

React



INHALT

- Allgemeines zu React
 - Entstehung
 - Was ist React
 - Wer benutzt React
- Installationsanweisung
 - React IDE'S
 - Vorgehen
 - Folder Struktur
 - Verwenden der Web-App
- Beispiel APP
- Offlinefähig, Responsive, Zugriff auf Hostsystem
 - Wichtige Begriffe
 - Offlinefähigkeit
 - Redux-Persist
 - Responsive
- Developer Experience
 - Developer Bewertungen
 - Communities
 - Testtools
 - Lernkurve
 - Unsere Erfahrung

ALLGEMEINES ZU REACT

ALLGEMEINES ZU REACT

ENTSTEHUNG

- 2011 Probleme bei Facebook (Codewartung)
 - Lösung: Neues Framework
 - Spart enorm Aufwand und Kosten
- 2013 Erschienen

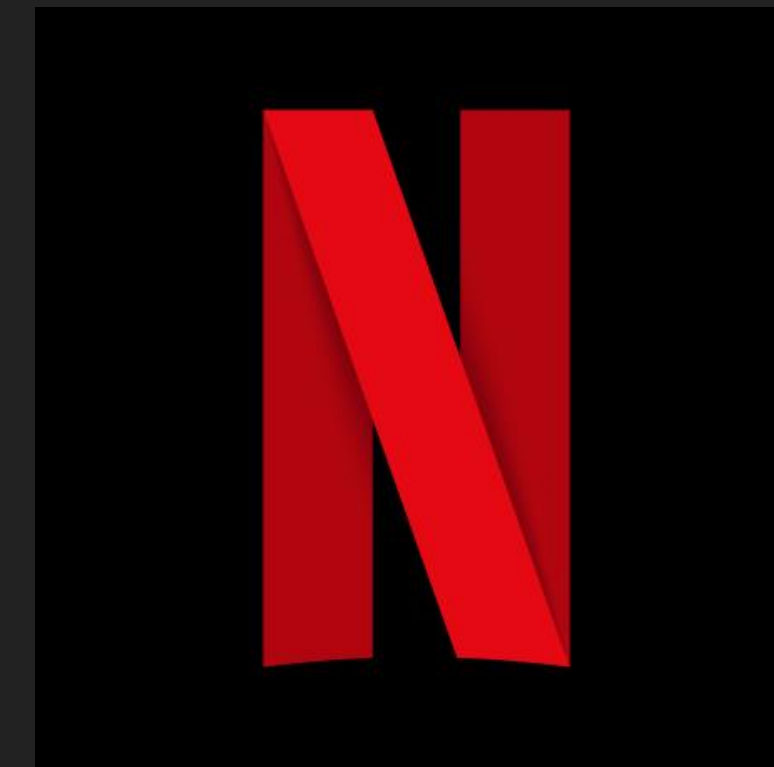
ALLGEMEINES ZU REACT

WAS IST REACT

- JavaScript basierte UI Entwicklungsbibliothek
- Am häufigsten verwendete Frontend Bibliothek der Welt für Webentwicklung
- Unterstützt nativ plattformübergreifende Entwicklung
- Macht Webanwendungen schneller, leistungsfähiger und suchmaschinenfreundlicher

ALLGEMEINES ZU REACT

WER BENUTZT REACT



INSTALLATIONS ANWEISUNG

REACT IDE'S

INSTALLATIONSANWEISUNG

REACT IDE'S

- Komfort der Benutzeroberfläche
- Anzahl der Feature
- Sprachen Unterstützung → long term use
- Preis

INSTALLATIONSANWEISUNG

REACT IDE'S (VS CODE)

PROS

- Unterstützt React.js IntelliSense
- Mehrsprachig
- 20000 Plugins
- Kostenlos

CONS

- Code-Check nicht optimal
- gelegentliche Performanceprobleme
- Probleme beim debuggen

INSTALLATIONSANWEISUNG

REACT IDE'S (WEBSTORM)

PROS

- Wahrscheinlich meist ausgereiftes und benutztes web-Tool
- Excelente Tipps zum vertiefen/vereinfachen von code
- Überprüfung des Codes auf Fehler

CONS

- 129€ pro Jahr
- Keine gute Performance
- Komplexe Einstellungen

INSTALLATIONSANWEISUNG

REACT IDE'S (REACTIDE)

PROS

- Nahtlose Browserintegration
- Live Visualisierung (kein hin und her schalten)
- Integrierter Node Server

CONS

- Nur für React nutzbar
- Keine Plugins
- Keine Community

INSTALLATIONSANWEISUNG

REACT IDE'S (REACTIDE)

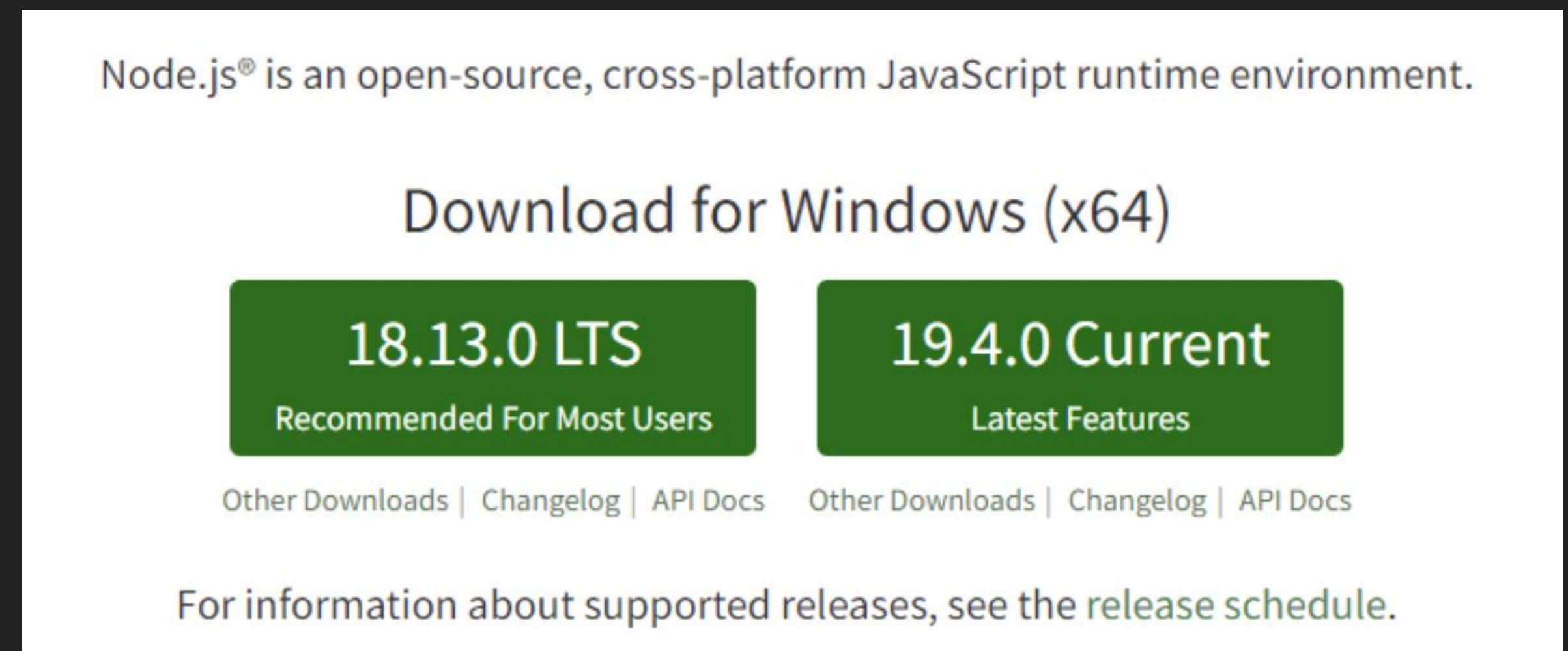


VORGEHEN

INSTALLATIONSANWEISUNG

VORGEHEN 1. NODE.JS

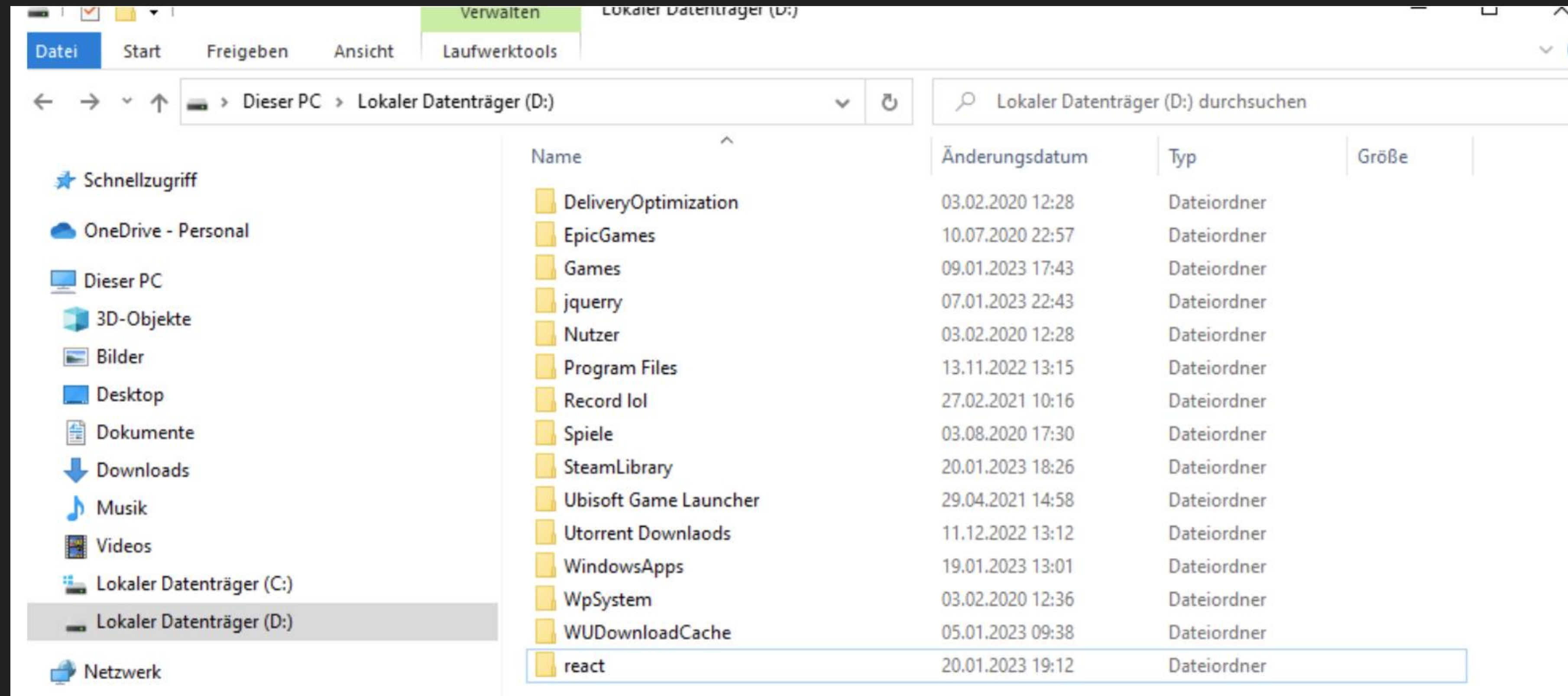
- Download von offizieller Seite
- JavaScript-Laufzeitumgebung
- Gleiche Sprache Front/Backend
- Erstellung Anwendungen
- Datenbankverwaltung
- HTTP-Verkehr etc.



INSTALLATIONSANWEISUNG

VORGEHEN 2. VERZEICHNIS FÜR APP ERSTELLEN

- Ordner Projektmappe erstellen



INSTALLATIONSANWEISUNG

VORGEHEN 3. CREATE WEB APP - GITHUB

- Download Gitprojekt
- Enthält notwendige Dateien
- Basisaufbau eines Programms

To create a new app, you may choose one of the following methods:

npx

```
npx create-react-app my-app
```

(npx is a package runner tool that comes with npm 5.2+ and higher, see [instructions for older npm versions](#))

npm

```
npm init react-app my-app
```

npm init <initializer> is available in npm 6+

Yarn

```
yarn create react-app my-app
```

yarn create <starter-kit-package> is available in Yarn 0.25+

It will create a directory called `my-app` inside the current folder.

Inside that directory, it will generate the initial project structure and install the transitive dependencies:

INSTALLATIONSANWEISUNG

VORGEHEN 4. NPM

```
C:\> npm install react react-dom react-scripts cra-template

Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\Nutzer>D:

D:\>cd react

D:\react>npm init react-app my-app

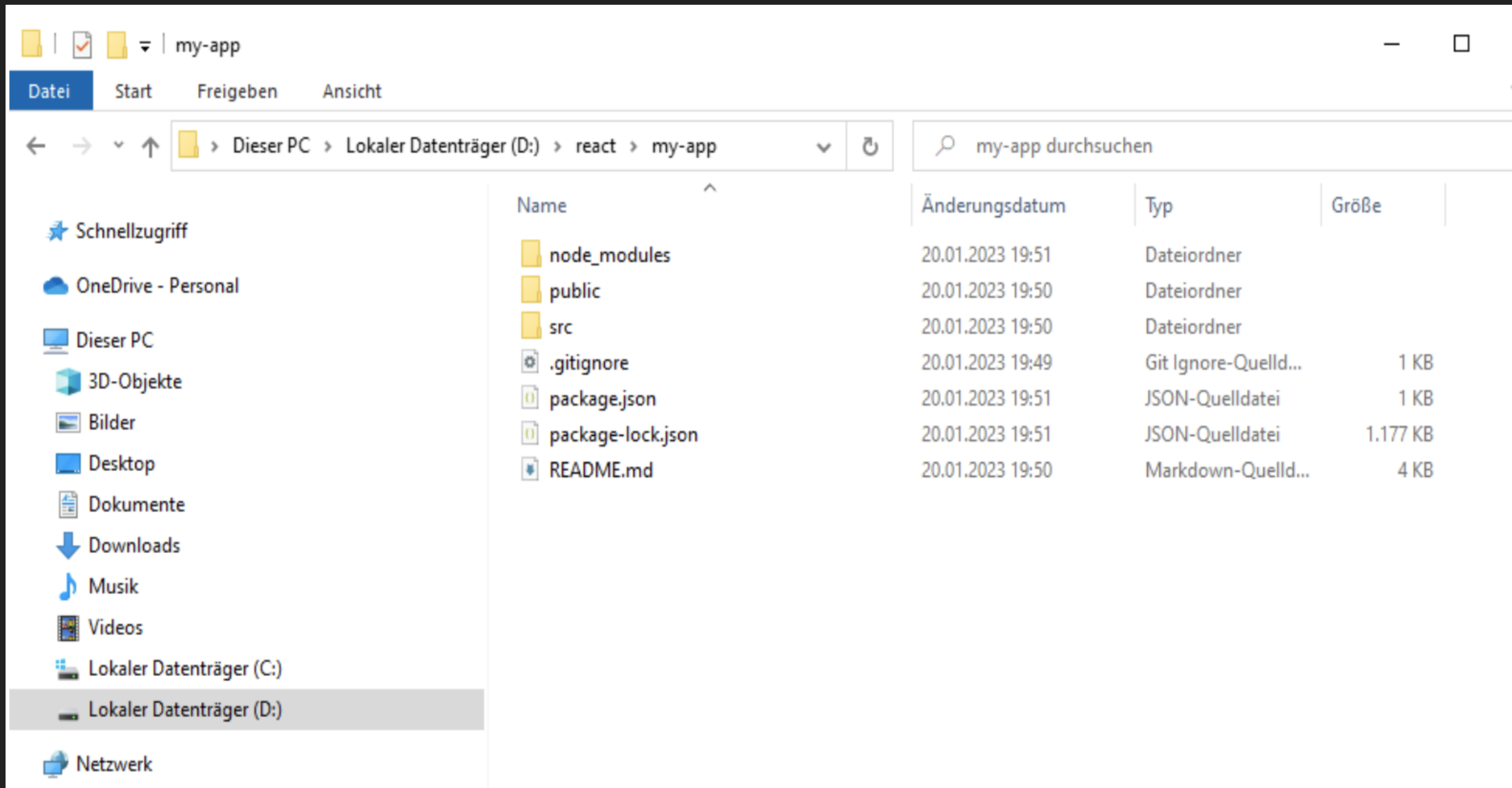
Creating a new React app in D:\react\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

[██████████] - idealTree:css-minimizer-webpack-plugin: timing idealTree:node_modules/css-minimizer-webpack-plugin
```

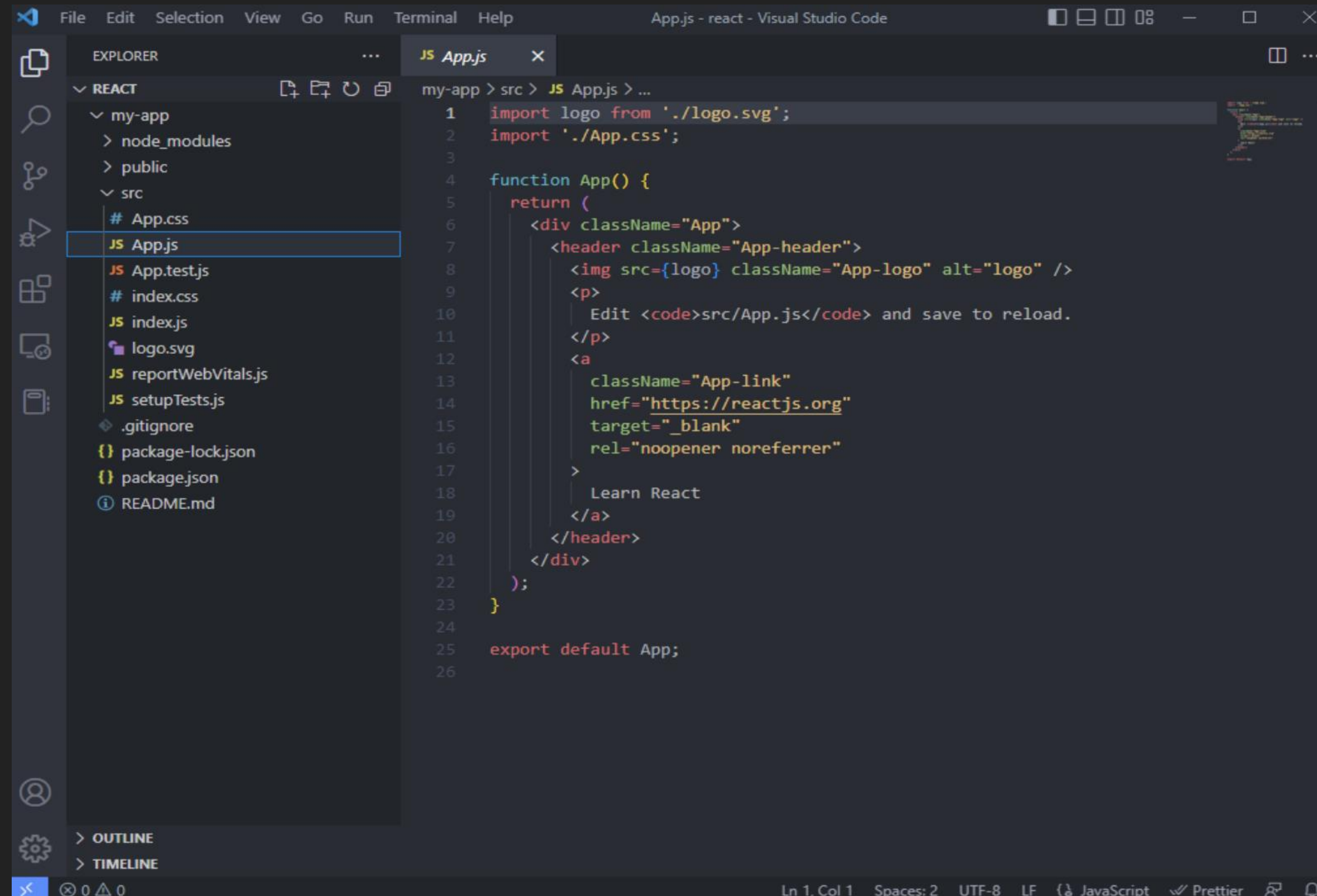
INSTALLATIONSANWEISUNG

VORGEHEN 5. MY-APP



INSTALLATIONSANWEISUNG

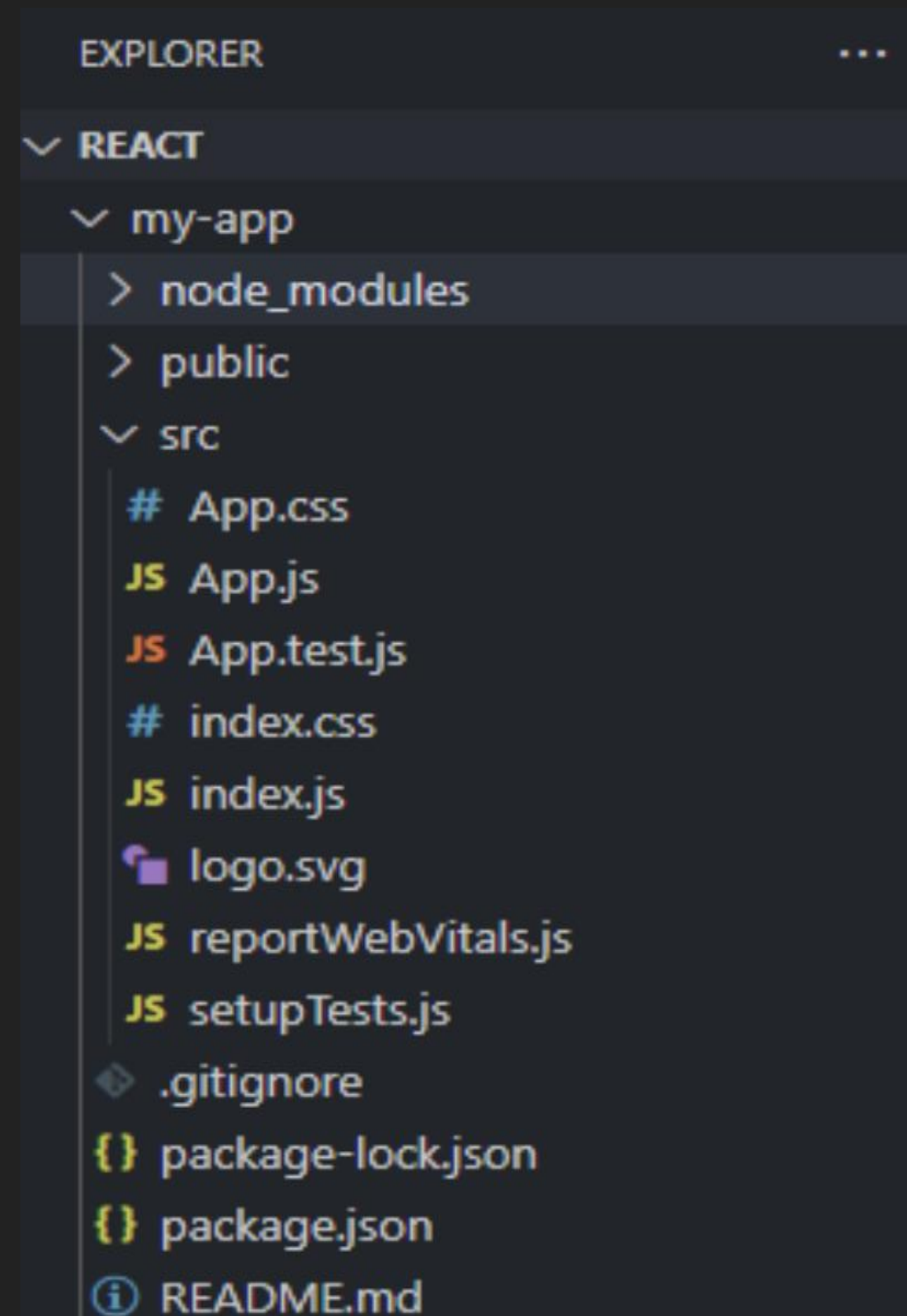
VORGEHEN 6. ÖFFNEN IN VS CODE



FOLDER STRUKTUR

INSTALLATIONSANWEISUNG

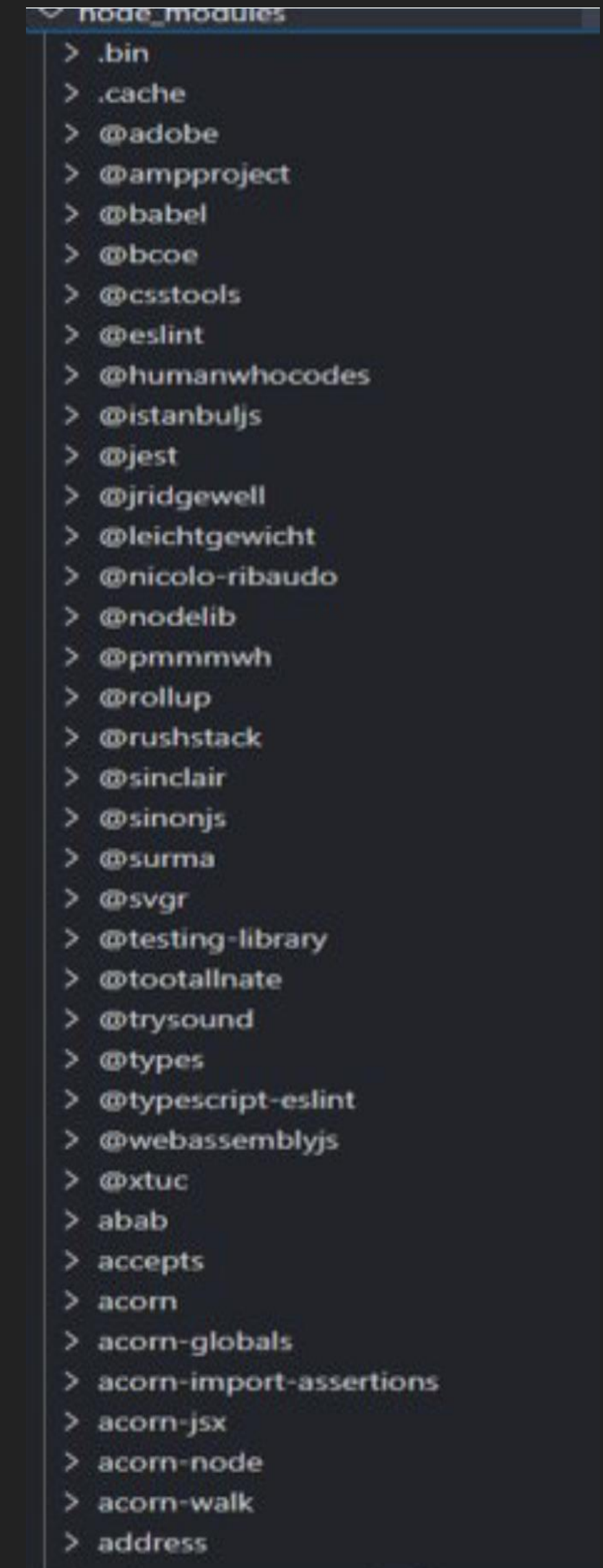
FOLDER STRUKTUR (NEUE APP)



INSTALLATIONSANWEISUNG

FOLDER STRUKTUR (NODE_MODULES)

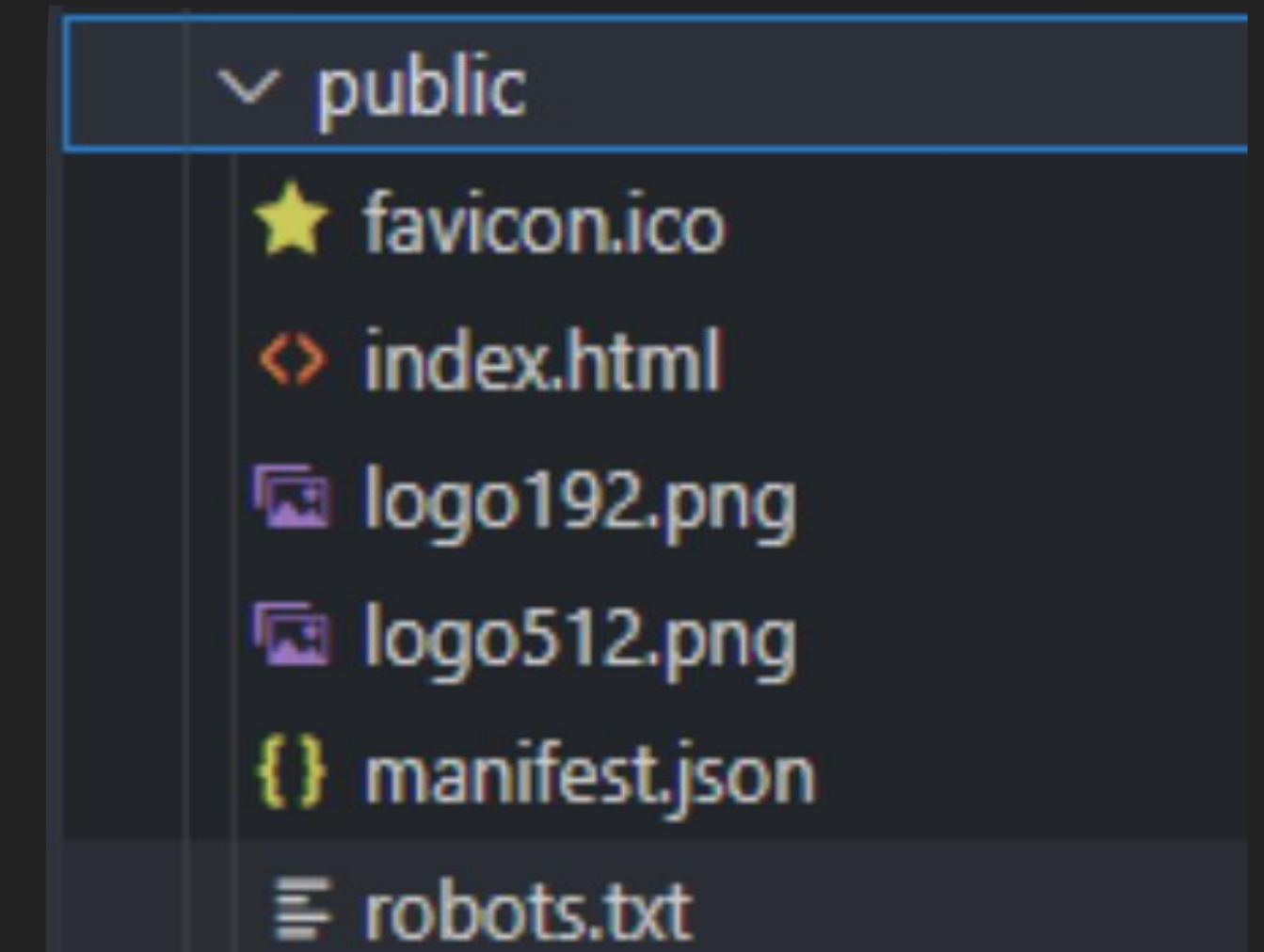
- Node Moduls
 - Enthält alle von npm installierten Abhängigkeiten für die App
 - Kann von eigenem Code aus importiert werden
 - Sollte automatisch von NPM verwaltet werden
 - Hochladen ist nicht nötig, da package.json diese speichert



INSTALLATIONSANWEISUNG

FOLDER STRUKTUR (PUBLIC)

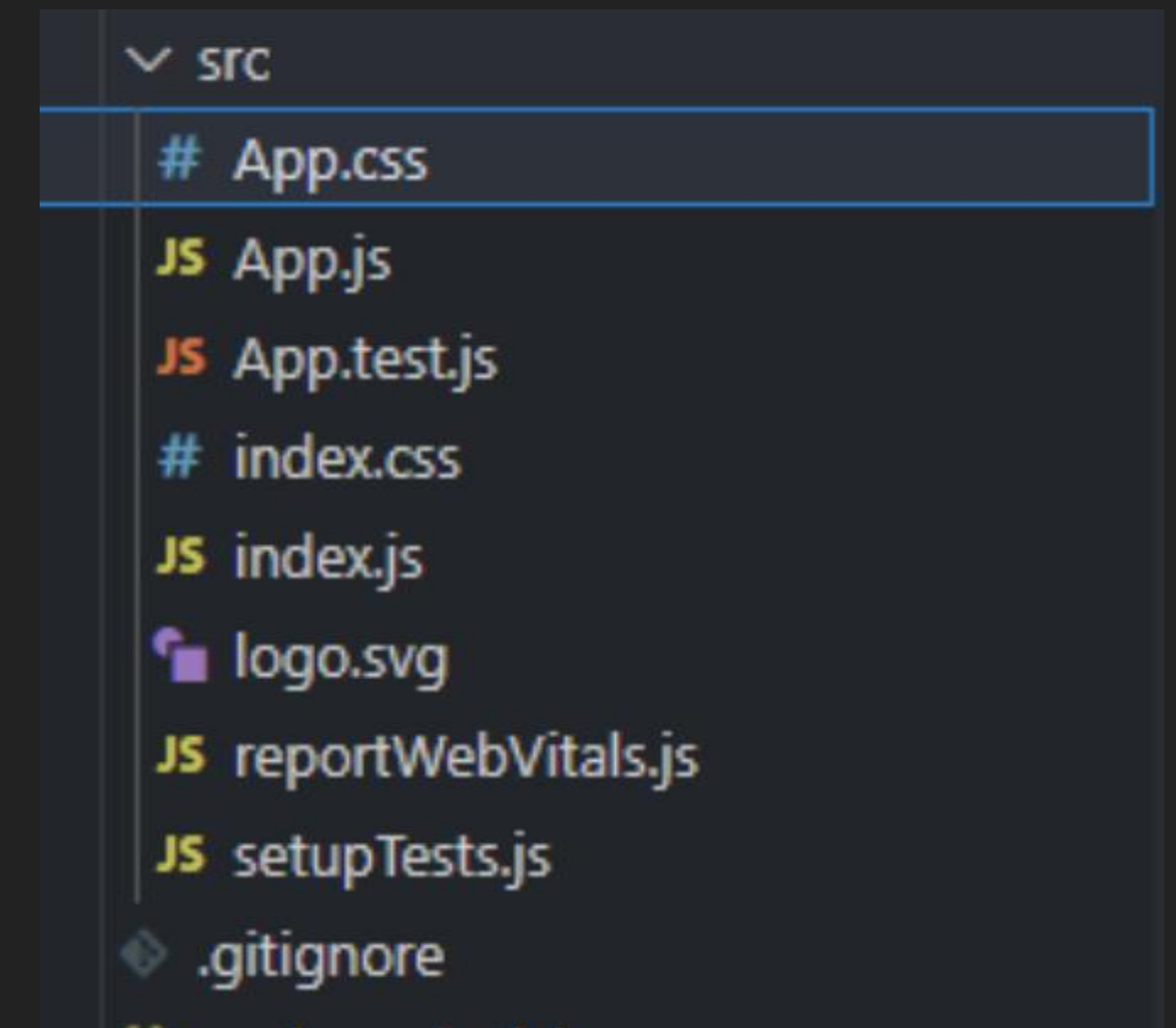
- Besitzt statische Daten
- Bilder, Icons etc.
- Ansprechen mit relativem Pfad möglich
- Index.html -> Startdokument für die App
- Dateien werden nicht von Build-Tools angesprochen



INSTALLATIONSANWEISUNG

FOLDER STRUKTUR (SRC)

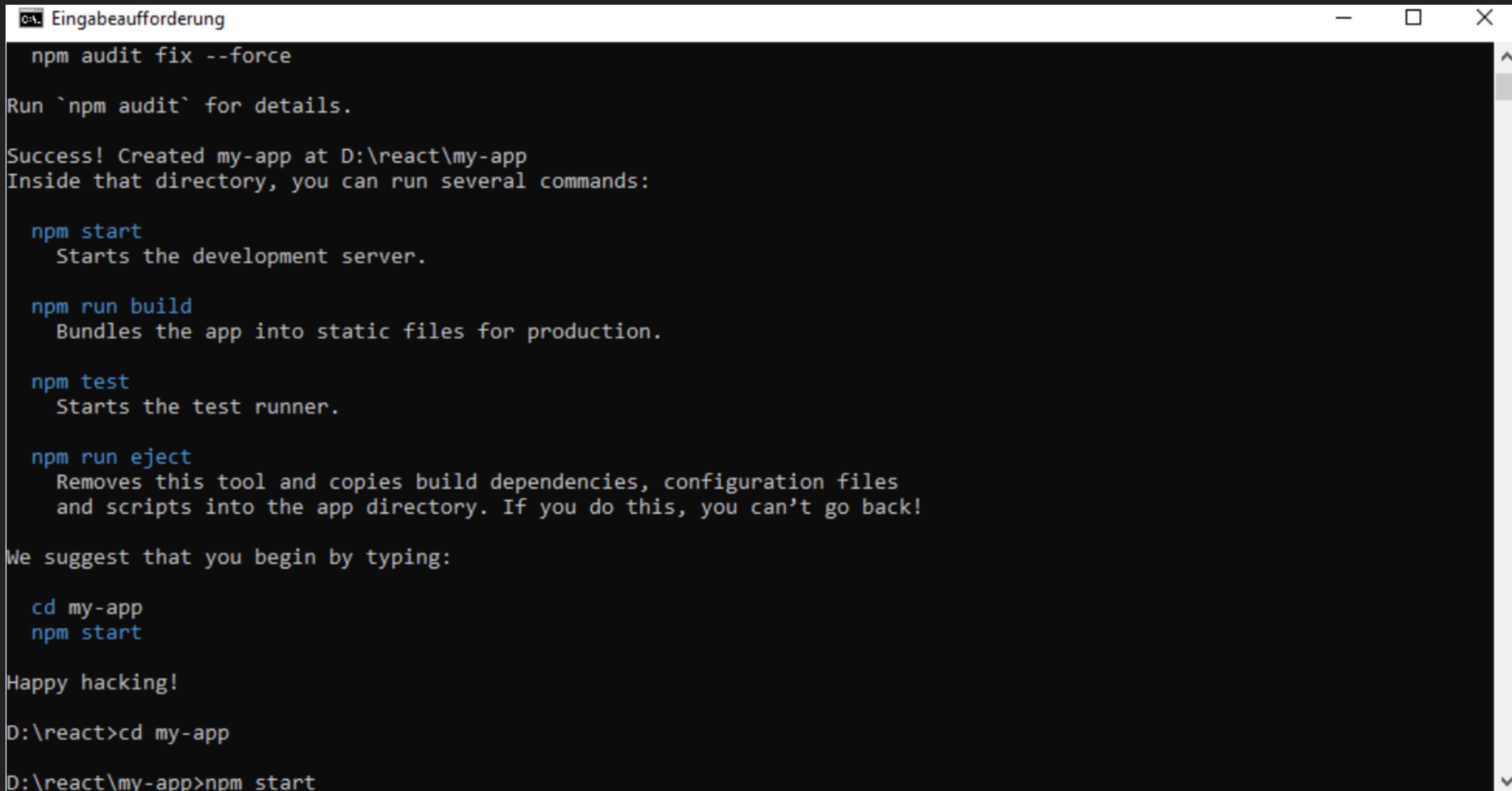
- Enthält häufig Unterordner wie Styles, Pages
- Index.js ist Einstiegsdatei für die App
 - Abhängigkeiten importiert, Routen konfiguriert
- Dateien im Ordner werden von Build-Tools verarbeitet
 - Wie Webpack
 - Nur Dateien im Ordner die relevant sind
 - → keine generierten



VERWENDEN DER WEB-APP

INSTALLATIONSANWEISUNG

VERVENDEN DER WEB-APP (START)



```
C:\> npm audit fix --force

Run `npm audit` for details.

Success! Created my-app at D:\react\my-app
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-app
  npm start

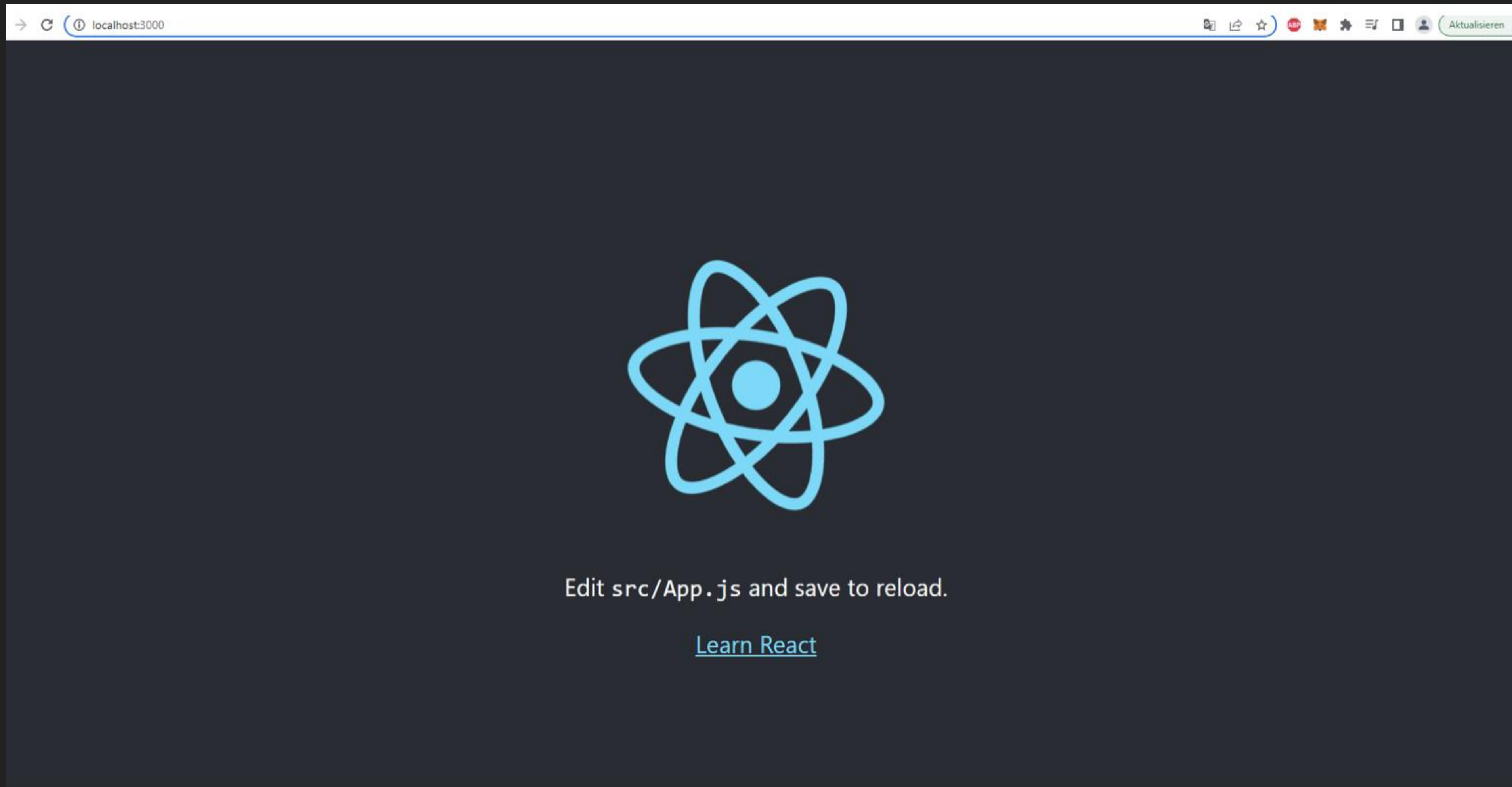
Happy hacking!

D:\react>cd my-app

D:\react\my-app>npm start
```

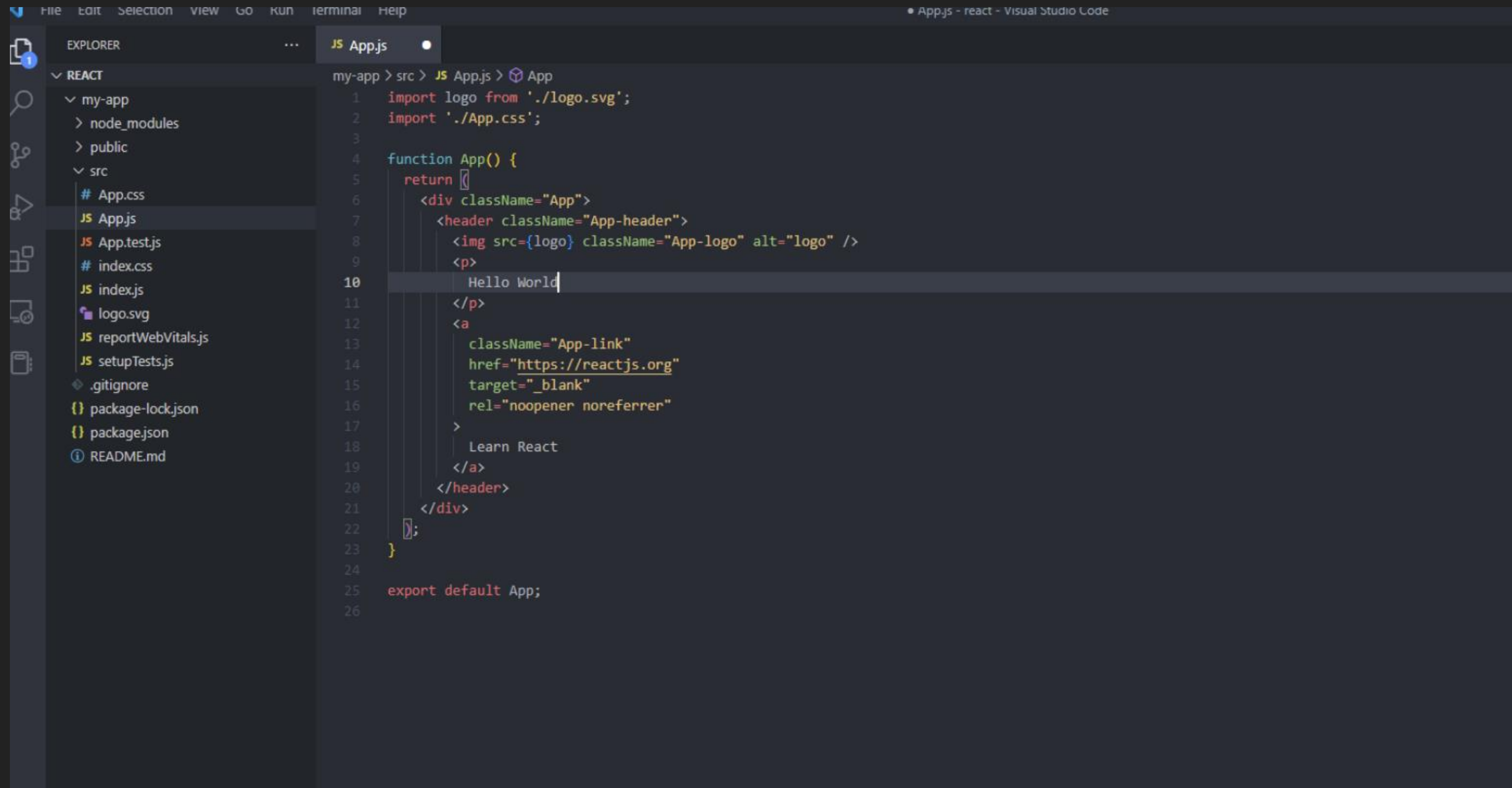
INSTALLATIONSANWEISUNG

VERVENDEN DER WEB-APP (ÖFFNEN)



INSTALLATIONSANWEISUNG

VERVENDEN DER WEB-APP (ERSTE ÄNDERUNGEN)



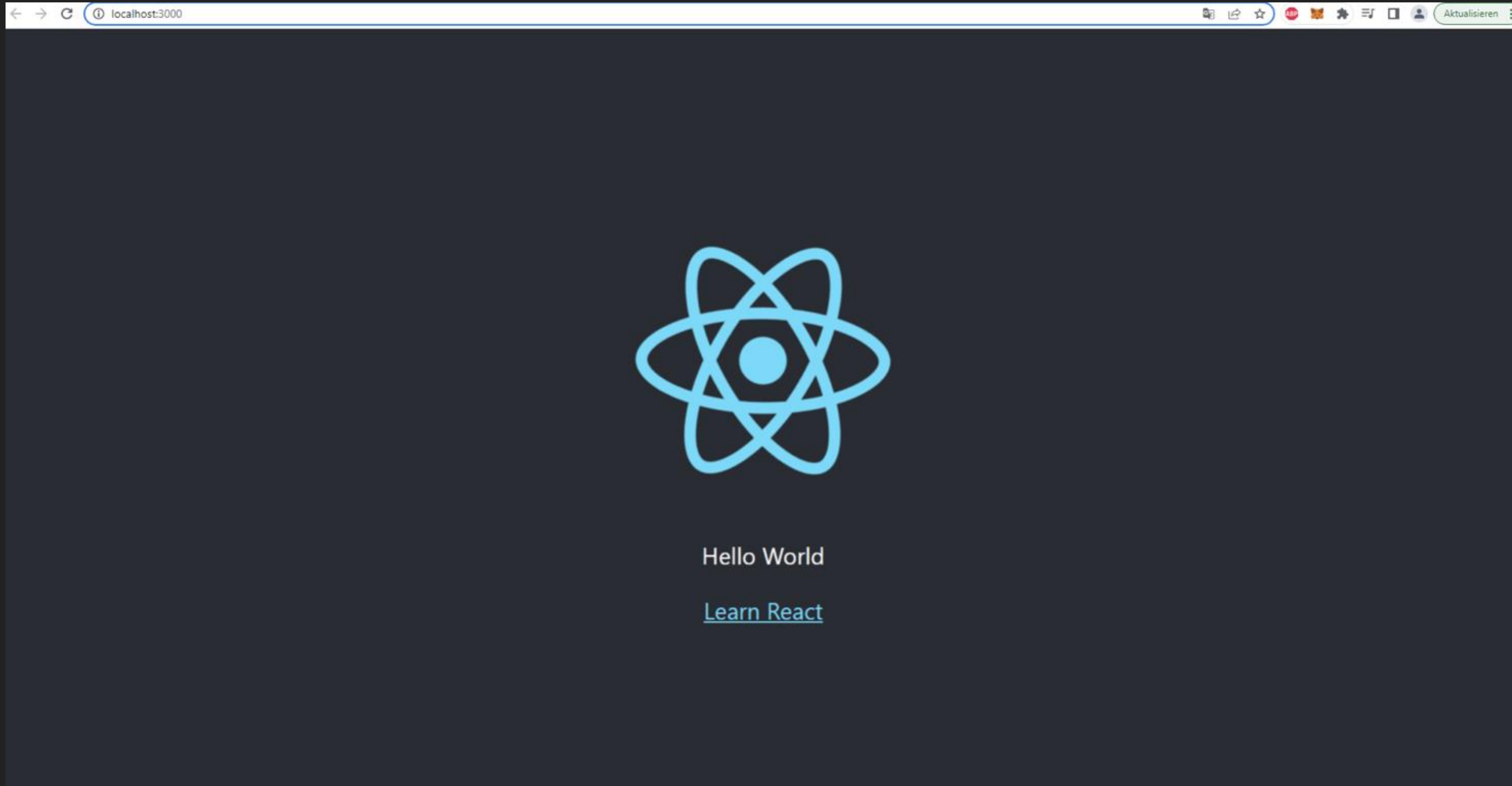
```
File Edit Selection View Go Run Terminal Help
App.js - react - Visual Studio Code

EXPLORER
REACT
  my-app
    node_modules
    public
    src
      App.css
      App.js
      App.test.js
      index.css
      index.js
      logo.svg
      reportWebVitals.js
      setupTests.js
      .gitignore
      package-lock.json
      package.json
      README.md

my-app > src > JS App.js > App
1 import logo from './logo.svg';
2 import './App.css';
3
4 function App() {
5   return (
6     <div className="App">
7       <header className="App-header">
8         <img src={logo} className="App-logo" alt="logo" />
9         <p>
10          Hello World
11        </p>
12        <a
13          className="App-link"
14          href="https://reactjs.org"
15          target="_blank"
16          rel="noopener noreferrer"
17        >
18          Learn React
19        </a>
20      </header>
21    </div>
22  );
23 }
24
25 export default App;
26
```


INSTALLATIONSANWEISUNG

VERVENDEN DER WEB-APP (HELLO WORLD)



BEISPIEL APP

BEISPIEL APP



```
export default function Card({ card, handleChoice, flipped, disabled }) {

  const handleClick = () => {
    if (!disabled) {
      handleChoice(card)
    }
  }

  return (
    <div className="card" key={card.id}>
      <div className={flipped ? "flipped" : null}>
        <img
          className="front"
          src={card.src}
          draggable="false"
        />
        
      </div>
    </div>
  )
}
```


BEISPIEL APP

```
export default function Card({ card, handleChoice, flipped, disabled }) {

  const handleClick = () => {
    if (!disabled) {
      handleChoice(card)
    }
  }

  return (
    <div className="card" key={card.id}>
      <div className={flipped ? "flipped" : null}>
        <img
          className="front"
          src={card.src}
          draggable="false"
        />
        
      </div>
    </div>
  )
}
```

```
<Card
  card={card}
  handleChoice={handleChoice}
  flipped="true"
  disabled="false"
/>
```

OFFLINEFÄHIG

RESPONSIVE

ZUGRIFF AUF HOSTSYSTEM

WICHTIGE BEGRIFFE

WICHTIGE BEGRIFFE

- Caching: zwischenspeichern
- Redux Store: Container für den Zustand der Anwendung
- Reduzierfunktion: Menge von Daten wird zu einem Schlüsselwert
- Fetch API: JavaScript Interface für Serveranfragen
- Axios: On Top von Fetch API um noch mächtigere und flexiblere Anfragen stellen zu können
- Apollo: Client, arbeitet mit Graph ql(query language), automatisiert caching, error handling
- Redux: Bibliothek zum managen des Status einer Applikation, zentraler Speicher für den Zustand

OFFLINEFÄHIGKEIT

OFFLINEFÄHIGKEIT

- Die Offlinefähigkeit ist ein essenzielles Merkmal für eine gute kaufmännische App. Denn sonst wäre ohne Internet keine volle Einsatzfähigkeit und Datenverfügbarkeit gegeben.
- Applikationen in React nicht von Anfang an Offlinefähig
- Offlinefähigkeit kann ermöglicht werden bedarf aber Planung und Architekturüberlegungen

OFFLINE PLUGIN

EINFÜHRUNG

- JavaScript Bibliothek die offline Funktion ermöglicht
- Features: network Status Updates, Offline Ausweichoptionen z.B cached Daten zeigen statt live Daten
- ermöglicht somit Verwendung ohne Internetverbindung

OFFLINE PLUGIN

WIE FUNKTIONIERT DAS?

- Nutzt Browser-APIs z.B. Service Worker und die Cache API um Ressourcen zu speichern und die App offline laufen zu lassen
- Stellt Hooks und Komponenten bereit, mit denen der Offline Zustand verwaltet wird und Status anzeigen kann (Hook ist eine Funktion welche die Nutzung von Features ohne Klassen ermöglicht)
- Einfach zu installieren und konfigurieren zu sein, deshalb beliebte Wahl

REDUX-PERSIST

REDUX-PERSIST

EINFÜHRUNG

- Bibliothek, mit der der Zustand eines Redux-Store in einem Speichermedium wie dem Speicher des Browsers gespeichert wird. Auch nach schließen oder aktualisieren
- Damit werden Fortschritt und Sitzungsdaten gesichert
- Nach Neustart stellt Redux-Persist automatisch den Zustand wieder her und verbessert somit die Benutzererfahrung

REDUX-PERSIST

WIE FUNKTIONIERT DAS?

- Angeben welche Zustände gespeichert werden sollen(z.B Anmeldedaten, Warenkorb)
- Wahl des Speichermediums(lokal, indexedDB kann man sich als Browser Datenbank vorstellen)
- Nach Zustandsänderung speichert Redux-Persist automatisch im Speichermedium
- Beim Neu laden wird Zustand geladen und an Reduzier (Zustand) Funktion übergeben
- Man kann als Nutzer dort weitermachen wo man aufgehört hat

RESPONSIVE

RESPONSIV

- Ansatz im Webdesign um Webseiten auf einer Vielzahl von Geräten und Auflösungen gut darzustellen
- Benutzererfahrung soll verbessert werden in dem Layout und Schriftgröße zum Gerät passen
- React hat keine Responsivität eingebaut kann aber in Verbindung mit CSS erreicht werden

REACT RESPONSIV MACHEN

- Media Queries, Viewports
 - Elemente skalieren basierend auf Bildschirmgröße und Auflösung
- Flexbox
- CSS Grid
 - Gitter basierte Layouts die Größe und Position nach Bildschirmgröße skalieren

BEISPIEL MEDIA QUERIES

- Fenstergröße tracken

```
@media only screen and (min-width: 1024px) {}
```

- Trackt Veränderungen der Fenstergröße und setzt die Minimalweite auf 1024 Pixel
- Startet erst wenn das Fenster mindestens 1024 Pixel groß ist

DOM ELEMENTE UPDATEN

WAS SIND DOM ELEMENTE

- Document Object Model
- Verbindet JavaScript mit HTML Elementen, welche die Eigenschaften(meist Attribute HTML Element) und Methoden für JavaScript definiert
- Elemente werden zu Objekten die dynamisch aufgerufen verändert, hinzugefügt und gelöscht werden können
- `getElementById()`: kann auf Elemente zugreifen, die ein dokumentweit eindeutiges ID-Attribut erhalten
- `getElementsByTagName()`: kann auf Elemente zugreifen, die einen Namen besitzen (er muss nicht unbedingt eindeutig sein)

DOM ELEMENTE UPDATEN

IMPLEMENTIERUNG

```
@media only screen and( min.width: 1024px
{
    .container header
    .header-nax-area
    #nav_container { display:flex; }
}
```

VORTEILE

- einfach zu implementieren
- jeder mit Basis CSS Skills kann wahrscheinlich Media Queries implementieren

NACHTEILE

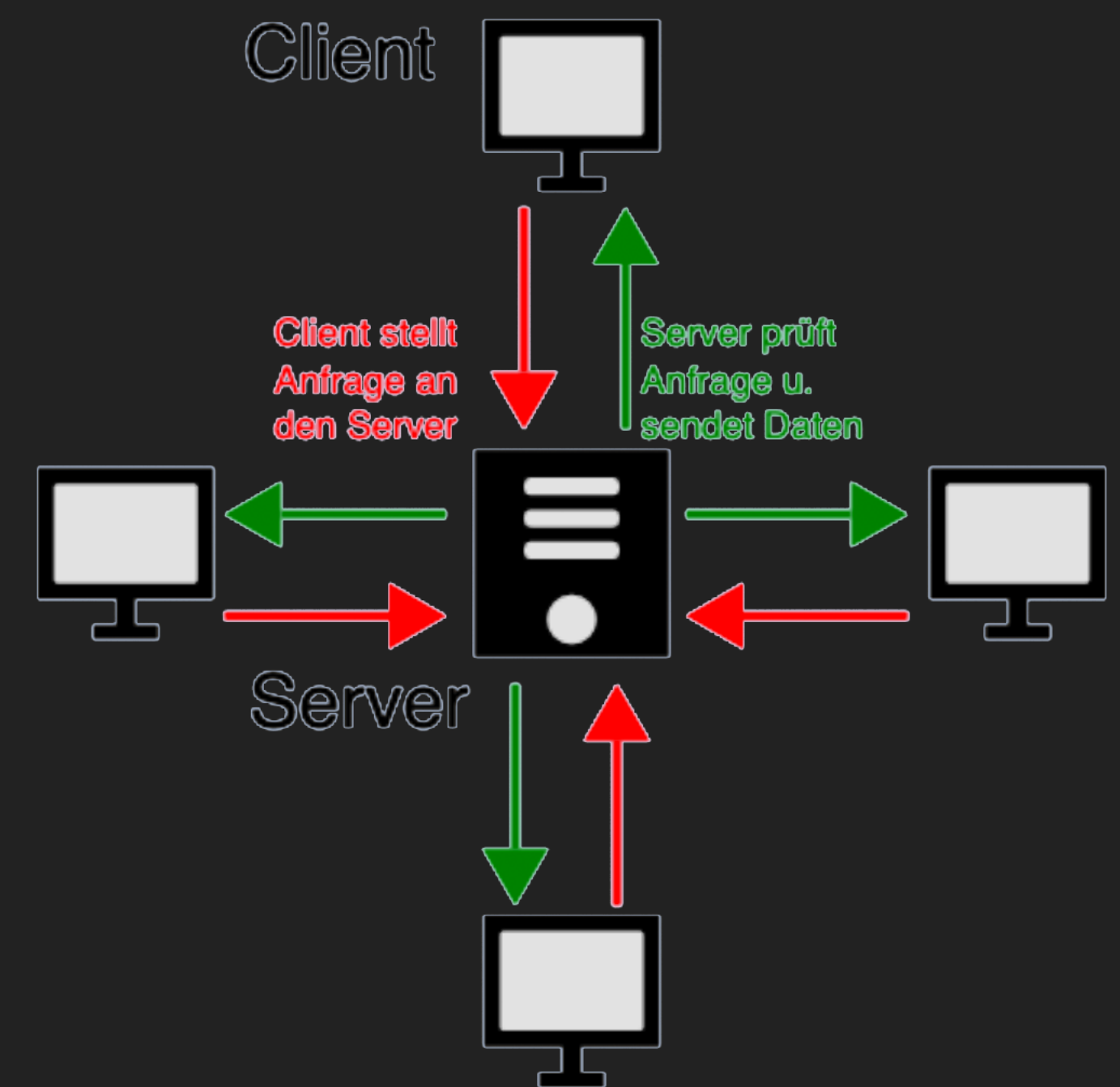
- React wird extra DOM Elemente rendern welcher nie genutzt werden aber versteckt von CSS
- CSS ist getrennt von der Komponente
- Extra Import für CSS File wird benötigt

BEISPIEL ILLUSTRATION



ZUGRIFF AUF HOSTSYSTEM

- WebApps typischerweise mit Client Server Architektur realisiert
- Client im Webbrowser kommuniziert mit Hostsystem(Webserver)
- Kommunikation kann über verschiedene dinge realisiert werden
- z.B. Frameworks, libraries, fetch API, axios, Apollo oder Redux




BROWSER KOMPATIBILITÄT

BEISPIEL: FETCH_API BROWSER VERSION

`fetch()`

The `fetch()` method used to fetch a resource.

													
	 Chrome	 Edge	 Firefox	 Opera	 Safari	 Chrome Android	 Firefox for Android	 Opera Android	 Safari on iOS	 Samsung Internet	 WebView Android	 Deno	 Node.js
<div>fetch</div>	< 42	< 14	< 39	< 29	< 10.1	< 42	< 39	< 29	< 10.3	< 4.0	< 42	< 1.0 *	< 18.0.0 ...

DEVELOPER EXPERIENCE

DEVELOPER BEWERTUNGEN

DEVELOPER EXPERIENCE

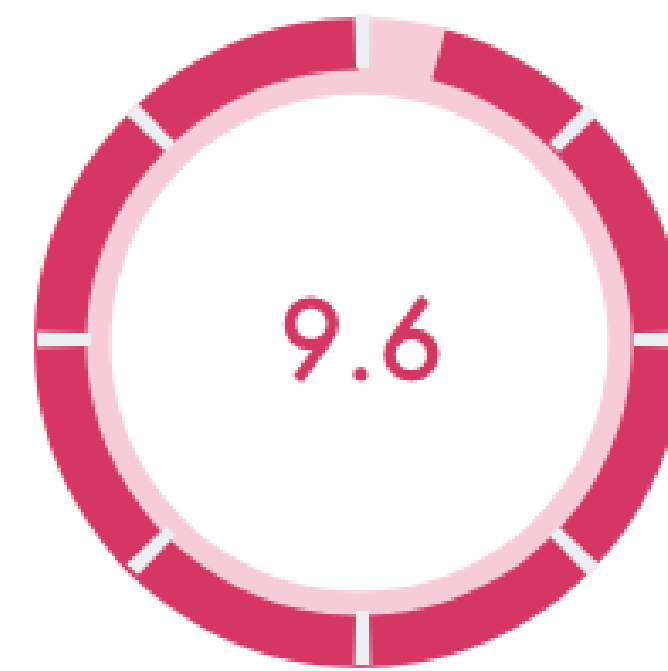
DEV BEWERTUNGEN



Benutzerfreundlichkeit

Content-Management-Systeme
(CMS)

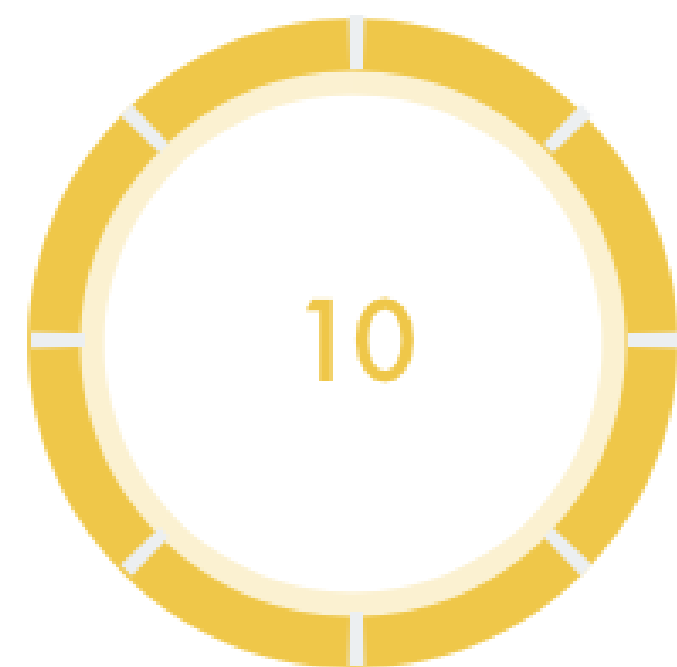
Kategorie-Durchschnitt: 8



Erfüllung der Anforderungen

Content-Management-Systeme
(CMS)

Kategorie-Durchschnitt: 8.7



Kundensupport

Content-Management-Systeme
(CMS)

Kategorie-Durchschnitt: 8.2



Einfache Einrichtung

Content-Management-Systeme
(CMS)

Kategorie-Durchschnitt: 7.7

DEVELOPER EXPERIENCE

DEV BEWERTUNGEN (POSITIVES)

- Spart code Wiederholung ein
- Sehr gut für UIs durch React componets
- Leichter verwendbar als z.b. jQuery, Ajax
- Schneller und effizienter als Vanilla JavaScript
- Leichtes lernen der Syntax (JSX)
- Weit verbreitet und etabliert

DEVELOPER EXPERIENCE

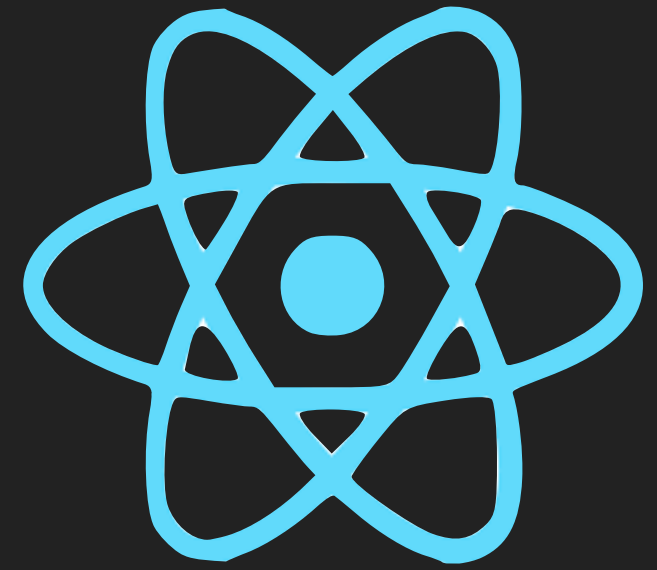
DEV BEWERTUNGEN (NEGATIVES)

- Kein Data Binding, routing
- Veraltete Dokumentation der Sprache auf Website
- Verbesserungswürdige Dokumentation
- Abhängig von zusätzlichen Paketen

COMMUNITIES

DEVELOPER EXPERIENCE

COMMUNITIES (DEV'S REACT COMMUNITY)



- Eine der größten der Welt
- Mehr als 800k Benutzer
- Über 25k Beiträge zu React
- Fragen stellen möglich
- Verwendet von Entwicklern

DEVELOPER EXPERIENCE

COMMUNITIES (HASHNODE'S REACT COMMUNITY)



- Über 13k Beiträge zu React
- Beiträge von Entwicklern verfasst
- Viele Tutorials, Projekte und Konzepte

DEVELOPER EXPERIENCE

COMMUNITIS (REACTIFLUX)



- Discord Community
- 180k Devs
- Monatliche Q&A Sessions
- Sprachchats
- Auch off-topic Räume

DEVELOPER EXPERIENCE

COMMUNITIES (STACK OVERFLOW)



- 390k fragen zu React
- Jeder Anfängerfehler abgedeckt
- Spart Zeit und Nerven
- Lösung in Form von Code

DEVELOPER EXPERIENCE

COMMUNITIES (REDDIT)



- 300k React Devs
- Personenbezogen
- Gut für fragen zur Karriere
- Feedback zu Projekten

TESTTOOLS

DEVELOPER EXPERIENCE

TESTTOOLS (AUSWAHLPROZESS)

- Wahl zwischen kommerziell, Open-Source, Eigenentwickelt
- Spezifizieren der Anforderungen
 - Kriterien: Methoden, Dokumentation der Ergebnisse, Metriken, Entwickler Skills

DEVELOPER EXPERIENCE

TESTTOOLS (JEST)



- Open-Source
- Von meisten Entwicklern und Facebook empfohlen
- Funktioniert mit vielen JavaScript Projekten
- UI Snapshot-Tests und Complete-API
- Paralleles testen
- 16mio Downloads

DEVELOPER EXPERIENCE

TESTTOOLS (MOCHA)



- Open-Source
- Unterstützt Browser
- Asynchrones testen
- Code coverage Berichte
- Kombinierbar mit anderen Frameworks (z.B. Chai, Enzyme)

DEVELOPER EXPERIENCE

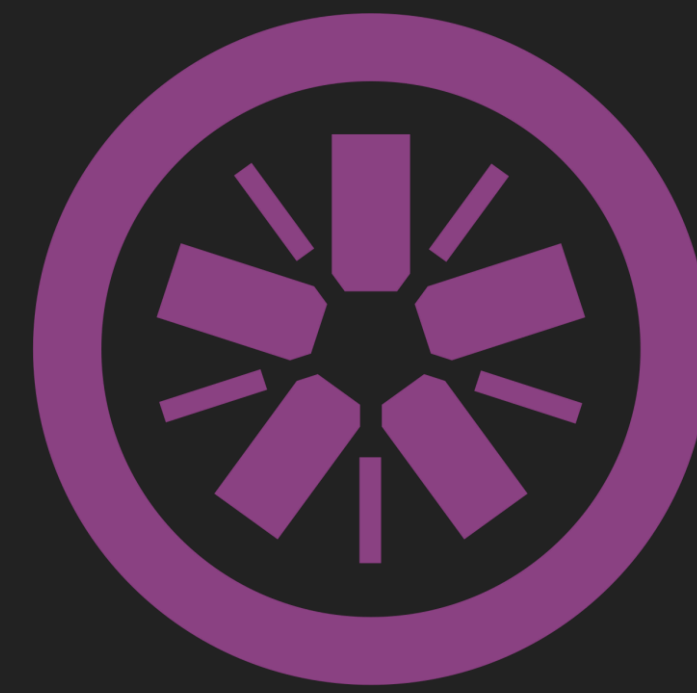
TESTTOOLS (KARMA)



- Open-Source
- Erstellt test html Datei + Server
- Testet in mehreren verschiedenen Browsern, Geräten gleichzeitig
- Vereinfacht Workflow da testen auf Website und Feedback in IDE

DEVELOPER EXPERIENCE

TESTTOOLS (JASMINE)



- Open-Source
- Unabhängig von Browsern, Document Object Model (DOM), JS-Frameworks
- Es gibt spezielle Testlibrary extra für React
- Paralleles Testen
- Schneller als Jest

DEVELOPER EXPERIENCE

TESTTOOLS (ENZYME)



- Open-Source
- Simuliert Verwendung durch Benutzer
- Sehr einfach Output von React Komponenten zu testen
 - DOM, Manipulation und traverse funktionieren wie bei jQuery

LERNKURVE

DEVELOPER EXPERIENCE

LERNKURVE

- Relativ leicht zu lernen
- Benötigt keine überragenden JavaScript Kenntnisse
- Jedoch teilweise gewöhnungsbedürftig
- Meistern erfordert viel Aufwand

Unsere Erfahrung

Unsere Erfahrung

MIT REACT

- Bekannter Syntax macht programmieren einfach
- Verwendung der Dokumentation kann frustrierend sein
 - Legacy Komponenten teilweise nicht als solche zu erkennen
- Dynamisches Data Binding ist sehr praktisch und spart viel Code
- Nützliche Hilfsfunktionen wie useEffect
- Vorteile gegenüber reinem HTML/CSS/JavaScript schon bei kleinen Apps erkennbar

**VIELEN DANK FÜR
DIE
AUFMERKSAMKEIT!!!**