

Web Engineering

Dr. Michael Lesniak
mlesniak@micromata.de

03. Juni 2019

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



A form for creating a post. It has three input fields: 'title', 'url', and 'text'. The 'title' and 'url' fields are single-line text boxes. The 'text' field is a larger multi-line text box. Below the 'text' field is a 'submit' button. The word 'or' is placed between the 'url' and 'text' fields, indicating that either a URL or a text description can be provided.

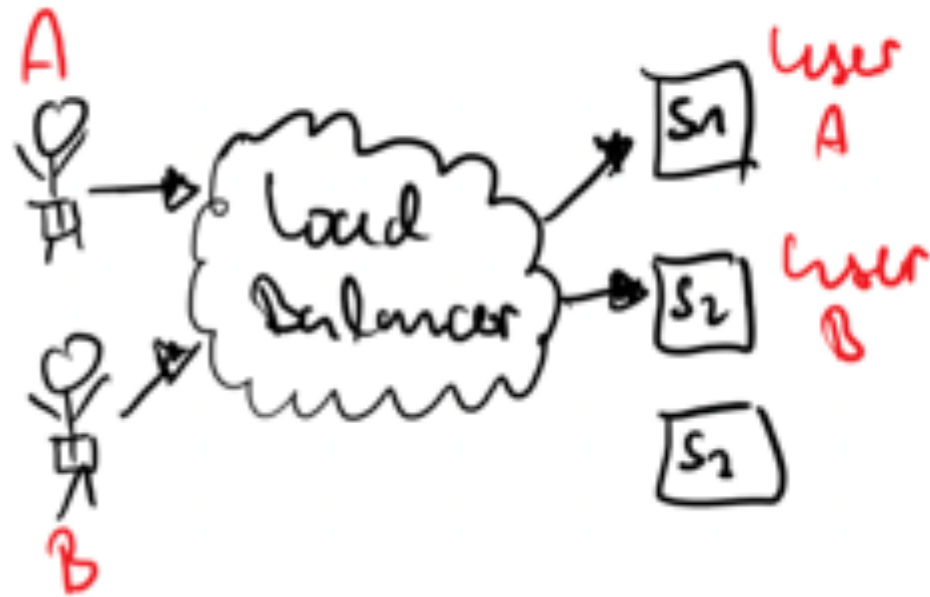
Authentifizierung in Web-Anwendungen

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

Sessions

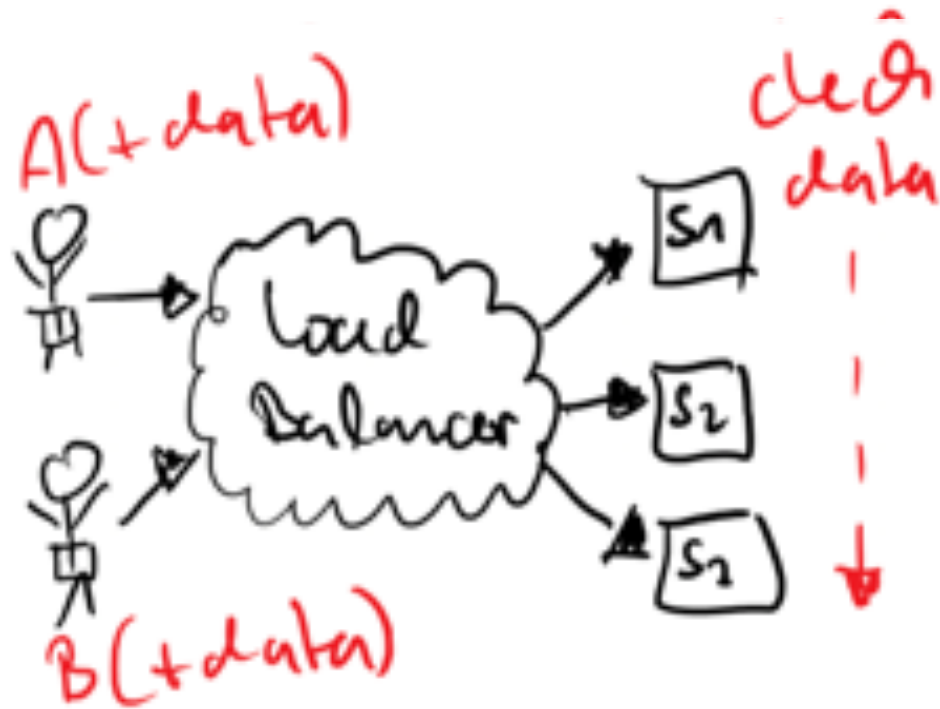


Sessions und Load Balancing



- data stored on **server**
- **Sticky sessions**
- multiple clients?

Tokens



- data stored on **client**
- Send in a http header
- on each request, **only** **newer** is possible

JWT

Header

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
```

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

Payload

```
eyJzdWIiOiJtbGVzbnR5cCI6IkpXVCJ9
```

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

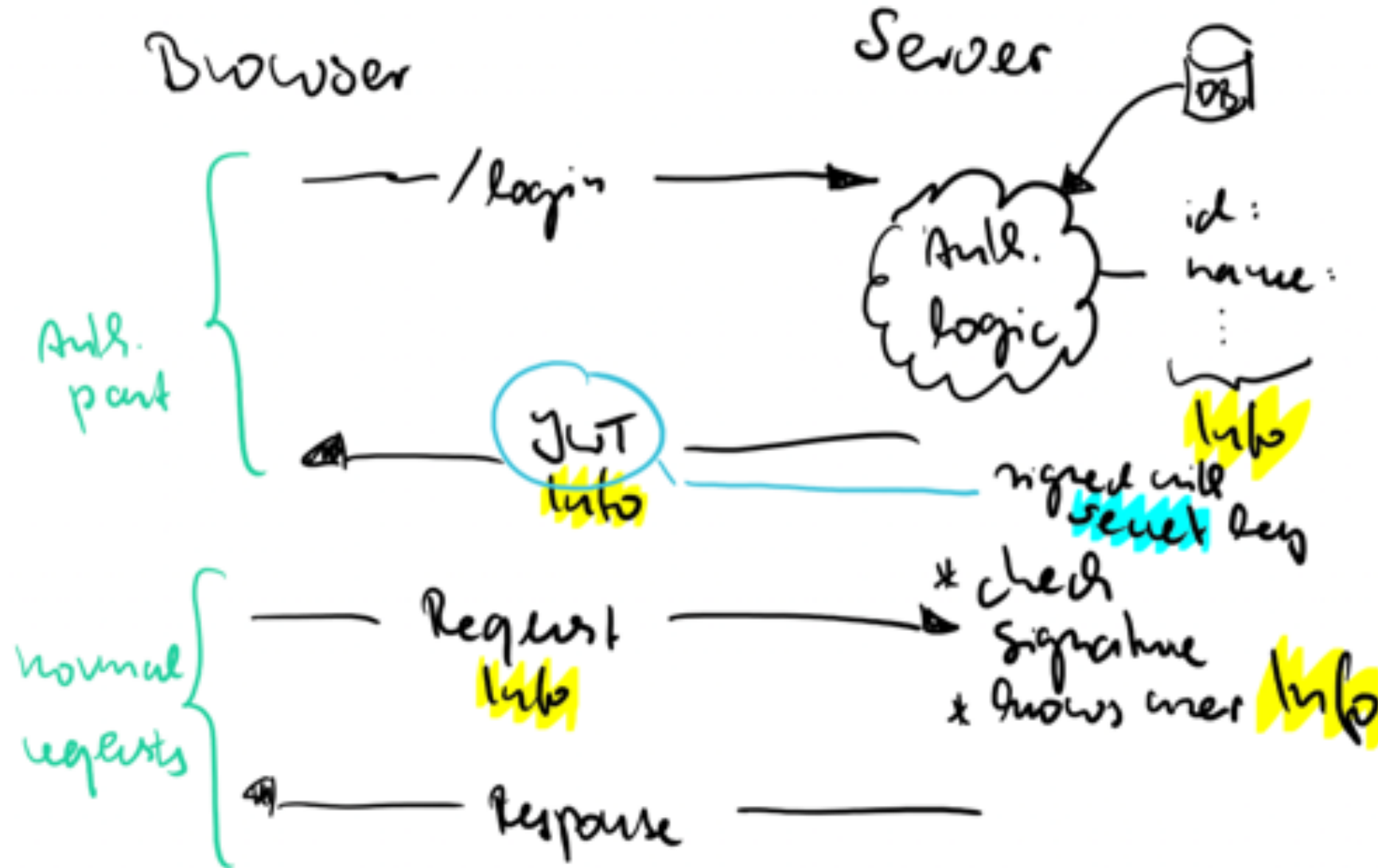
Signature

```
bdxg_mDa3knK_bZCw9o  
EGCjupvSeycaDfb37Dt  
31q9g
```

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

HTTP-Header **Authorization:** Bearer eyJhbeyJzdWIibdxg_mDa3k ...

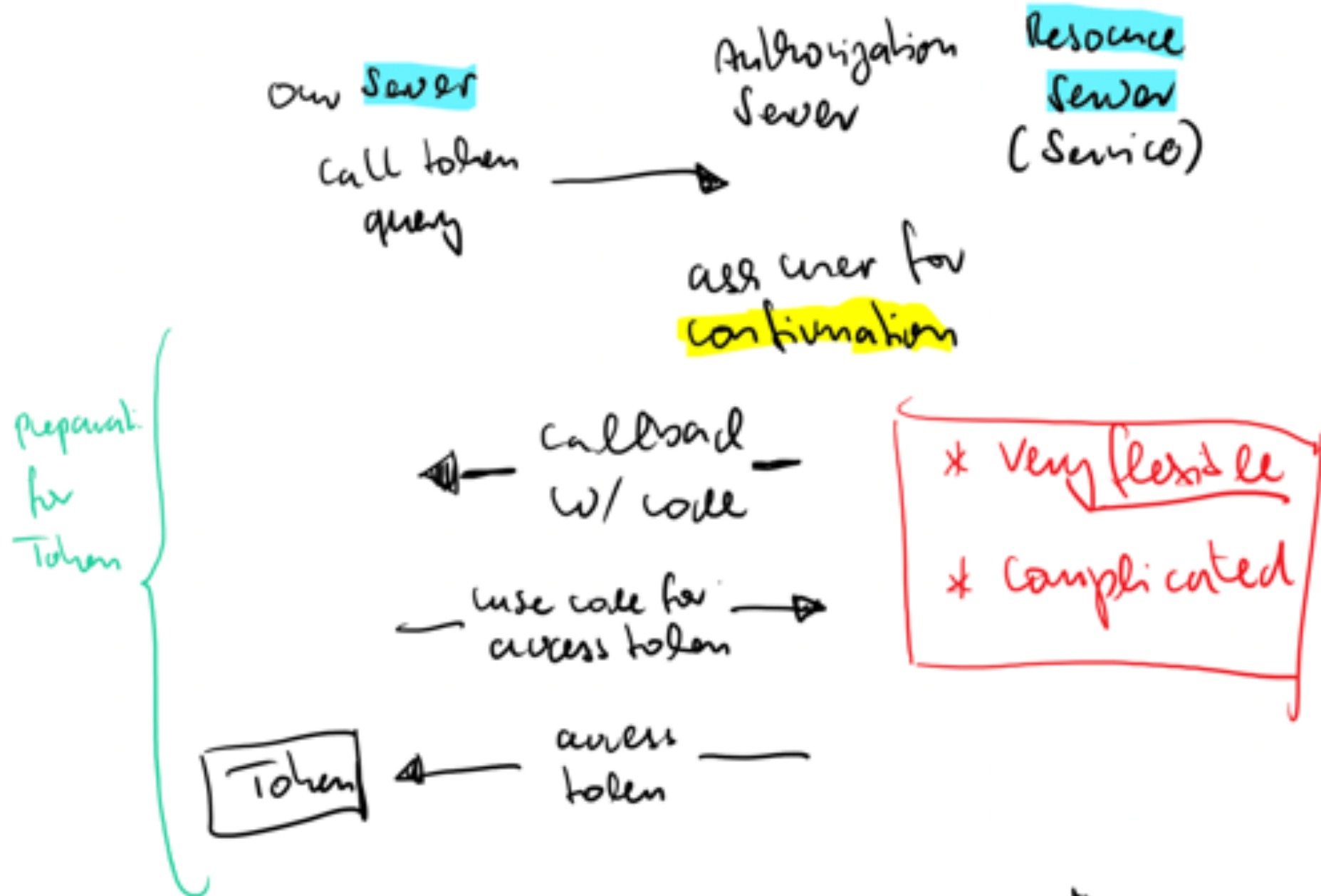
JWT („standard“ request and response)



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 **08.06.2019 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