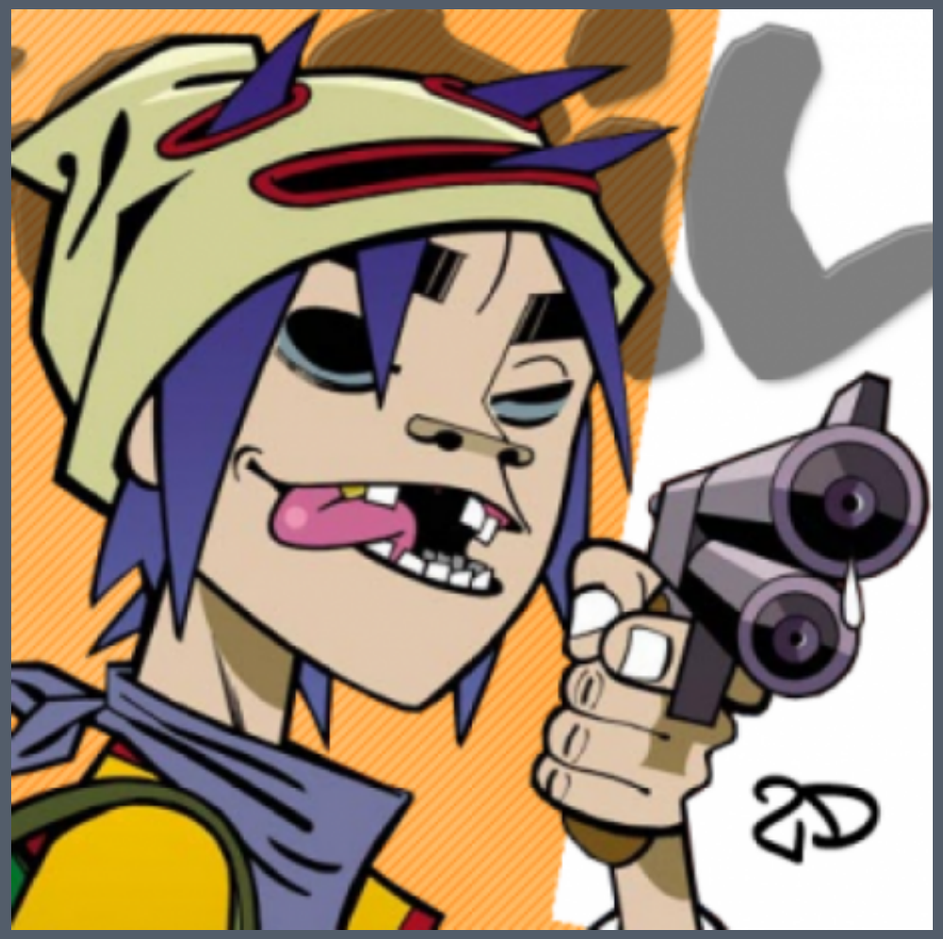
JSON Web Tokens

Thameera Senanayaka

@thameera



twitter.com/thameera

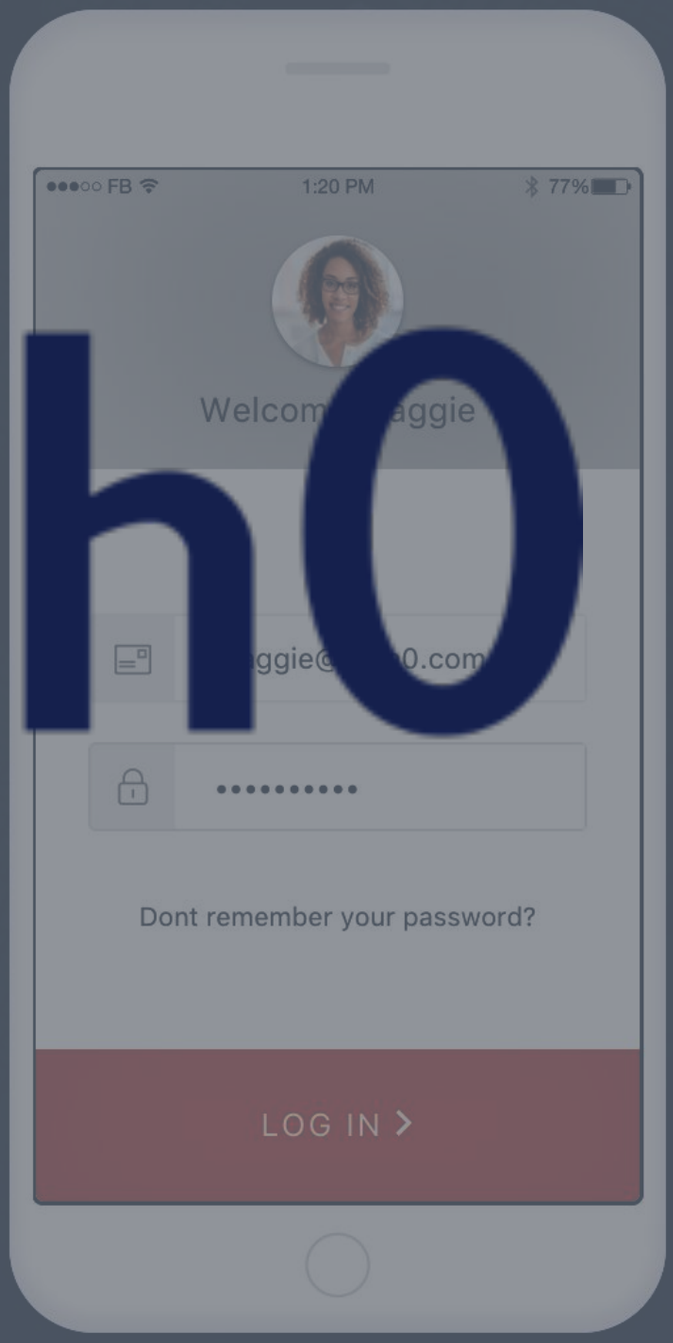
The new way to solve Identity

We solve complex identities with an extensible, easy to integrate platform that secures billions of users every

USE AUTH0 FOR FREE

WATCH VIDEO

No credit card required



OLIVLABS

LIBRARY

No. 44301 Expires NOV 18 1951

Pearce, Ralph M.

123 Main St.

San Jose Public Library

San Jose, California

Present this card each time you borrow a book


CLOSED
high
Feng Shui

No. 44301 Expires NOV 18 1951

Pearce, Ralph M.

123 Main St.

San Jose Public Library

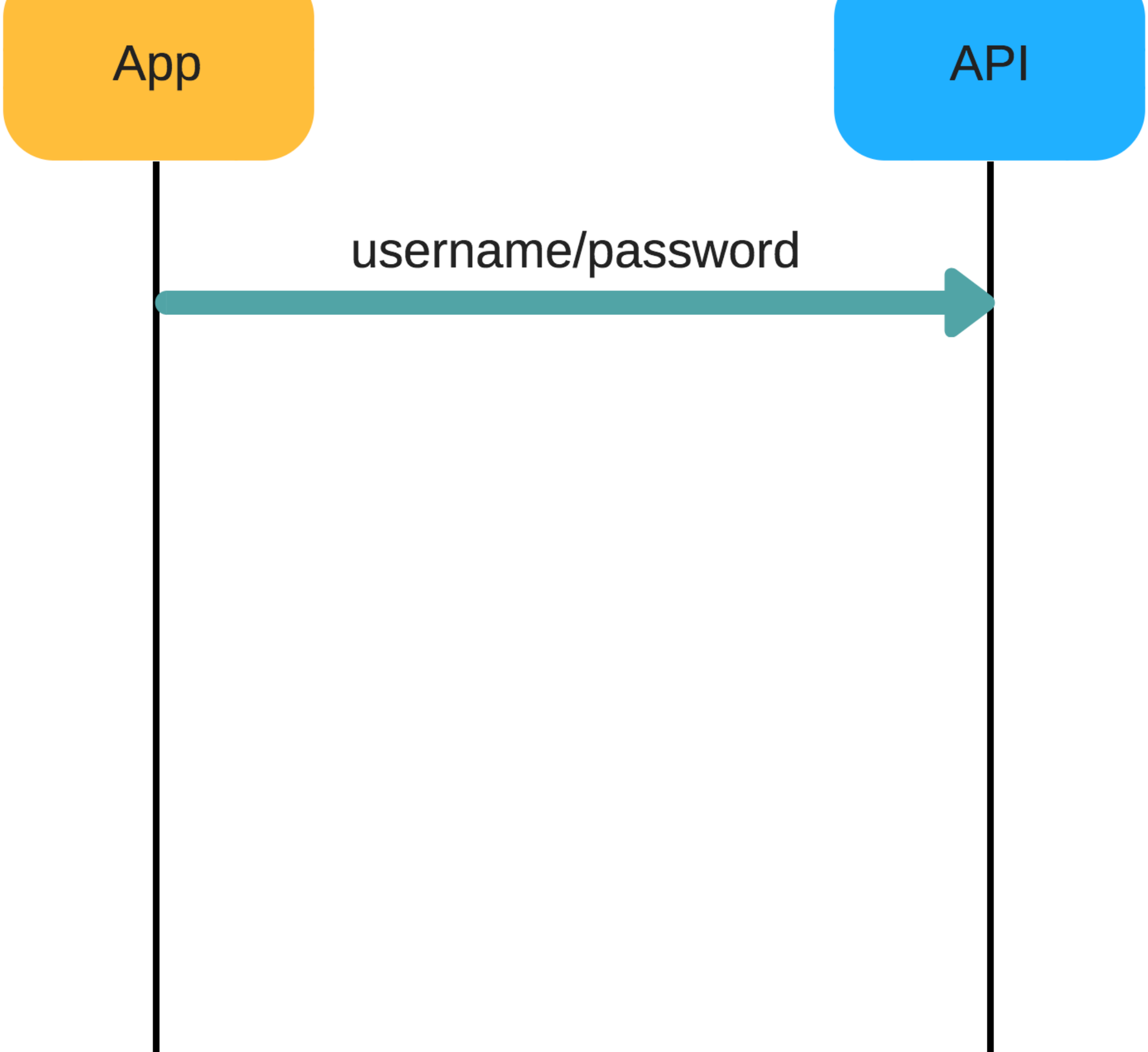
San Jose, California

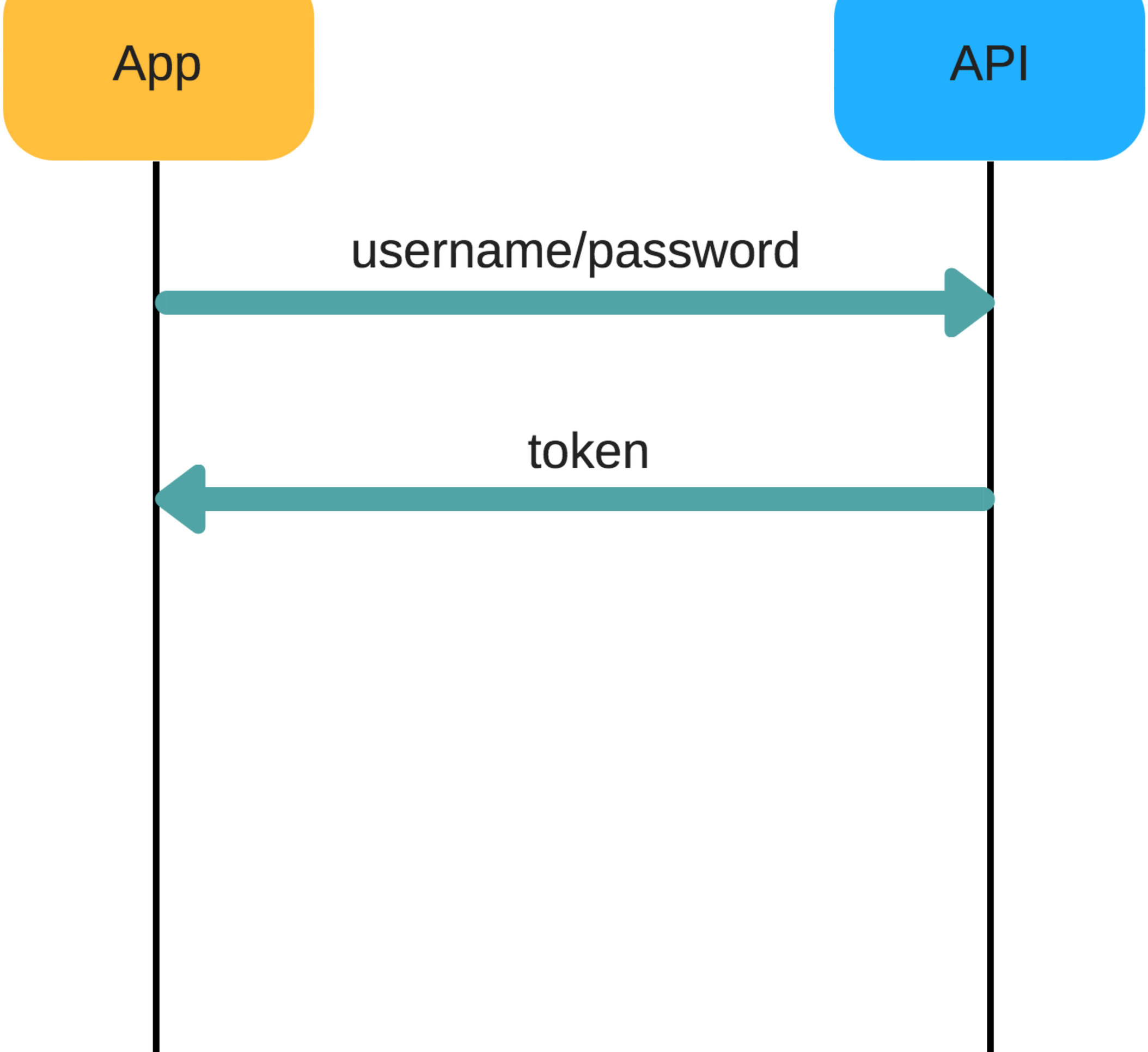
Present this card each time you borrow a book

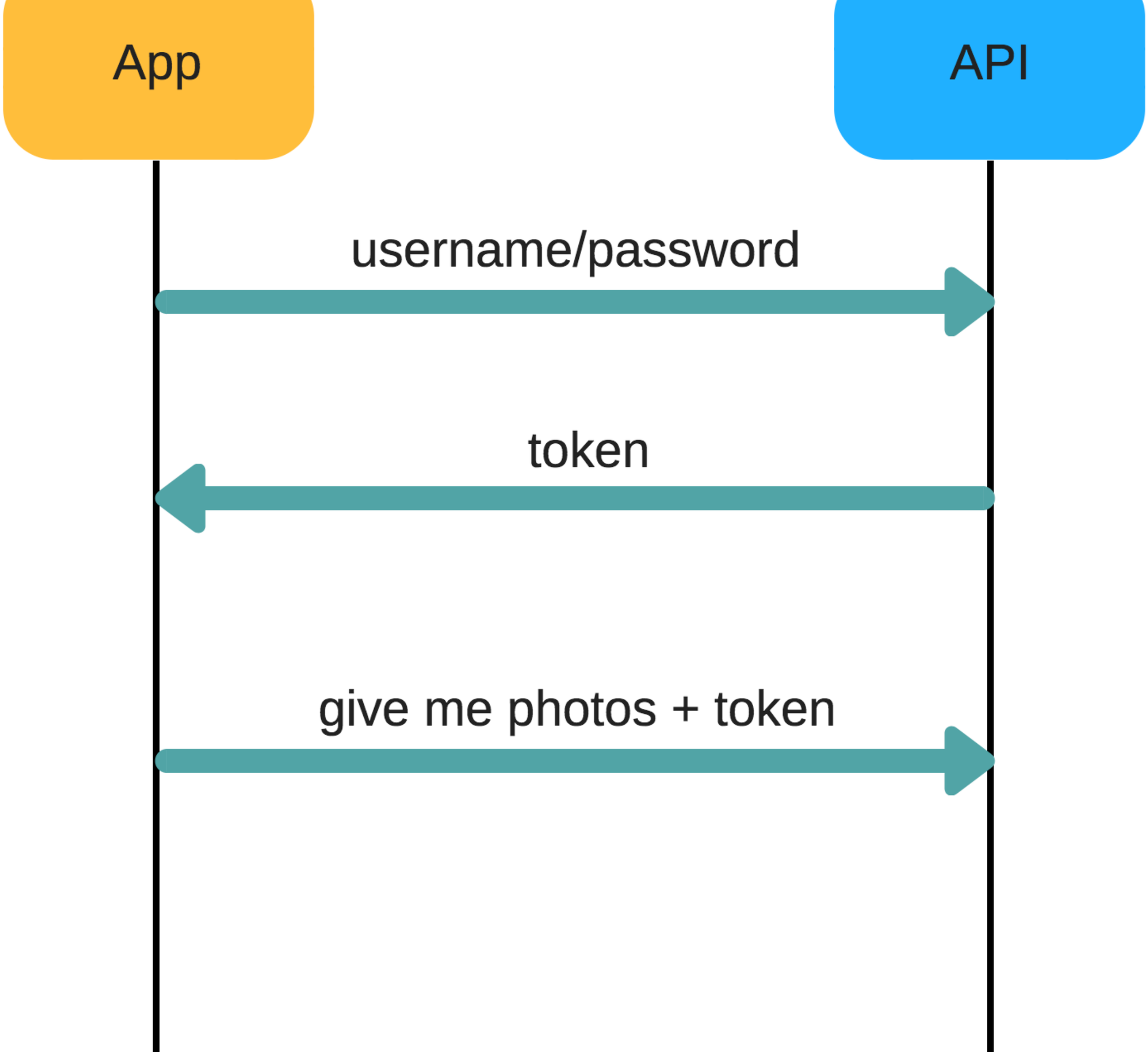

IDEA
NEWS
BUSINESS
TECHNOLOGY
SPORT
INTERNATIONAL

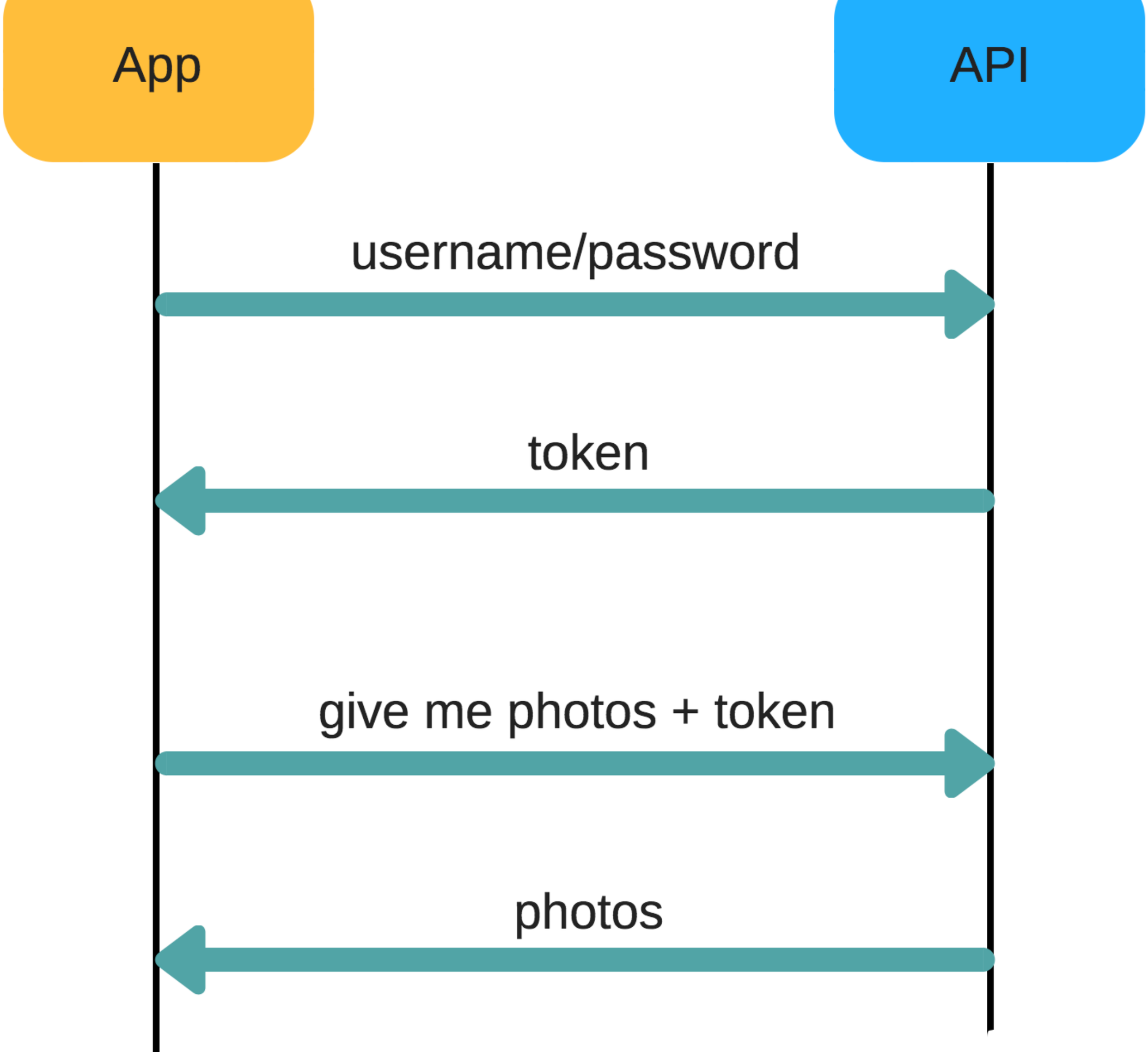
App

API


App
API
username/password


App
API
username/password
token


App
API
username/password
token
give me photos + token


App
API
username/password
token
give me photos + token
photos

JSON Web Tokens

aka JWT

RFC 7519

https://tools.ietf.org/html/rfc7519

An open standard for passing claims between two parties

JSON Web Token

```
{
    "name": "dinesh chandimal",
    "age": 27,
    "strengths": [],
    "weaknesses": ["captaincy"]
}
```

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJuY
W1lIjoidGhhbWVlcmEiLCJzdWIiOiJhdXRoMH
w1NzFkZmM4NzJmMWQ1ZTU2MDI2NzAyZjYi
LCJleHAiOjE1MDI5MTkwMTZ9.lmqptC83nKo
mEfsgQcmcgOydoJi5j80gOuU2ClWSA0Q

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.**eyJuY**
W1lIjoidGhhbWVlcmEiLCJzdWIiOiJhdXRoM
Hw1NzFkZmM4NzJmMWQ1ZTU2MDI2NzAy
ZjYiLCJleHAiOjE1MDI5MTkwMTZ9.*lmqptC8*
3nKomEfsgQcmcgOydoJi5j80gOuU2ClWSA0
Q

JWT.io

Demo

Signing algorithms

- → HMAC
- → RSA
- → ECDSA

Payload

Reserved claims

iss, sub, exp, aud, ...

How to build a JWT

payload


```
{
    "name": "jon snow",
    "house": "stark",
    "sub": "1234"
}
```

base64 encode the payload

```
bPayload = base64( payload )
```

eyJuYW1lIjoiam9uIHNub3ciLCJob3VzZSI6InN0YXJrIiwic3ViIjoiMTIzNCJ9

header

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

base64 encode the header

```
bHeader = base64( header )
```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9

signature

```
signature = sign( bHeader + '.' +
bPayload, secret )
```

```
sign( 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYW1lIjoiam9uIHNub3ciLCJob3VzZSI6InN0YXJrIiwic3ViIjoiMTIzNCJ9', 'mySecret123' )
```

```
bSignature = base64( signature )
```

TiMShk7JvK4zR3Kn4It5+H8N4KrGdVL3f/FTw4WTUXM=

Add everything together

```
jwt = bHeader.bPayload.bSignature
```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.**eyJuY**
W1lIjoiam9uIHNub3ciLCJob3VzZSI6InN0YX
JrIiwic3ViIjoiMTIzNCJ9.*TiMShk7JvK4zR3Kn4I*
t5+H8N4KrGdVL3f/FTw4WTUXM=

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuY
W1lIjoiam9uIHNub3ciLCJob3VzZSI6InN0YXJr
IiwiciI6IjoiMTIzNCJ9.TiMShk7JvK4zR3Kn4It5+
H8N4KrGdVL3f/FTw4WTUXM=

Live coding

Is the JWT encrypted?

JWTs are signed,

not encrypted

How does the server know that we didn't mess with the JWT?

Don't Reinvent

The Wheel

JWT libraries are available for almost every language and framework

Creating a JWT with `jsonwebtoken`

```
const jwt = require('jsonwebtoken')
const token = jwt.sign({ name: 'thameera' }, 'mySecret123')
```

Verifying a JWT

```
const jwt = require('jsonwebtoken')
try {
    const decoded = jwt.verify(token, 'mySecret123')
} catch(e) {
    console.log('Invalid token!!!')
}
```

Advantages of JWTs 😎

Compact

Stateless

Scalable

Decoupled

Cross Domain

Sessions

vs

Tokens

Pass by Reference **vs** Pass by Value

Where to go from here?

JSON Web Token Specification

RFC 7519

Internet Engineering Task Force (IETF)
Request for Comments: 7519
Category: Standards Track
ISSN: 0000-0000

M. Jones
Microsoft
J. Bradley
Ping Identity
N. Sakimura
NRI

May 2015

JSON Web Token (JWT)

Abstract

JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted.

https://tools.ietf.org/html/rfc7519

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

JWT Handbook

https://goo.gl/HyzEZA

VERSION 0.12.0

JWT Handbook

Ever wondered how JWT came to be and what problems it was designed to tack
curious about the plethora of algorithms available for signing and encrypting JW
you interested in getting up-to-speed with JWTs as soon as possible? Then this
for you.

What's new in this version

- Added descriptions of signature verification algorithms for HS256, RS256 an
- With clean-room sample implementations

PAY WITH A TWEET

DOWNLOAD VIA EMAIL

We would like to request permissions to post only this tweet, but Twitter OAuth scopes are very broad. You
always revoke permissions after downloading the book. If you don't feel comfortable granting these
permissions, feel free to contact us and we will send you a private download link.

Thank you!