits-of-server-side-rendering-over-client-side-rendering-5d07ff2cefe8

CSR



From https://medium.com/walmartlabs/the-benefits-of-server-side-rendering-over-client-side-rendering-5d07ff2cefe8

SERVER-SIDE RENDERING

PROS:

SEO

Performance

CONS:

Added complexity

TTFB slower than client-side rendering

```
import express from 'express';
import fs from 'fs';
import path from 'path';
import React from 'react';
import ReactDOMServer from 'react-dom/server';
import Hello from './Hello.js';

function handleRender(req, res) {
  // Renders our Hello component into an HTML string
  const html = ReactDOMServer.renderToString(<Hello />);

  // Load contents of index.html
  fs.readFile('./index.html', 'utf8', function (err, data) {
    if (err) throw err;

    // Inserts the rendered React HTML into our main div
    const document = data.replace(/<div id="app"><\/div>/, `<div id="app">${html}</div>

    // Sends the response back to the client
    res.send(document);
  });
}

const app = express();

// Serve built files with static files middleware
app.use('/build', express.static(path.join(__dirname, 'build')));

// Serve requests with our handleRender function
app.get('*', handleRender);

// Start server
app.listen(3000);
```

```
import React from 'react';
import Butter from 'buttercms'

const butter = Butter('b60a008584313ed21803780bc9208557b3b49fbb');

var Hello = React.createClass({
  getInitialState: function() {
    return {loaded: false};
  },
  componentWillMount: function() {
    butter.post.list().then((resp) => {
      this.setState({
        loaded: true,
        resp: resp.data
      })
    });
  },
  render: function() {
    if (this.state.loaded) {
      return (
        <div>
          {this.state.resp.data.map((post) => {
            return (
              <div key={post.slug}>{post.title}</div>
            )
          })}
        </div>
      );
    } else {
      return <div>Loading...</div>;
    }
  }
});

export default Hello;
```

From CSS Tricks: https://css-tricks.com/server-side-react-rendering/

LAMBDA

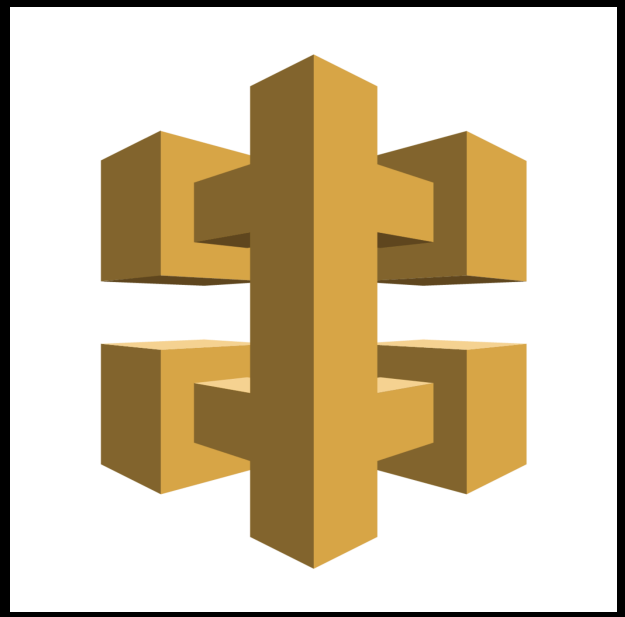
PROS:

- On-demand execution
- Lower cost than an instance (in some cases)
- Hassle-free scalability

CONS:

- Initial load time may be slow, but subsequent invocations will be faster
- Stage environments

API GATEWAY



API

- Endpoints
- Methods
- Event Object
- Proxy to Lambda

The Recipe

1. Create zip file of React app
2. Upload zip file to Lambda
3. Create API using API Gateway
4. Create an endpoint in the API
5. Create a GET method for that endpoint
6. Go back to Lambda and choose your new API Gateway as trigger
7. Stage the API
8. Make a request to that endpoint
9. Woohoo!


HashiCorp
Terraform


SERVERLESS

Terraform Lambda Config

From https://www.terraform.io/docs/providers/aws/r/lambda_function.html

Example Usage

```
resource "aws_iam_role" "iam_for_lambda" {
  name = "iam_for_lambda"

  assume_role_policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Effect": "Allow",
      "Sid": ""
    }
  ]
}
EOF
}

resource "aws_lambda_function" "test_lambda" {
  filename         = "lambda_function_payload.zip"
  function_name    = "lambda_function_name"
  role             = "${aws_iam_role.iam_for_lambda.arn}"
  handler          = "exports.test"
  source_code_hash = "${base64sha256(file("lambda_function_payload.zip"))}"
  runtime          = "nodejs4.3"

  environment {
    variables = {
      foo = "bar"
    }
  }
}
```

TWITTER: @natqab

EMAIL: natalieqabazard@gmail.com

GITHUB: https://github.com/natqab/ssr-react-lambda