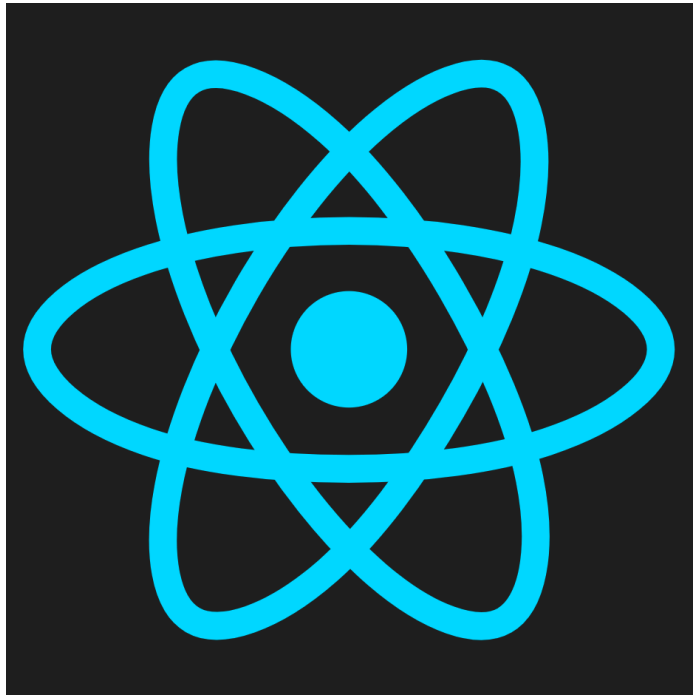


SERVERLESS SERVER-SIDE RENDERING

Hello 👋

I work with…

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

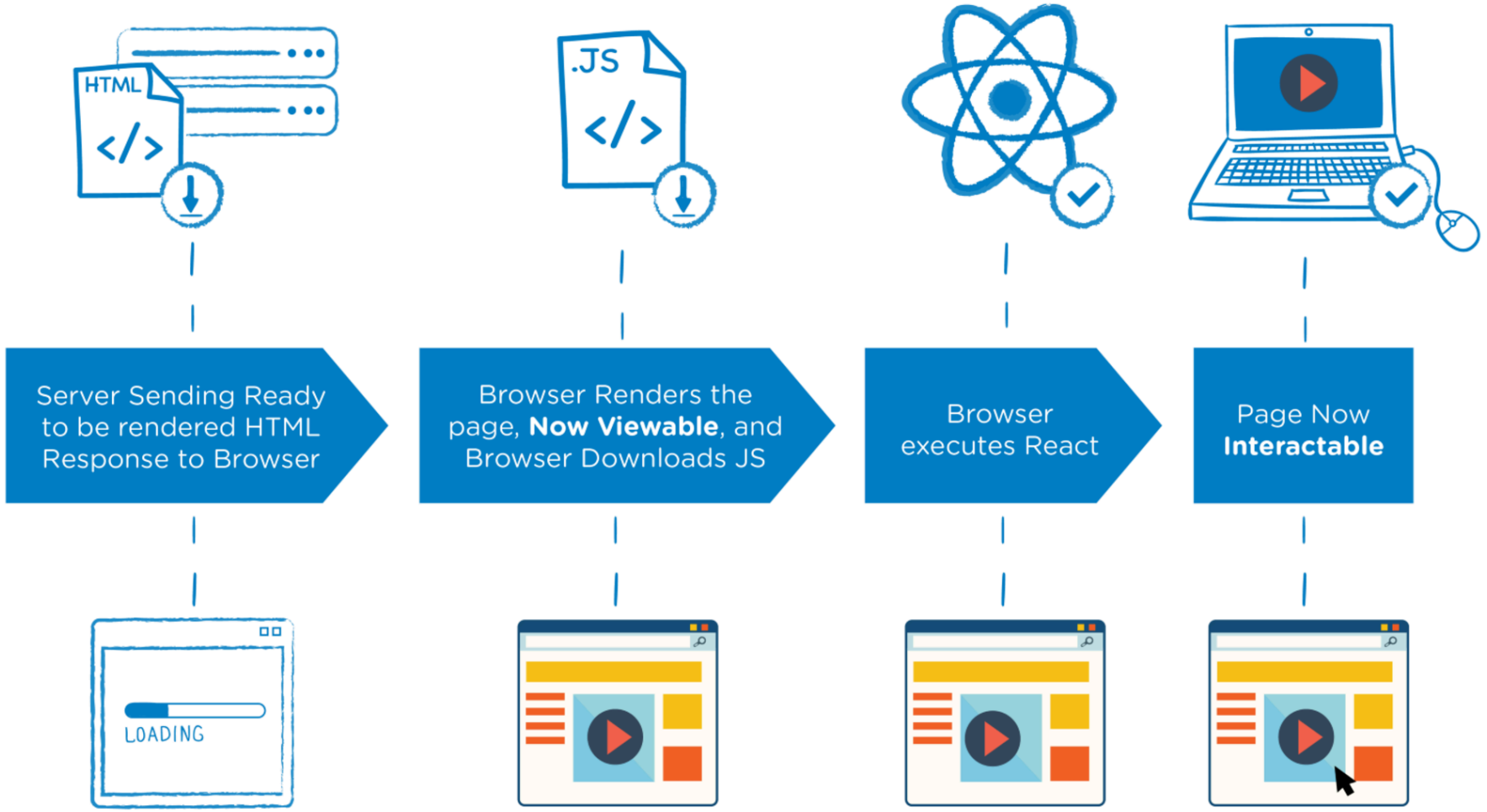

THIS IS ME WAITING
FOR YOUR SLOW WEBSITE TO LOAD

PERFORMANCE ⚡

USER EXPERIENCE ✨

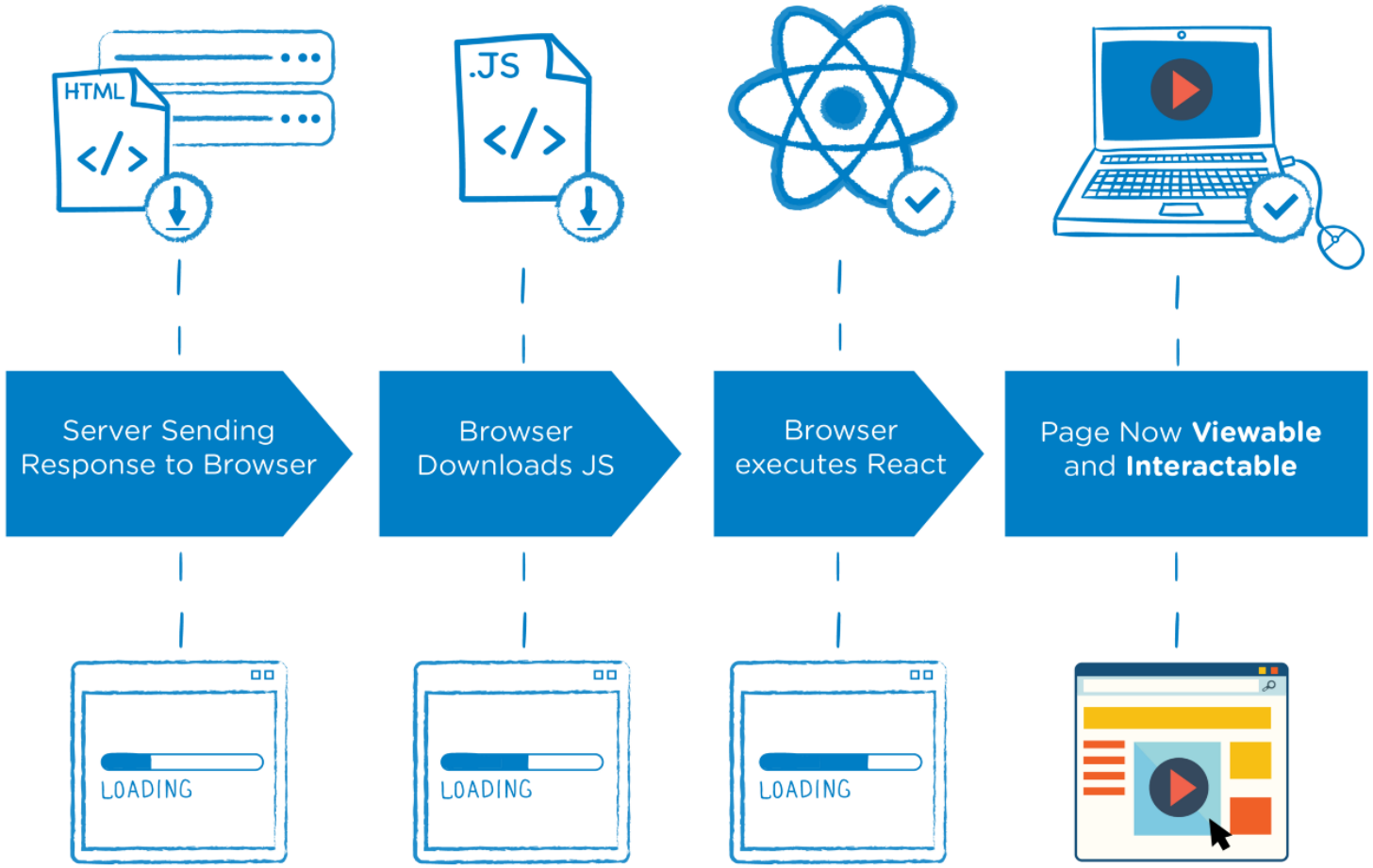
SEO 

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

Improved perceived performance

CONS:

Added complexity

TTFB slower than client-side rendering

```
1   import * as e from 'express';
2   import { Application } from 'express';
3   import * as ServerlessExpress from 'aws-serverless-express';
4   import { MarkupHandler } from './src/handlers/MarkupHandler';
5
6   let express: any = (<any>e).default || e;
7   const app: Application = express()
8
9   app.use('/', MarkupHandler);
10
```

```
1   import { createElement } from 'react';
2   import { renderToString } from 'react-dom/server';
3   import HelloWorld from '../components/HelloWorld';
4   import { Router, Request, Response, NextFunction } from 'express';
5   const MarkupHandler = Router();
6
7   MarkupHandler.get('/', function(request: Request, response: Response, next: NextFunction) {
8     const html = renderToString(createElement(HelloWorld as any));
9     request = request;
10    response.send(html);
11    next();
12  });
13
14  export { MarkupHandler };
```

```tsx
import * as React from 'react';

interface HelloWorldState {
  buttonClicked: boolean;
}

class HelloWorld extends React.Component<void, HelloWorldState> {
  constructor(props: any) {
    super(props);
    this.state = {
      buttonClicked: false
    };
  }

  render() {
    const backgroundStyles = {
      backgroundColor: 'black',
      height: '100%',
      width: '100%'
    };
    const textStyles = {
      color: 'white',
      textAlign: 'center',
      paddingTop: '25%',
      paddingBottom: '25%'
    };

    return (
      <div style={backgroundStyles}>
        <h1 style={textStyles}>hello boston!</h1>
      </div>
    );
  }
}

export default HelloWorld;
```

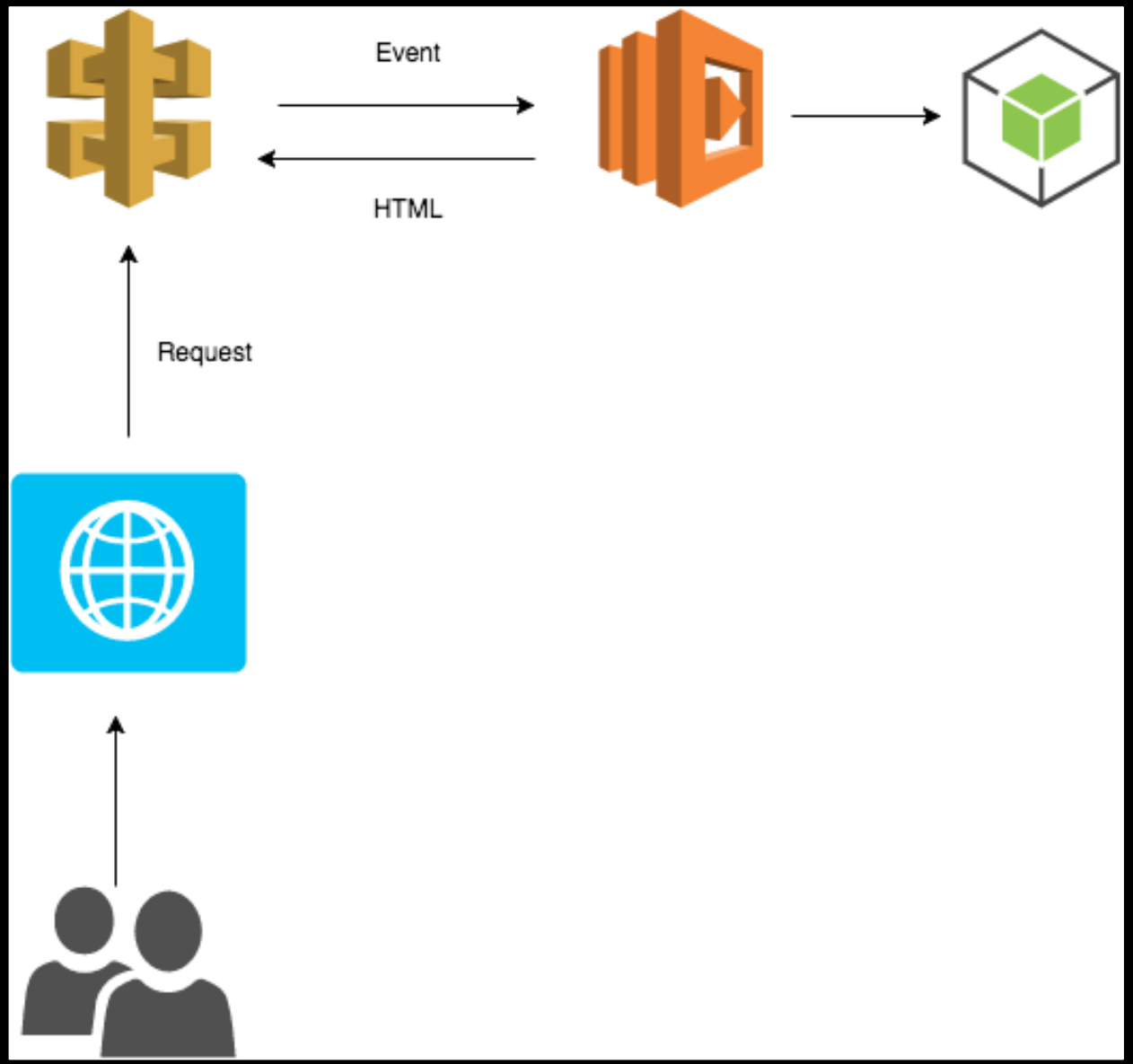
LAMBDA

- 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


Event
HTML
Request

How to Get Started

1. Create a Lambda function
2. Upload React app zip file to Lambda
3. Create an API
   - w/ Lambda Proxy integration
4. Create a GET method
5. Deploy API to a stage
6. Invoke stage URL
7. Woohoo!


HashiCorp
Terraform


SERVERLESS

Terraform Lambda Config

```
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"
    }
  }
}
```

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

twitter: @natqab

email: natalieqabazard@gmail.com

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

slides: https://github.com/natqab/react-boston-2017