Fullstack Development

Authentication / Authorization

Authentication - `authen`

- A process of verifying user identity.
- Who is the user?
- Is the user really who he/she represents himself to be?

Authorization - `author`

- A process of verifying a user’s access level.
- Is user `X` authorized to access resource `R`?
- Is user `X` authorized to perform operation `P`?

Note

`authen` and `author` do not exist separately.

- Users try to access protected APIs:
  - Applications might need to allow user based on role (`author`) but also need to know user identities (`authen`).
- Social login (i.e. Google):
  - Users verify themselves to Google (`authen`) but authorize applications (`author`) to access their resources.

Approach

Rather than talking about `authen` vs `author`, let's focus on requirements:

- How do users sign up/in with credentials?
- How do users sign up/in with social accounts?
- How do we persist users' auth states?
    - So that users don't need to sign in at every request.

Part 1: Signing up/in with credential

Situation

- User fill in username and password.
- Your app creates user entry in database.
- *How do you store password?*
    - (and also compare it?)

Part 1: Signing up/in with credential

Section 1A: How to store password

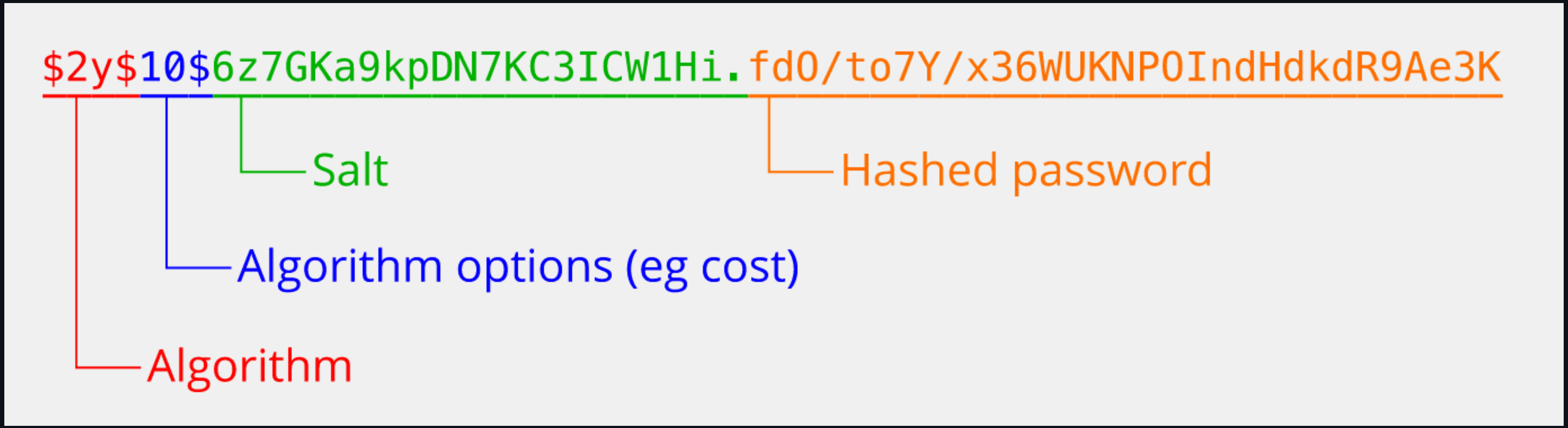
6 levels of safety

Technique	Ranking	Vunerability
Plain text	F	All
Encryption	D	Stolen key
Hashing	C	Rainbow table attack
Salting	B	Fast computer
Salting + Cost Factor (`bcrypt`)	B+	*Infinity stone* 🤣
?	A	

Adapted from source

Note

- SHA256
- Rawinbow table attack
- `bcrypt` hash



$2y$10$6z7GKa9kpDN7KC3ICW1Hi.fd0/to7Y/x36WUKNPOIndHdkdR9Ae3K
Salt
Hashed password
Algorithm options (eg cost)
Algorithm

`bcrypt` example

- `git clone -b bcrypt https://github.com/fullstack-67/auth-mpa-v2.git auth-bcrypt`
- `pnpm i`
- `npx tsx ./src/hash.ts`
- `npx tsx ./src/compare.ts`

Note

- Promisify the callback.
- Increasing time to generate (and compare) hash with incrasing `saltRounds`.
- `bcrypt.compare`
- Use of `debug` package.

Part 1: Signing up/in with credential

Section 1B: Implementation with `passport`

`passport`

- Most popular authentication middleware for `express`.
- Minimal and modular
- 500+ strategies (click at button)
- Confusing and poor documented 🤣
    - Hidden manual

Let's see it

- `git clone -b signin-credential https://github.com/fullstack-67/auth-mpa-v2.git auth-signin-credential`
- `pnpm i`
- `npm run db:reset`
- `npm run dev`

Side note about the project

- MPA - HTMX
- Use `SQLite` + `drizzle`.
  - Checkout the schema.
- Try debugging in VSCode.
  - See `launch.json`.

Highlighed packages

`package.json`

```json
{
  "passport": "^0.7.0",
  "passport-local": "^1.0.0"
}
```

Middleware

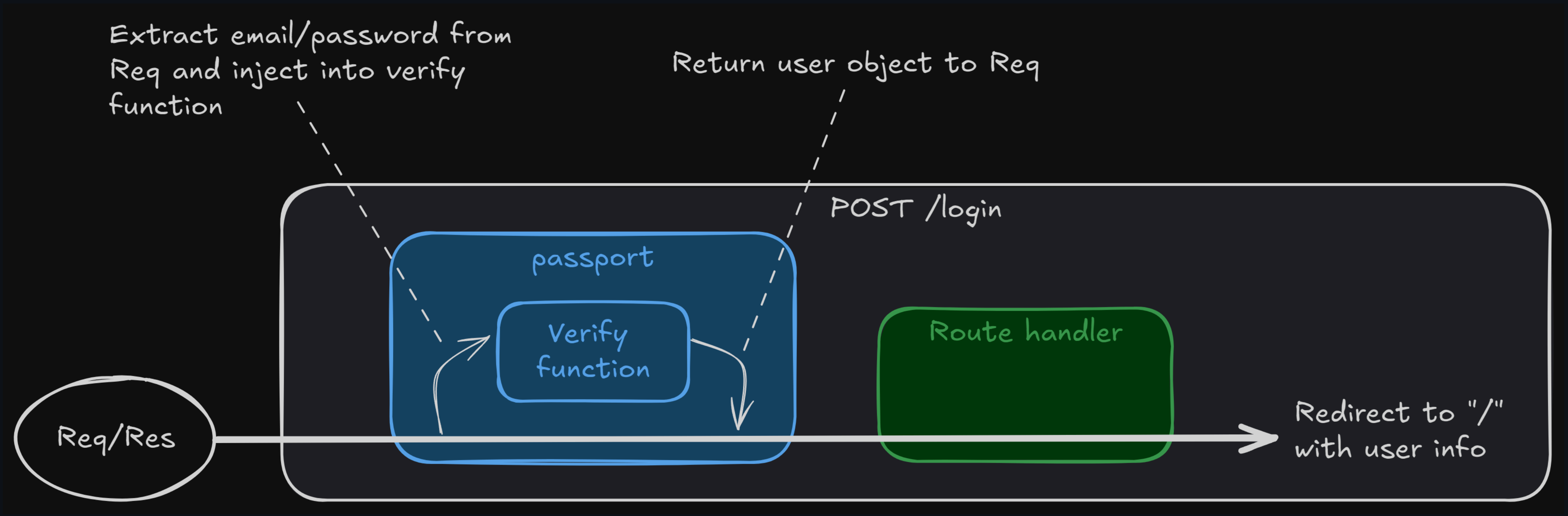
`src/index.ts`

```
passport.use(
  new LocalStrategy(
    {
      // Options
    },
    async function (email, password, done) {
      // Verify email / password
    }
  )
);
//
app.use(passport.initialize());
```

Available options

Route

```
app.post(
  "/login",
  passport.authenticate("local", { session: false }),
  function (req, res) {
    // * Passport will attach user object in the request
  }
);
```


Extract email/password from Req and inject into verify function
Return user object to Req
POST /login
passport
Verify function
Route handler
Req/Res
Redirect to "/" with user info

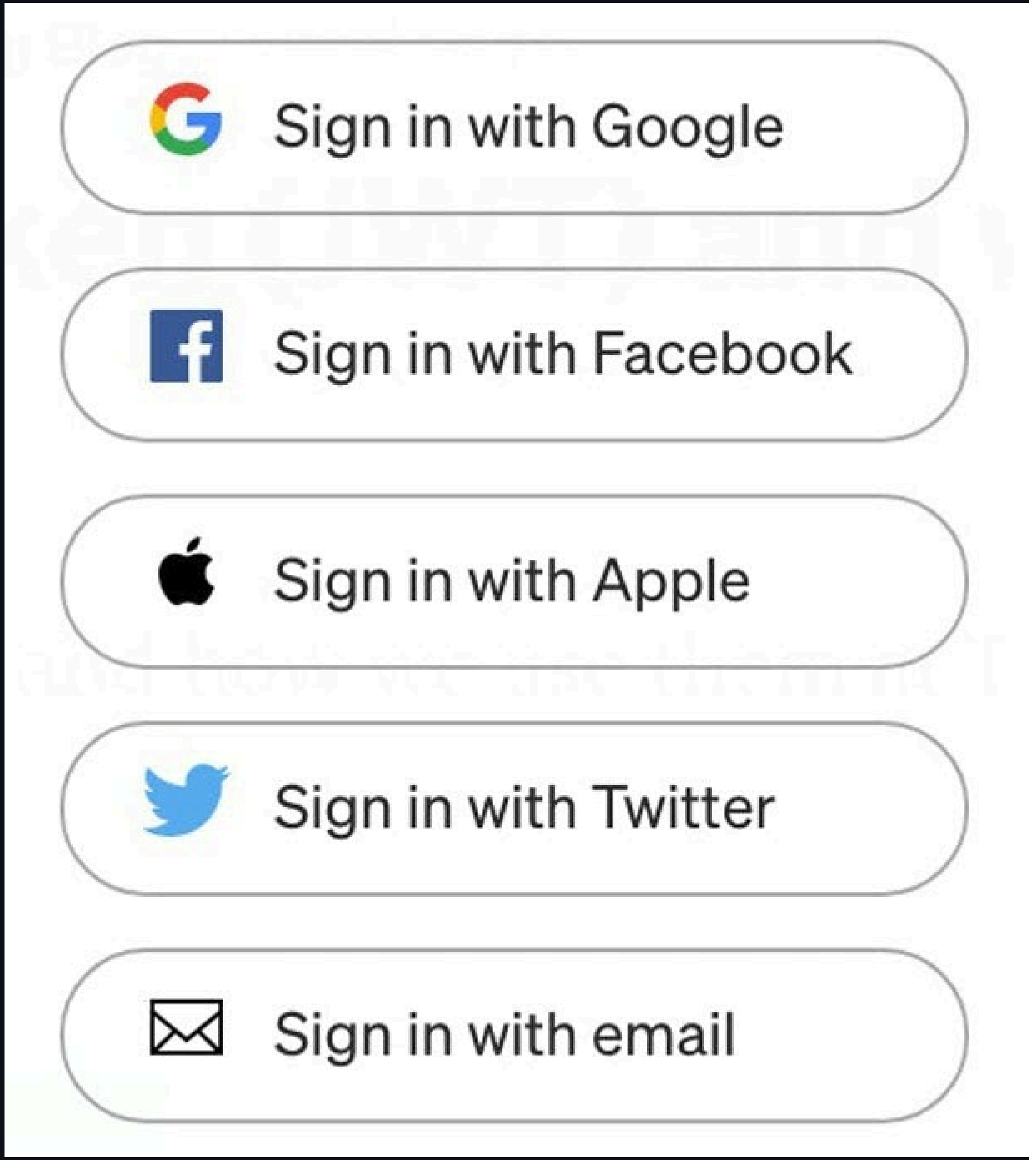
Can we do better?

Technique	**Ranking**	**Vunerability**
Plain text	F	All
Encryption	D	Stolen key
Hashing	C	Rainbow table attack
Salting	B	Fast computer
Salting + Cost Factor (`bcrypt`)	B+	*Infinity stone*
Not storing password	A	👉👉👉

Part 2: Social signing up/in

Something like this

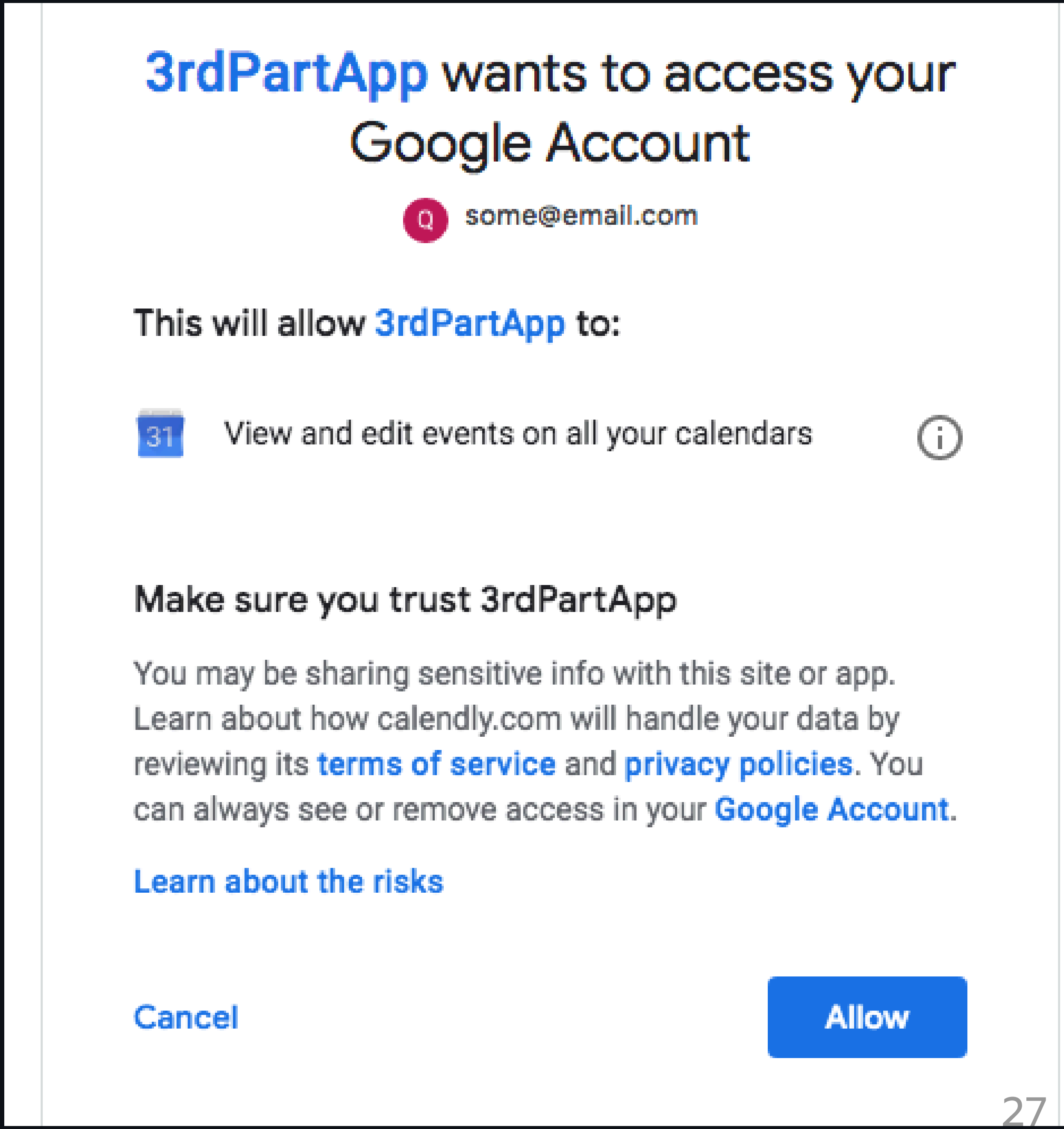
We need OAuth 2.0.



Part 2: Social signing up/in

Section 2A: OAuth 2.0

OAuth 2.0



3rdPartApp wants to access your Google Account

Q some@email.com

This will allow 3rdPartApp to:

31 View and edit events on all your calendars

Make sure you trust 3rdPartApp

You may be sharing sensitive info with this site or app. Learn about how calendly.com will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.

OAuth 2.0

- "Open Authorization"
- Standard designed to allow application to access resources hosted by other web apps on behalf of a user.
  - Standard for `author`
  - Not for `authen`
- Replaced OAuth 1.0 in 2012.

OAuth 2.0

- Specifies many "flows"
  - **Authorization Code Flow**
  - Client Credentials Flow
  - Refresh Token Flow
  - JWT Bearer Flow
  - Device Code Flow
- We will use "Authorization Code Flow" for social login.

Recommended resources

- https://engineering.backmarket.com/oauth2-explained-with-cute-shapes-7eae51f20d38
- https://developer.okta.com/blog/2019/10/21/illustrated-guide-to-oauth-and-oidc?utm_source=pocket_shared
- https://youtu.be/8aCyojTIW6U?si=YPxkcLPcAoK5jixl
- https://youtu.be/t18YB3xDfXI?si=pD1JnFP0GrnBXW2v

Wait

> Are we using OAuth (standard for `author`) and **authorization** code flow for `authen`?

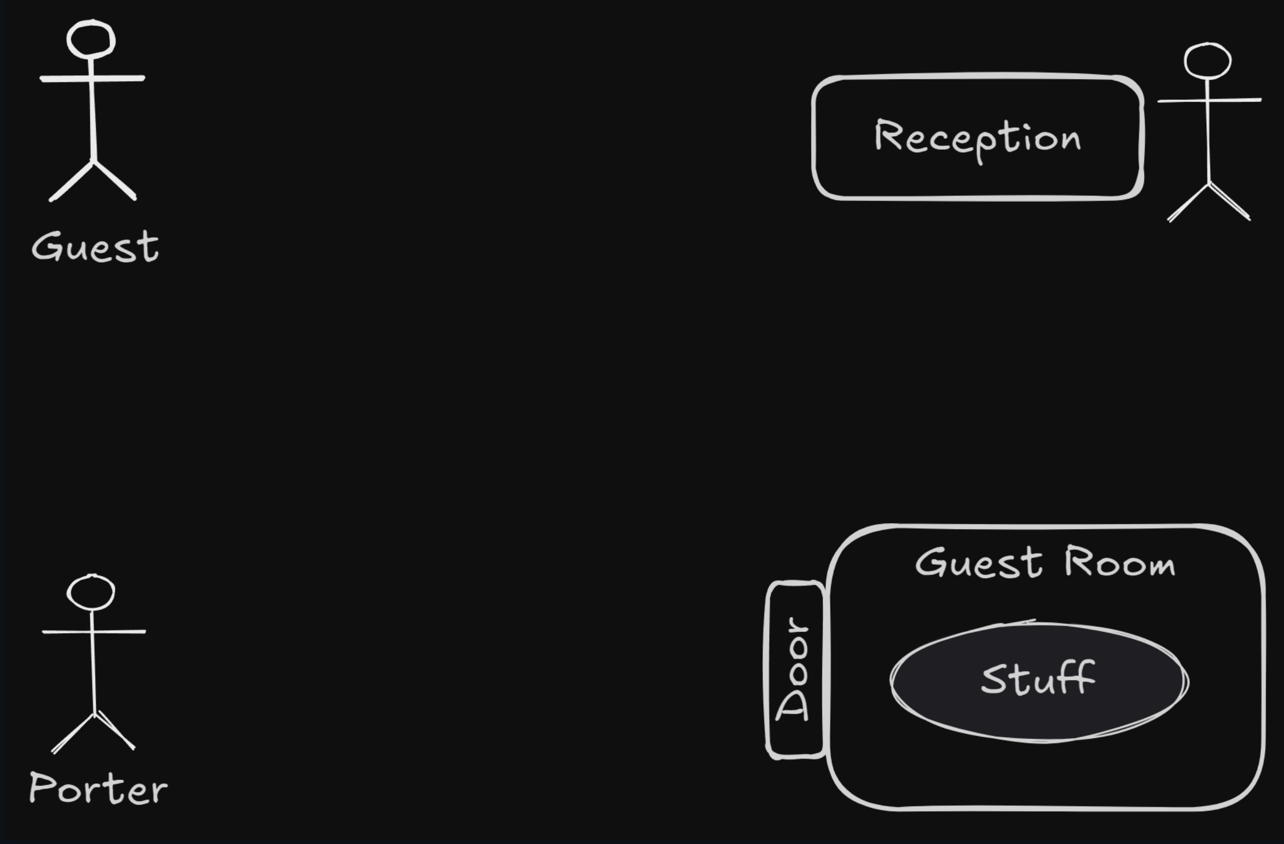
> Yes, we kind of "misusing" it.

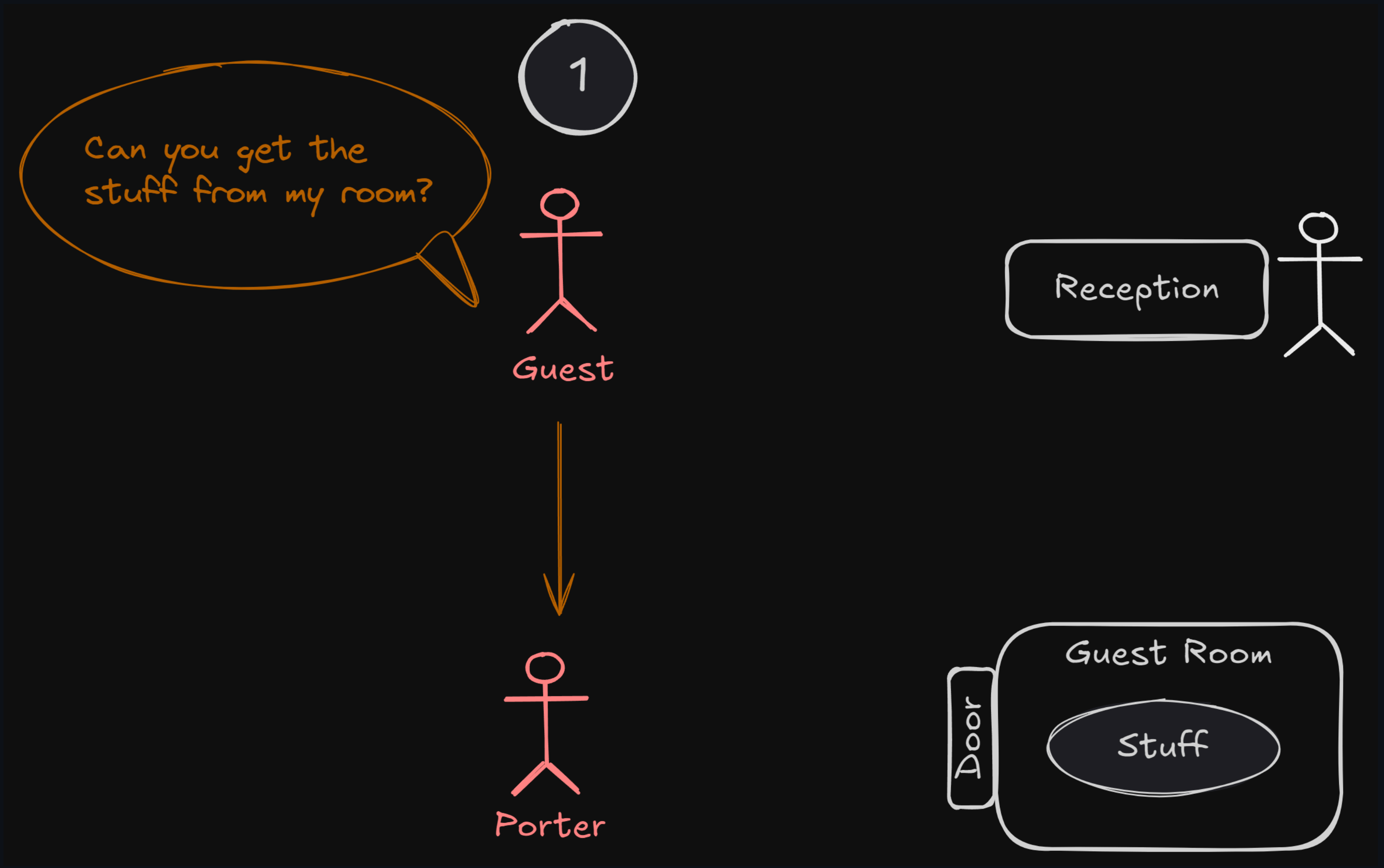
Authorization code flow

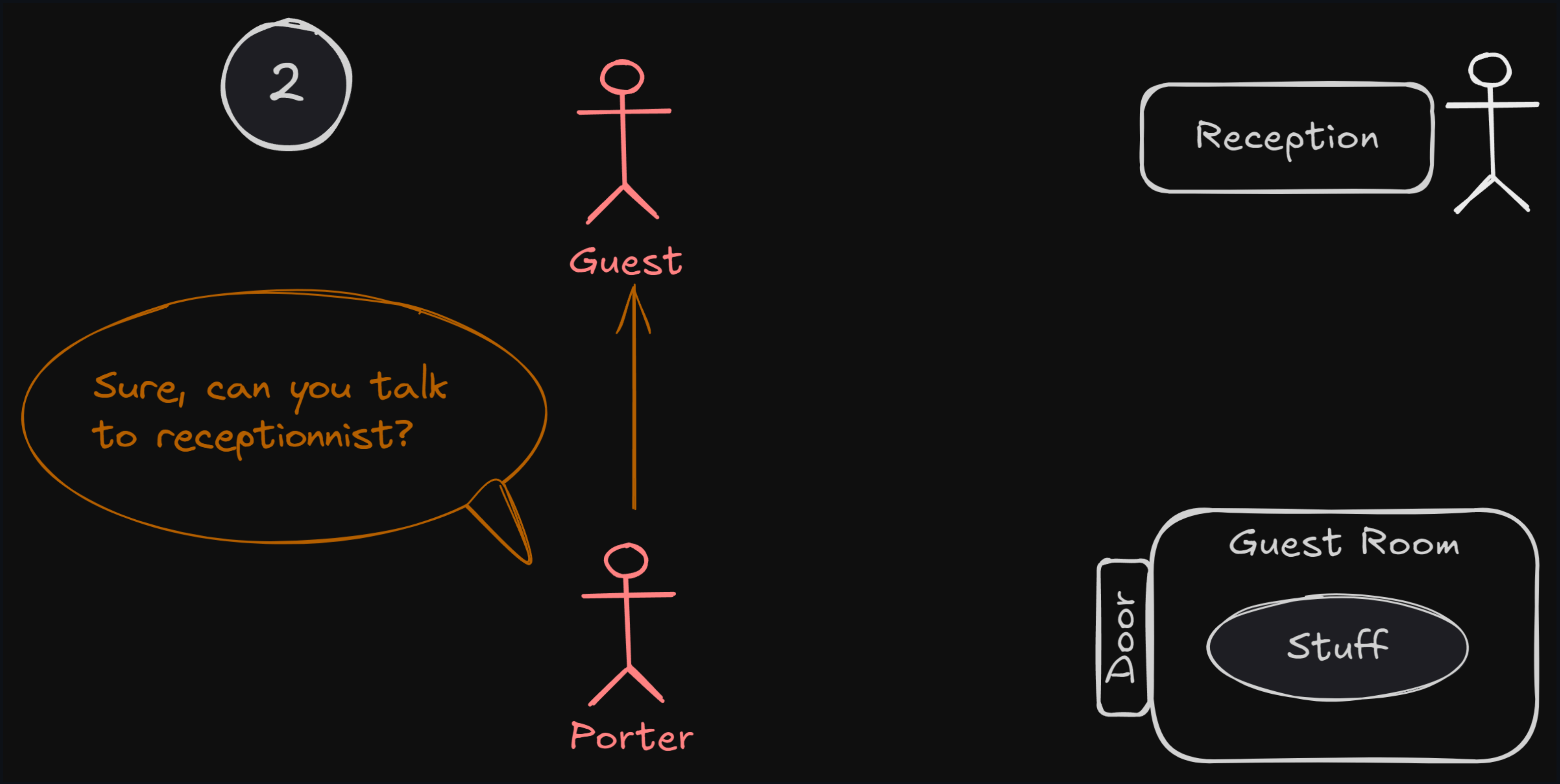
> In real life

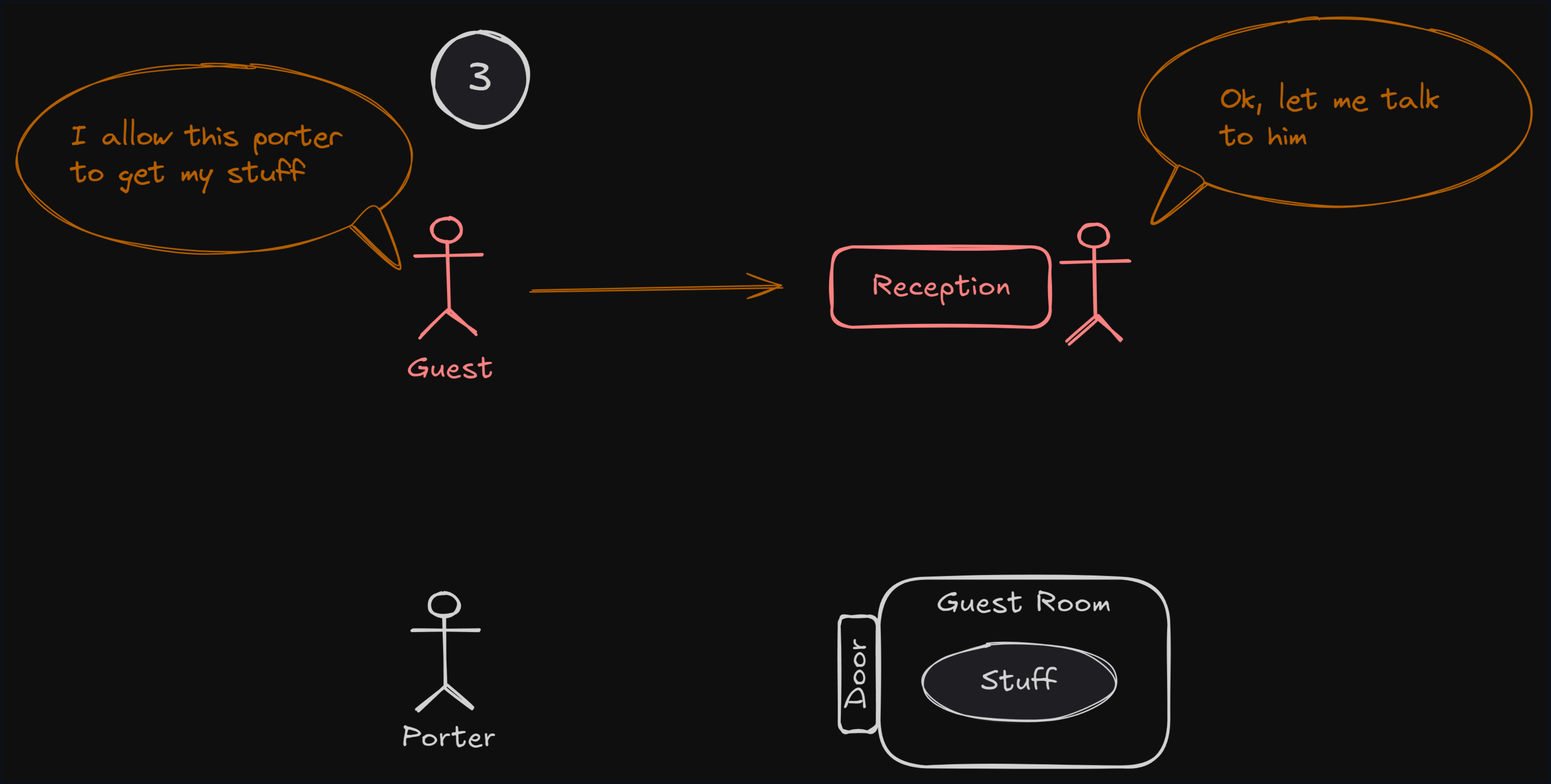
Setup

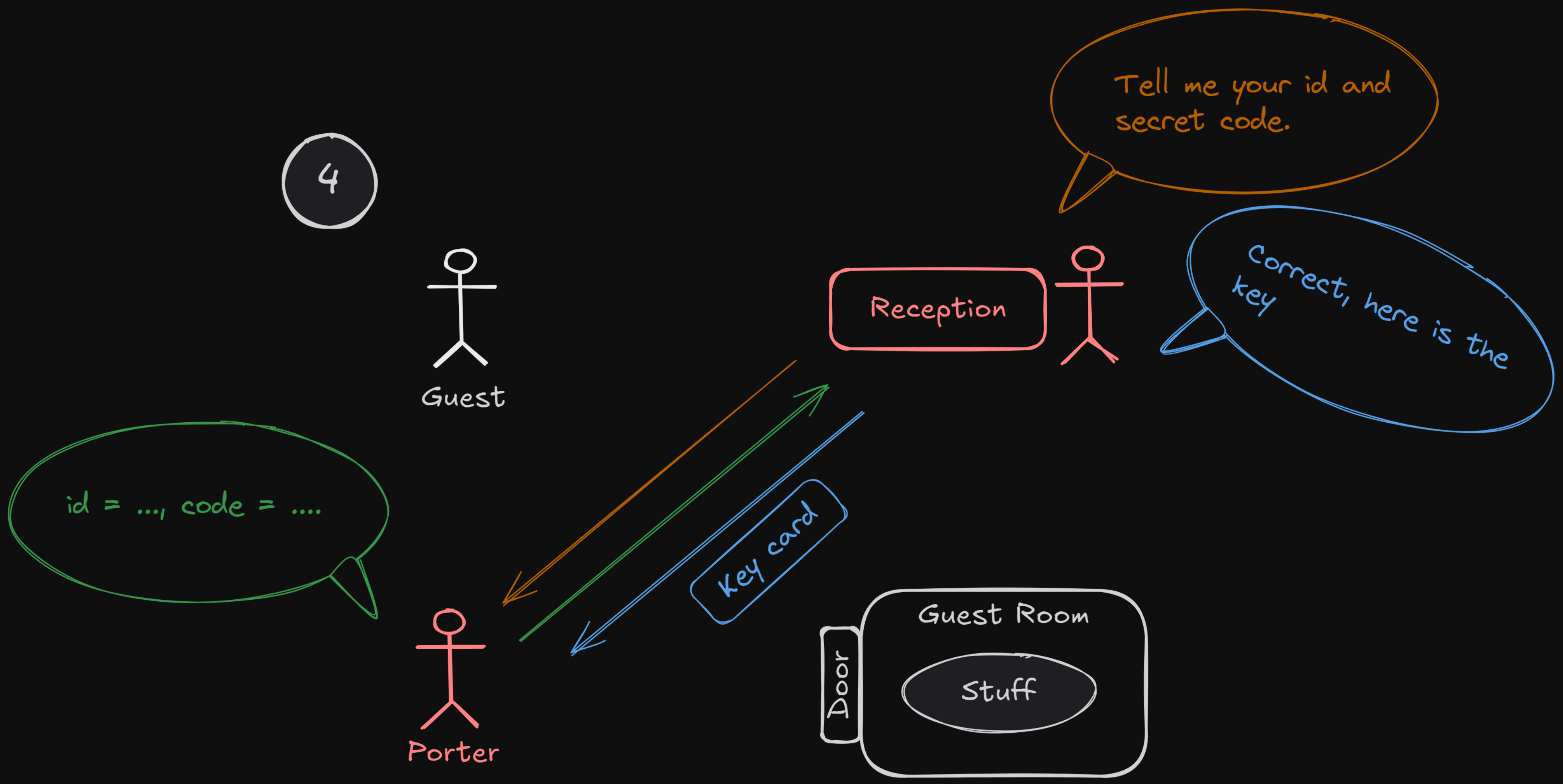
- You are a guest at a hotel.
- You already checked out.
- You forgot your stuff in the room.
- You want a porter to get your stuff for you.

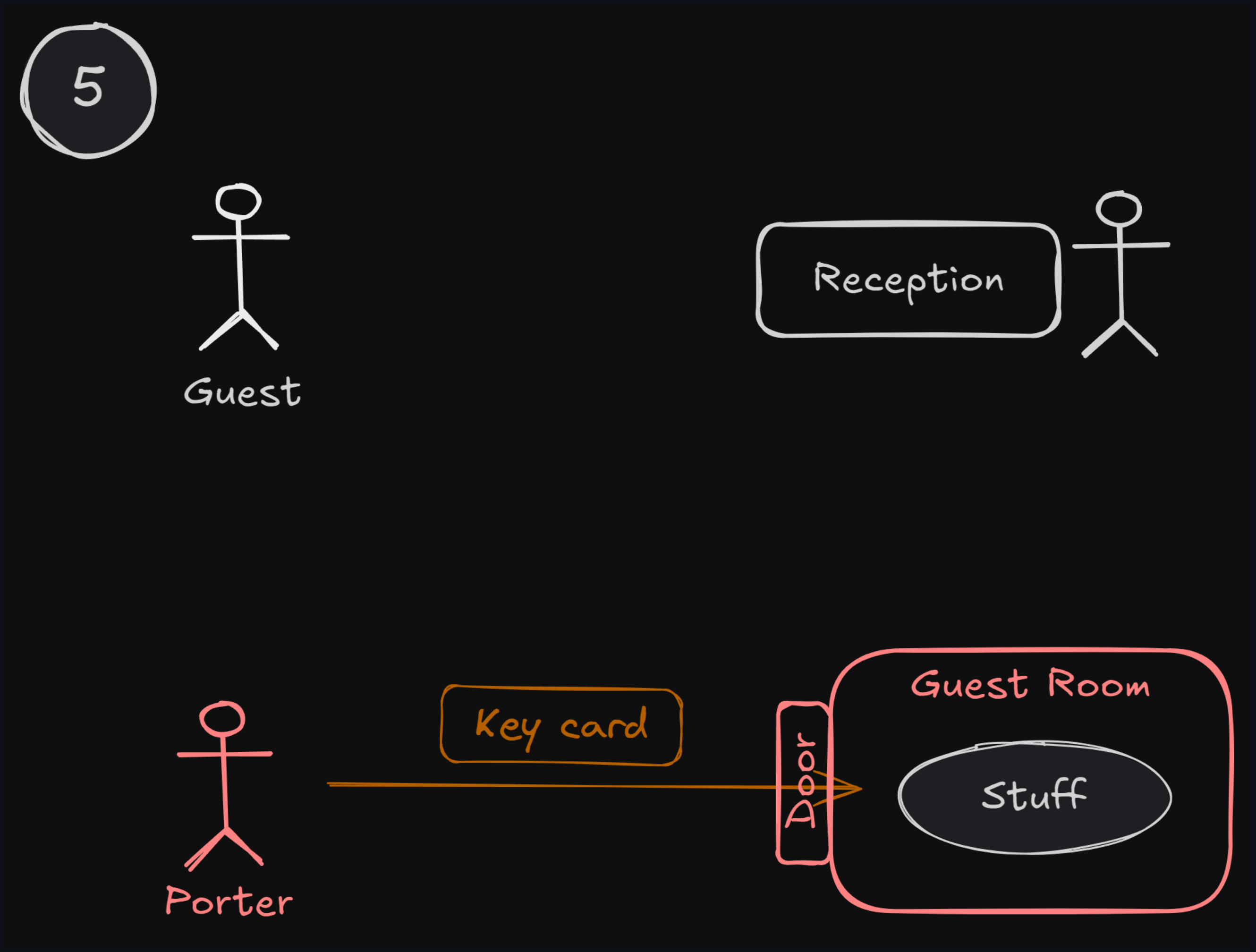



1
Can you get the stuff from my room?
Guest
Porter
Reception
Guest Room
Door
Stuff


2
Guest
Sure, can you talk to receptionnist?
Porter
Reception
Guest Room
Door
Stuff


3
I allow this porter to get my stuff
Guest
Reception
Ok, let me talk to him
Porter
Door
Guest Room
Stuff


4
Guest
Tell me your id and secret code.
Reception
Correct, here is the key
id = ..., code = ...
Key card
Porter
Guest Room
Door
Stuff

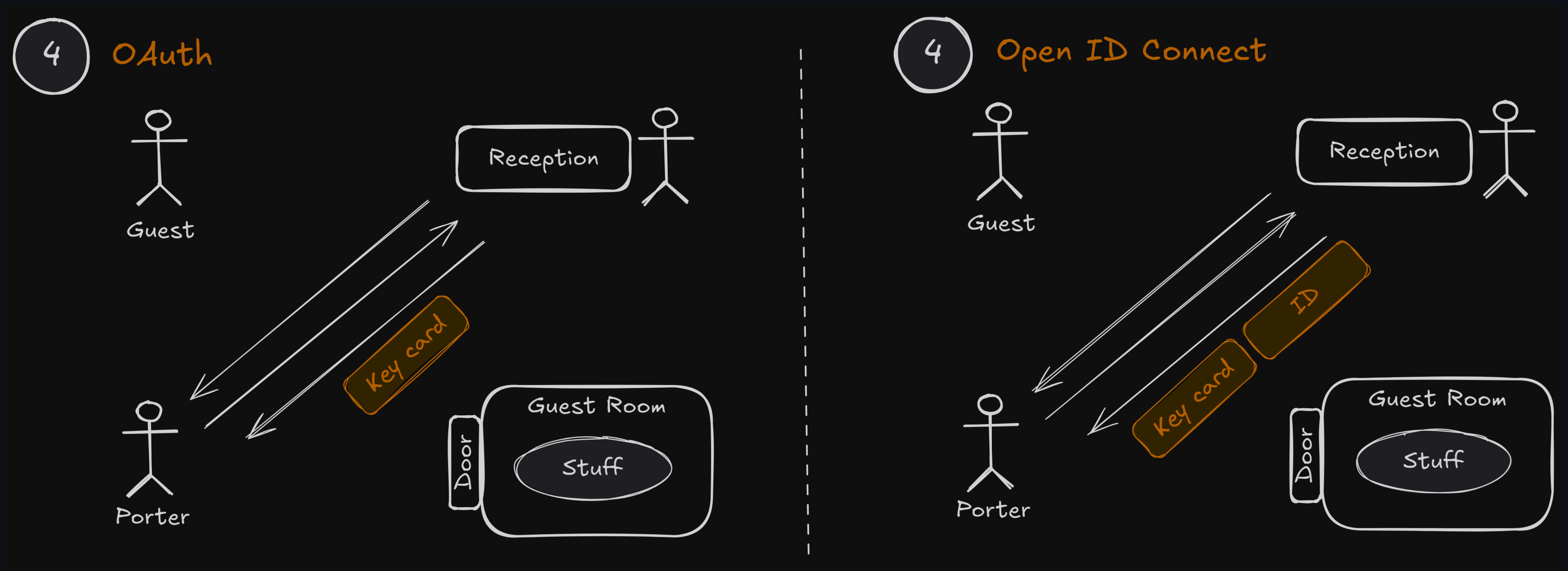

5
Guest
Reception
Key card
Door
Guest Room
Stuff
Porter

Authorization code flow

- You (`guest`) authorize `porter` to access your resource.
- `porter` do not need to know who you are.
- The keycard reader at the door also don't need to have your information.

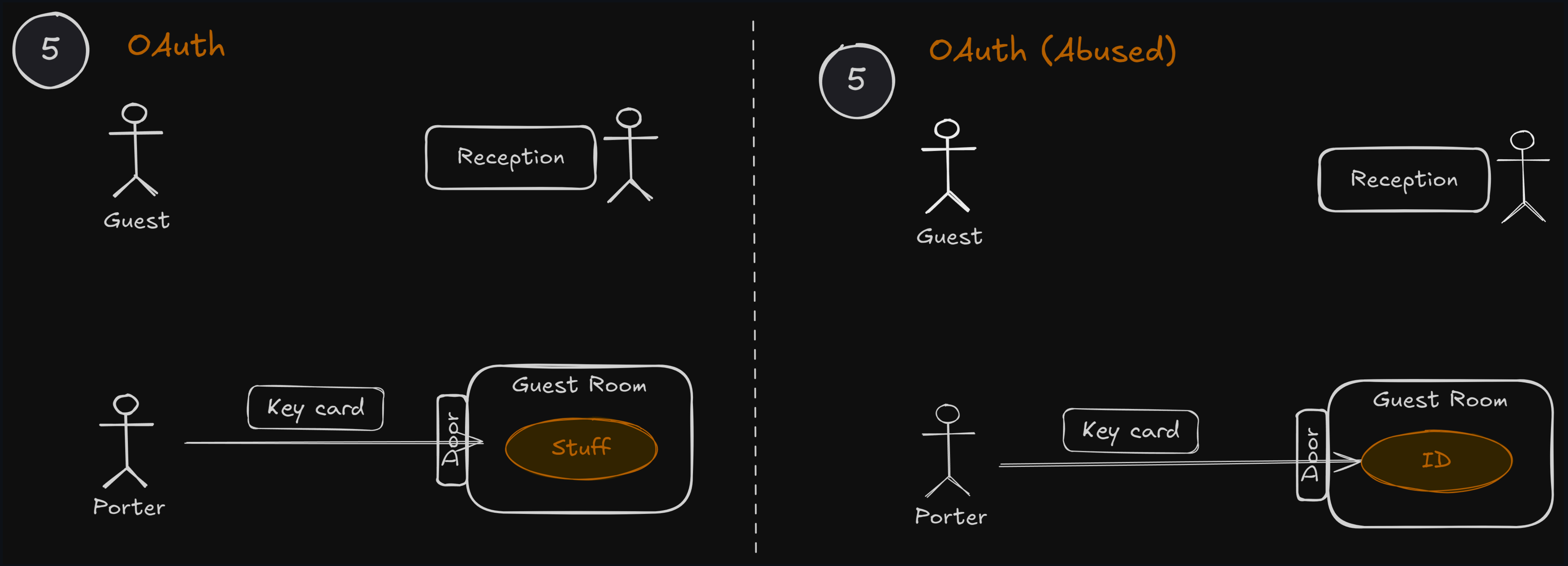
Authentication?

- But what if the porter wants to know who you are.
- There are two ways.


4
OAuth
Guest
Reception
Key card
Porter
Guest Room
Door
Stuff
4
Open ID Connect
Guest
Reception
ID
Key card
Porter
Guest Room
Door
Stuff

OpenID Connect (OIDC)

- Thin layer that sits on top of OAuth 2.0
    - Adds login and profile information about the person who is logged in.
- When a "Authorization Server" supports OIDC, it is sometimes called an "Identity Provider".
- Not all servers support OIDC.



This is what we are using.