

LU16c - Bundling, Minify & Deploy

Bevor eine Vue.js-App veröffentlicht werden kann, muss sie **gebundelt** werden. Dieser Schritt optimiert alle Dateien für den Produktionseinsatz und erstellt einen `dist`-Ordner, der auf einem Server bereitgestellt werden kann.

npm run build

Der Befehl `npm run build` startet den Vite-Build-Prozess.

```
npm run build
```

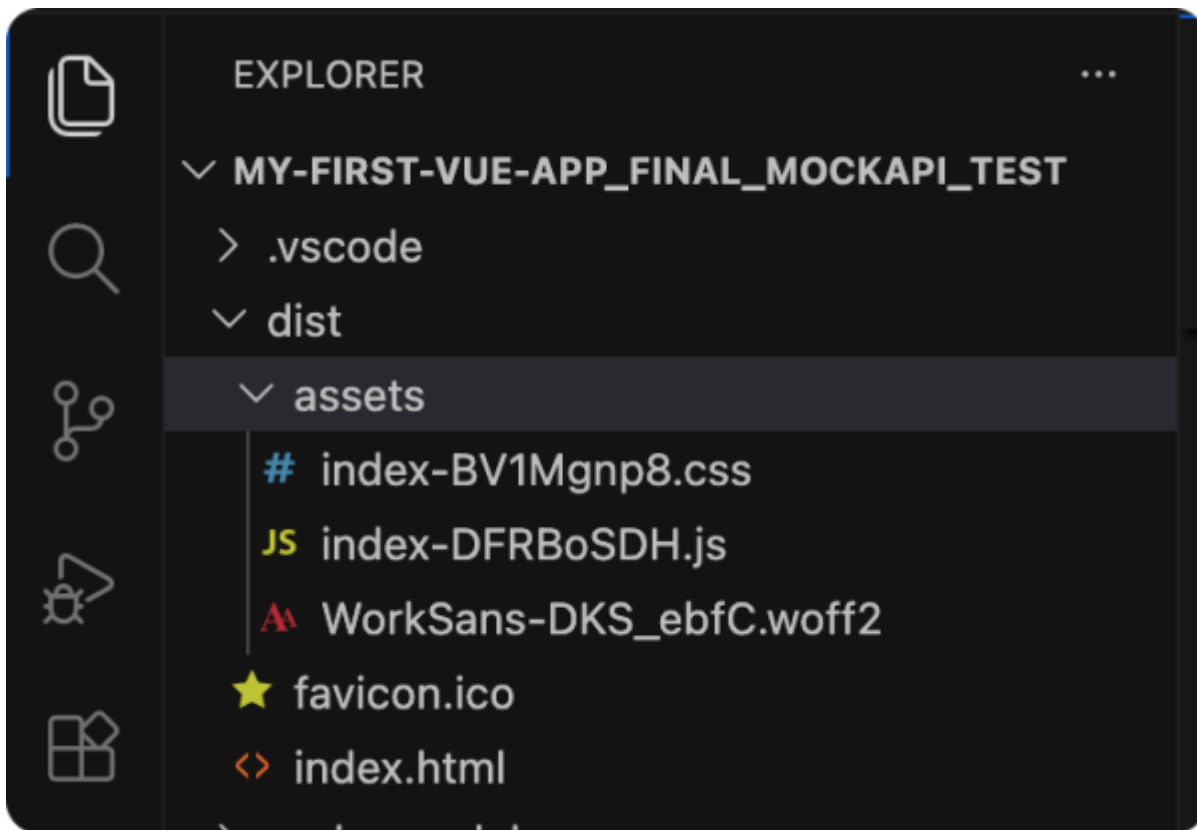
```
291_WebUI/16_LU/my-first-vue-app_final_mockapi_test via ● v24.12.0 using ☁ temp-config/
→ npm run build

> my-first-vue-app@0.0.0 build
> vite build

vite v7.3.6 building client environment for production...
✓ 16 modules transformed.
dist/index.html                0.43 kB | gzip: 0.28 kB
dist/assets/WorkSans-DKS_ebfC.woff2 131.31 kB
dist/assets/index-BV1Mgnp8.css    5.53 kB | gzip: 2.18 kB
dist/assets/index-DFRBoSDH.js    63.56 kB | gzip: 25.39 kB
✓ built in 297ms
```

Platzhalter: Screenshot Terminal mit Build-Output - Dateigrößen, Chunk-Namen, Dauer

Nach dem Build liegt ein neuer Ordner `dist/` im Projekt:



Screenshot VS Code Explorer mit dist-Ordner: index.html, assets/-Unterordner mit .js und .css Dateien
Dieser dist/-Ordner enthält alles, was für den Betrieb der App nötig ist – kein Node.js, kein Vite, keine Vue-Komponenten. ===== Minification ===== **Minification** bedeutet: überflüssige Zeichen aus dem Code entfernen, ohne die Funktion zu verändern.


Vorher (Entwicklung)	Nachher (Production)
Zeilenumbrüche, Einrückungen	Alles in einer Zeile
Lange Variablennamen	Kurze Namen (a, b, fn)
Kommentare	Entfernt

```

src > components > AccordionItem.vue > {} template > div.accordion-item > button.accordion-
1 <script setup>
2 import { ref } from 'vue';
3 import { truncateText } from '../utils/textHelpers.js';
4
5 const props = defineProps({
6   faq: Object,
7 });
8
9 let isOpen = ref(false);
10
11 function toggleOpen() {
12   isOpen.value = !isOpen.value;
13 }
14 </script>
15
16 <template>
17   <div class="accordion-item">
18     <button
19       class="accordion-btn"
20       v-on:click="toggleOpen"
21       :aria-expanded="isOpen"
22     >
23       {{ isOpen ? faq.question : truncateText(faq.question, 35) }}
24     </button>
25
26     <div class="panel accordion-body" v-bind:class="{ open: isOpen }">
27       {{ faq.answer }}
28     </div>
29   </template>
30
31 <style scoped>
32 .panel {
33   height: 0;
34   opacity: 0;
35   overflow: hidden;
36   transition: all 800ms ease-in-out;
37   font-size: 1rem;
38   font-weight: 400;
39
dist > assets > JS index-DFRBoSDH.js > ...Minified Vue-Projekt
1 (function(){const t=document.createElement("link").relList;if(t&&t.
supports&&t.supports("modulepreload"))return;for(const r of
document.querySelectorAll('link[rel="modulepreload"]'))n(r);new
MutationObserver(r=>{for(const i of r)if(i.type==="childList")for
(const o of i.addedNodes)o.tagName==="LINK"&&o.
rel==="modulepreload"&&n(o)}.observe(document,{childList:!0,
subtree:!0});function s(r){const i={};return r.integrity&&(i.
integrity=r.integrity),r.referrerPolicy&&(i.referrerPolicy=r.
referrerPolicy),r.crossOrigin==="use-credentials"?i.
credentials="include":r.crossOrigin==="anonymous"?i.
credentials="omit":i.credentials="same-origin",i}function n(r){if(r.
ep)return;r.ep=!0;const i=s(r);fetch(r.href,i)}();function Fs(e)
{const t=Object.create(null);for(const s of e.split(","))t[s]=1;
return s>s in t}const V={},tt=[],Ee={},Fn={}>!1,Qt=e>e.
charCodeAt(0)==111&&e.charCodeAt(1)==110&&(e.charCodeAt(2)>122|e.
charCodeAt(2)<97),Xt=e>e.startsWith("onUpdate:");e=Object.assign,
Ds=(e,t)>(const s=e.indexOf(t);s>=1&&e.splice(s,1)),Vr=Object.
prototype.hasOwnProperty,j=(e,t)>Vr.call(e,t),P=Array.isArray,
st=e>Pt(e)=="[object Map]",Dn=e>Pt(e)=="[object Set]",nn=e>Pt
(e)=="[object Date]",Ie=e>typeof e=="function",Ge=e>typeof
e=="string",Ae=e>typeof e=="symbol",Ne=e>e!=null&&typeof
e=="object",jn=e>(N(e)||I(e))&&I(e.then)&&I(e.catch),Hn=Object.
prototype.toString,Pt=e>Hn.call(e),Wr=e>Pt(e).slice(8,-1),Ln=e>Pt
(e)=="[object Object]",js=e>G(e)&&e!="NaN"&&e[0]!="-"&&""
+parseInt(e,10)==e,_t=Fs("key,ref,ref_for,ref_key,
onVnodeBeforeMount,onVnodeMounted,onVnodeBeforeUpdate,
onVnodeUpdated,onVnodeBeforeUnmount,onVnodeUnmounted"),Zt=e>{(const
t=Object.create(null);return(s=>t[s]||(t[s]=e(s))))},Br=/-\/w/g,Fe=Zt
(e>e.replace(Br,t=>t.slice(1).toUpperCase())),qr=/\B([A-Z])/g,Ze=Zt
(e>e.replace(qr,"-$1").toLowerCase()),Nn=Zt(e>e.charAt(0).
toUpperCase()+e.slice(1)),fs=Zt(e>e?"on${Nn(e)}":""),Te=(e,t)
=>!Object.is(e,t),us=(e,...t)>{for(let s=0;s<e.length;s++)e[s](...
t)},$n=(e,t,s,n=!1)>{Object.defineProperty(e,t,{configurable:!0,
enumerable:!1,writable:n,value:s})},Gr=e>{const t=parseFloat(e);
return isNaN(t)?e:t};let rn;const kt=()=>rn||(rn=typeof
globalThis<"u"?globalThis:typeof self<"u"?self:typeof window<"u"?
window:typeof global<"u"?global:{});function es(e){if(P(e)){const t=
{};for(let s=0;s<e.length;s++){const n=e[s],r=G(n)?Qr(n):es(n);if(r)
for(const i in r)t[i]=r[i]}return t}else if(G(e)||N(e))return e}

```

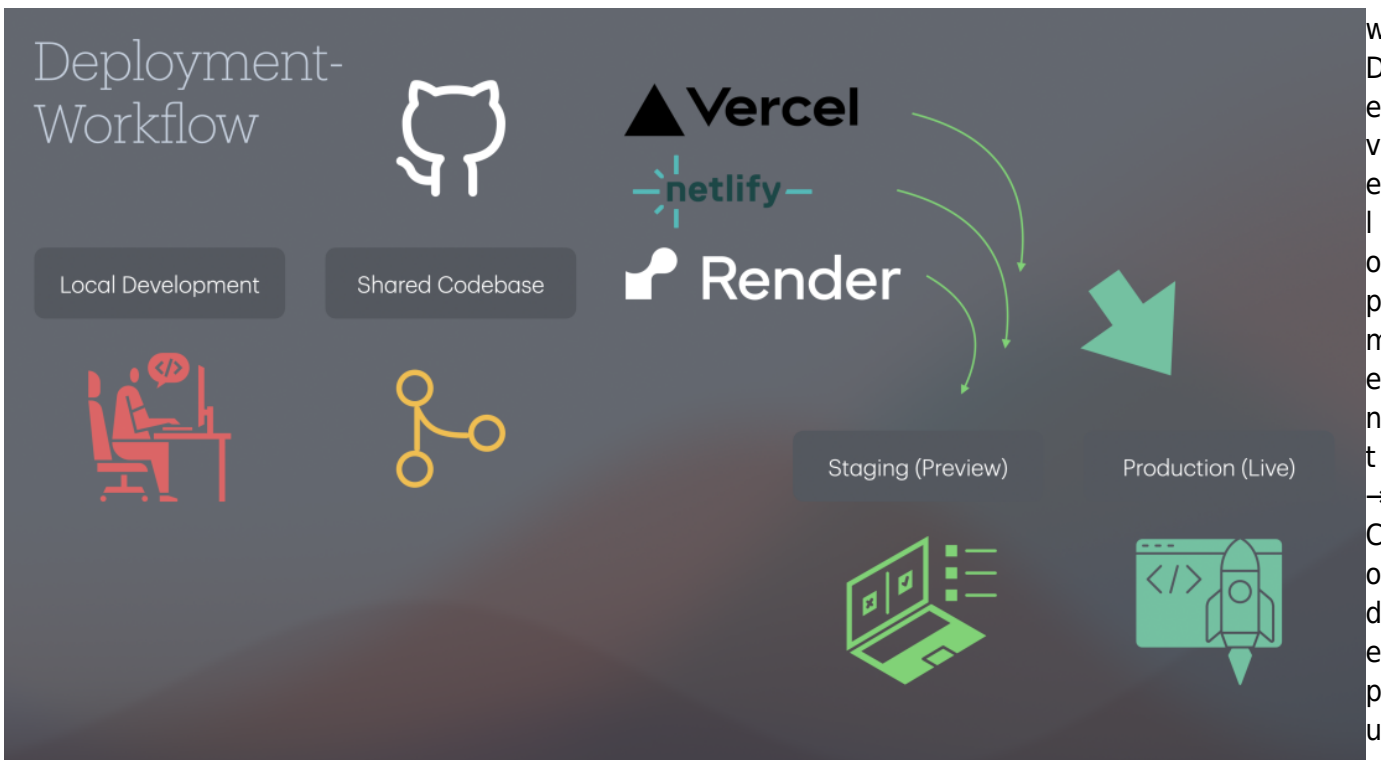
inander - links normal, rechts minifiziert **Ziel:** Kleinere Dateigrösse → kürzere Ladezeit für die Nutzerin / den Nutzer.



Vite minifiziert automatisch bei `npm run build`. Sie müssen nichts zusätzlich konfigurieren.

==== Chunking & Tree Shaking ==== Vite teilt den JavaScript-Code automatisch in **mehrere kleine Dateien (Chunks)** auf, anstatt alles in eine grosse Datei zu schreiben. **Chunking:** Der Browser lädt nur den Code, den er gerade braucht - nicht alles auf einmal. **Tree Shaking:** Code, der im Projekt importiert aber nie verwendet wird, wird automatisch aus dem Build entfernt. ====

Deploy: Mit Git (Vercel, Netlify, Render) ==== Die meisten modernen Hosting-Plattformen verbinden sich mit einem GitHub-Repository und deployen automatisch, sobald neuer Code gepusht



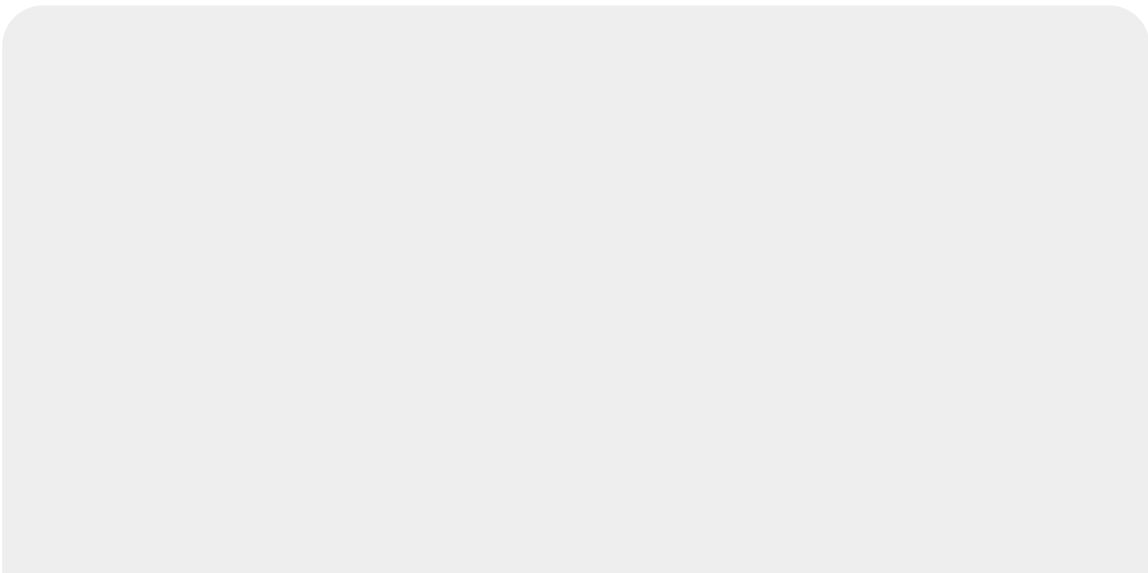
wird.
 D
 e
 v
 e
 l
 o
 p
 m
 e
 n
 t
 →
 C
 o
 d
 e
 p
 u
 s

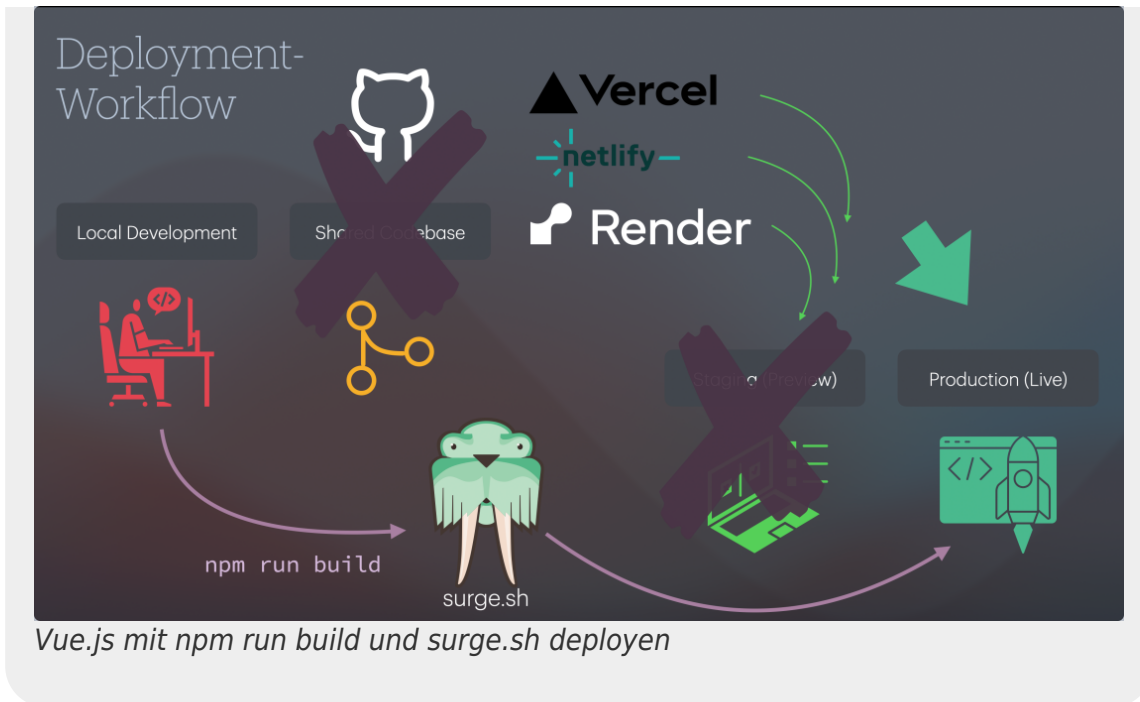
h zu GitHub → Vercel/Netlify/Render → Preview → Production (Live)

Anbieter	Kostenlos	Besonderheit
Vercel	☐ (Hobby-Plan)	Sehr schnell, optimiert für Vue/React/Next.js
Netlify	☐ (Starter-Plan)	Netlify Drop für direktes Hochladen möglich
Render	☐ (mit Einschränkungen)	Auch für Backend-Services geeignet

Voraussetzung: Ein GitHub-Account und ein Repository mit dem Projektcode. Diese Plattformen eignen sich ideal für Teams, die mit Git arbeiten.

===== Deploy: Ohne Git (Surge.sh & Netlify Drop) ===== Für den Unterricht - ohne Git-Kenntnisse
 - gibt es zwei einfachere Wege:





==== Option A: Surge.sh (Terminal-Weg) ====

```
npm run build
cd dist
npx surge
```

Surge fragt beim ersten Start nach E-Mail und Passwort (kostenlose Registrierung direkt im Terminal).
Anschliessend wählen Sie eine Domain:

```
domain: vorname-faq.surge.sh
```

```
dist — -zsh > -zsh — 105x59
Last login: Fri Jun 26 18:13:53 on ttys000

holiday_explorer on [?]main [?] via ● v24.12.0 using ☁ temp-config/
[→ code .]

holiday_explorer on [?]main [?] via ● v24.12.0 using ☁ temp-config/
[→ cd dist]

holiday_explorer/dist on [?]main via ● v24.12.0 using ☁ temp-config/
[→ npx surge]
Need to install the following packages:
surge@0.27.4
[Ok to proceed? (y) y]

Welcome to surge! (surge.sh)
Login or create surge account by entering email & password.

[ email: guido.koch@bzz.ch ]
[ password: ]

Running as guido.koch@bzz.ch (Student)

[ project: /Users/guidokoch/Documents/03_BZZ/291_WebUI/13_LU/holiday_explorer/dist/ ]
[ domain: holiday-explorer-gk.surge.sh ]
[ size: 6 files, 85.0 KB ]
[ upload: [=====] 100% ]
[ CDN: [=====] 100% ]
[ encryption: [=====] 100% ]

Certificate: issuer=C = GB, O = Sectigo Limited, CN = Sectigo Public Serv... Valid
*.surge.sh, surge.sh 172 more days

NS | ns1.surge.world | ns2.surge.world | or CNAME... | |
   | ns3.surge.world | ns4.surge.world | geo.surge.world | |
---|---|---|---|---|
HTTP | sfo.surge.sh | US, San Francisco | 138.197.235.123 | D.Ocean | ✓ ●
HTTP | lhr.surge.sh | GB, London | 46.101.67.123 | D.Ocean | ✓ ●
HTTP | yyz.surge.sh | CA, Toronto | 159.203.50.177 | D.Ocean | ✓ ●
HTTP | jfk.surge.sh | US, New York | 159.203.159.100 | D.Ocean | ✓ ●
HTTP | ams.surge.sh | NL, Amsterdam | 188.166.132.94 | D.Ocean | ✓ ●
HTTP | fra.surge.sh | DE, Frankfurt | 138.68.112.220 | D.Ocean | ✓ ●
HTTP | sgp.surge.sh | SG, Singapore | 139.59.195.30 | D.Ocean | ✓ ●
HTTP |blr.surge.sh | IN, Bangalore | 139.59.50.135 | D.Ocean | ✓ ●
HTTP |syd.surge.sh | AU, Sydney | 45.76.126.95 | Vultr | ✓ ●
HTTP |nrt.surge.sh | JP, Tokyo | 172.104.96.133 | Linode | ✓ ●

Live preview ..... 1782641569002-holiday-explorer-gk.surge.sh
Production ..... holiday-explorer-gk.surge.sh

Success! - Published to holiday-explorer-gk.surge.sh

holiday_explorer/dist on [?]main via ● v24.12.0 using ☁ temp-config/ took 4m 50.3s
→ □
```

Screenshot Terminal – surge-Befehl, Registrierung, gewählte Domain, grüne Erfolgsanzeige □ Die App ist live unter <https://vorname-faq.surge.sh> ==== Option B: Netlify Drop (Drag-and-Drop) ==== - npm run build ausführen - app.netlify.com/drop im Browser öffnen - Den dist/-Ordner per Drag-and-Drop ins Browserfenster ziehen - Fertig - Netlify erstellt sofort eine öffentliche URL

Hinweis: Ohne Konto ist die Netlify-Drop-URL nur ca. 1 Stunde aktiv. Mit kostenlosem Konto (keine Kreditkarte nötig) bleibt sie dauerhaft erreichbar.

===== Vergleich: Deploy-Optionen =====

	Surge.sh	Netlify Drop	Vercel / Netlify / Render
Git nötig?	Nein	Nein	Ja
Terminal nötig?	Ja	Nein	Nein
Aufwand	Gering	Sehr gering	Mittel (Einrichtung)
Geeignet für	Unterricht, Demos	Unterricht, schnelle Tests	Echte Projekte
Automatisches Re-Deploy	Nein	Nein	Ja (bei Git-Push)
Gratis	☐	☐	☐ (mit Limits)

===== Zusammenfassung: Der komplette Ablauf =====

```
# 1. App für Production optimieren
npm run build

# 2. In den dist-Ordner wechseln
cd dist

# 3. Deployen mit Surge
npx surge
```

Nach diesen drei Befehlen ist die Vue.js-App unter einer echten, öffentlichen URL erreichbar.

From:
<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/m291/learningunits/lu16/theorie/c_deploy?rev=1782682475

Last update: **2026/06/28 23:34**

