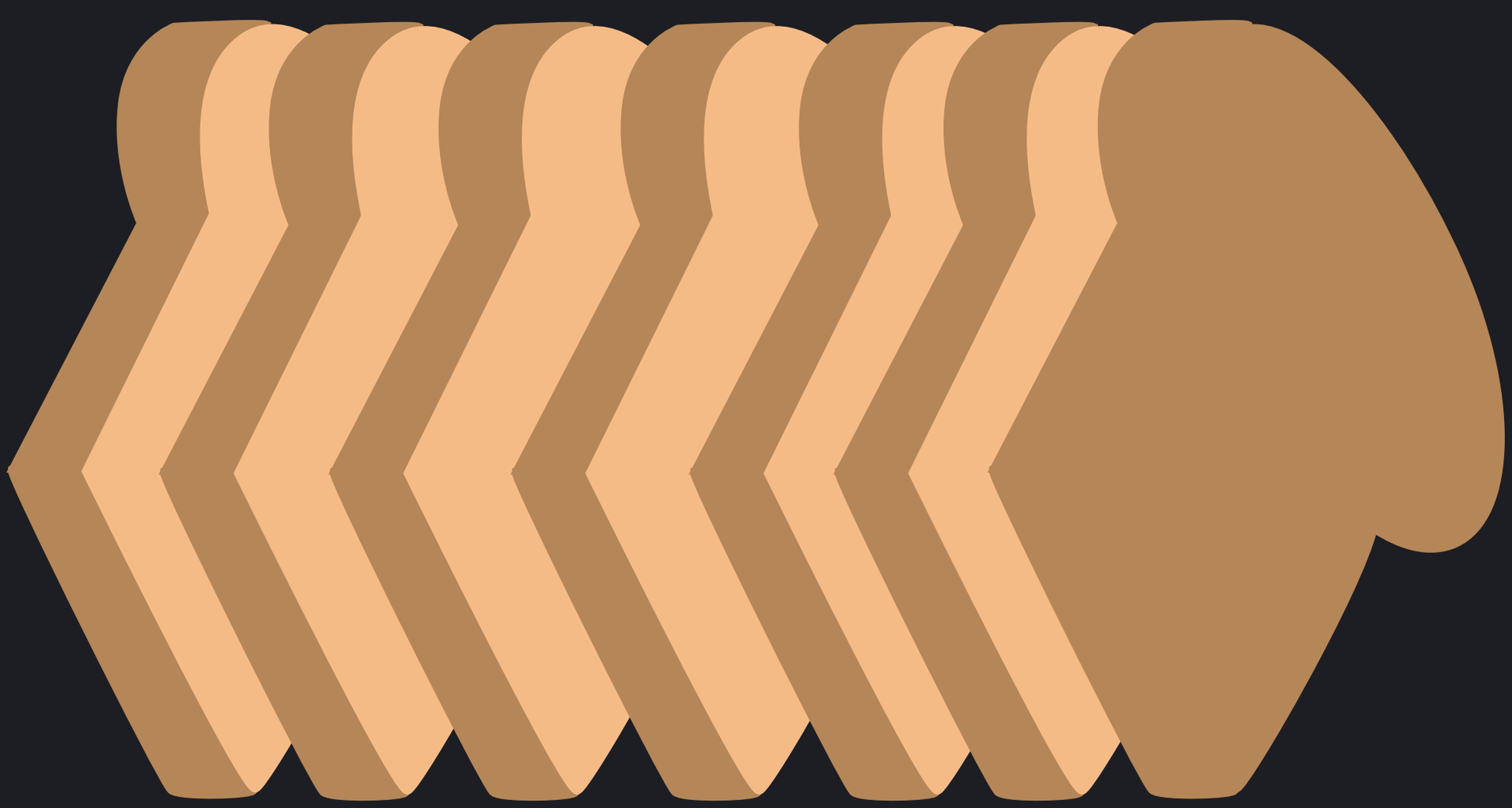
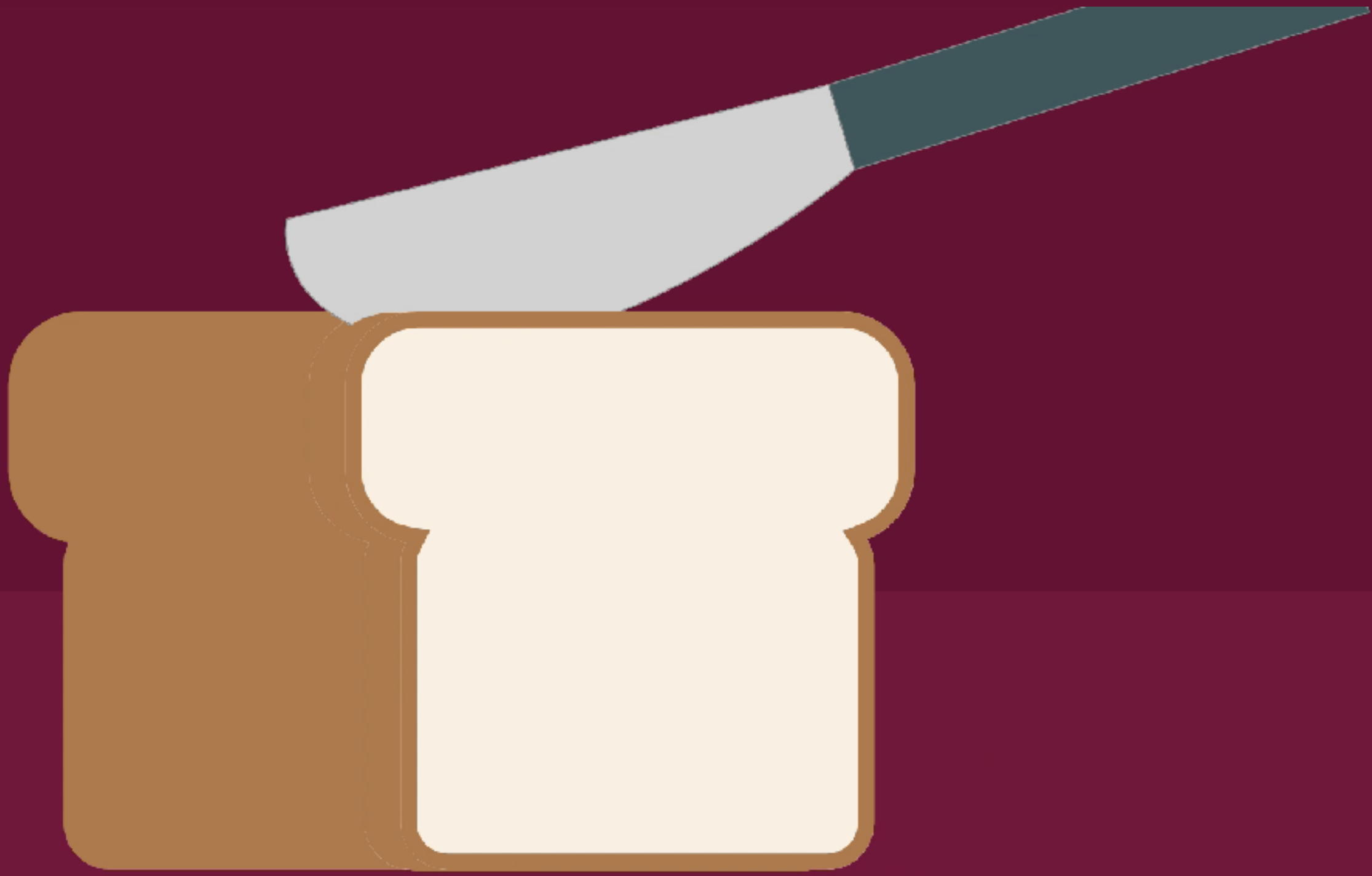
JAM stack

hits the spot

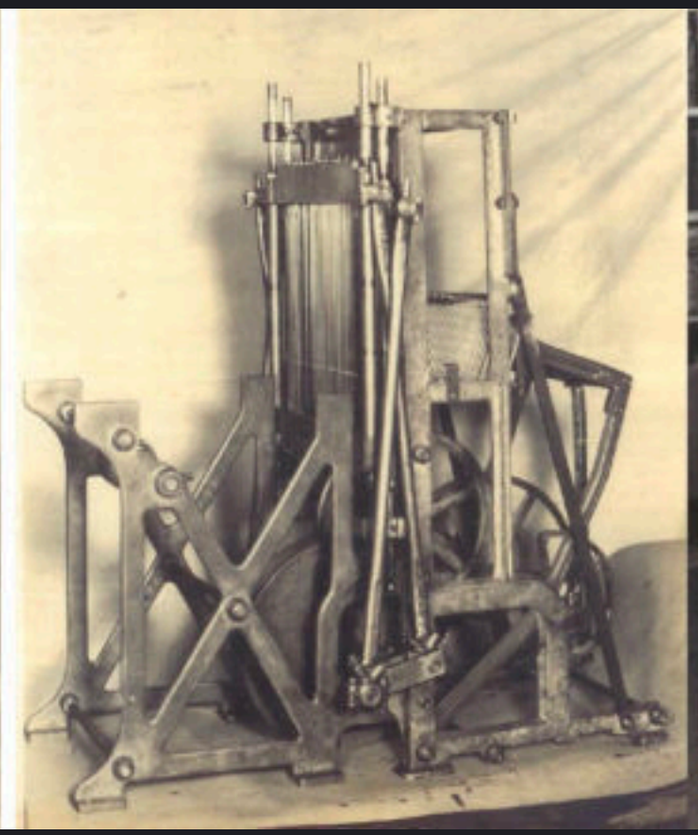
Divya Sasidharan

@shortdiv





https://codepen.io/shortdiv/pen/wvawjvX

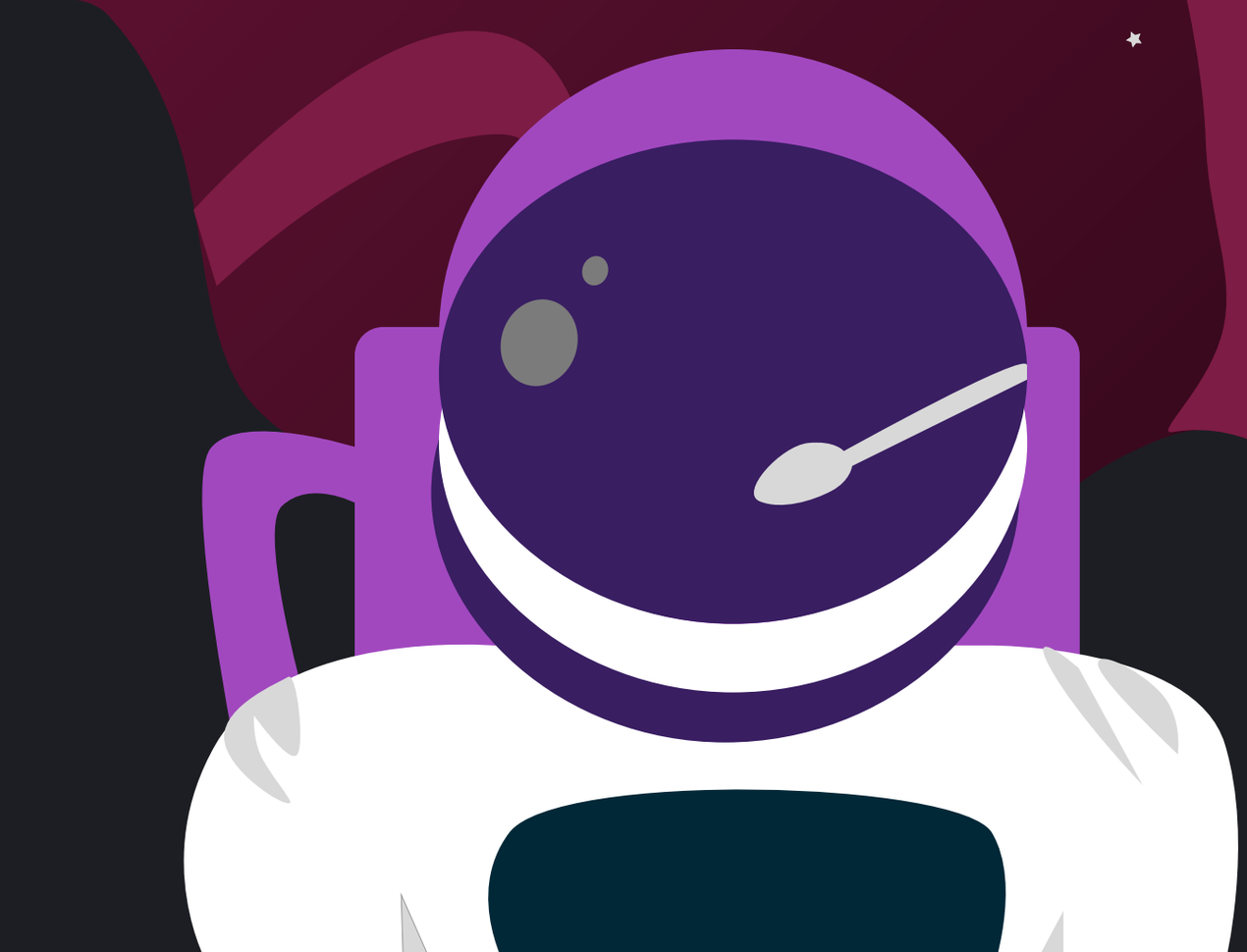



PIONEER BREAD
Colonial Bread

January 18, 1943

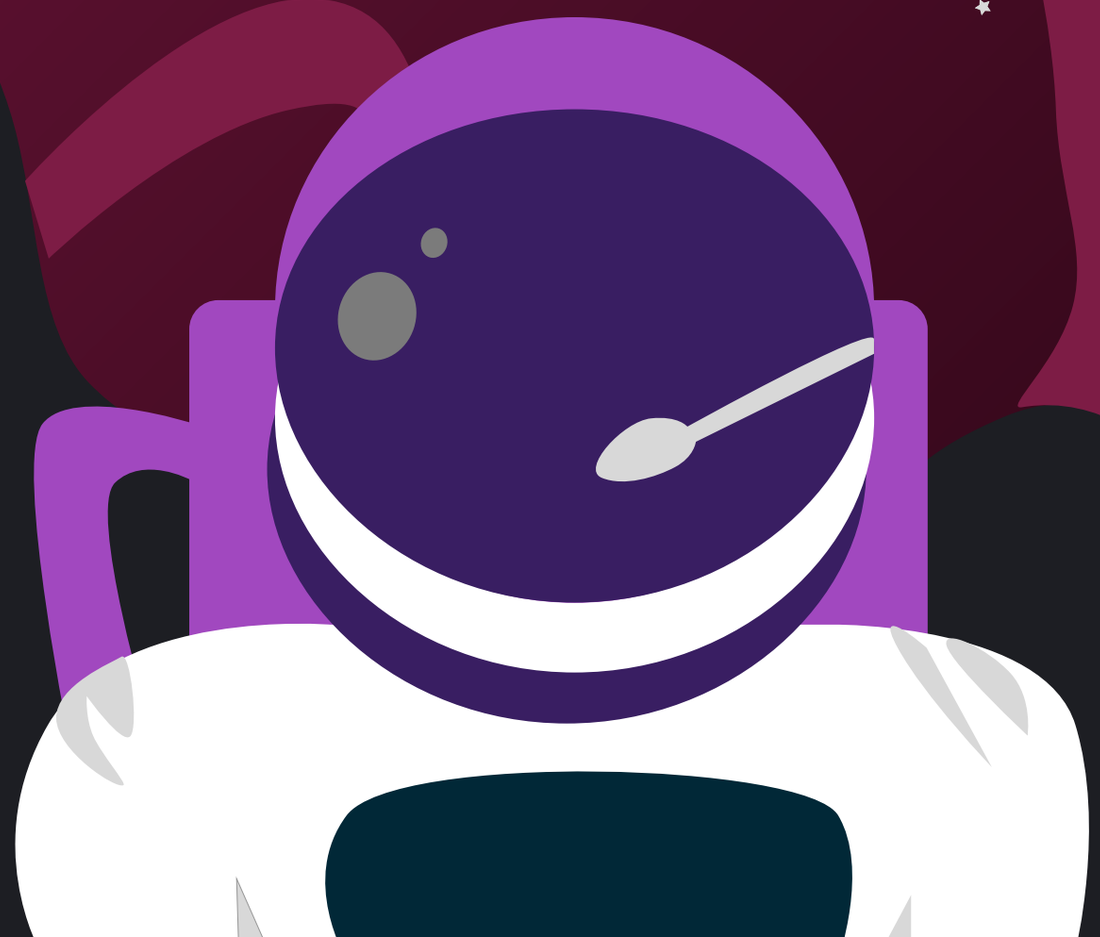
Get Out the Bread Knife

WASHINGTON, Jan. 18.—(AP)—Housewives will have to start slicing their own bread today. A government ban on bakery-sliced loaves for home consumption came simultaneously with an increase in flour prices which sent the old ceilings up an average of 10 per cent.

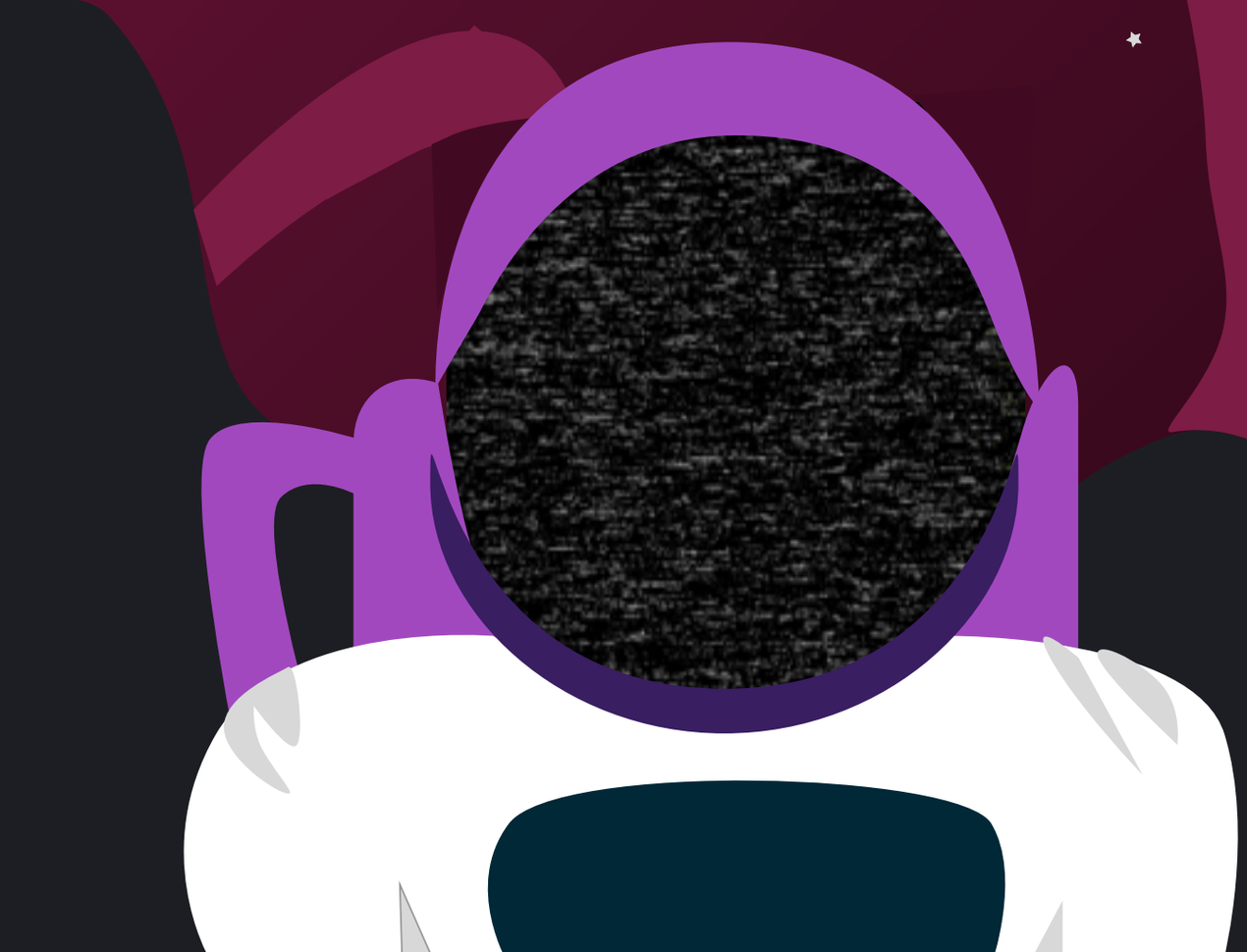


JAM stack

A modern architecture —
Create fast and secure sites and dynamic apps with JavaScript, APIs, and prerendered Markup

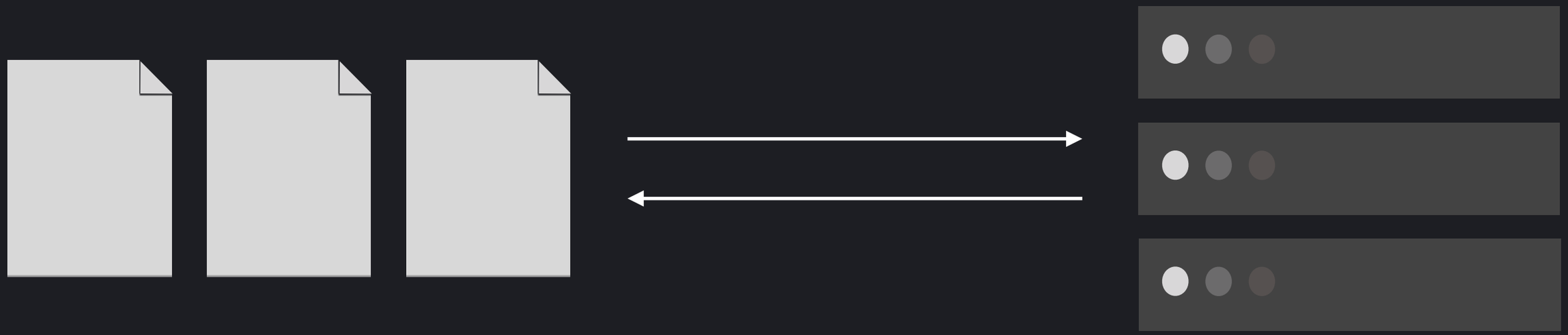


JAM stack



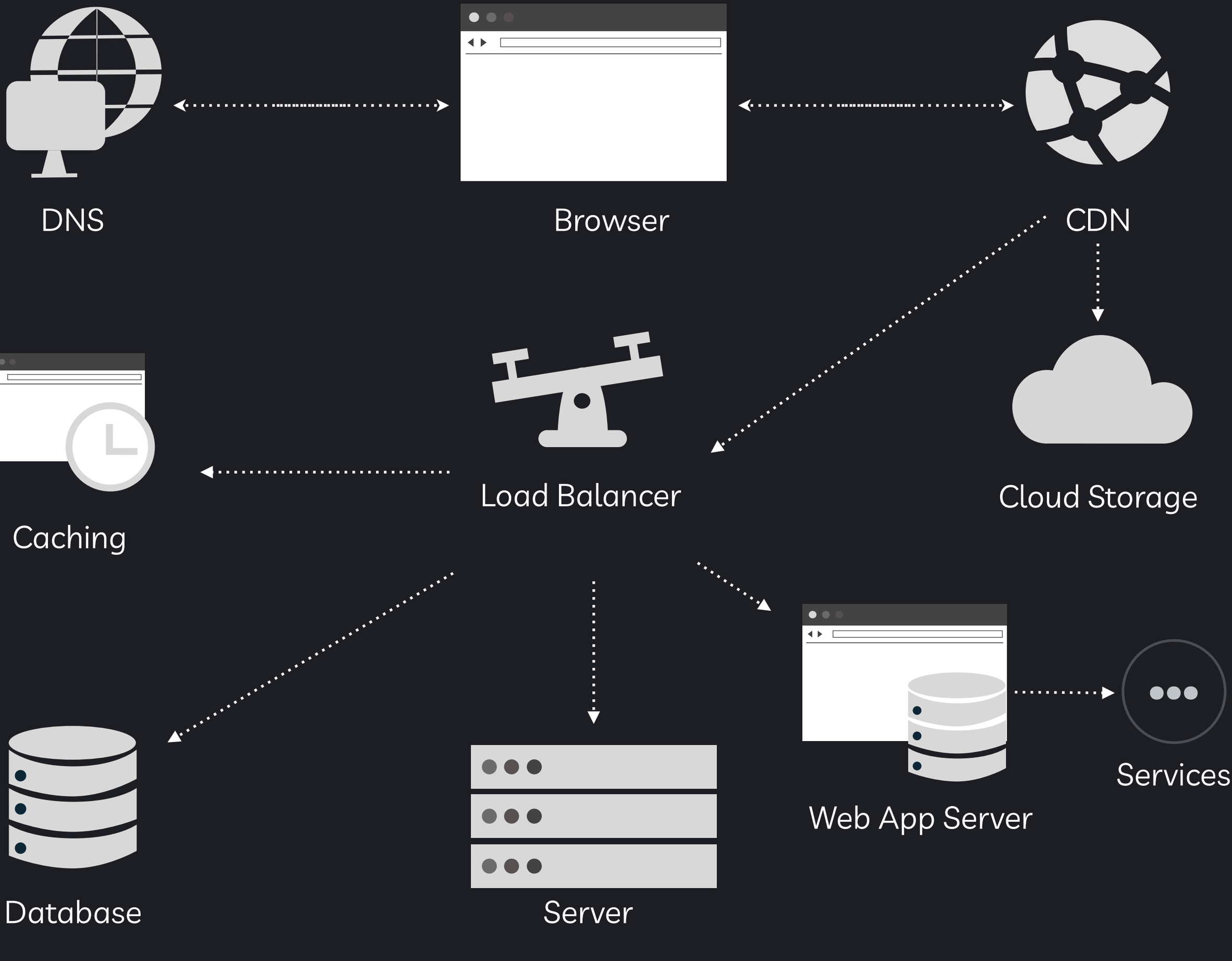
Static

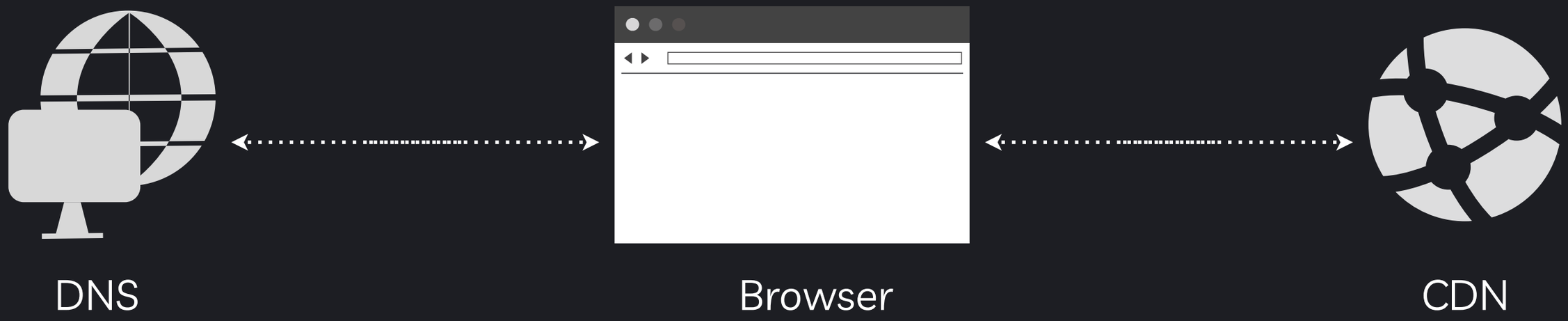
Dynamic

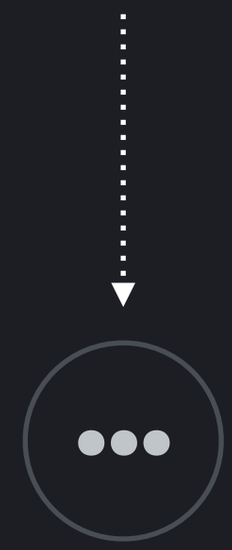


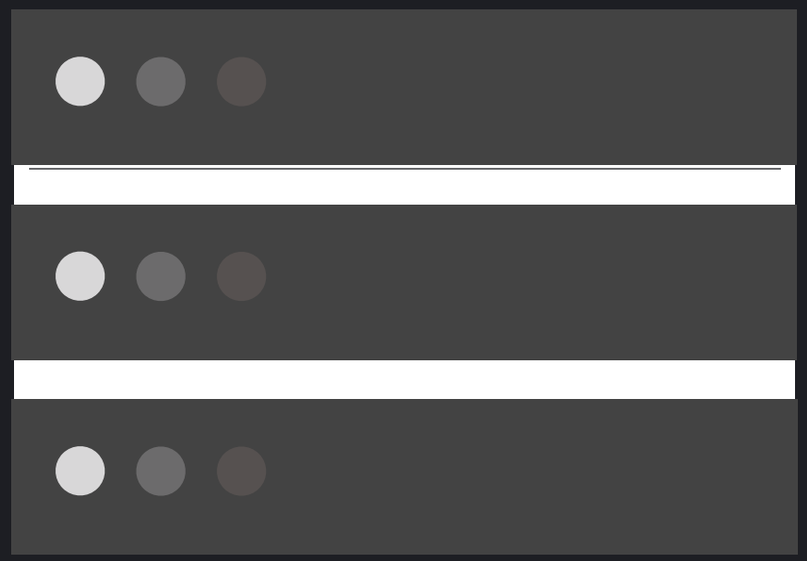
Markup

Server

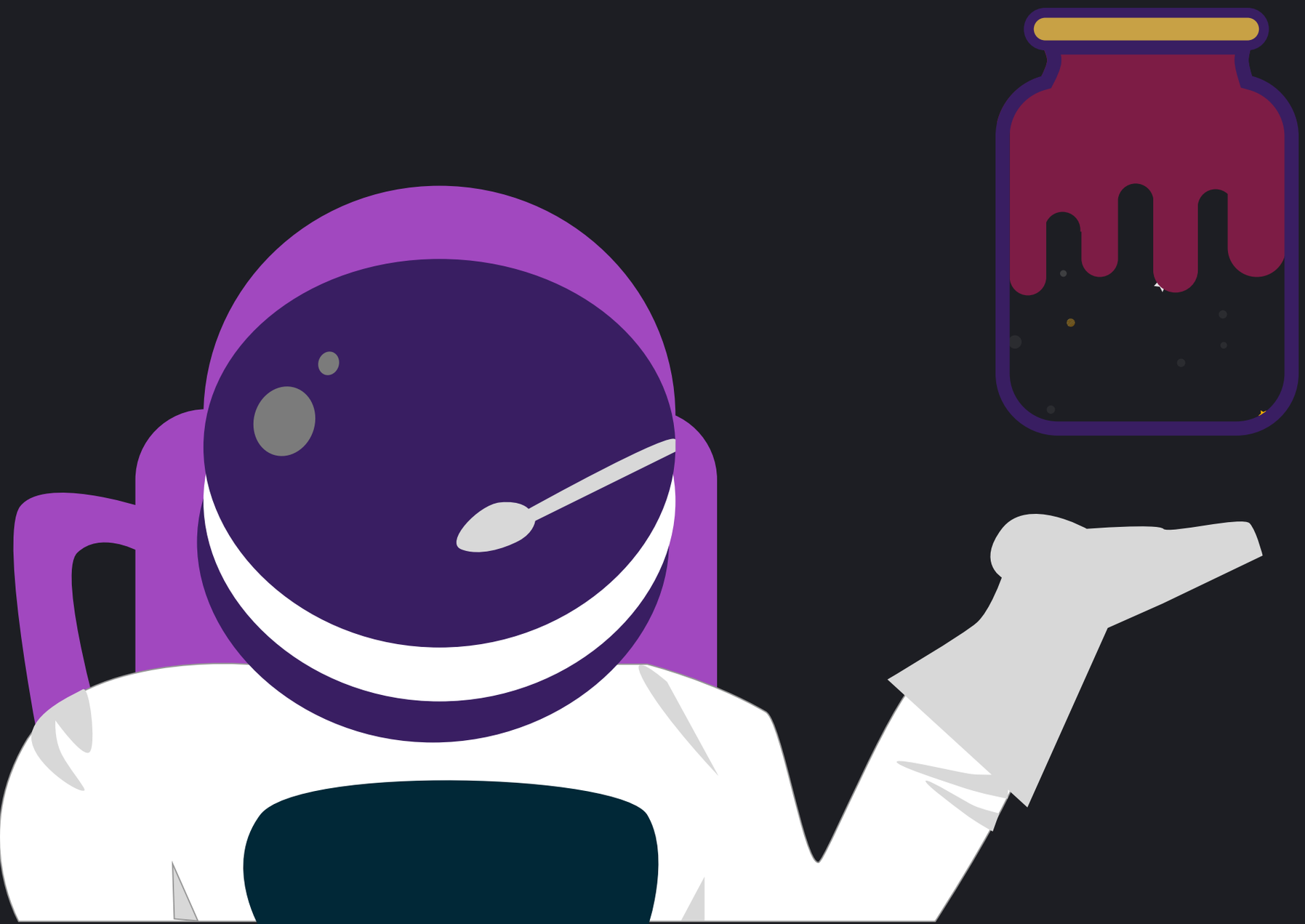

DNS
Browser
CDN
Caching
Load Balancer
Cloud Storage
Database
Server
Web App Server
Services


DNS
Browser
CDN


Services



Is this JAMstack?



- ✓ Pre-rendered Site/SPA
- ✗ Isomorphic SSR SPA
- ✗ Traditional CMS site
- ✗ Monolithic Fullstack App

Immutable Deploys

Build Automation

Event Triggers

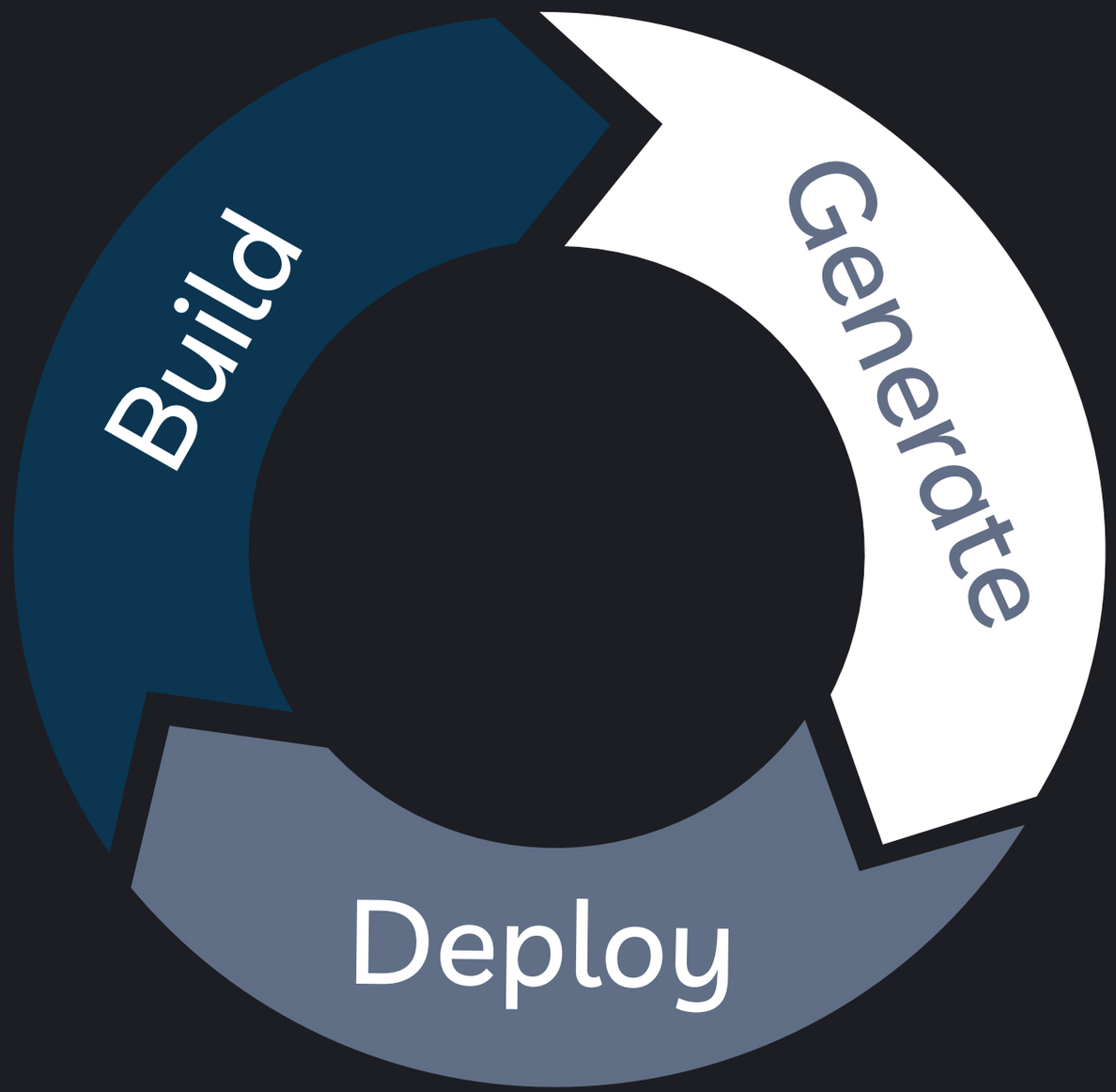
Serverless

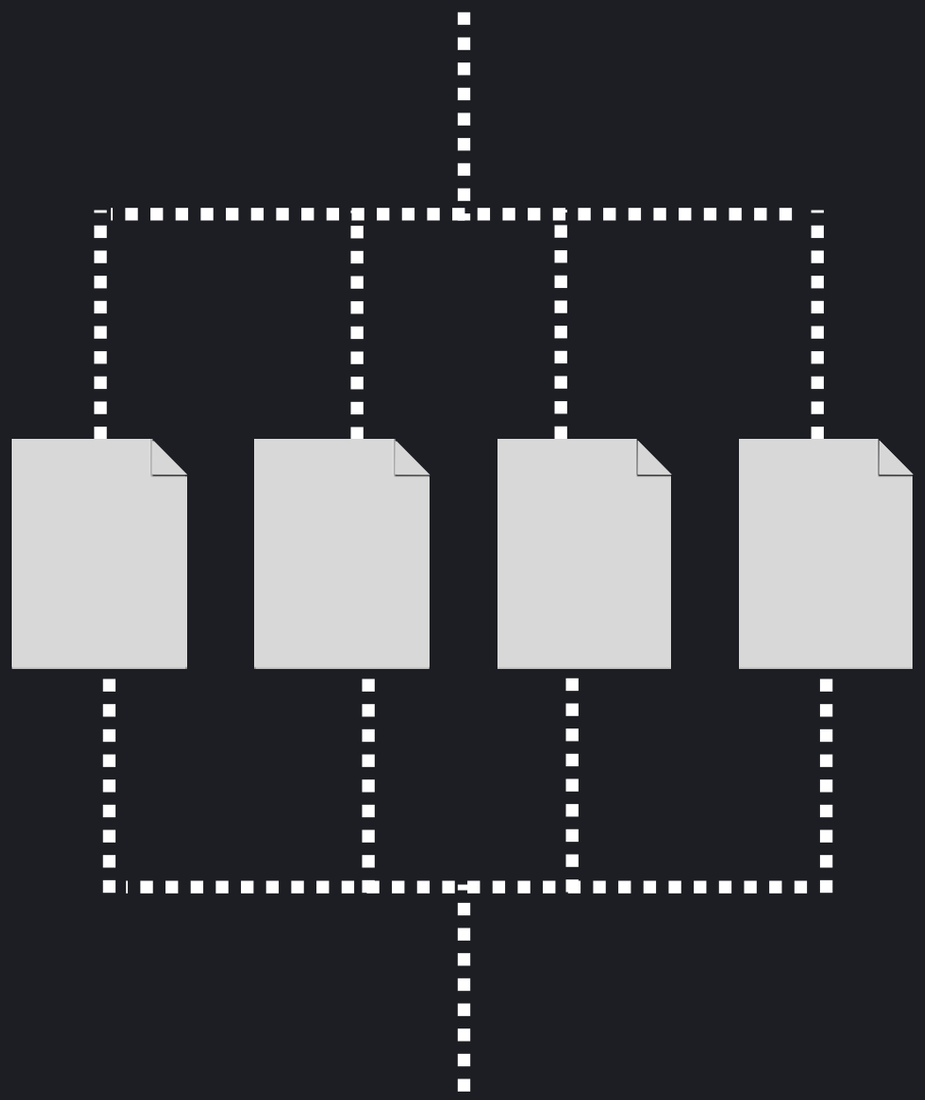
Immutable Deploys

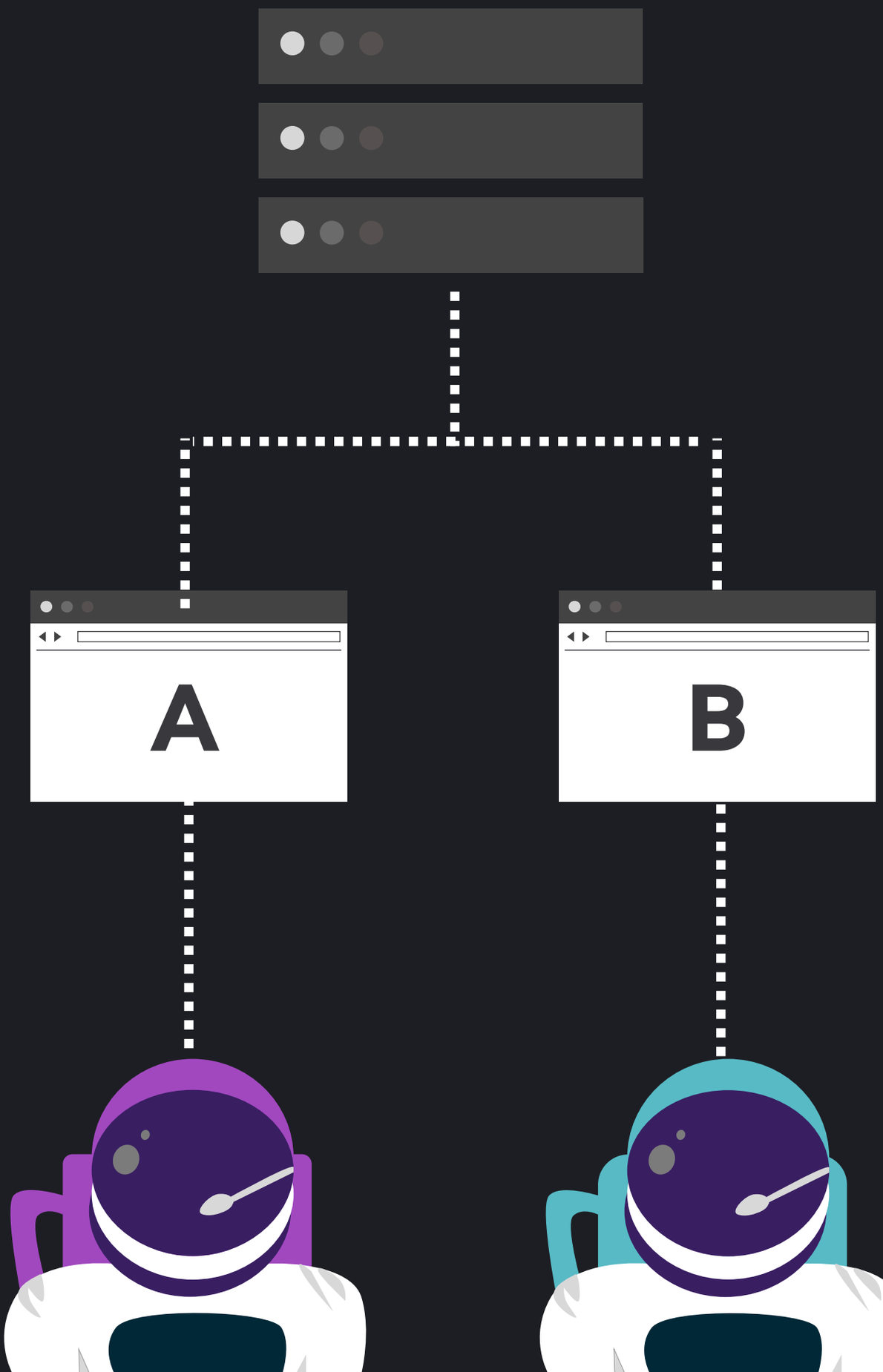
Build Automation

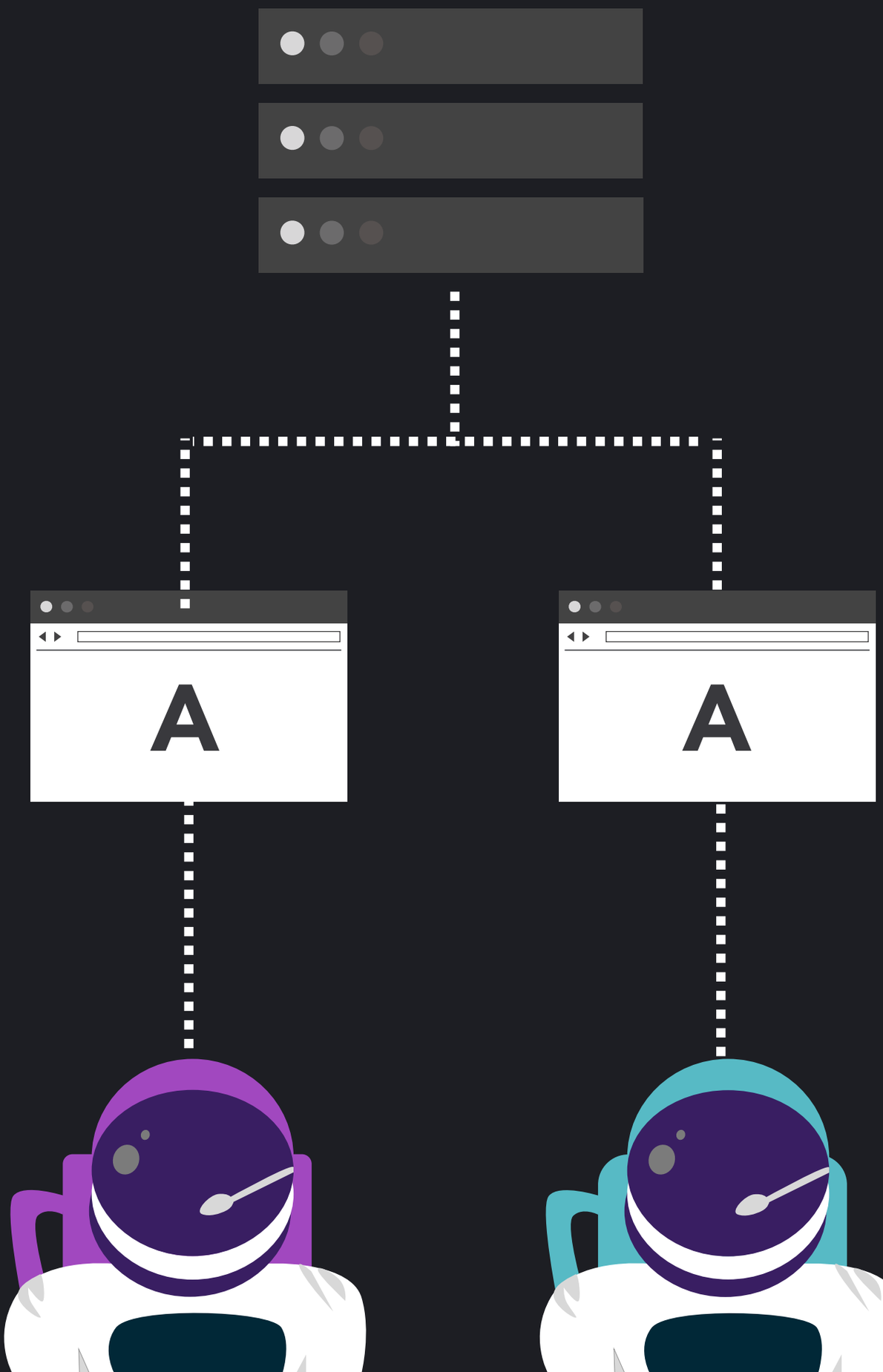
Event Triggers

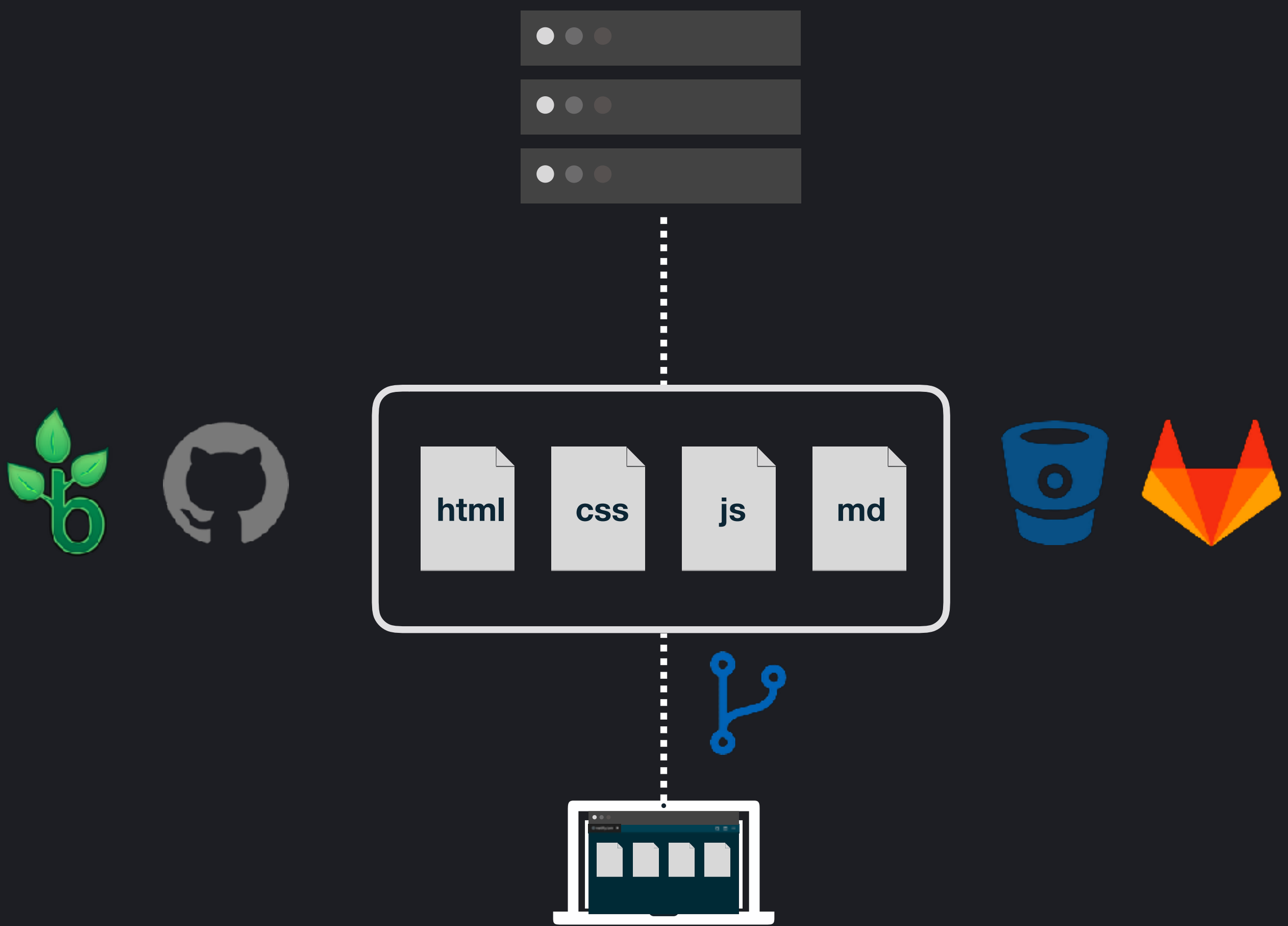
Serverless


Build
Generate
Deploy

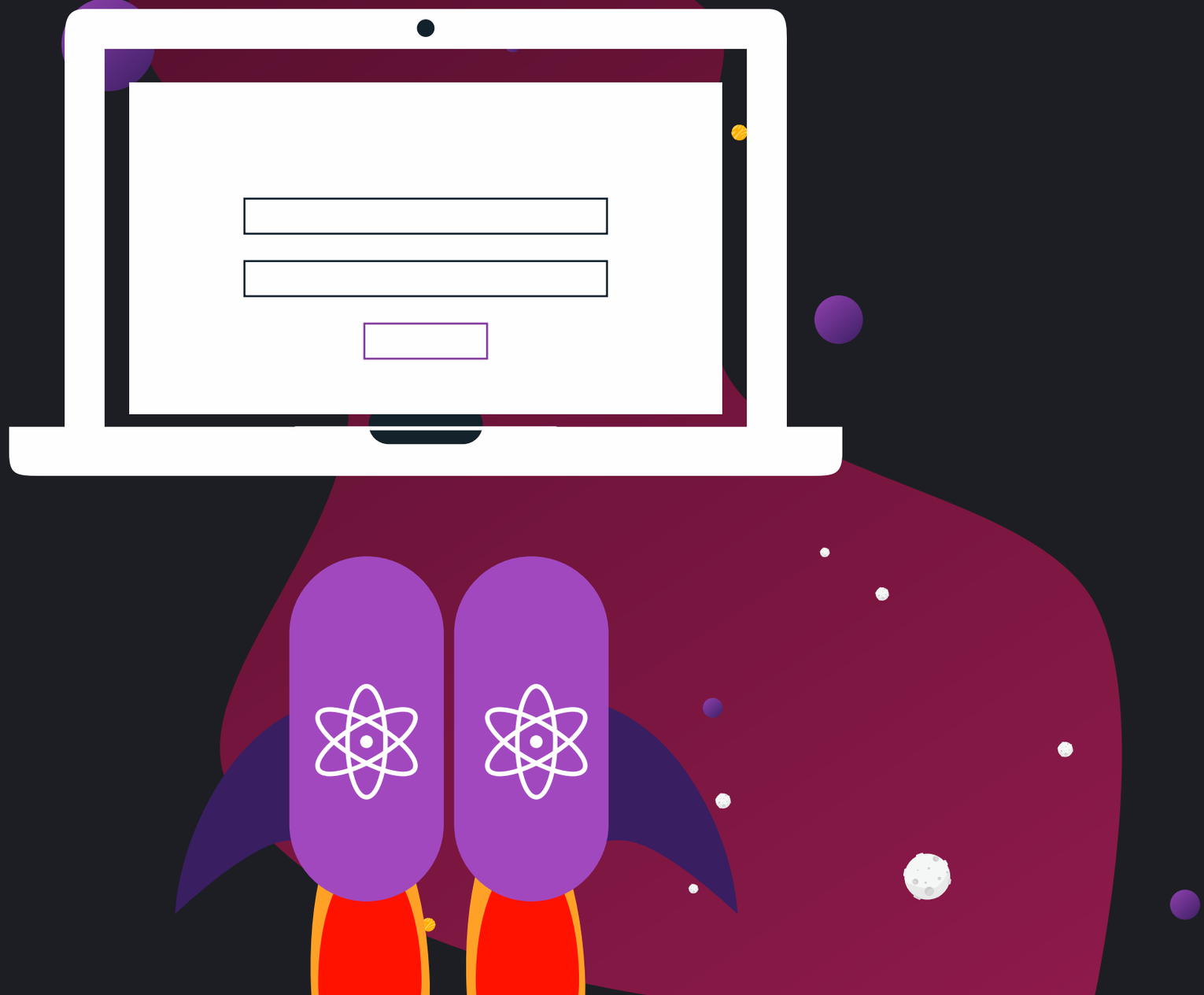



A
B


A
A


html
css
js
md

Atomic Deploys



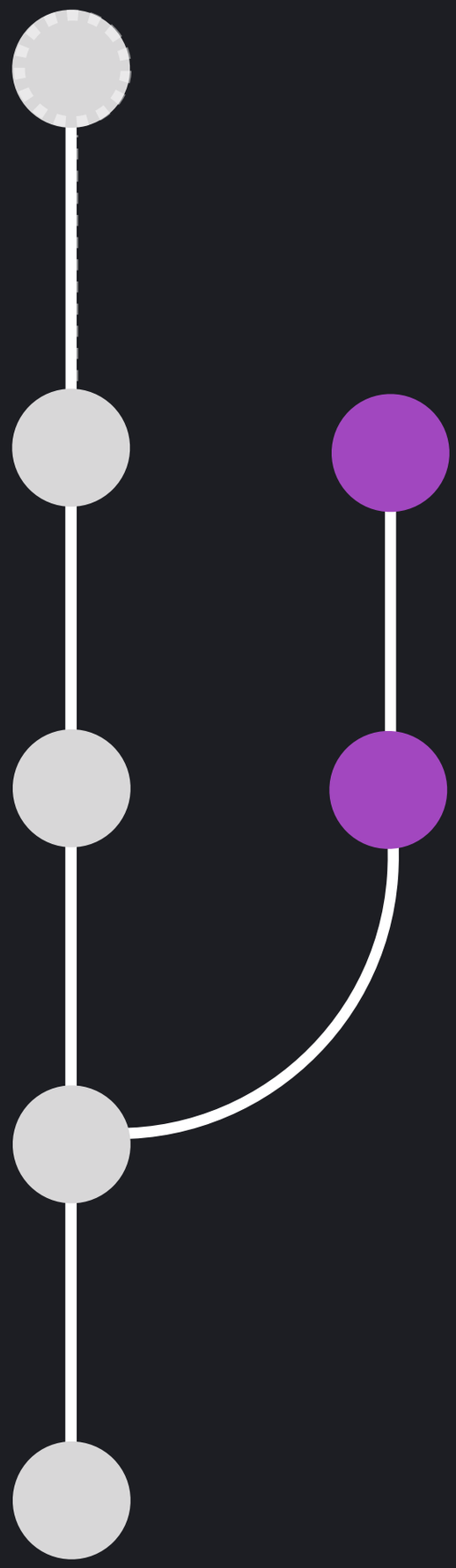
This works?

f**k I have no idea why this works

Why is this not working

Work in Progress

Initial

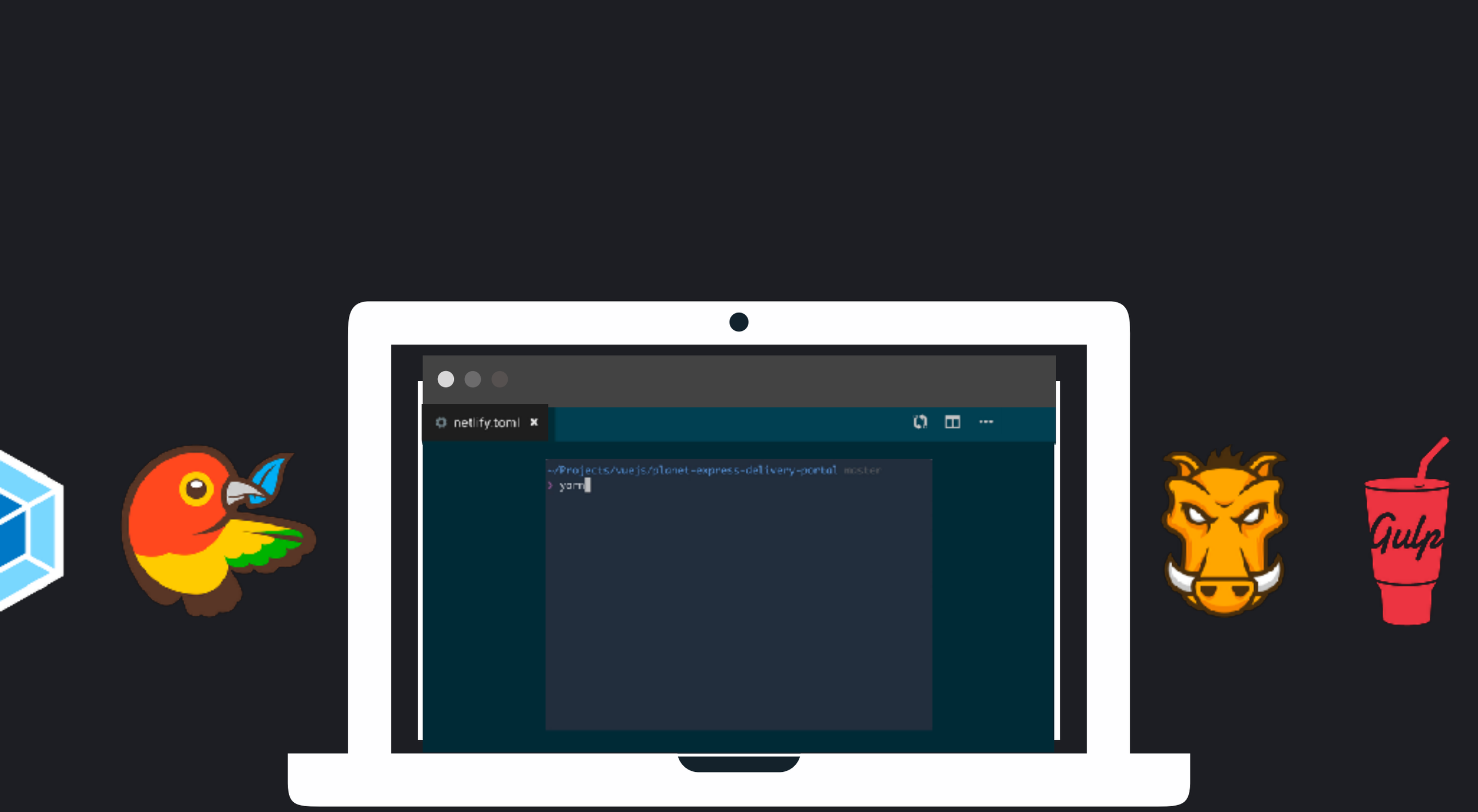


Immutable Deploys

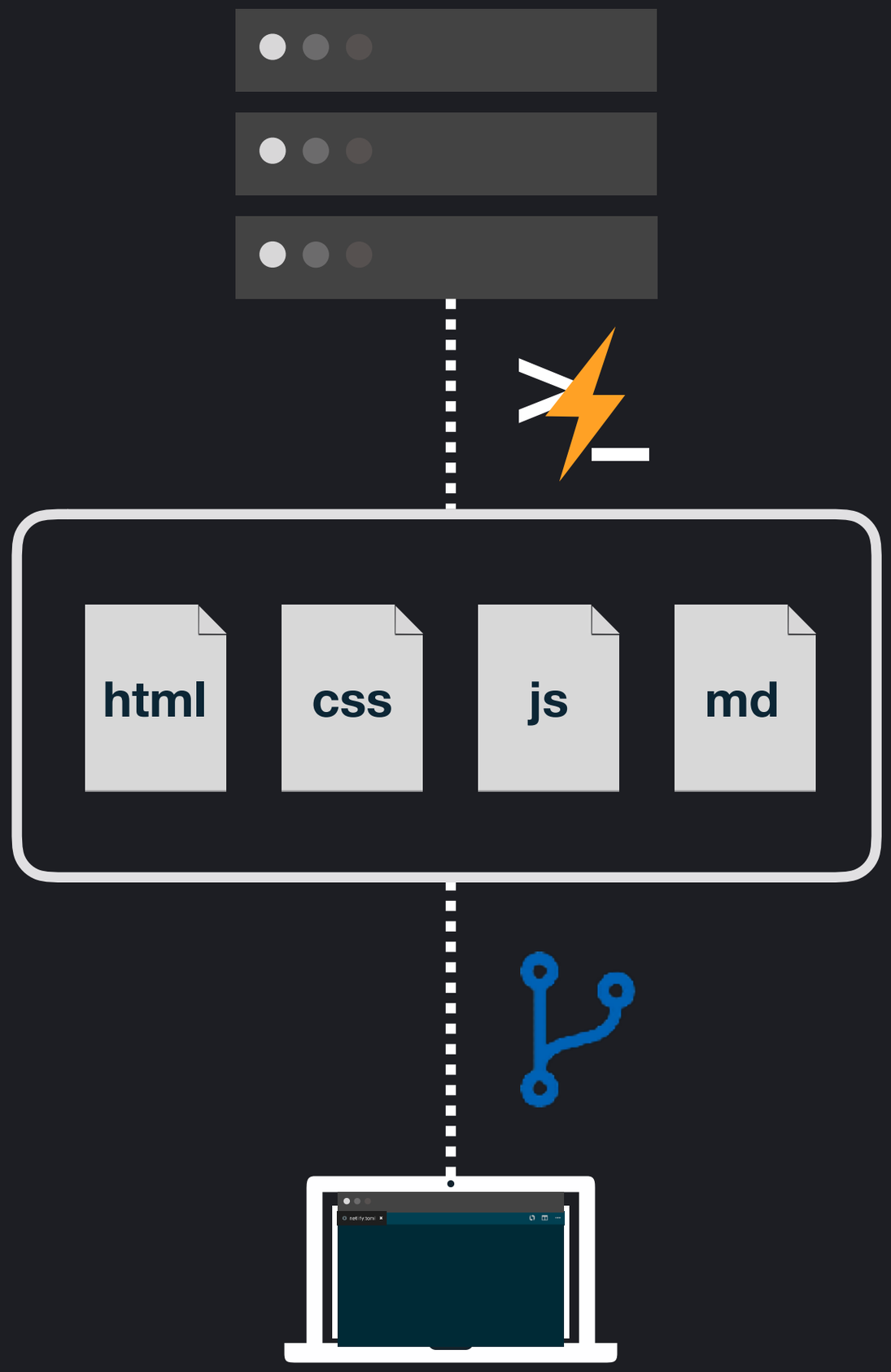
Build Automation

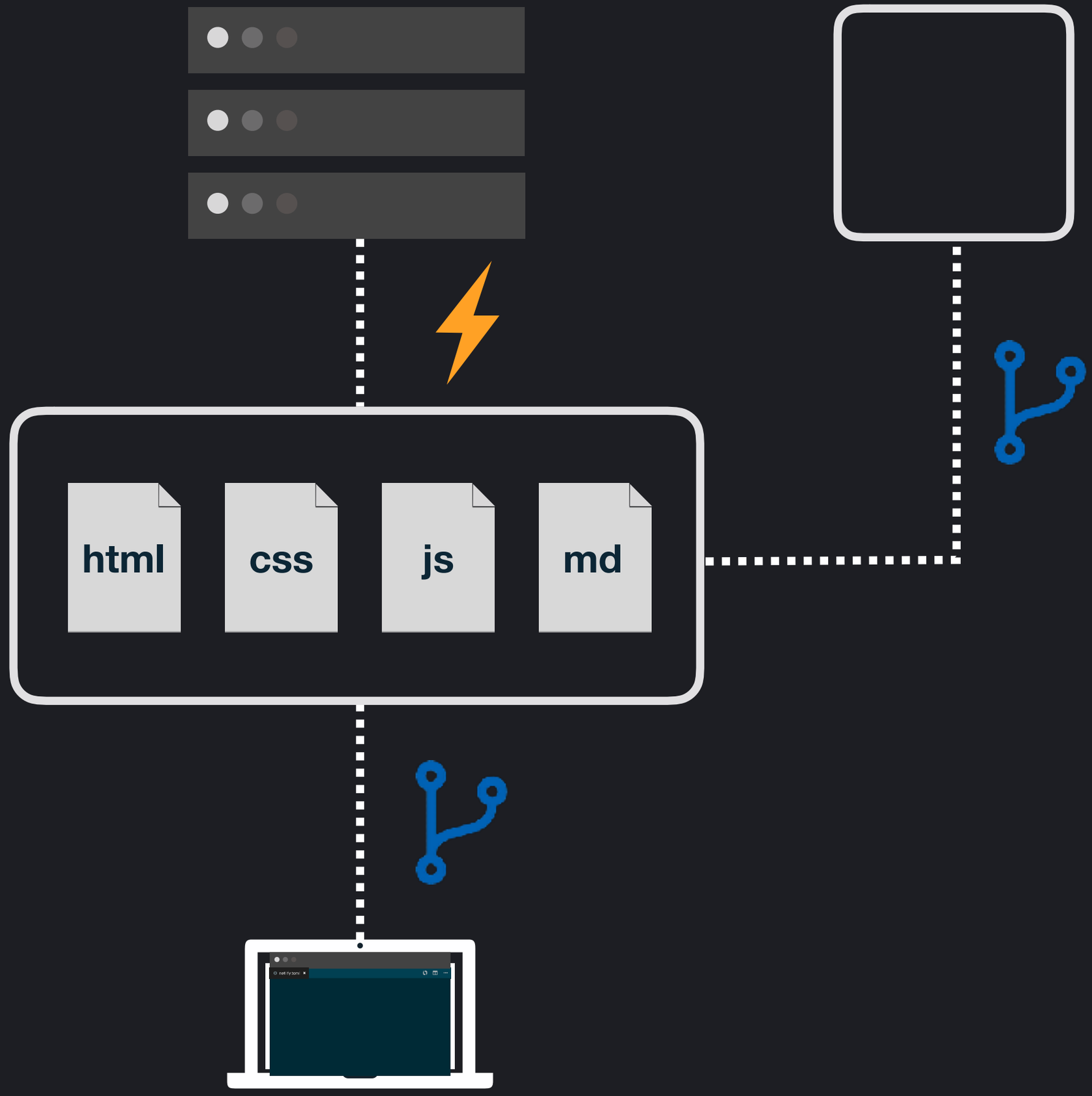
Event Triggers

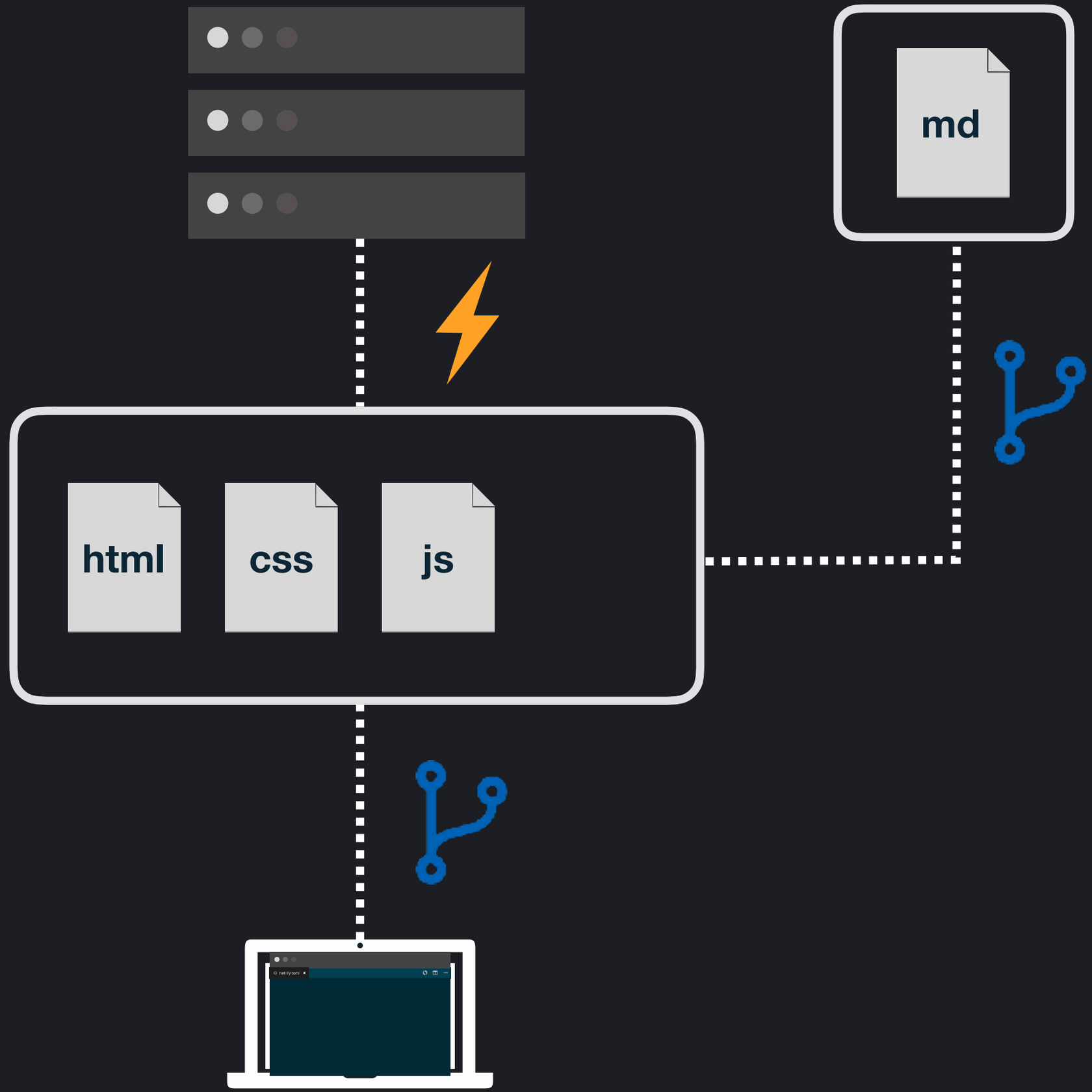
Serverless

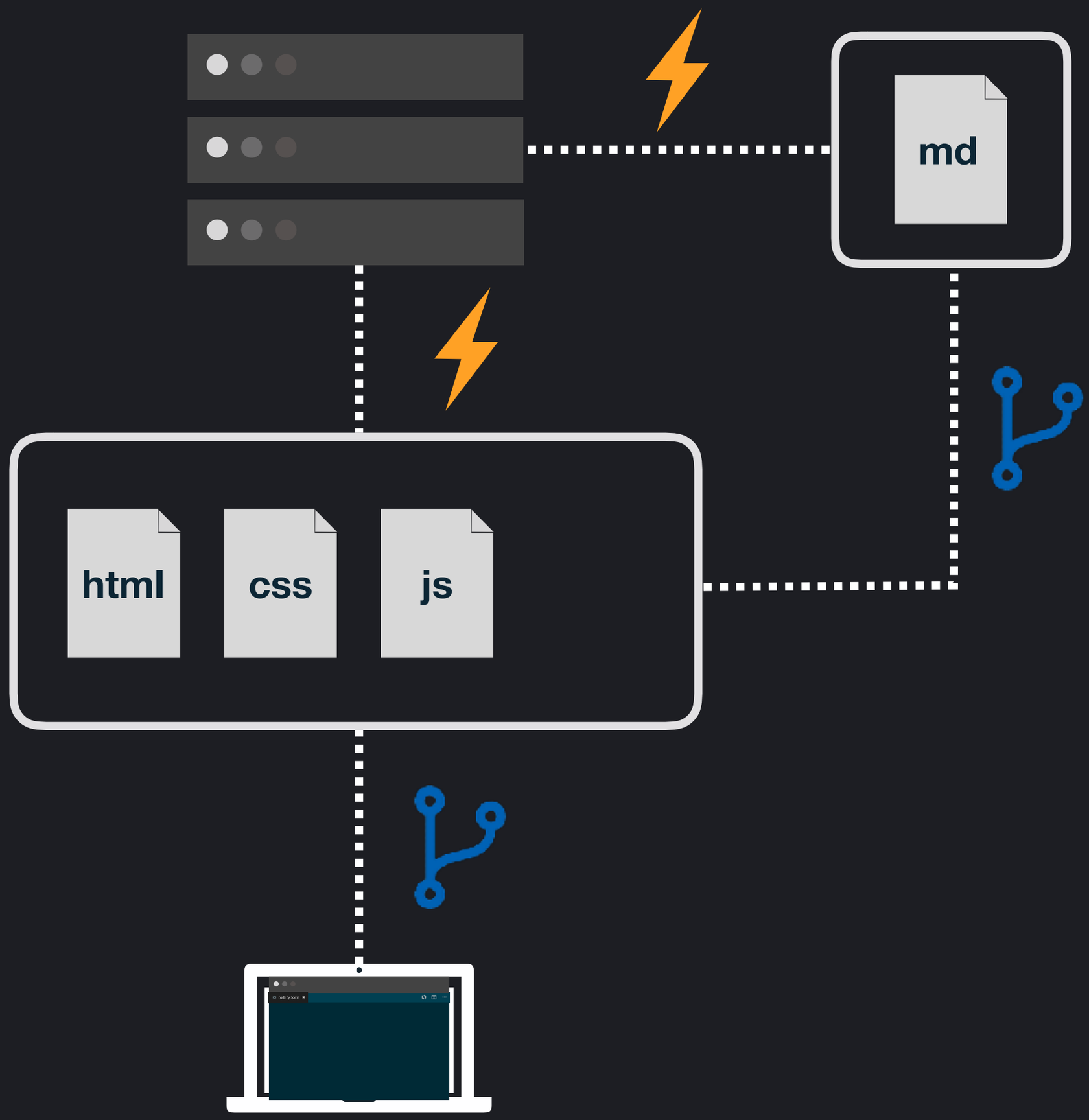

netlify.toml
~/Projects/vuejs/planet-express-delivery-portal master
> yarn
Gulp

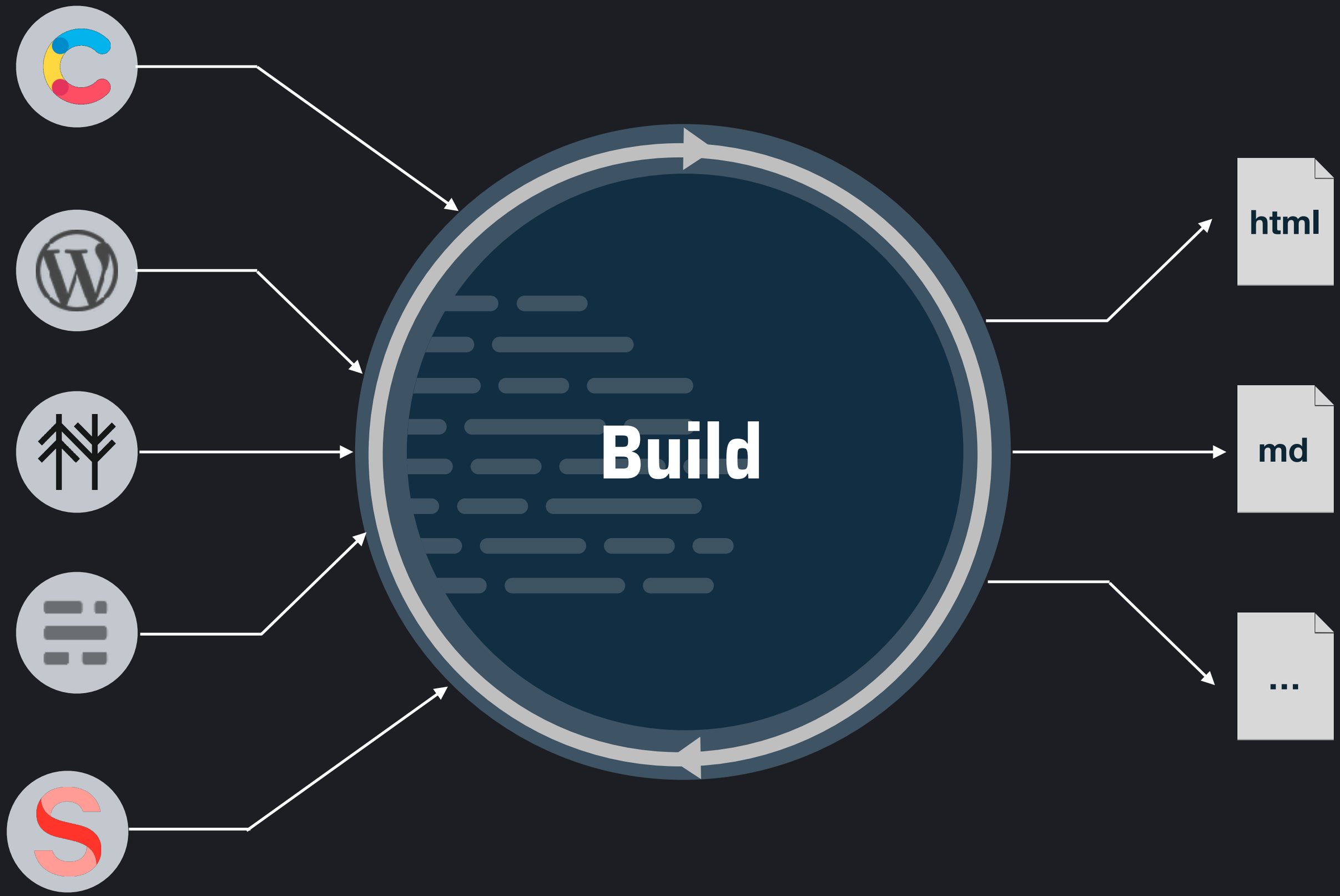

netlify.toml
~/Projects/vuejs/planet-express-delivery-portal master
> yarn
Gulp


html
css
js
md


html
css
js
md


md
html
css
js


md
html
css
js


Build
html
md
...

Immutable Deploys

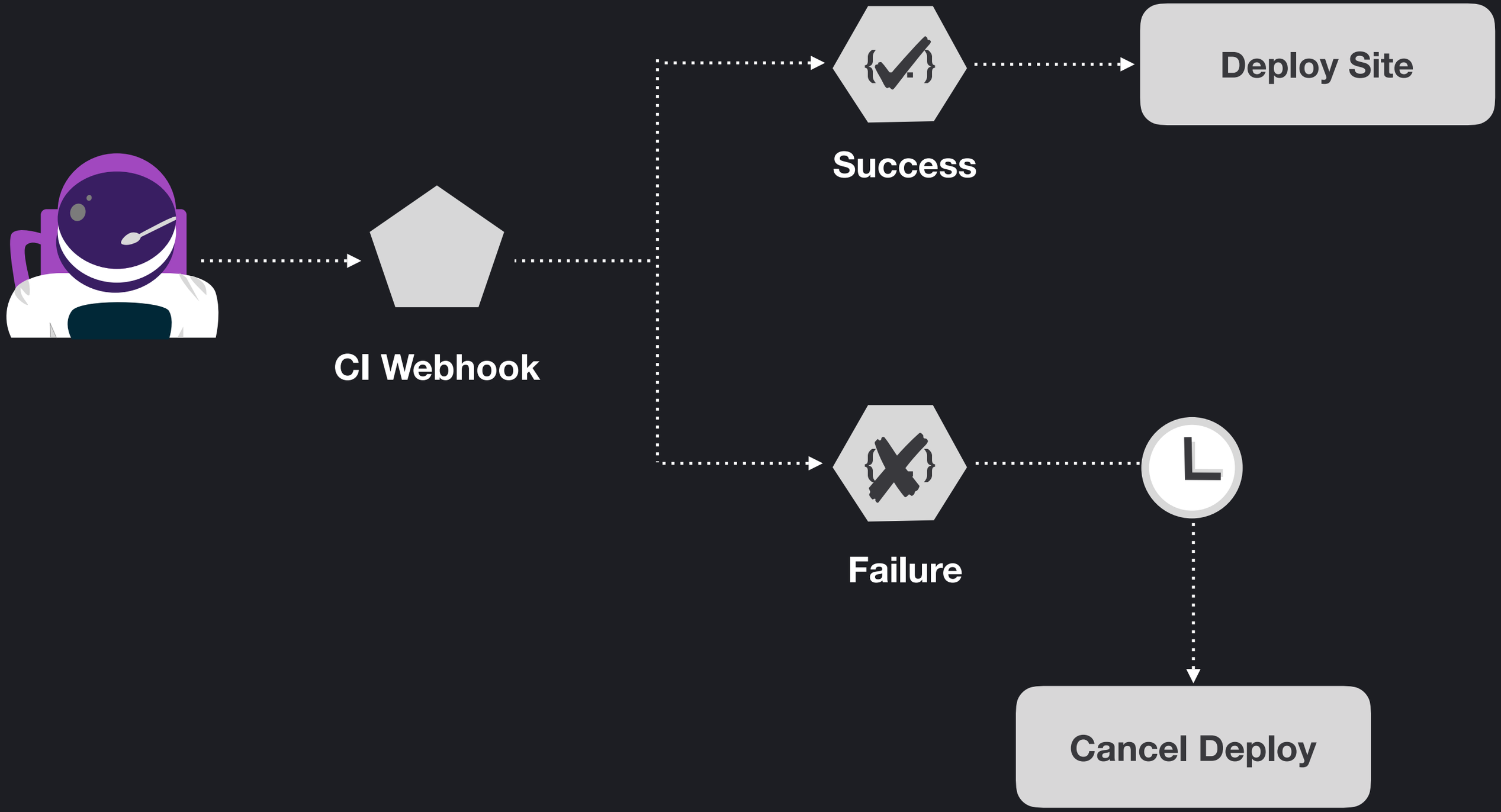
Build Automation

Event Triggers

Serverless

Web Hooks

If  then 


Deploy Site
Success
CI Webhook
Failure
Cancel Deploy

Git Activity

Split Test Events

Build/Deploy Events

Form Submissions

Content Updates

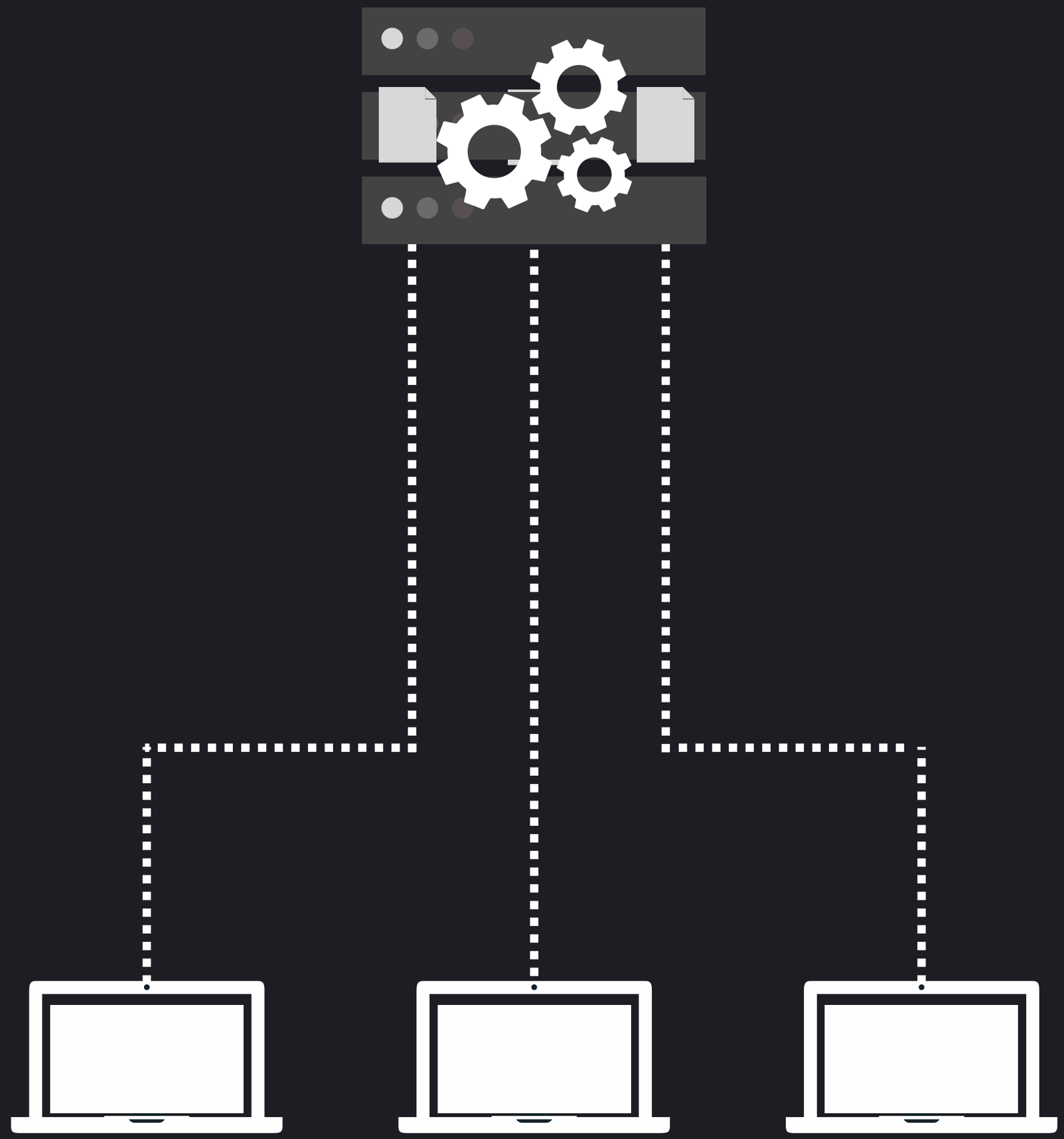
Auth Events

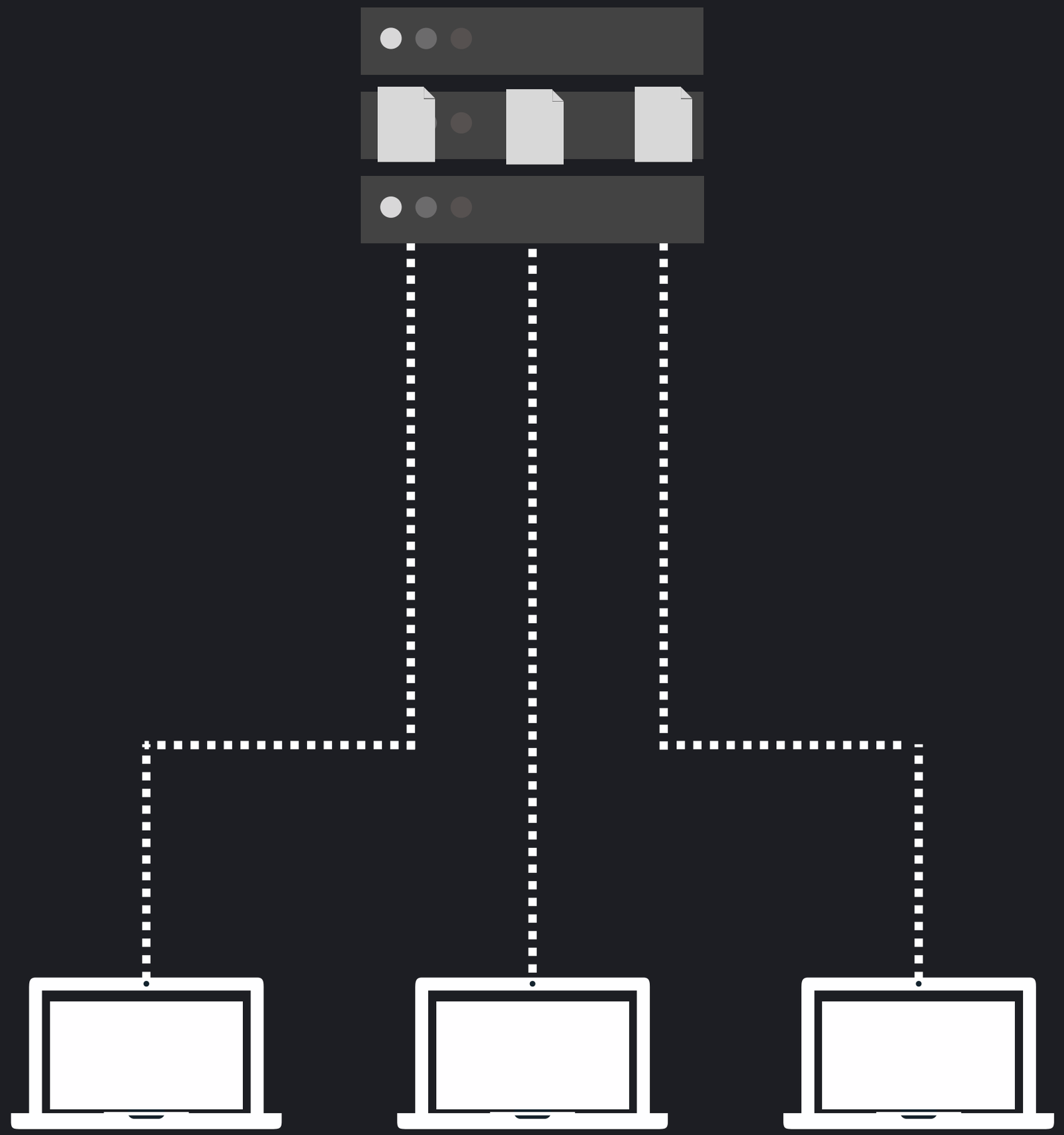
Immutable Deploys

Build Automation

Event Triggers

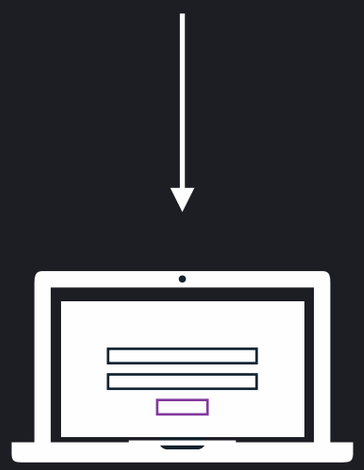
Serverless





Frontend Framework

Server



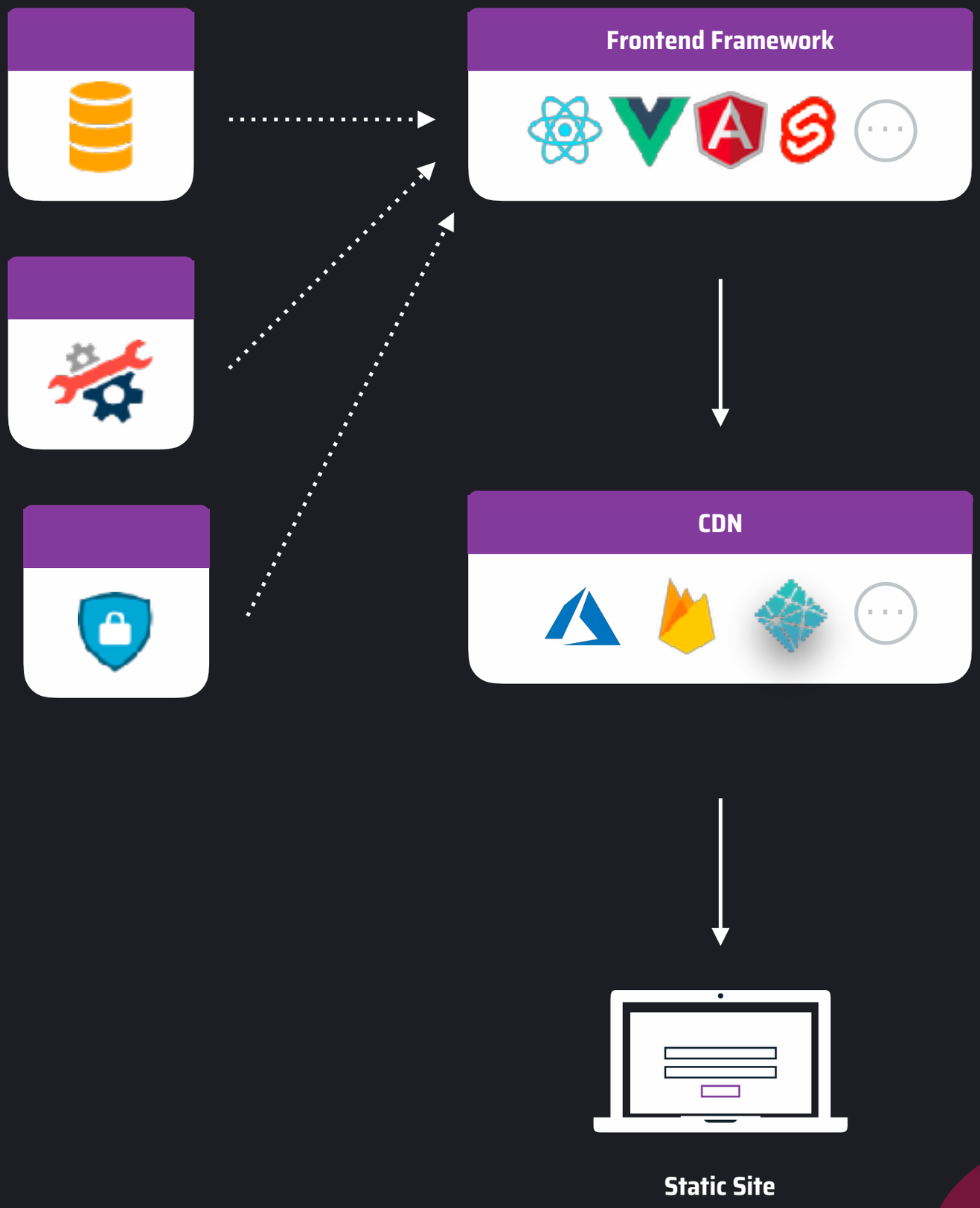
~~Static Site~~ Full Stack Application

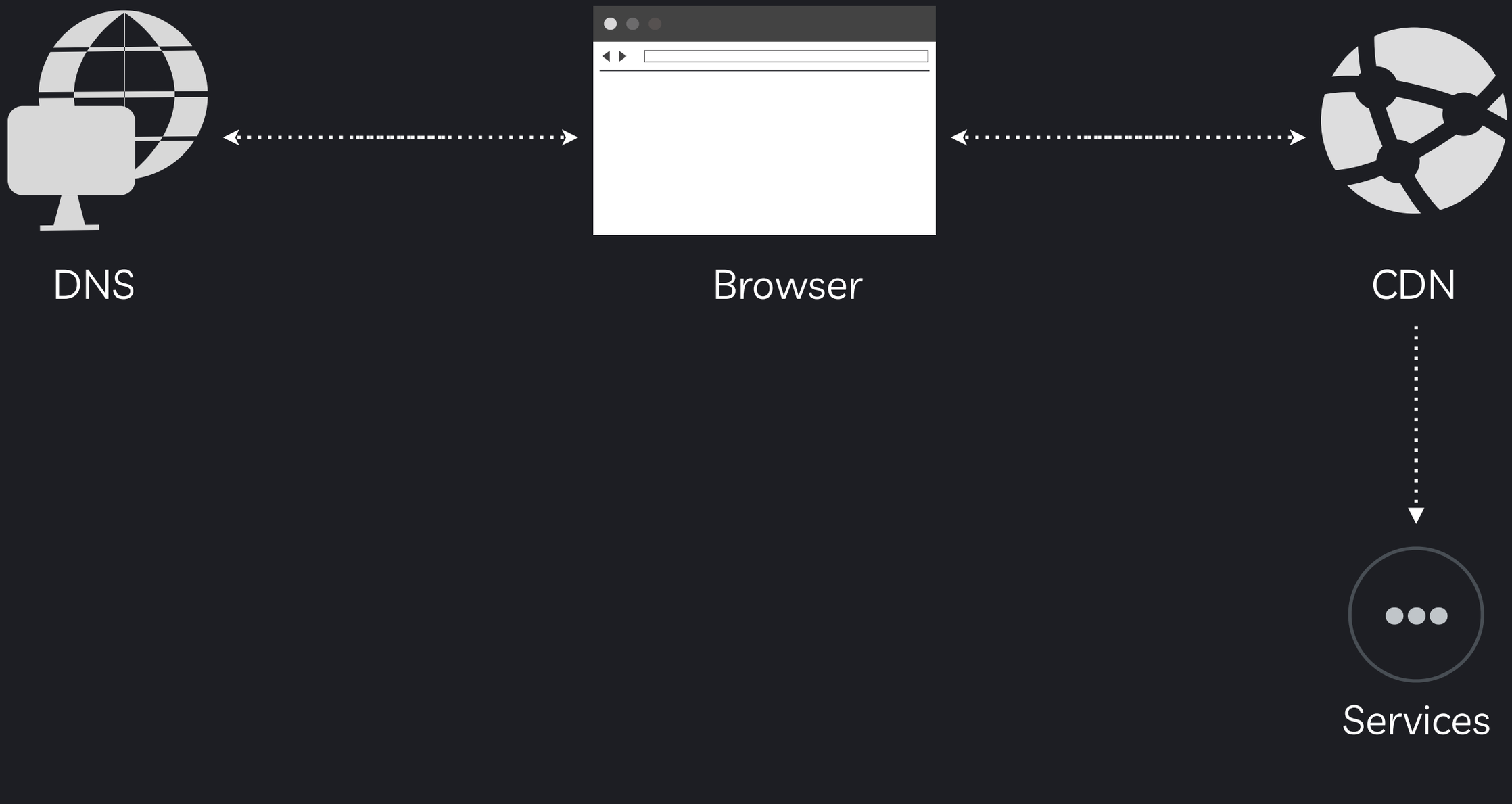
Backend Application

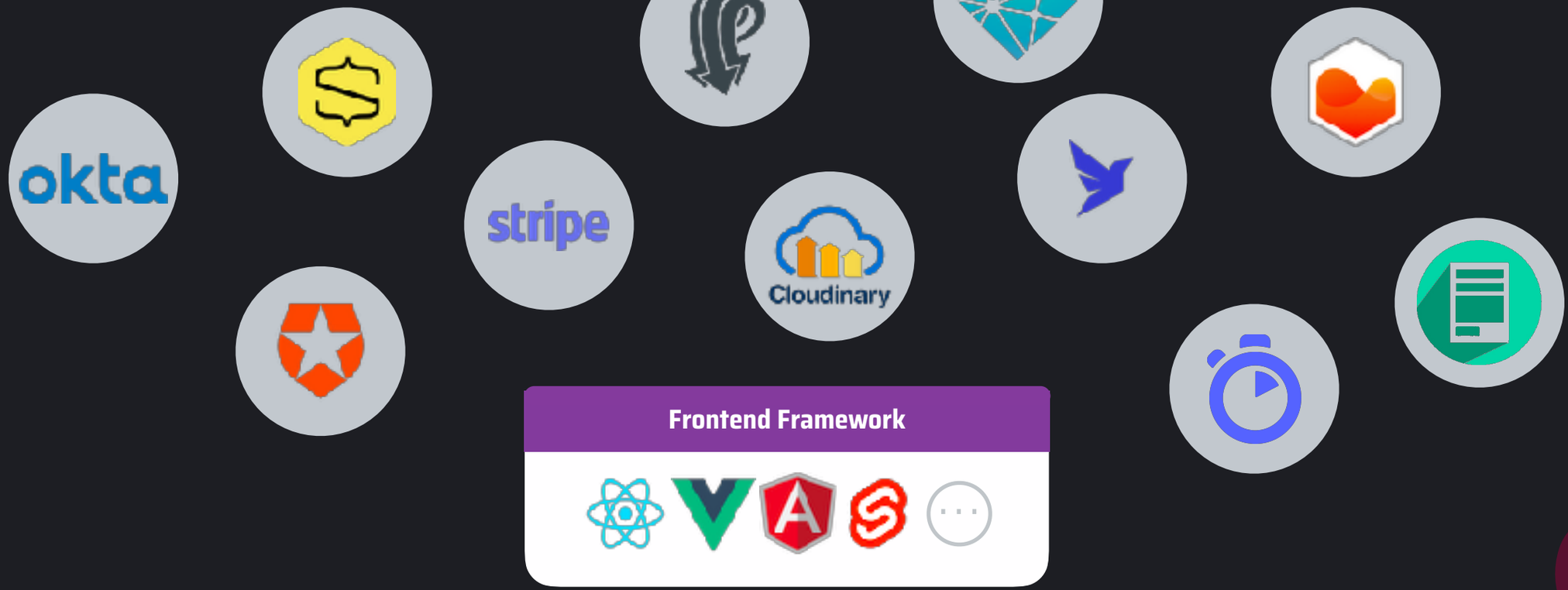
Frontend Framework

CDN

Static Site


Frontend Framework
CDN
Static Site


DNS
Browser
CDN
Services


okta
stripe
Cloudinary
Frontend Framework
CDN

Functions as a Service

Good News

Everyone!

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

mo Time! Demo Time! Demo Time! Demo Tim

Planet Express Delivery Request

Contents

Crew

- [ ] Leela
- [ ] Fry
- [ ] Bender
- [ ] Amy
- [ ] Zoidberg
- [ ] Prof. Farnsworth
- [ ] Hermes

Recipient

Destination

Notes

Appearances

Request Job

Forms

Functions

Authentication

Event triggers ✨

Planet Express Delivery Request

Contents

Crew

- [ ] Leela
- [ ] Fry
- [ ] Bender
- [ ] Amy
- [ ] Zoidberg
- [ ] Prof. Farnsworth
- [ ] Hermes

Recipient

Destination

Notes

Appearances

Request Job

divtag > delivery-request

< Forms

Submissions from delivery-request

1 verified submission. Last submission at 11:53 AM (a few seconds ago).

Extra spam prevention enabled via honeypot field.

Learn more about form handling in the docs ↗

Download as CSV

Verified submissions

Delete | Mark as spam | Expand all

☐ **Moon Rocks** 11:53 AM

Contents **Moon Rocks**

Chosen Crew **Leela,Fry,Bender,Amy,Zoidberg**

Recipient **Hydroponic Farmer**

Destination **The Moon**

Notes

Appearances

Received today at **11:53 AM** from 207.229.178.97

```
<form
  name=“delivery-request“
  method=“POST"
  netlify-honeypot=“bot-field"
  data-netlify=“true"
>
  <p class="hidden">
    <label>
      Don’t fill this out if you're human:
      <input name="bot-field" />
    </label>
  </p>
  <p>
    <label>Contents:
      <input type="text" name="contents" />
    </label>
  </p>
  <fieldset>
    <label>
      <input
        name="chosenCrew"
        type="checkbox"
      />
      <span>Leela</span>
    </label>
    <label>
      <input
        name="chosenCrew"
```

```
<template>
  <div class="hello">
    <form
      name="delivery-request"
      method="post"
      data-netlify="true"
      data-netlify-honeypot="bot-field"
      @submit.prevent="handleSubmit"
    >
      <header>
        <h2>Planet Express Delivery Request</h2>
      </header>
      <input type="hidden" name="form-name" value="delivery-request" />
      <input type="hidden" id="bot" name="bot" />
      <div>
        <label>
          <span>Contents</span>
          <input
            id="contents"
            name="contents"
            type="text"
            v-model="form.contents"
          />
        </label>
        …
      </div>
      <button type="submit" class="submit-button">Request Job</button>
    </form>
  </div>
```

```
  },
  methods: {
    encode(data) {
      return Object.keys(data)
        .map(
          key => `${encodeURIComponent(key)}=${encodeURIComponent(data
        )
        .join("&");
    },
    handleSubmit() {
      const axiosConfig = {
        header: { "Content-Type": "application/x-www-form-urlencoded"
      };
      const payload = {
        "form-name": "delivery-request",
        ...this.form
      };
      axios
        .post("/", this.encode(payload), axiosConfig)
        .then(() => {
          this.$router.push("thanks");
        })
        .catch(() => {
          this.$router.push("404");
        });
    }
  }
};
</script>
```

Forms

Functions

Authentication

Event triggers ✨

Log me out

The Series Has Landed (1ACV02)

Contents:
Prizes for the claw crane

Crew:

Recipient:
Sal

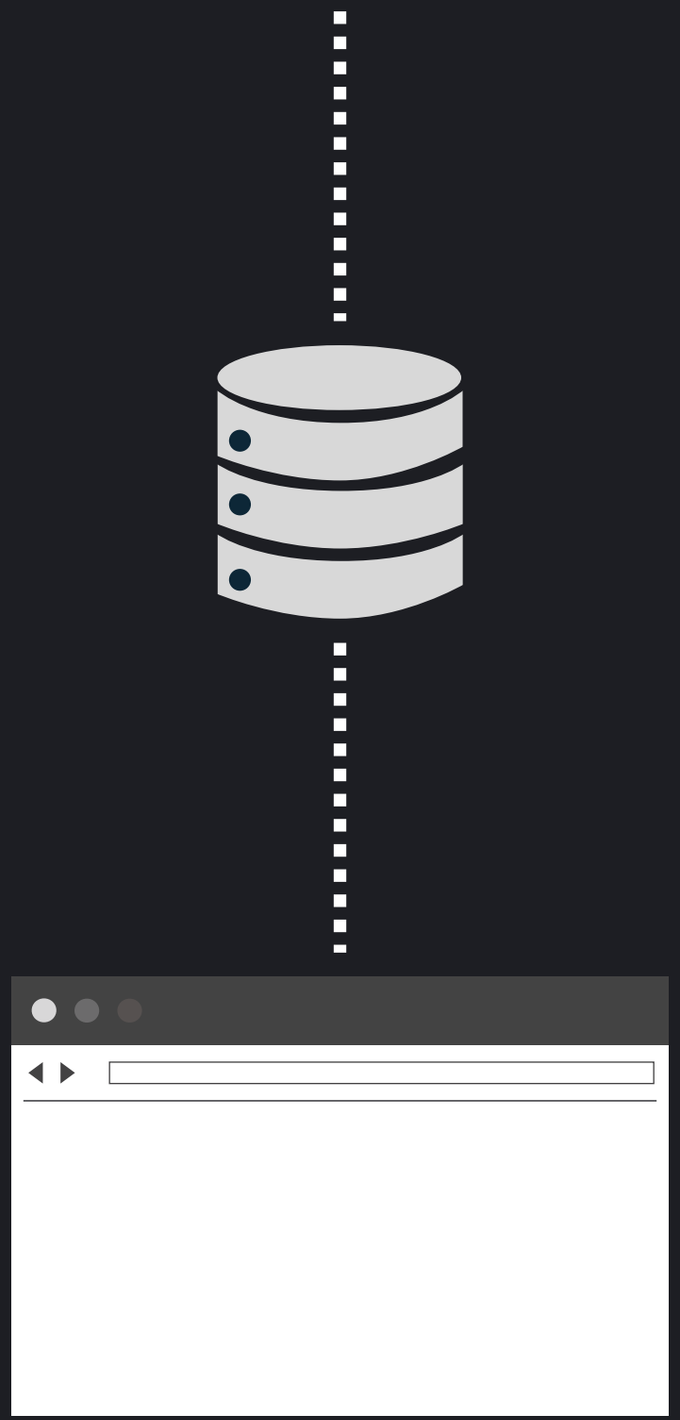
Destination:
Luna Park, Moon

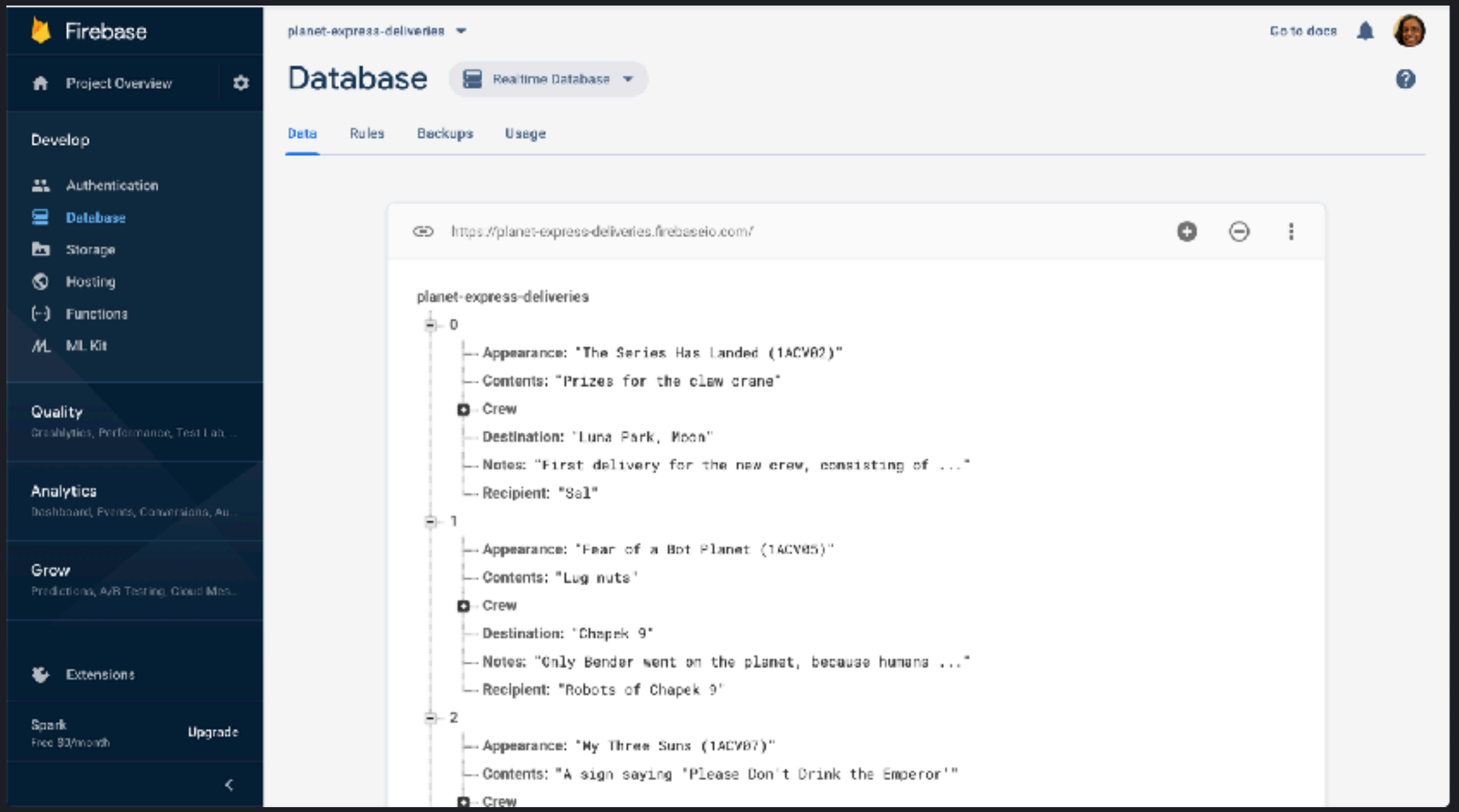
Notes:
First delivery for the new crew, consisting of Leela, Fry and Bender.

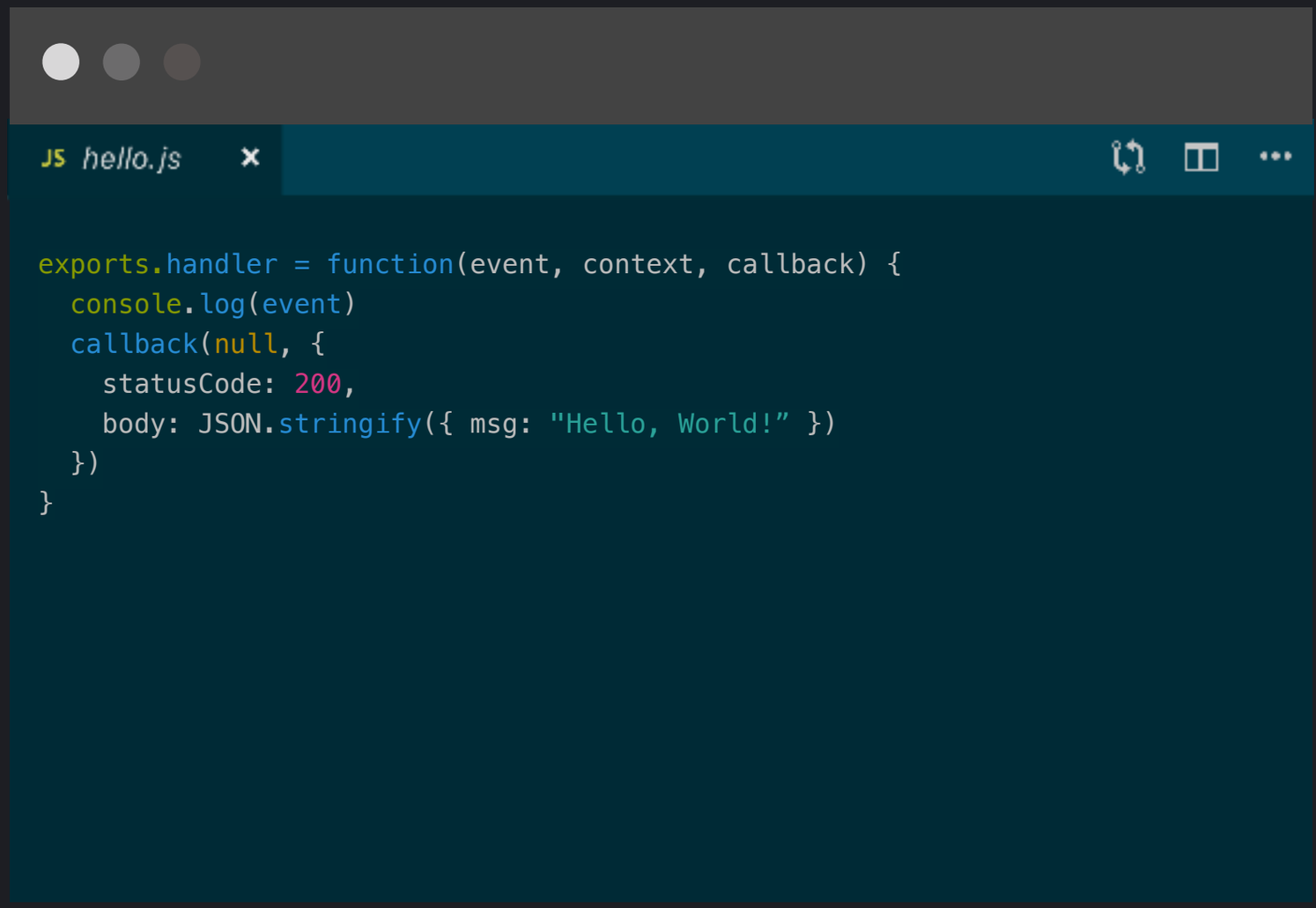
Fear of a Bot Planet (1ACV05)

Contents:
Lug nuts

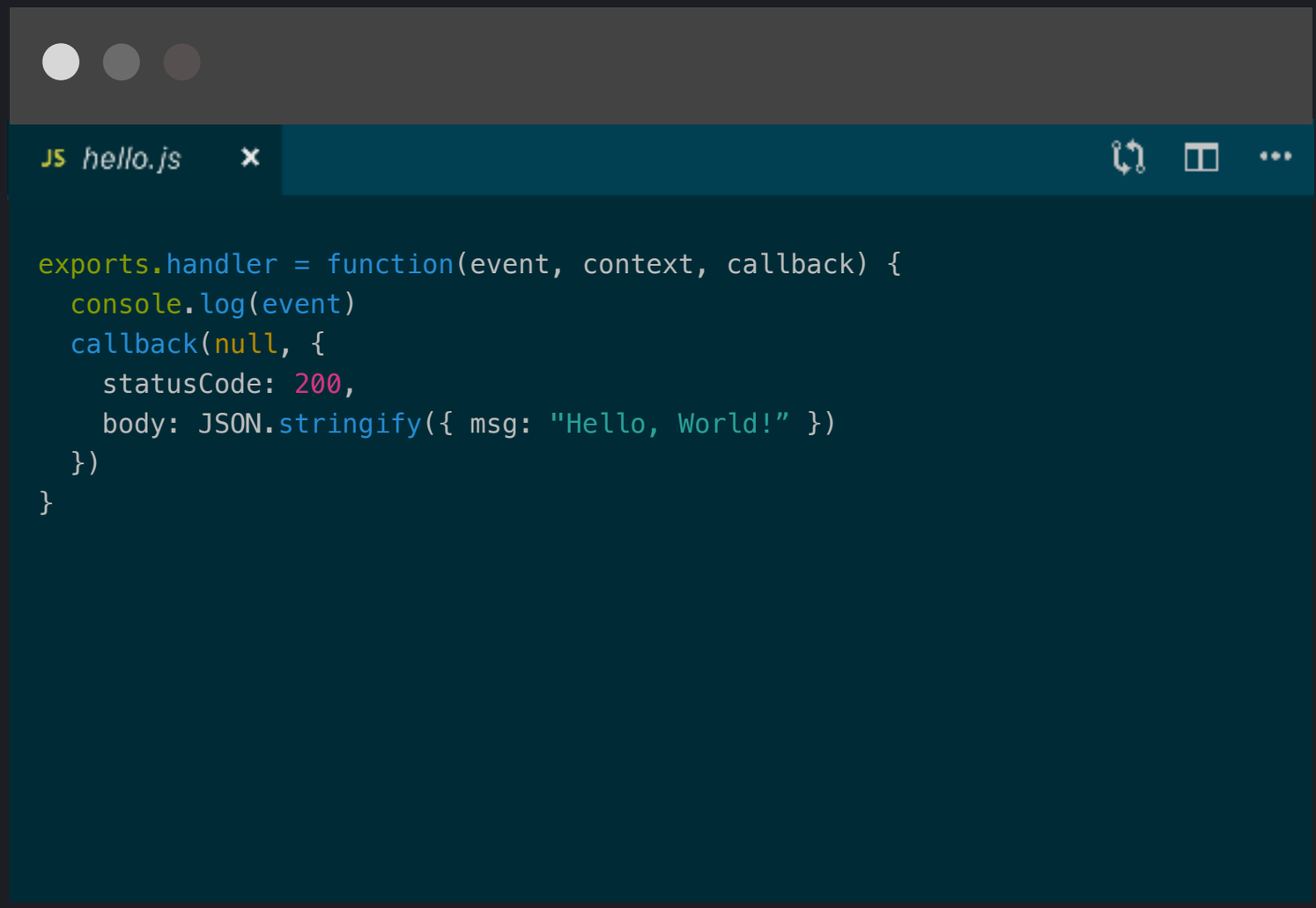
Crew:




Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics, Performance, Test Lab ...
Analytics
Dashboard, Events, Conversions, Au...
Grow
Predictions, A/B Testing, Cloud Mes...
Extensions
Spark
Free $0/month
Upgrade
planet-express-deliveries
Go to docs
Database
Realtime Database
Data
Rules
Backups
Usage
https://planet-express-deliveries.firebaseio.com/
planet-express-deliveries
0
Appearance: "The Series Has Landed (1ACV02)"
Contents: "Prizes for the claw crane"
Crew
Destination: "Luna Park, Moon"
Notes: "First delivery for the new crew, consisting of ..."
Recipient: "Sal"
1
Appearance: "Fear of a Bot Planet (1ACV05)"
Contents: "Lug nuts"
Crew
Destination: "Chapek 9"
Notes: "Only Bender went on the planet, because humans ..."
Recipient: "Robots of Chapek 9"
2
Appearance: "My Three Suns (1ACV07)"
Contents: "A sign saying 'Please Don't Drink the Emperor'"
Crew



functions/hello.js



functions/submission-created.js

```
/* Triggered when a form submission is posted to your site. */
const faunadb = require("faunadb");
const q = faunadb.query;

const client = new faunadb.Client({ secret: process.env.FAUNA_SECRET })

exports.handler = (event, context, callback) => {
  const body = JSON.parse(event.body).payload;
  const {
    contents,
    chosenCrew,
    recipient,
    destination
  } = body.data;
  client
    .query(
      q.Create(q.Class("deliveries"), {
        data: {
          Contents: contents,
          Crew: chosenCrew.split(","),
          Recipient: recipient,
          Destination: destination,
          status: "pending"
        }
      })
    )
    .then(ret => {
      console.log(ret);
      return callback(null, {
```

```
exports.handler = (event, context, callback) => {
  const body = JSON.parse(event.body).payload;
  const {
    contents,
    chosenCrew,
    recipient,
    destination
  } = body.data;
  client
    .query(
      q.Create(q.Class("deliveries"), {
        data: {
          Contents: contents,
          Crew: chosenCrew.split(","),
          Recipient: recipient,
          Destination: destination,
          status: "pending"
        }
      })
    )
    .then(ret => {
      console.log(ret);
      return callback(null, {
        statusCode: 200,
        body: JSON.stringify(body)
      });
    })
    .catch(err => {
      return callback(null, {
```

```
exports.handler = (event, context, callback) => {
  const body = JSON.parse(event.body).payload;
  const {
    contents,
    chosenCrew,
    recipient,
    destination
  } = body.data;
  client
    .query(
      q.Create(q.Class("deliveries"), {
        data: {
          Contents: contents,
          Crew: chosenCrew.split(","),
          Recipient: recipient,
          Destination: destination,
          status: "pending"
        }
      })
    )
    .then(ret => {
      console.log(ret);
      return callback(null, {
        statusCode: 200,
        body: JSON.stringify(body)
      });
    })
    .catch(err => {
      return callback(null, {
```

```
exports.handler = (event, context, callback) => {
  const body = JSON.parse(event.body).payload;
  const {
    contents,
    chosenCrew,
    recipient,
    destination
  } = body.data;
  client
    .query(
      q.Create(q.Class("deliveries"), {
        data: {
          Contents: contents,
          Crew: chosenCrew.split(","),
          Recipient: recipient,
          Destination: destination,
          status: "pending"
        }
      })
    )
    .then(ret => {
      console.log(ret);
      return callback(null, {
        statusCode: 200,
        body: JSON.stringify(body)
      });
    })
    .catch(err => {
      return callback(null, {
```

```
exports.handler = (event, context, callback) => {
  const body = JSON.parse(event.body).payload;
  const {
    contents,
    chosenCrew,
    recipient,
    destination
  } = body.data;
  client
    .query(
      q.Create(q.Class("deliveries"), {
        data: {
          Contents: contents,
          Crew: chosenCrew.split(","),
          Recipient: recipient,
          Destination: destination,
          status: "pending"
        }
      })
    )
    .then(ret => {
      console.log(ret);
      return callback(null, {
        statusCode: 200,
        body: JSON.stringify(body)
      });
    })
    .catch(err => {
      return callback(null, {
```

Forms

Functions

Authentication

Event triggers ✨

Log In:

Email:

Password:

Login

GoTrue

```
import Vue from 'vue'
import Vuex from 'vuex'
import GoTrue from "gotrue-js";

Vue.use(Vuex)

const auth = new GoTrue({
  APIUrl: "https://planet-express-deliveries.netlify.com/.netlify/iden
  audience: "",
  setCookie: false
});

export default new Vuex.Store({
  state: {
    currentUser: getSavedState("auth.currentUser"),
    loggedIn: false,
    deliveries: []
  },
  mutations: {
    SET_CURRENT_USER(state, value) {
      state.currentUser = value;
    }
  },
  getters: {
    loggedIn(state) {
      return !!state.currentUser;
    }
  },
  actions: {
```

```
  },
  attemptLogin({ commit, dispatch }, credentials) {
    return new Promise((resolve, reject) => {
      auth
        .login(credentials.email, credentials.password)
        .then(response => {
          resolve(response);
          commit("SET_CURRENT_USER", response);
        })
        .catch(error => {
          reject(error.json);
        });
    });
  },
  attemptLogout({ commit }) {
    return new Promise((resolve, reject) => {
      const user = auth.currentUser();
      user
        .logout()
        .then(response => {
          console.log(response);
          resolve(response);
          commit("SET_CURRENT_USER", null);
        })
        .catch(error => {
          reject(error);
          console.log("Could not log out", error);
        });
    });
  },
```

```
35    },
36    attemptLogin({ commit, dispatch }, credentials) {
37      return new Promise((resolve, reject) => {
38        auth
39          .login(credentials.email, credentials.password)
40          .then(response => {
41            resolve(response);
42            commit("SET_CURRENT_USER", response);
43          })
44          .catch(error => {
45            reject(error.json);
46          });
47      });
48    },
49    attemptLogout({ commit }) {
50      return new Promise((resolve, reject) => {
51        const user = auth.currentUser();
52        user
53          .logout()
54          .then(response => {
55            console.log(response);
56            resolve(response);
57            commit("SET_CURRENT_USER", null);
58          })
59          .catch(error => {
60            reject(error);
61            console.log("Could not log out", error);
62          });
63      });
64    },
```

Forms

Functions

Authentication

Event triggers ✨

The Road to

Get to the CDN

Design for immutable deploys

Automate all the things

Serverless all the things

JAMuary

https://dev.to/t/jamuary

Thanks!

@shortdiv