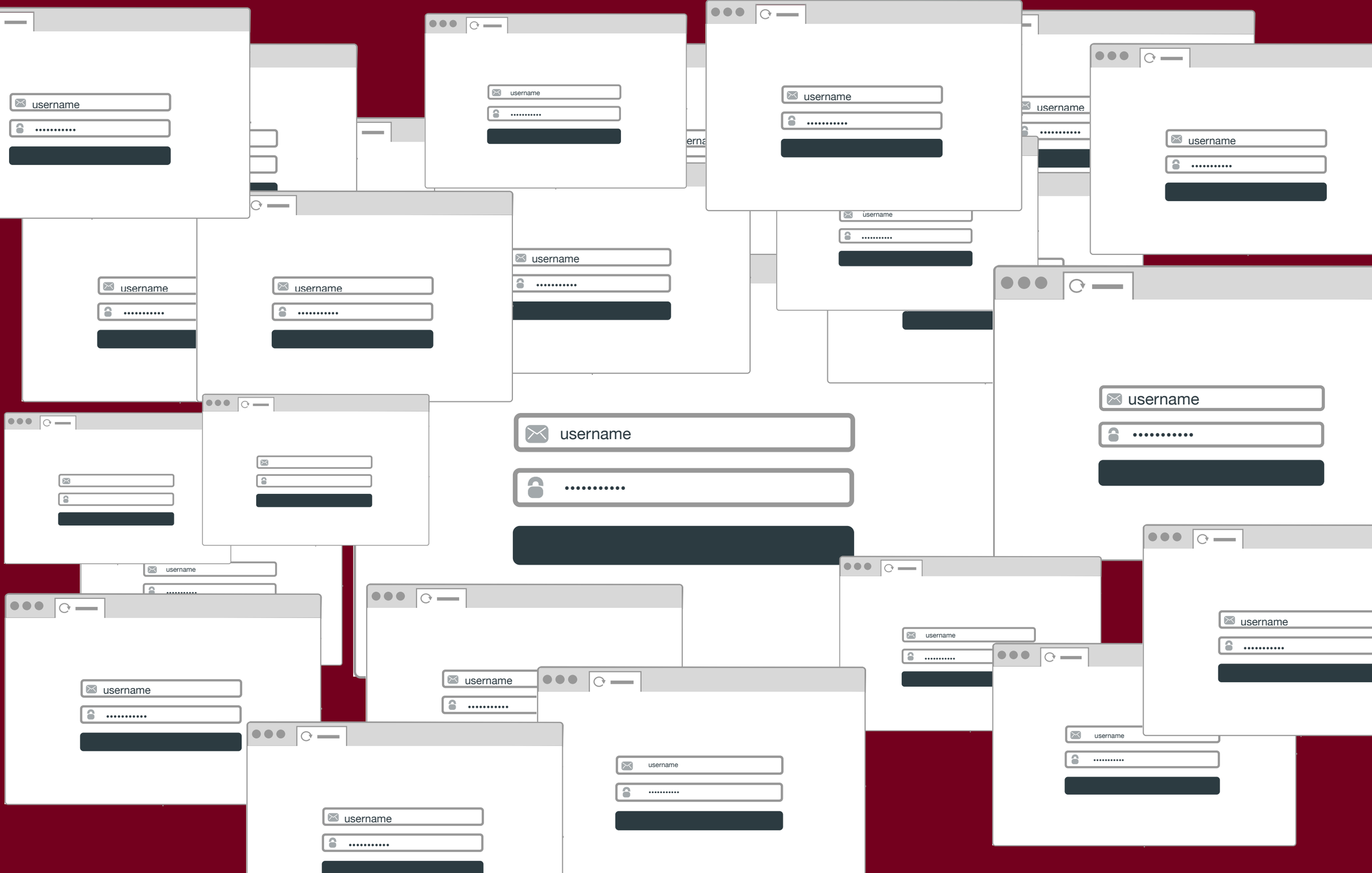
THE INTERNET IN THE BEGINNING...

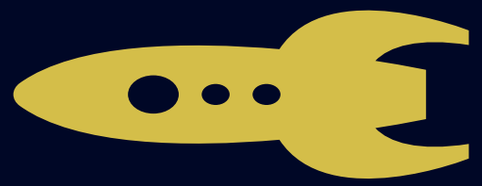


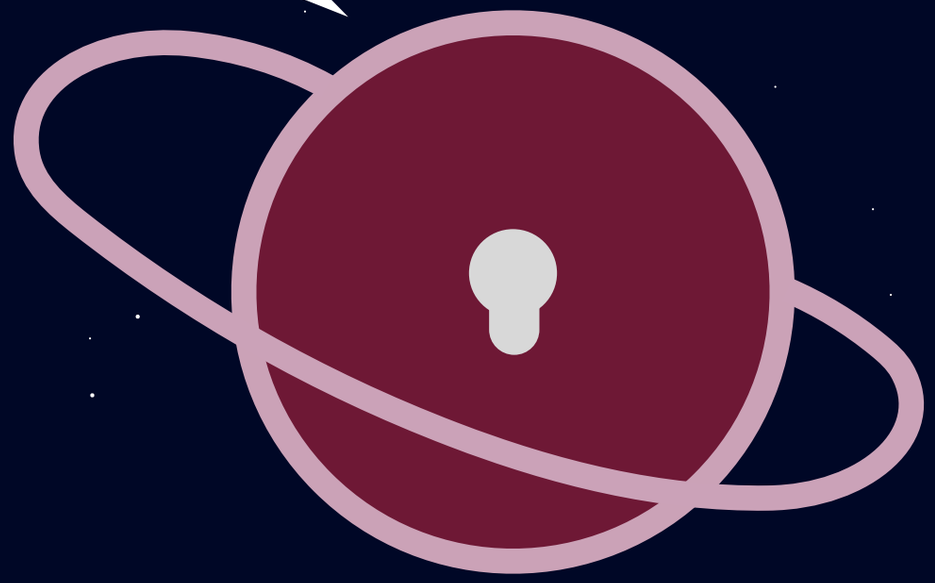
THE INTERNET

IN THE BEGINNING...

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVC
J9.eyJhcHBfbWV0YWRhdGEiOnsiYXV0aG9
yaXphdGlvbiI6eyJyb2xlcyI6WyJhZG1pb
iIsImVkaXRvciJdfX19.4IKFHH33EXWseN
jNIRO4-u5IlSlJOLyibG20qPA4Djs

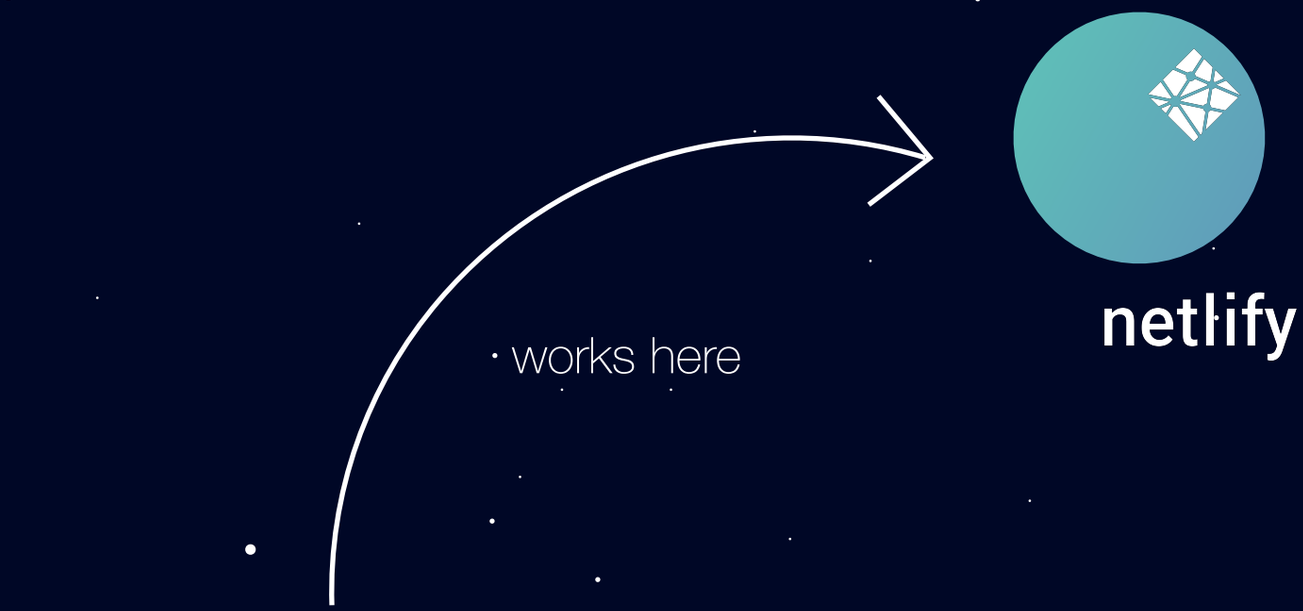
?????????
?????????
?????????





Authentication for the REST of us

Divya Sasidharan @shortdiv



Divya Sasidharan

DON'T PANIC

WHY DOES IT MATTER?

INTRODUCTION

SECURITY

STRATEGIES

AUTHENTICATION

DON’T PANIC

INTRODUCTION

AUTHENTICATION

WHO
WHAT
WHY
WHERE
WHEN
HOW
INTRODUCTION

INTRODUCTION
SECURITY
STRATEGIES
AUTHENTICATION
DON’T PANIC

AUTHENTICATION
WHY DOES IT MATTER?

WHY DOES IT MATTER?

AUTHENTICATION WHY DOES IT MATTER?

AUTHENTICATION WHY DOES IT MATTER?

AUTHENTICATION

CHARACTERISTICS

HISTORY

HOW TO HAVE FUN WITH

HOW TO NOT CONFUSE YOURSELF

WHAT TO AVOID

AUTHENTICATION

AUTHORIZATION

AUTHENTICATION

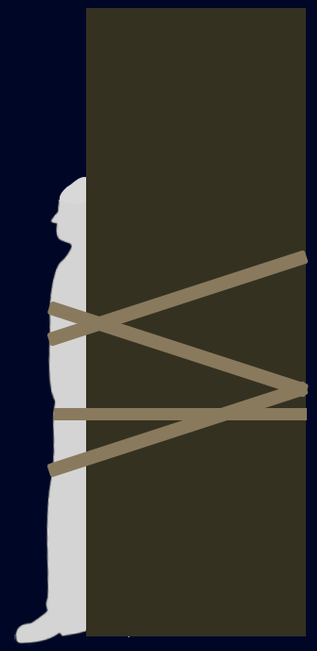
AUTHORIZATION

AUTHENTICATION AUTHORIZATION

User Authenticated:
Hitchhiker

AUTHENTICATION AUTHORIZATION

ERROR:
Hitchhiker NOT AUTHORIZED

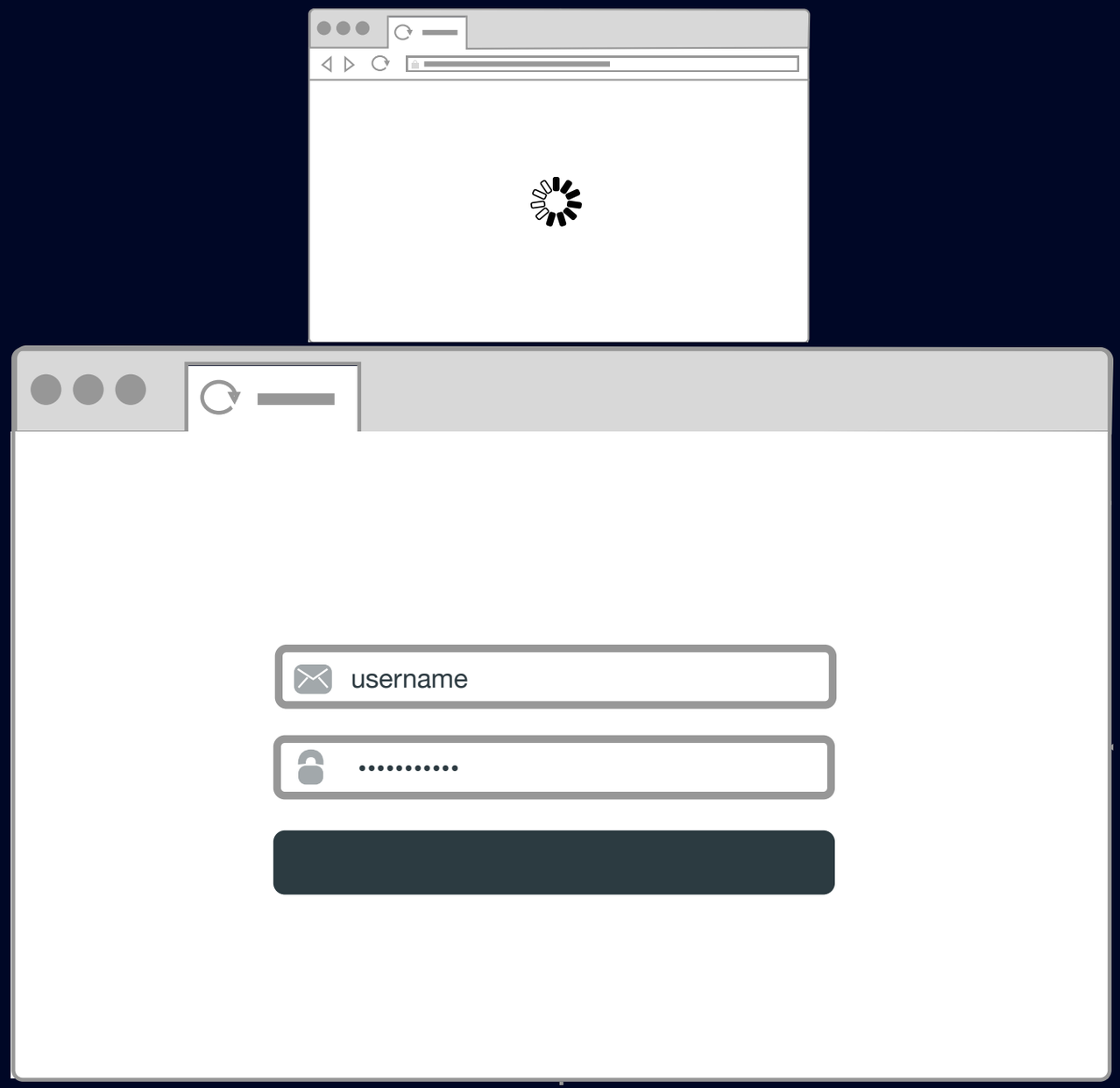


AUTHENTICATION AUTHORIZATION

HOW IT WORKS

AUTHENTICATION

AUTHENTICATION HOW IT WORKS



AUTHENTICATION HOW IT WORKS

Username: FPrefect
Password: thisisn0t@dri11

1

2 RlByZWZlY3Q6dGhpc2lzbjB0QGRya
TEx

AUTHENTICATION HOW IT WORKS BASIC AUTH

Username: Password

Username:Password

3

Authorization: Basic
RlByZWZlY3Q6dGhpc2lzbjB0
QGRyaTEx

4

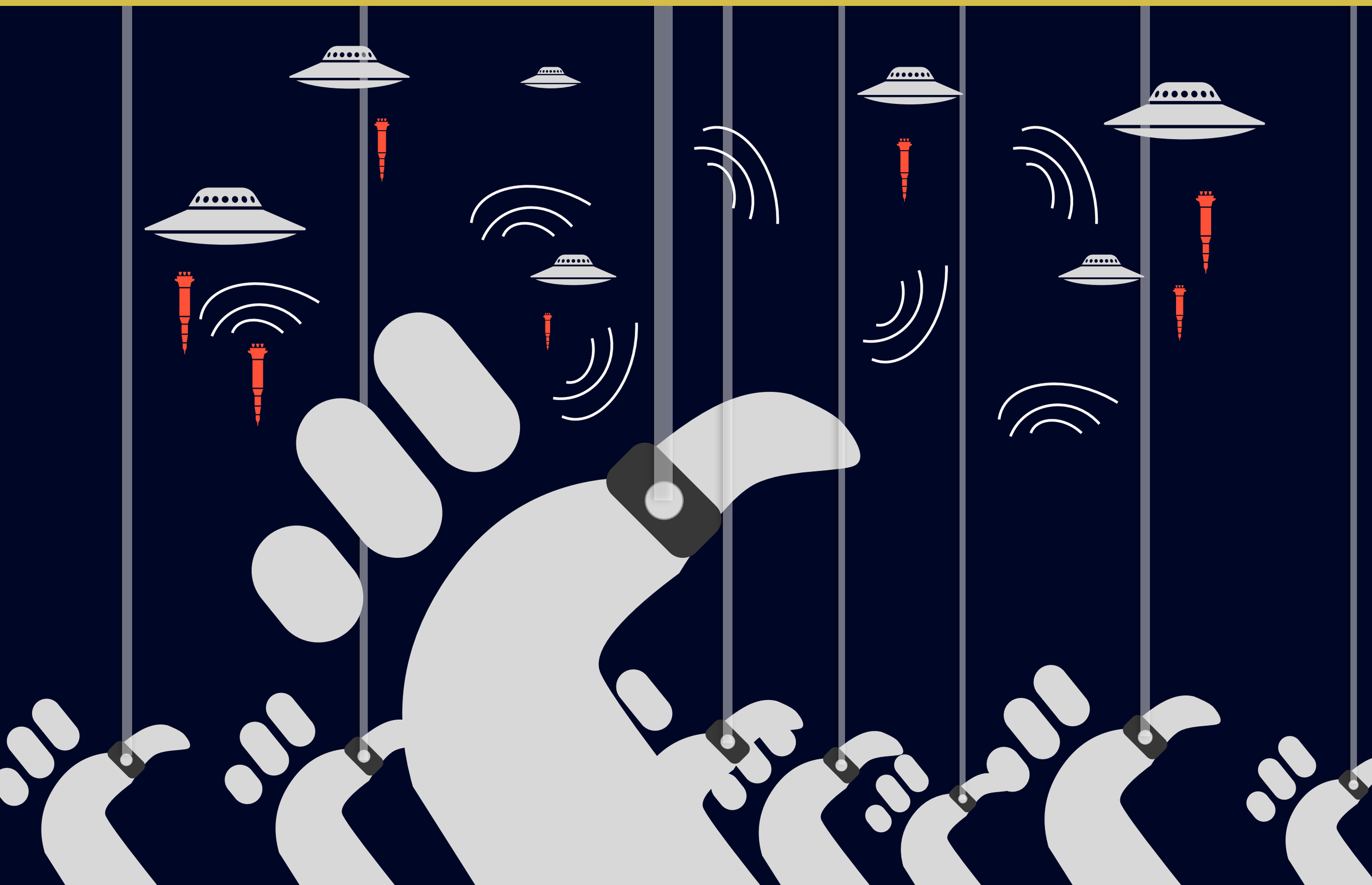
```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  “email”:”ford@infinidim.com”,
  …
}
```

AUTHENTICATION | HOW IT WORKS | BASIC AUTH

Authorization: Basic
RlByZWZlY3Q6dGhpc2lzbjB0
QGRyaTEx

```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  “email”:”ford@infinidim.com”,
  …
}
```

AUTHENTICATION HOW IT WORKS BASIC AUTH



AUTHENTICATION HOW IT WORKS BASIC AUTH

AUTHENTICATION HOW IT WORKS

```
Username: FPrefect
Password: thisisn0t@dri11
```

1

2

```
set-
cookie:hiker="SESSION_ID;
path=/; Secure; HttpOnly;
SameSite"
```

AUTHENTICATION HOW IT WORKS SESSION BASED AUTH

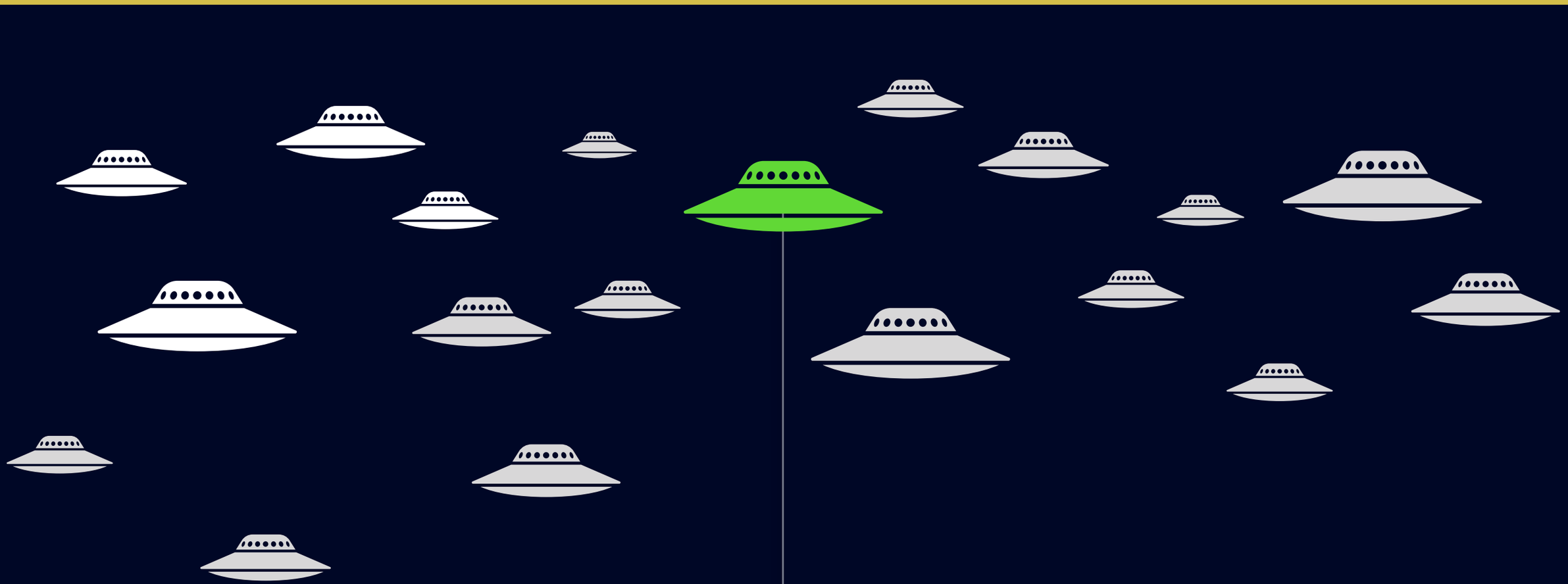
```
cookie:“hiker=XHTGubdsnHFJ
KHHJGFJHGKHhgjgJHNJGHGJHjs
ddsBHJGGFJKHtcbjBUY”
```

3

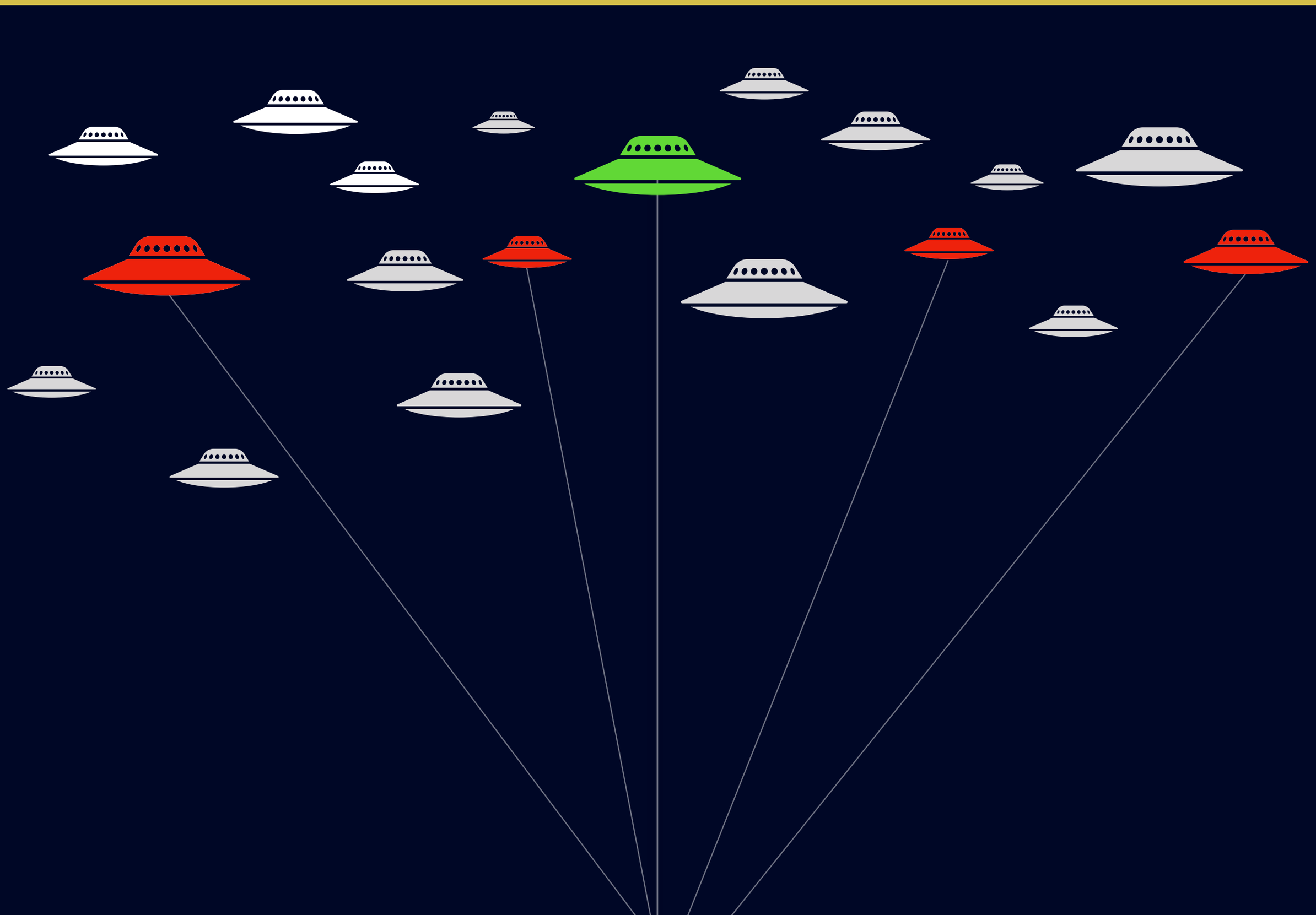
4

```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  “email”:”ford@infinidim.com”,
  …
}
```

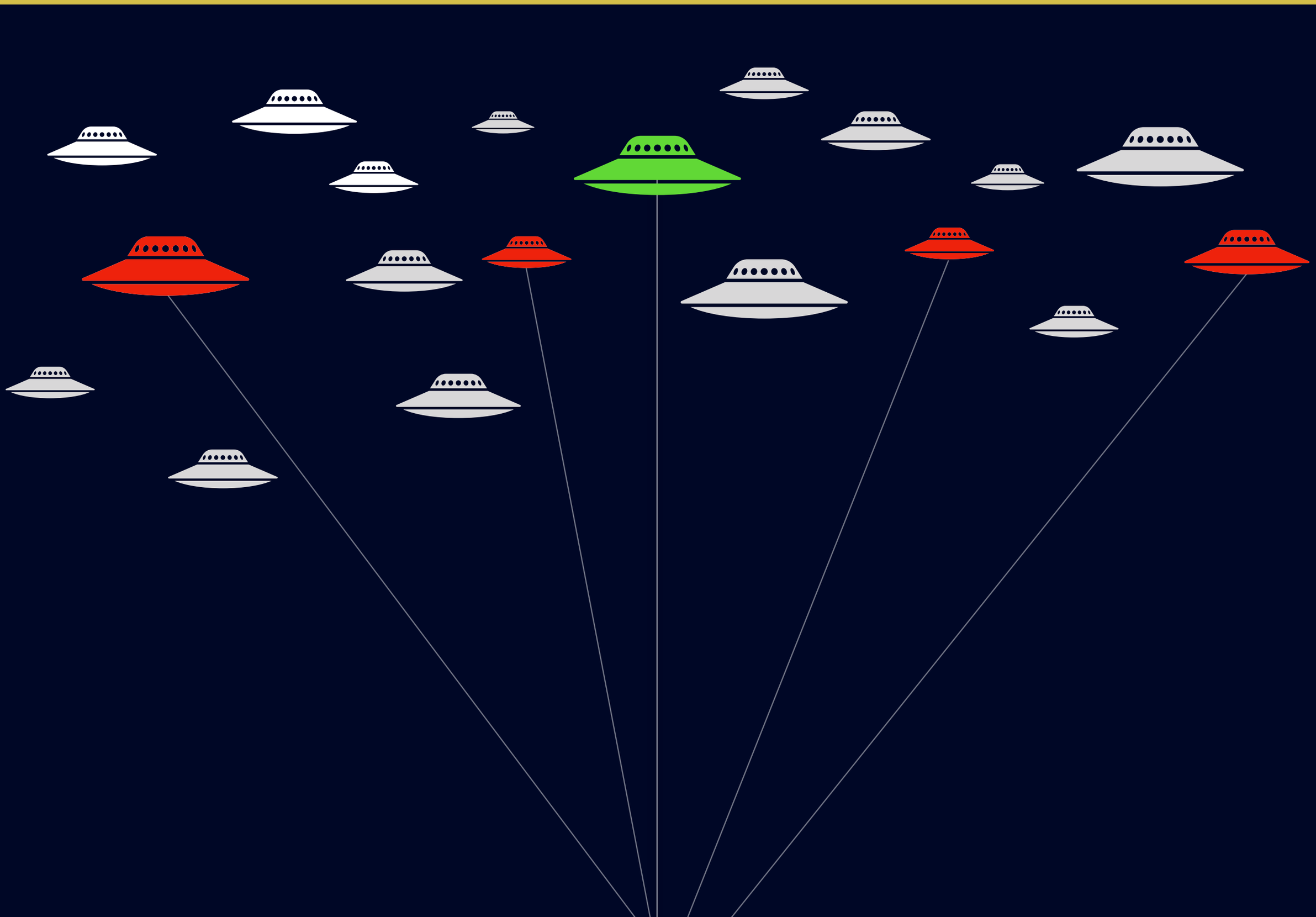
AUTHENTICATION HOW IT WORKS SESSION BASED AUTH



AUTHENTICATION HOW IT WORKS SESSION BASED AUTH



AUTHENTICATION HOW IT WORKS SESSION BASED AUTH



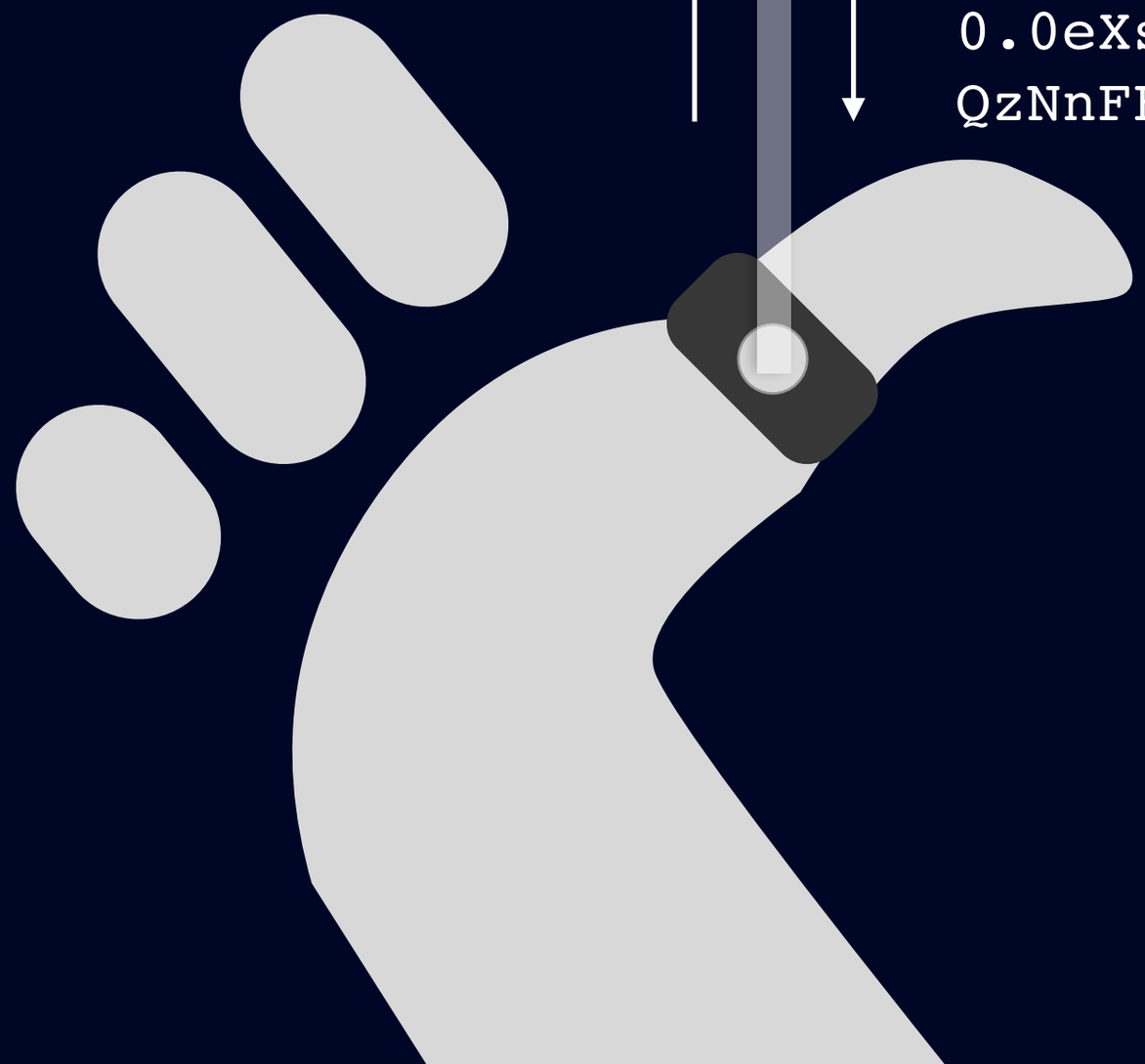
AUTHENTICATION HOW IT WORKS TOKEN BASED AUTH

Username: FPrefect
Password: thisisn0t@dri11

1

2

```
eyJhbGciOiJIUzI1NiIsInR5cCI6I
kpXVCJ9.eyJuYW1lIjoiRm9yZCBQc
mVmZWN0Iiwic3ViIjoiZm9yZEBpbm
ZpbmlkaW0uY29tIiwiaWF0IjoxNTE
2MjM5MDIyLCJhcHBfbWV0YWRhdGEi
OnsiYXV0aG9yaXphdGlvbiI6eyJyb
2xlcyI6WyJqb3VybmFsaXN0Il19fX
0.0eXsZauX7hjEB9oGjx7cg6LdyU8
QzNnFFS7tevWnqCA
```



```
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5c
CI6IkpXVCJ9.eyJuYW1lIjoiR
m9yZCBQcmVmZWN0Iiwic3ViIj
oiZm9yZEBpbmZpbmlkaW0uY29
tIiwiaWF0IjoxNTE2MjM5MDIy
LCJhcHBfbWV0YWRhdGEiOnsiY
XV0aG9yaXphdGlvbiI6WyJyb2
xlcyI6WyJqb3VybmFs
9fX0.0eXsZauX7b
g6LdyU8QzNnFF
```

3

4

```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  “email”:”ford@infinidim.com”,
  …
}
```

JWT

AUTHENTICATION

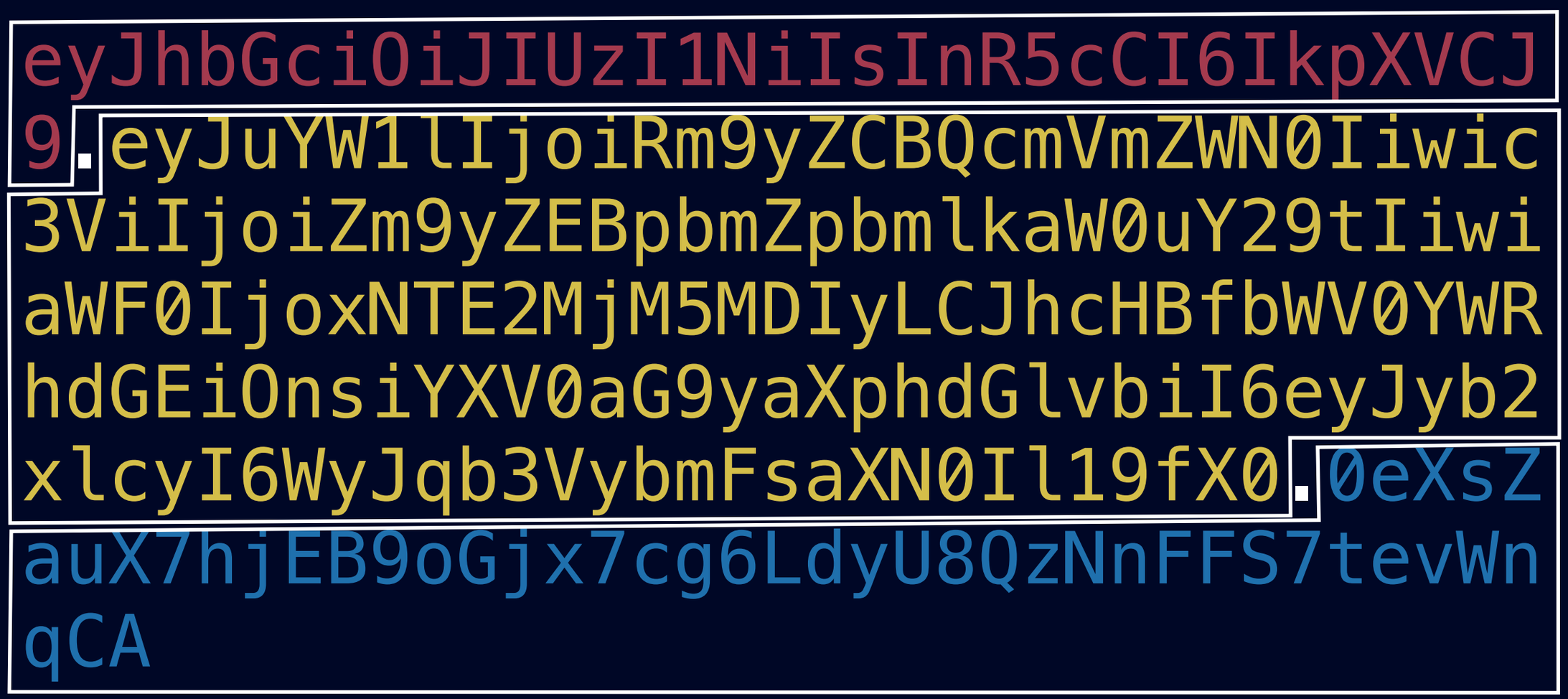
HOW IT WORKS

TOKEN BASED AUTH

AUTHENTICATION TOKEN BASED AUTH JWT


```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ
9.eyJuYW1lIjoiRm9yZCBQcmVmZWN0Iiwic
3ViIjoiZm9yZEBpbmZpbmlkaW0uY29tIiwi
aWF0IjoxNTE2MjM5MDIyLCJhcHBfbWV0YWR
hdGEiOnsiYXV0aG9yaXphdGlvbiI6eyJyb2
xlcyI6WyJqb3VybmFsaXN0Il19fX0.0eXsZ
auX7hjEB9oGjx7cg6LdyU8QzNnFFS7tevWn
qCA
```

AUTHENTICATION TOKEN BASED AUTH JWT



AUTHENTICATION TOKEN BASED AUTH JWT


```
{
  "alg": "HS256",
  "typ": "JWT"
}
{
  "name": "Ford Prefect",
  "sub" : "ford@infinidim.com",
  "exp" : "1577836800",
  "app_metadata": {
    "authorization": {
      "roles": [ "journalist" ]
    }
  },
}
```


header

AUTHENTICATION TOKEN BASED AUTH JWT

AUTHENTICATION TOKEN BASED AUTH JWT


```
{
  "alg": "HS256",
  "typ": "JWT"
}
{
  "name": "Ford Prefect",
  "sub" : "ford@infinidim.com",
  "exp" : "1577836800",
  "app_metadata": {
    "authorization": {
      "roles": [ "journalist" ]
    }
  },
}
```


header

AUTHENTICATION TOKEN BASED AUTH JWT

```
{
  "alg": "HS256",
  "typ": "JWT"
}
{
  "name": "Ford Prefect",
  "sub" : "ford@infinidim.com",
  "exp" : "1577836800",
  "app_metadata": {
    "authorization": {
      "roles": [ "journalist" ]
    }
  },
}
{
```

payload

AUTHENTICATION TOKEN BASED AUTH JWT

```
    "app_metadata": {
      "authorization": {
        "roles": [ "journalist" ]
      }
    },
  }
  {
    HMACSHA256(
    base64UrlEncode(header) + "." + signature
    base64UrlEncode(payload),
    a-256-bit-super-secret-secret)
  }
```

AUTHENTICATION TOKEN BASED AUTH JWT

JWT

Debugger Libraries Introduction Ask Get a T-shirt!

Crafted by Auth0

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ
uYW1lIjoiRm9yZCBQcmVmZWN0Iiwic3ViIjoiZm9
yZEBpbmZpbmlkaW0uY29tIiwiaWF0IjoxNTE2MjM
5MDIyLCJhcHBfbWV0YWRhdGEiOnsiYXV0aG9yaXp
hdGlvbiI6eyJyb2xlcyI6WyJqb3VybmFsaXN0Il1
9fX0.0eXsZauX7hjEB9oGjx7cg6LdyU8QzNnFFS7
tevWnqCA
```

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

Subject (whom the token refers to)

```
{
  "name": "Ford Prefect",
  "sub": "ford@infinidim.com",
  "iat": 1516239022,
  "app_metadata": {
    "authorization": {
      "roles": ["journalist"]
    }
  }
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) ☐ secret base64 encoded
```

Signature Verified

SHARE JWT

AUTHENTICATION TOKEN BASED AUTH JWT

CHARACTERISTICS

HISTORY

HOW TO HAVE FUN WITH

HOW TO NOT CONFUSE YOURSELF

WHAT TO AVOID

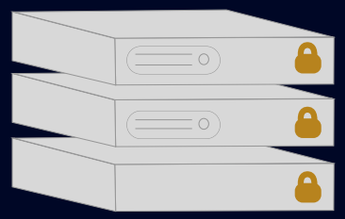
AUTHENTICATION

TOKEN BASED AUTH

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

DEMO

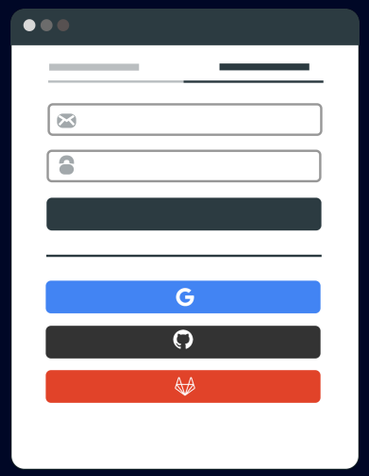
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION



Auth Server

Username: FPrefect
Password: thisisn0t@dri11

eyJhbGciOJ9.sfsfweHsefw9
.dgergeJGhg9HV…



Authorization: Bearer
eyJhbGciOJ9.sfsfweHsefw9
.dgergeJGhg9HV…

```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  …
}
```

Resource Server

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

Authentication with Functions as a Service

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

MYAWESOMESITE.netlify.com/.netlify/functions/jwt

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
const jwt = require("jsonwebtoken");
const uuidv4 = require("uuid/v4");
const axios = require("axios");

exports.handler = function(event, context, callback) {
  const getExpiryDate = () => {
    const exp = Math.floor(Date.now() / 1000) + 60 * 60;
    // const expReadable = new Date(exp);
    return exp;
  };
  const generateJWT = (exp, claims, roles, secret) =>
    jwt.sign(
      {
        exp,
        app_metadata: {
          user_id: uuidv4(),
          authorization: { roles }
        },
        user_metadata: claims
      },
      secret
    );
  const parsedBody = JSON.parse(event.body);
  const { claims, roles, secret } = parsedBody;
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
exports.handler = function(event, context, callback) {
  const getExpiryDate = () => {
    const exp = Math.floor(Date.now() / 1000) + 60 * 60;
    // const expReadable = new Date(exp);
    return exp;
  };
  const generateJWT = (exp, claims, roles, secret) =>
    jwt.sign(
      {
        exp,
        app_metadata: {
          user_id: uuidv4(),
          authorization: { roles }
        },
        user_metadata: claims
      },
      secret
    );
  const parsedBody = JSON.parse(event.body);
  const { claims, roles, secret } = parsedBody;

  const expiry = getExpiryDate();
  const token = generateJWT(expiry, claims, roles, secret);

  const response = {
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
23    const parsedBody = JSON.parse(event.body);
24    const { claims, roles, secret } = parsedBody;
25
26    const expiry = getExpiryDate();
27    const token = generateJWT(expiry, claims, roles, secret);
28
29    const response = {
30      jwt: token,
31    };
32
33    callback(null, {
34      statusCode: 200,
35      body: JSON.stringify(response)
36    });
37  };
38
39
40
41
42
43
44
45
46
47
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
  const getExpiryDate = () => {

    // const expReadable = new Date(exp);
    return exp;
  };
  const generateJWT = (exp, claims, roles, secret) =>
    jwt.sign(
      {
        exp,
        app_metadata: {
          user_id: uuidv4(),
          authorization: { roles }
        },
        user_metadata: claims
      },
      secret
    );
  const parsedBody = JSON.parse(event.body);
  const { claims, roles, secret } = parsedBody;

  const expiry = getExpiryDate();
  const token = generateJWT(expiry, claims, roles, secret);

  const response = {
    jwt: token,
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
  const getExpiryDate = () => {
    const exp = Math.floor(Date.now() / 1000) + 60 * 60;
    // const expReadable = new Date(exp);
    return exp;
  };
  const generateJWT = (exp, claims, roles, secret) =>
    jwt.sign(
      {
        exp,
        app_metadata: {
          user_id: uuidv4(),
          authorization: { roles }
        },
        user_metadata: claims
      },
      secret
    );
  const parsedBody = JSON.parse(event.body);
  const { claims, roles, secret } = parsedBody;

  const expiry = getExpiryDate();
  const token = generateJWT(expiry, claims, roles, secret);

  const response = {
    jwt: token,
```

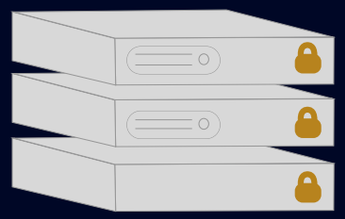
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
23   const parsedBody = JSON.parse(event.body);
24   const { claims, roles, secret } = parsedBody;
25
26   const expiry = getExpiryDate();
27   const token = generateJWT(expiry, claims, roles, secret);
28
29   const response = {
30     jwt: token,
31   };
32
33   callback(null, {
34     statusCode: 200,
35     body: JSON.stringify(response)
36   });
37 };
38
39
40
41
42
43
44
45
46
47
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
fetch(“/.netlify/functions/jwt”).then(res => {
  fetch(“/.netlify/functions/super-secret-function”, {
    method: "POST",
    headers: {
      "Authorization": `Bearer ${res.jwt}`
    },
    body: JSON.stringify({
      text: someDataIWanttoPost
    })
  })
})
```

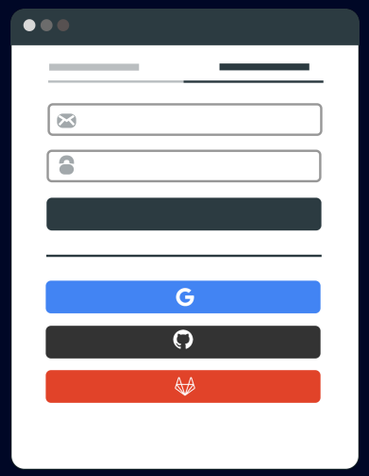
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION



Auth Server

```
Username: FPrefect
Password: thisisn0t@dri11
```

```
eyJhbGciOJ9.sfsfweHsefw9
.dgergeJGhg9HV…
```

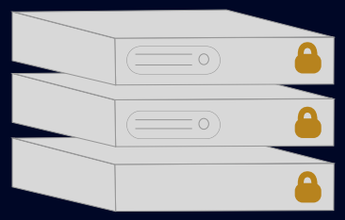


```
Authorization: Bearer
eyJhbGciOJ9.sfsfweHsefw9
.dgergeJGhg9HV…
```

```
{
  “id”:”89765-1”,
  “role”:”hitchhiker”,
  …
}
```

Resource Server

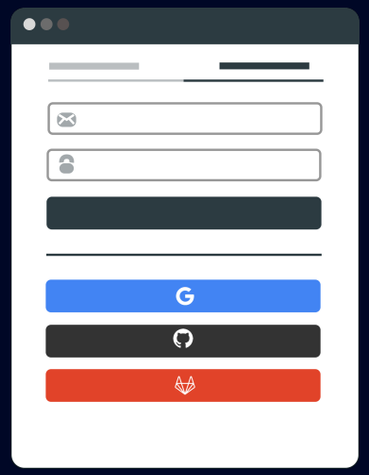
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION



Auth Server

Username: FPrefect
Password: thisisn0t@dri11

set-cookie:hiker="eyJhbGciOJ9.sfsfweHsefw9.dgergeJGhg9HV…; path=/; Secure; HttpOnly; SameSite"



cookie:"hiker=XHTGubdsnHFJKHHJGFJHGKHhgjgJHNJGHGJHjsddsBHJGGFJKHtcbjBUY"

```
{
  "id":"89765-1",
  "role":"hitchhiker",
  …
}
```

Resource Server

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
const jwt = require("jsonwebtoken");
const uuidv4 = require("uuid/v4");
const cookie = require("cookie");
const axios = require("axios");

exports.handler = function(event, context, callback) {
  const getExpiryDate = () => {
    const exp = Math.floor(Date.now() / 1000) + 60 * 60;
    // const expReadable = new Date(exp);
    return exp;
  };
  const generateJWT = (exp, claims, roles, secret) =>
    jwt.sign(
      {
        exp,
        app_metadata: {
          user_id: uuidv4(),
          authorization: { roles }
        },
        user_metadata: claims
      },
      secret
    );
  const parsedBody = JSON.parse(event.body);
  const { claims, roles, secret } = parsedBody;
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
  const expiry = getExpiryDate();
  const token = generateJWT(expiry, claims, roles, secret);

  const netlifyCookie = cookie.serialize("nf_jwt", token, {
    secure: true,
    path: "/",
    expires: new Date(expiry.toString())
  });

  const response = {
    jwt: token,
    exp: expiry
  };

  callback(null, {
    statusCode: 200,
    headers: {
      "Set-Cookie": netlifyCookie,
      "Cache-Control": "no-cache"
    },
    body: JSON.stringify(response)
  });
};
```

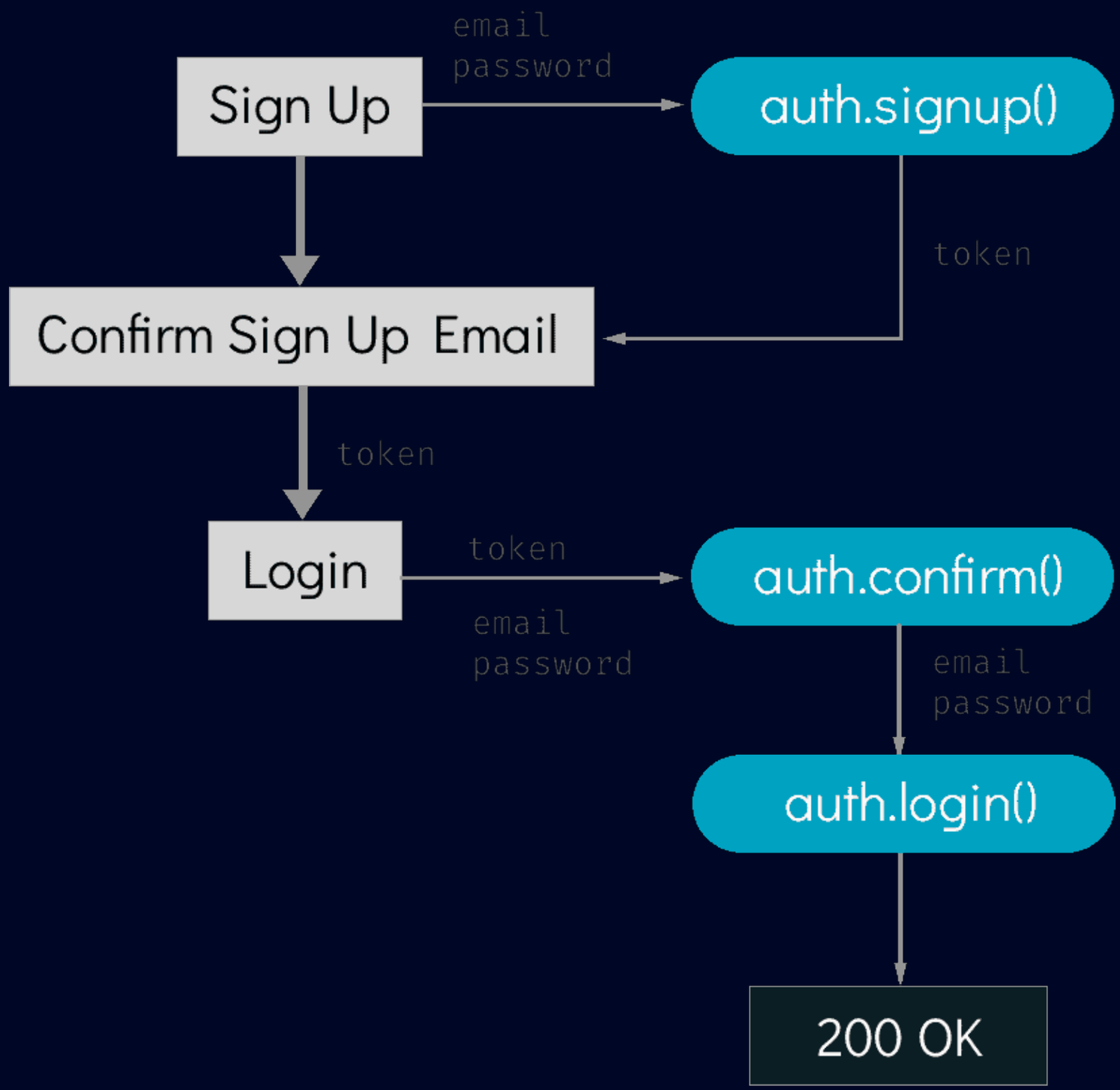
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
  const expiry = getExpiryDate();
  const token = generateJWT(expiry, claims, roles, secret);

  const netlifyCookie = cookie.serialize("nf_jwt", token, {
    secure: true,
    path: "/",
    expires: new Date(expiry.toString())
  });

  const response = {
    jwt: token,
    exp: expiry
  };

  callback(null, {
    statusCode: 200,
    headers: {
      "Set-Cookie": netlifyCookie,
      "Cache-Control": "no-cache"
    },
    body: JSON.stringify(response)
  });
};
```

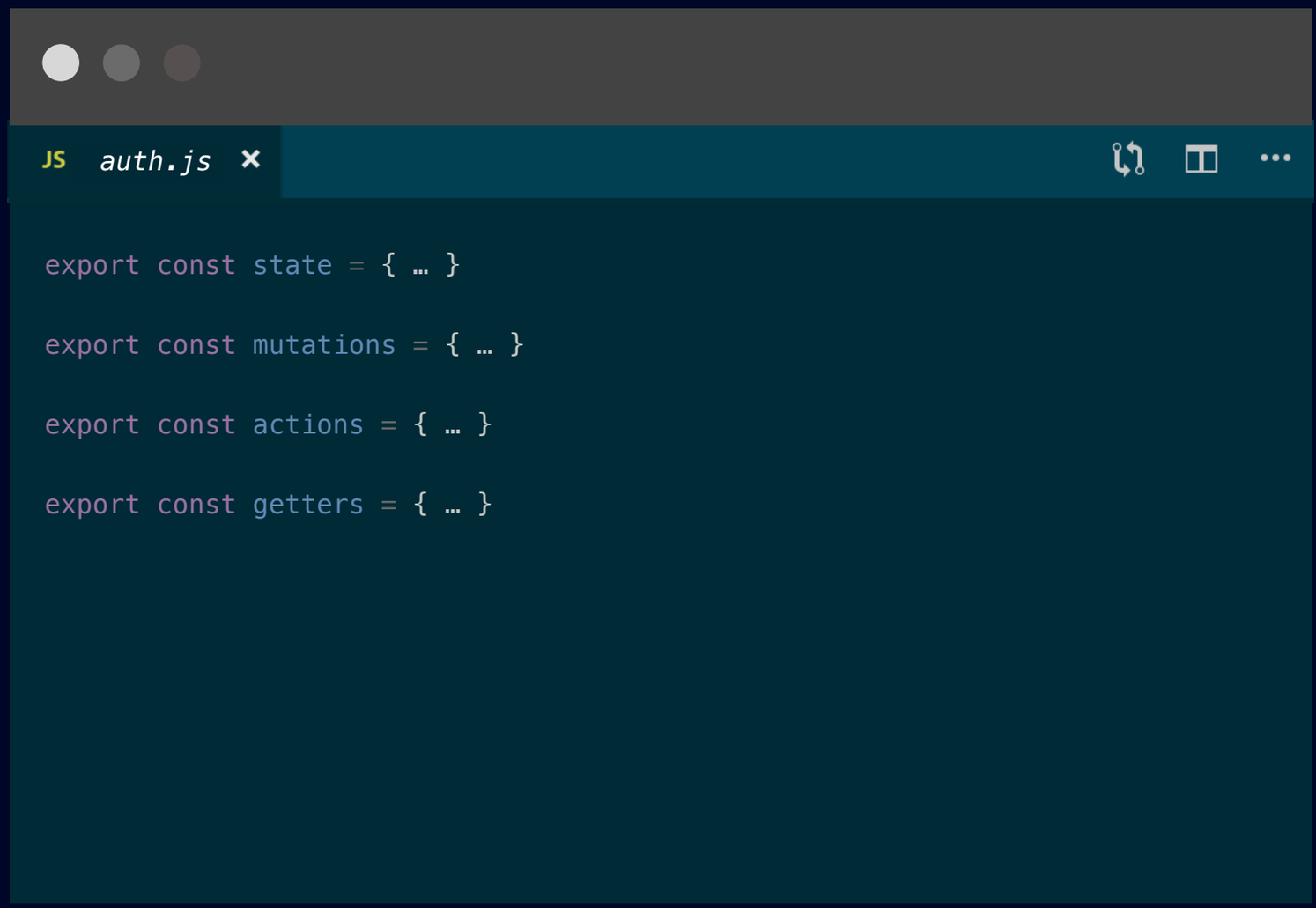
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

DEMO

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION



AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION



src/state/auth.js

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
import GoTrue from "gotrue-js";
import axios from "axios";

const auth = new GoTrue({
  APIUrl: "https://chipie.netlify.com/.netlify/identity",
  audience: "",
  setCookie: false
});


export const state = {
  currentUser: getSavedState("auth.currentUser"),
  loading: false,
  token: null,
  notifications: []
};

export const mutations = {
  SET_CURRENT_USER(state, value) {
    state.currentUser = value;
    saveState("auth.currentUser", value);
  },
  TOGGLE_LOAD(state) {
    state.loading = !state.loading;
  }
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
import GoTrue from "gotrue-js";
import axios from "axios";

const auth = new GoTrue({
  APIUrl: "https://chipie.netlify.com/.netlify/identity",
  audience: "",
  setCookie: false
});


export const state = {
  currentUser: getSavedState("auth.currentUser"),
  loading: false,
  token: null,
  notifications: []
};

export const mutations = {
  SET_CURRENT_USER(state, value) {
    state.currentUser = value;
    saveState("auth.currentUser", value);
  },
  TOGGLE_LOAD(state) {
    state.loading = !state.loading;
  }
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
export const actions = {
  init() {
    localStorage.removeItem("auth.currentUser");
  },
  validate({ commit, state }) {
    if (!state.currentUser) return Promise.resolve(null);
    const user = auth.currentUser();
    commit("SET_CURRENT_USER", user);
    return user;
  },
  attemptLogin({ commit, dispatch }, credentials) {
    return new Promise((resolve, reject) => {
      dispatch("attemptConfirmation", credentials).then(() => {
        auth
          .login(credentials.email, credentials.password)
          .then(response => {
            resolve(response);
            commit("SET_CURRENT_USER", response);
          })
          .catch(error => {
            reject(error.json);
          });
      });
    });
  },
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
export const actions = {
  init() {
    localStorage.removeItem("auth.currentUser");
  },
  validate({ commit, state }) {
    if (!state.currentUser) return Promise.resolve(null);
    const user = auth.currentUser();
    commit("SET_CURRENT_USER", user);
    return user;
  },
  atteмptLogin({ commit, dispatch }, credentials) {
    return new Promise((resolve, reject) => {
      dispatch("atteмptConfirmation", credentials).then(() => {
        auth
          .login(credentials.email, credentials.password)
          .then(response => {
            resolve(response);
            commit("SET_CURRENT_USER", response);
          })
          .catch(error => {
            reject(error.json);
          });
      });
    });
  },
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
16 export const mutations = {
17   SET_CURRENT_USER(state, value) {
18     state.currentUser = value;
19     saveState("auth.currentUser", value);
20   },
21   TOGGLE_LOAD(state) {
22     state.loading = !state.loading;
23   }
24 };
25
26 export const getters = {
27   isLoggedIn(state) {
28     return !!state.currentUser;
29   }
30 };
31
32 export const actions = {
33   init() {
34     localStorage.removeItem("auth.currentUser");
35   },
36   validate({ commit, state }) {
37     if (!state.currentUser) return Promise.resolve(null);
38     const user = auth.currentUser();
39     commit("SET_CURRENT_USER", user);
40     return user;
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
<template>
  <div class="login-screen">
    <div class="account-login">
      <form @submit.prevent="login()">
        <label>
          <span>Email:</span>
          <input
            type=“text"
            placeholder=“name"
            v-model=“loginCreds.email"
          />
        </label>
        <label>
          <span>Password:</span>
          <input
            type="password"
            placeholder="password"
            v-model=“loginCreds.password"
          />
        </label>
        <button type="submit" class=“account-button">Login</button>
      </form>
    </div>
  </div>
</template>
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
<template>
  <div class="login-screen">
    <div class="account-login">
      <form @submit.prevent="login()">
        <label>
          <span>Email:</span>
          <input
            type=“text"
            placeholder=“name"
            v-model=“loginCreds.email"
          />
        </label>
        <label>
          <span>Password:</span>
          <input
            type="password"
            placeholder="password"
            v-model=“loginCreds.password"
          />
        </label>
        <button type="submit" class=“account-button">Login</button>
      </form>
    </div>
  </div>
</template>
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
<script>
import { mapActions } from "vuex";

export default {
  name: “LoginAccount”,
  data() {
    return {
      isNewUser: true,
      loginCreds: {
        email: null,
        password: null
      }
    }
  },
  methods: {
   ...mapActions("auth", [“attemptLogin”]),
    transferToDashboard() {
      this.$router.push(this.$route.query.redirect || "/");
    },
    login() {
      let token = decodeURIComponent(window.location.search)
       .substring(1)
       .split("confirmation_token=")[1];
      this.attemptLogin({ token, ...this.loginCreds })
        .then(res => {
```

AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
    }
  },
  methods: {
    ...mapActions("auth", [“attemptLogin”]),
    transferToDashboard() {
      this.$router.push(this.$route.query.redirect || "/");
    },
    login() {
      let token = decodeURIComponent(window.location.search)
       .substring(1)
       .split("confirmation_token=")[1];
      this.attemptLogin({ token, ...this.loginCreds })
       .then(res => {
         this.transferToDashboard();
       })
       .catch(err => {
         console.log(err);
       });
    },
  }
}
</script>
```

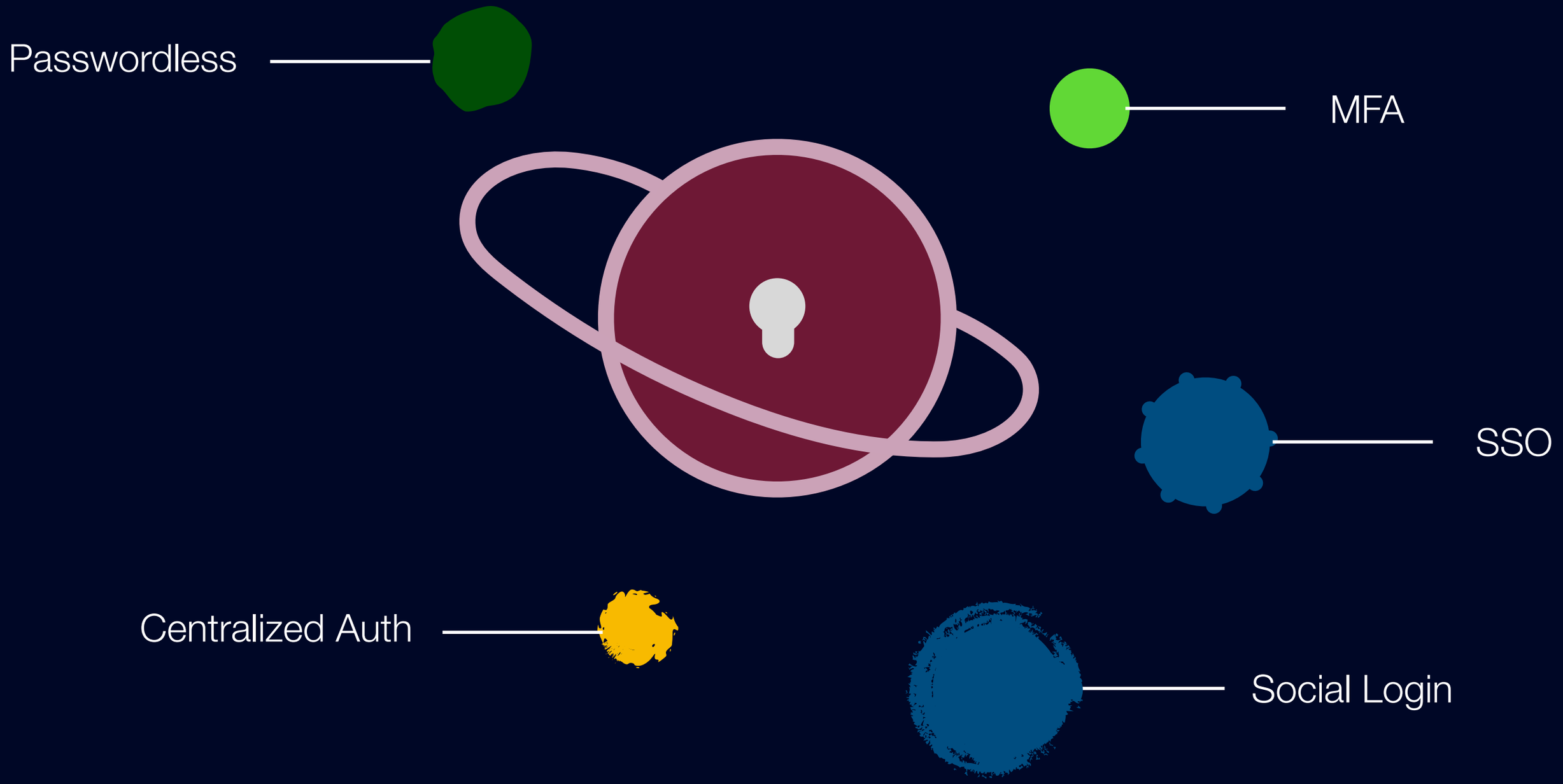
AUTHENTICATION TOKEN BASED AUTH IMPLEMENTATION

```
39      }
40    },
41    methods: {
42      ...mapActions("auth", [“attemptLogin”]),
43      transferToDashboard() {
44        this.$router.push(this.$route.query.redirect || "/");
45      },
46      login() {
47        let token = decodeURIComponent(window.location.search)
48          .substring(1)
49          .split('confirmation_token=')[1];
50        this.attemptLogin({ token, ...this.loginCreds })
51          .then(res => {
52            this.transferToDashboard();
53            console.log(res);
54          })
55          .catch(err => {
56            console.log(err);
57          });
58      },
59    }
60  }
61  </script>
62
63
```

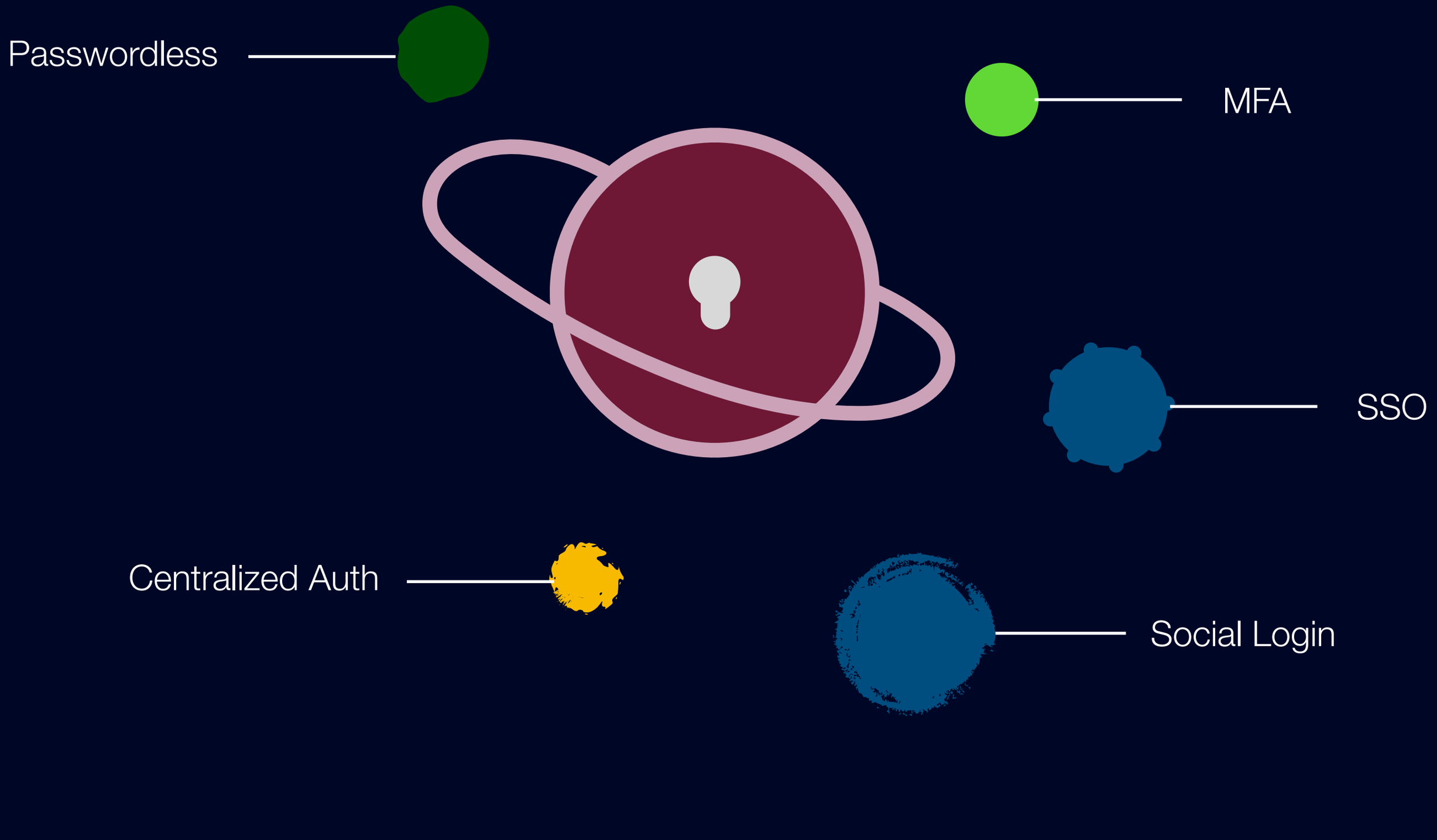
AUTHENTICATION HOW IT WORKS IMPLEMENTATION

AUTHENTICATION HOW IT WORKS IMPLEMENTATION

AUTHENTICATION DON’T ROLL YOUR OWN



AUTHENTICATION DON’T ROLL YOUR OWN



Roll your own JWT
code: https://github.com/shortdiv/jwt-generate
example: https://login-to-gated-site.netlify.com/

Identity Example
code: https://github.com/shortdiv/gotruejs-in-vue
example: https://netlify-gotrue-in-vue.netlify.com/

So Long and Thanks for All the Auth!!!

Divya Sasidharan @shortdiv