Zwischenprojekt

Kleinanzeigen-Frontend

Aufgabenstellung

Entwickelt ein Frontend für das bereitgestellte Backend.

Geht anhand der User-Stories vor. Haltet euch an die priorisierte Reihenfolge (von oben zuerst).

Verwendet React.

Ziel: Für bearbeitete User Stories effektive Lösung liefern.

"Lieber wenige effektive, als viele schlechte Stories."

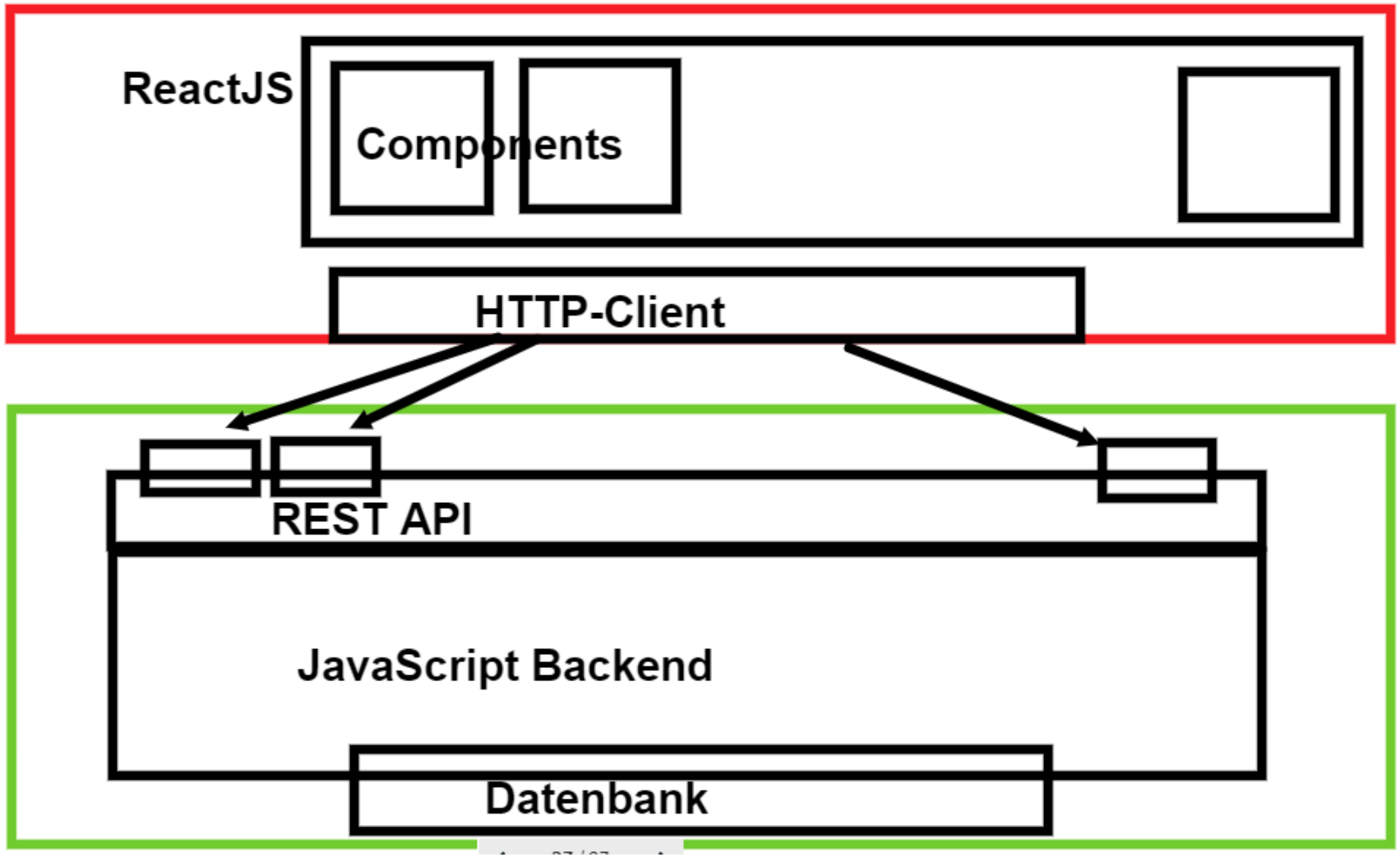
Kleinanzeigen Backend

Das Backend steht online zur Verfügung

Dokumentation (API Explorer) unter

https://awacademy-kleinanzeigen.azurewebsites.net/explorer/

Architektur



Teams

Team	Members		
1	Ceren	Richard	Sascha
2	Cristina	Marvin	Matthäus
3	Patrick	Philipp	Vivien
4	Dan	Paddy	Tim
5	Lin	Osama	Ricky

Wiederholung: REST API

Klärt im Team:

Wiederholung:

- HTTP Methode
- Path Parameter (…/:id/...
- Query Parameter
- Request Body

Welchen HTTP-Client verwendet ihr im Team (fetch, jQuery.ajax, ...)?
Wie konfiguriert ihr den gewählten HTTP-Client?

Git-Repository

Issue-Board

Startschuss

Erste Schritte:

- Projekt anlegen
- Zusammenarbeit mit Git sicherstellen
- Verbindung mit REST-API testen
- Alle User Stories lesen und verstehen

Später heute: Details zur REST-API

- Query Parameter Encoding
- Authentifizierung (Login)

Ende des Projekts:

- Code complete am Dienstag, 26.5. EOB
- Präsentation der Ergebnisse Mittwoch, 13:30 Uhr

Code Beispiele

REST API: Query Parameter

Einige der API-Endpunkte können Anfragen mit komplexen Query-Parametern behandeln.

Beispiel:

`https://awacademy-kleinanzeigen.azurewebsites.net/ad?filter=...`

Filter-Parameter kann enthalten:

- limit (Maximale Anzahl an Einträgen in der Antwort)
- offset (Überspringe die ersten X Einträge)
- where (Bedingungen für die gesuchten Einträge)

Filter-Parameter: Komplexes Beispiel

```
{
   limit: 20,
   offset: 0,
   where: {
     and: [
       {location: "Berlin"},
       {
          or: [
            {title: {like: "%bike%", options: "i"}},
            {description: {like: "%bike%", options: "i"}},
          ]
       },
     ]
  },
}
```

Finde die ersten 20 Einträge (limit/offset)
Welche (where)
sowohl (and)
in Berlin sind
und entweder (or)
im Titel “bike” vorkommt
oder in der Beschreibung “bike” vorkommt

Detaillierte Beschreibung aller Optionen:
https://loopback.io/doc/en/lb4/Querying-data.html

REST-Api: Komplexe Query Parameter

Die REST-Api kann URL-Kodiertes JSON verarbeiten

```
const filter = { /* Beispiel von voriger Folie */ };

const filterParam = encodeURIComponent(JSON.stringify(filter));


// Beispiel mit fetch
fetch('https://awacademy-kleinanzeigen.azurewebsites.net/ad/?filter='+filterParam)
    .then(r => r.json())
    .then(ads => console.log(ads));
```

Login mit **J**SON **W**eb **T**oken

- Sende Login-Daten (E-Mail-Adresse und Passwort) an /user/login
- Bei Erfolg, erhalte “JSON Web Token” in Response
  - Beispiel:
    `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjIiLCJuYW1lIjoiVGVzdCIsImlhdCI6MTU4OTE4Mzc5OSwiZXhwIjoxNTg5MTg0Mzk5fQ.AfizXlSyj4vOatTfJKnCt1kGfZH73xTKFjGFiRf_P68`
- Schicke HTTP Header `Authorization: Bearer <Token hier>`
  - Beispiel mit fetch:

```
fetch('/user/me', {
    headers: {
        'Authorization': `Bearer ${token}`
    }
})
.then(res => res.json())
.then(data => { console.log(data) })
```