Web Engineering

Dr. Michael Lesniak
mlesniak@micromata.de

03. Juni 2019

MICROMATA

Heute...

- Organisatorisches
- Review Übungsaufgaben
- Authentifizierung in Web-Anwendungen
- Code
  - Frontend: Build-System
  - Backend & Frontend: Authentifizierung mit OAuth2 und JWT

---------- 5 Minuten Pause ----------

- Übung (optional, aber Pflicht-Aufgabe)

Organisatorisches

- ¯_(ツ)_/¯ ...

Review Übungsaufgabe

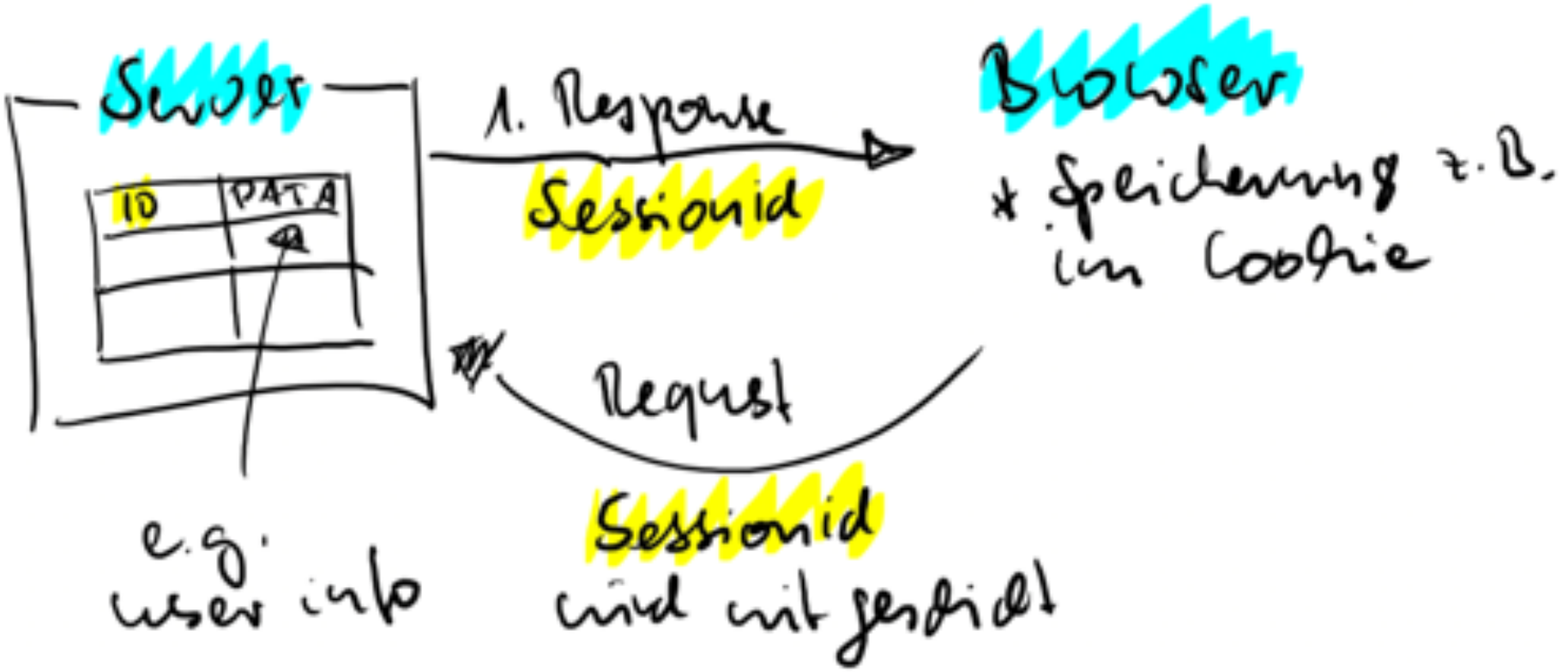
- Heroku
  - Account anlegen
  - CLI Tool installieren und konfigurieren (heroku login)
- Backend und Frontend:
  - Deployen
  - Kommentare für Post (statt Comment) ermöglichen
  - Alternativ zu URL auch reine Eingabe einer Beschreibung ermöglichen

Authentifizierung in Web-Anwendungen

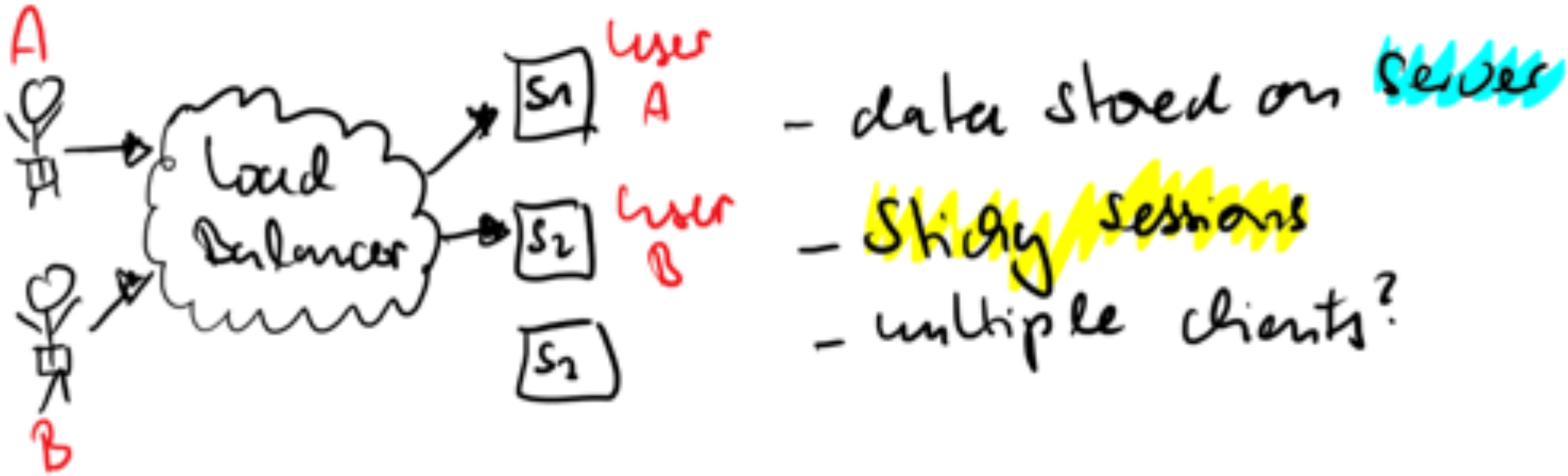
- User
  - Registrieren
  - Verwalten
  - Authentifizieren
  - Authorisieren
  - Anzeigen

- Welches Problem soll **eigentlich** gelöst werden?
  - „Wer bist Du?“ (HTTP ist zustandslos; über mehrere Requests)
  - „Darfst Du das?“

Sessions


Server
ID
DATA
1. Response
SessionId
Browser
* Speicherung z.B. im Cookie
Request
SessionId wird mitgeschickt
e.g. user info

Sessions und Load Balancing



Tokens

- data stored on client
- Send in a http header
- on each request, any server in possible

JWT

Header

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
```

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

Payload

```
eyJzdWIiOiJtbGVzbmlhayIsIm5hbWUiOiJNaWNoYWVsIExlc25pYWsiLCJpZCI6MX0
```

```
{
  "sub": "mlesniak",
  "name": "Michael Lesniak",
  "id": 1
}
```

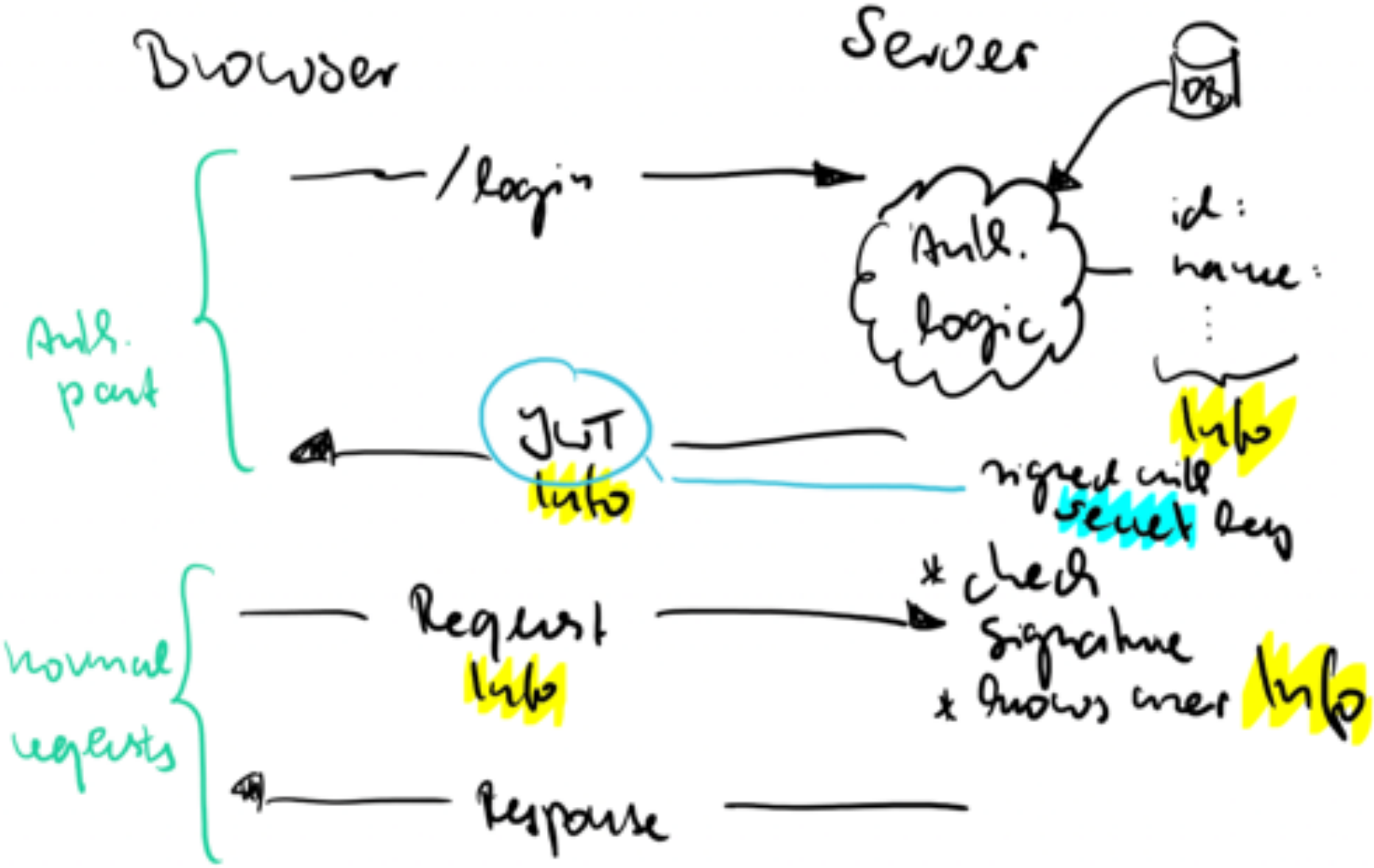
Signature

```
bdxg_mDa3knK_bZCw9oEGCjupvSeycaDfb37Dt31q9g
```

```
HMACSHA256(
    base64UrlEncode(header) +
    "." +
    base64UrlEncode(payload),
  <password>
)
```

```
HTTP-Header Authorization: Bearer eyJhb … .eyJzdWIi … .bdxg_mDa3k …
```

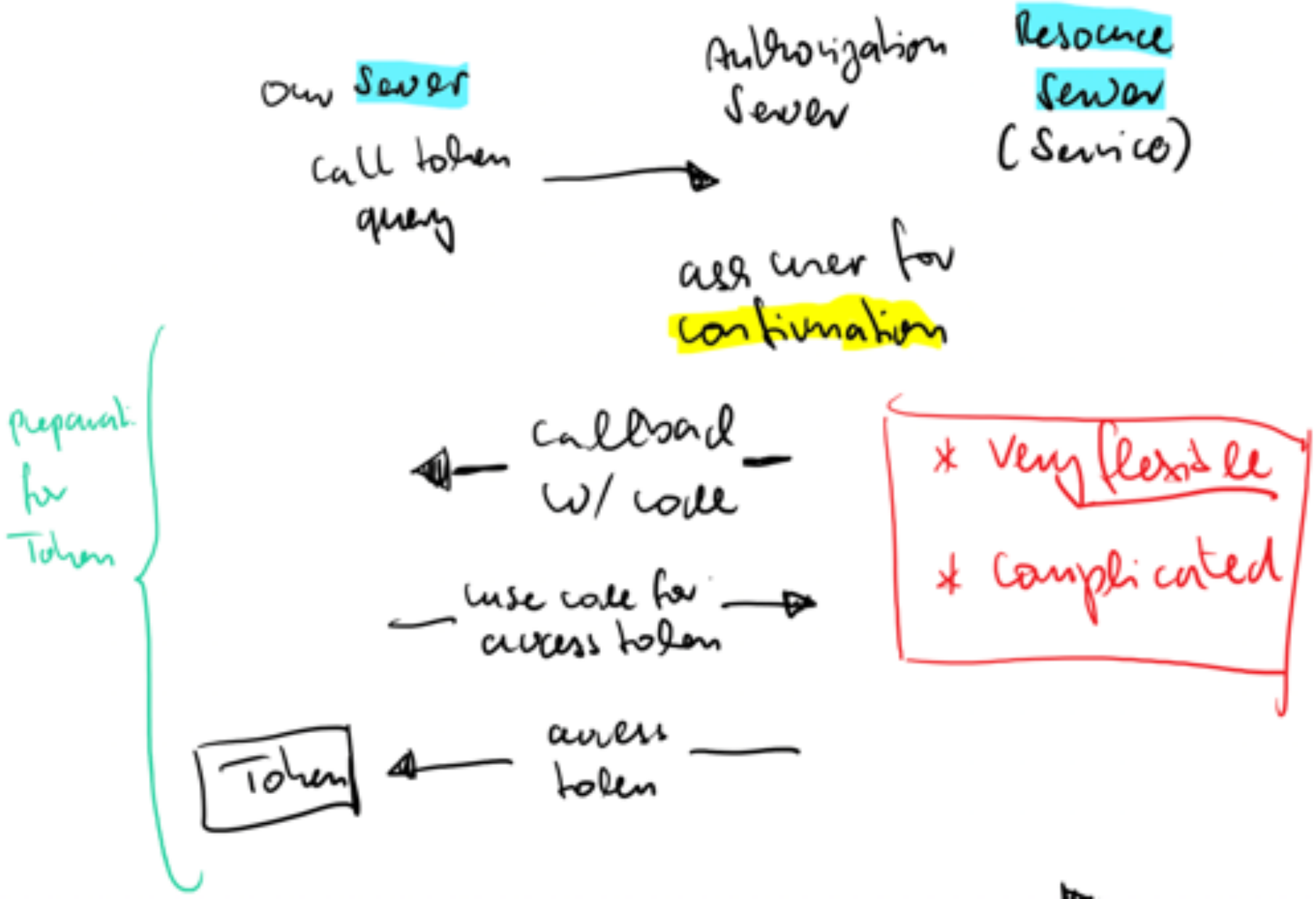
JWT („standard“ request and response)



Wie kommen User in das System?

- Klassisch über eigene Benutzerverwaltung
  - Registrierung
  - Passwort und Email Management
  - ...
- Extern, z. B. über GitHub, Facebook, ...
  - Idee: Abfrage von verifizierten Daten über OAuth
  - Speicherung im JWT Token für zukünftige Kommunikation

OAuth (protocol, simplified)



Parcel

- Problem
  - Frontend besteht aus vielen Dateien (.js(x), css, html, .png, …)
  - HTTP-Requests sind teuer (Erinnerung: GraphQL und REST)
- Paketierung baut alle benötigten Teile zusammen
- Optional: Transpilation, …

- Es gibt sehr viele Tools (Bower, Webpack, Parcel, …)
- Wir nehmen `parcel` (convention over configuration, YMMV!)

Code-Walkthrough

- Frontend: Build-System Parcel
- Backend & Frontend: Authentifizierung mit OAuth2 und JWT

Übung

- Commit für Commit auschecken und verstehen (ggf. fragen!)

- Refactoring Backend
  - Services
  - Kleinere Methoden, Kommentare, …
  - Debug-Mode für Authentifizierung („ohne Token“)
- Refactoring Frontend
  - authentication.jsx als „sauberes Objekt“
  - …

Optional

- Bis 09.06.2019 23:59 per Slack-Nachricht an @Michael Lesniak
  - eine konkrete Idee für ein Projekt
  - einen Wunschpartner (Vor- und Nachname) für das Projekt
  - ggf. Stärken und Vorlieben (aber: Full-Stack!)

Pflicht

Voraussetzung für Projekt-Teilnahme