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.

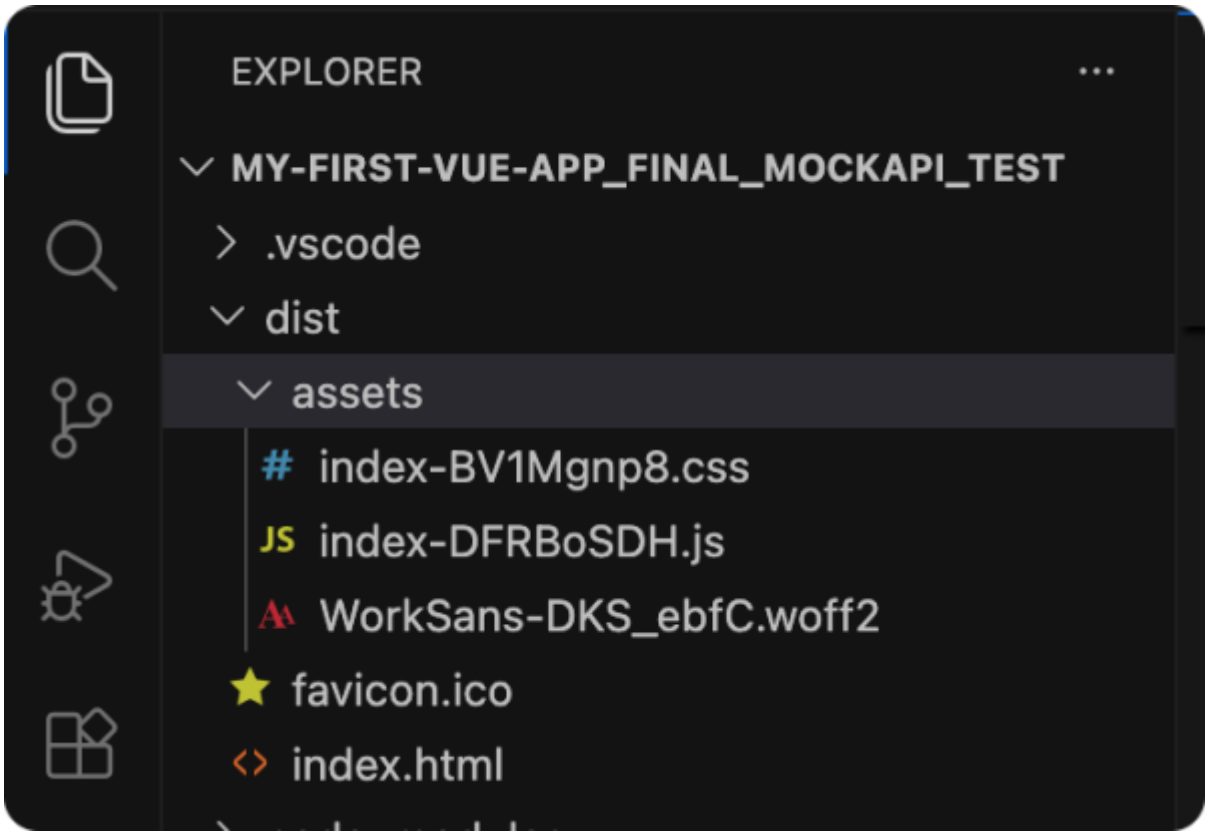
npm run build

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

```
npm run build
```

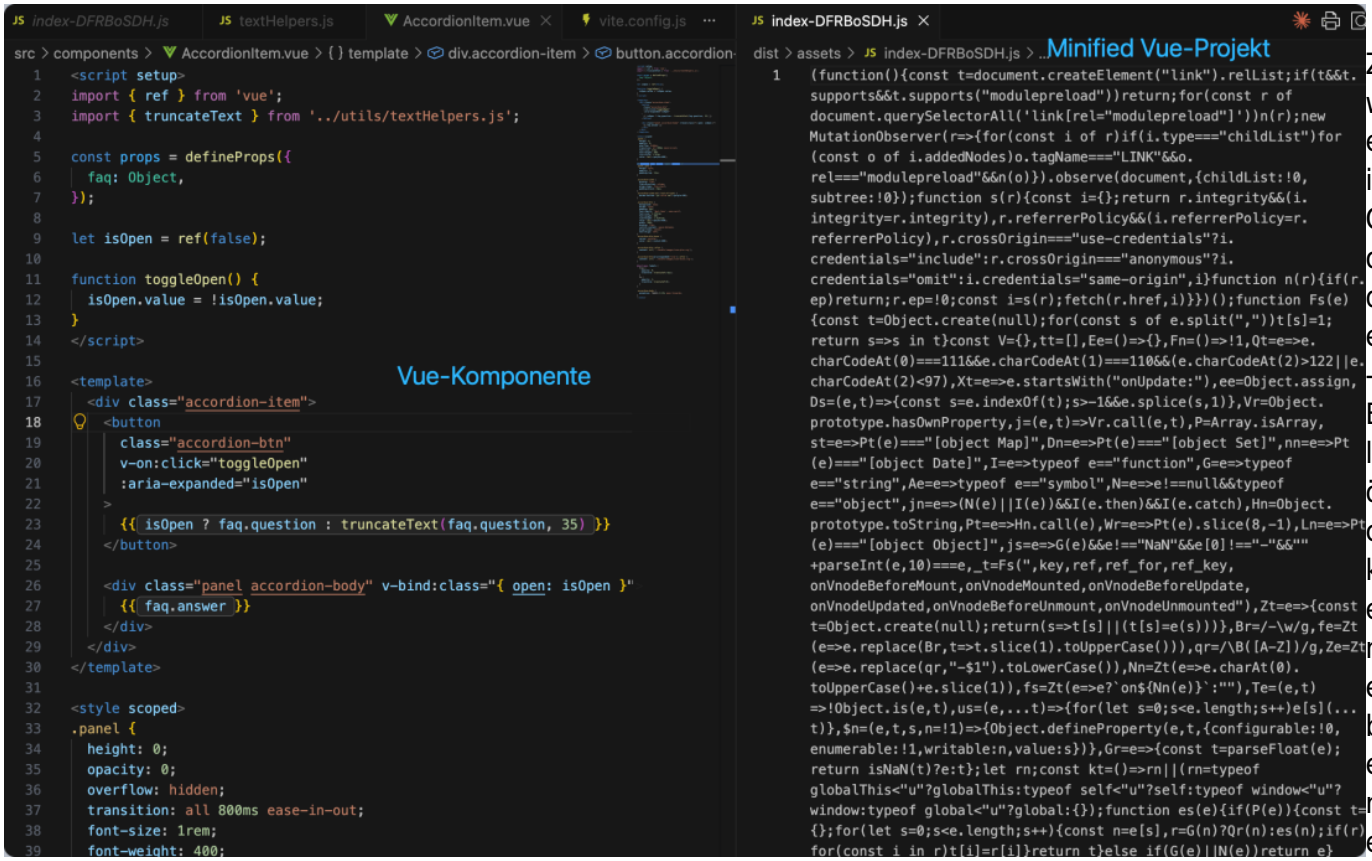
Platzhalter: Screenshot Terminal mit Build-Output – Dateigrössen, Chunk-Namen, Dauer

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.*

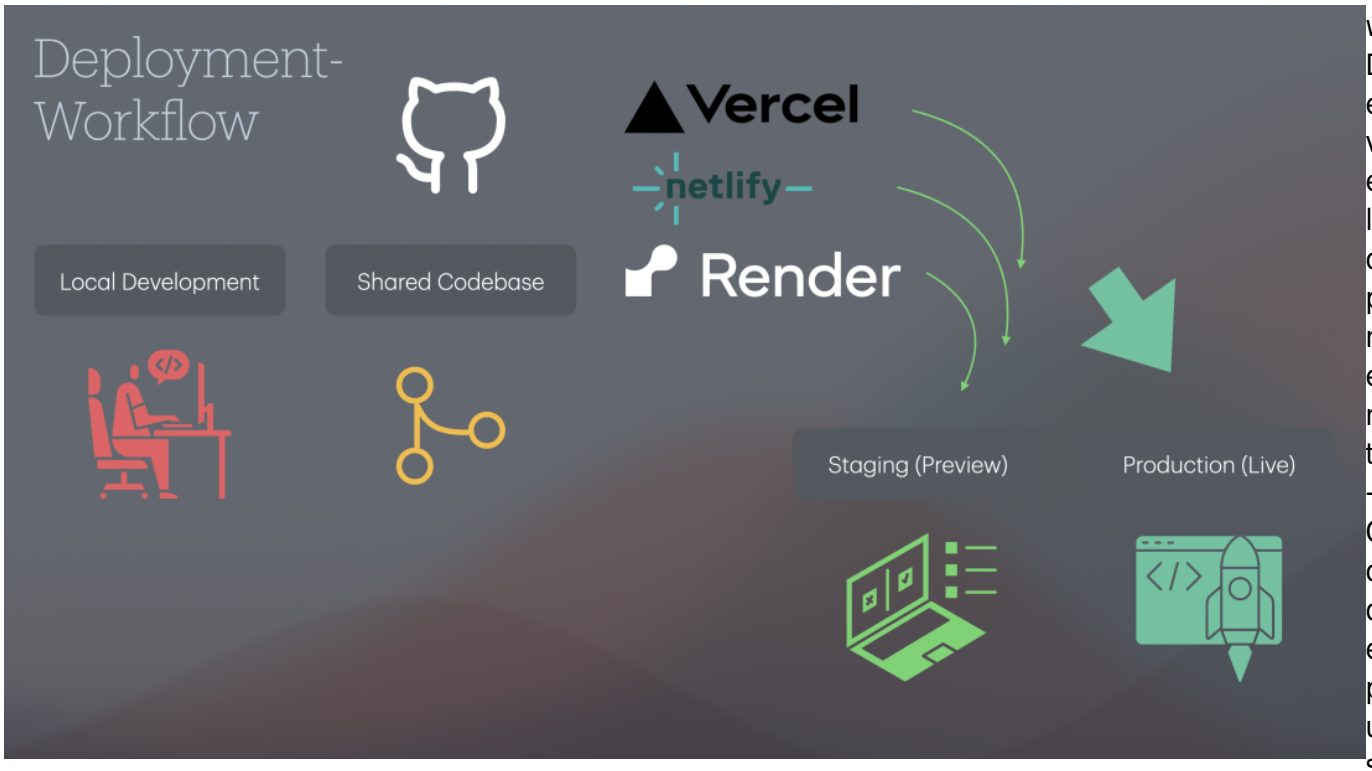
Vorher (Entwicklung)	**Nachher (Production)**
Zeilenumbrüche, Einrückungen	Alles in einer Zeile
Lange Variablennamen	Kurze Namen (`a`, `b`, `fn`)
Kommentare	Entfernt



Zweiter Block: 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

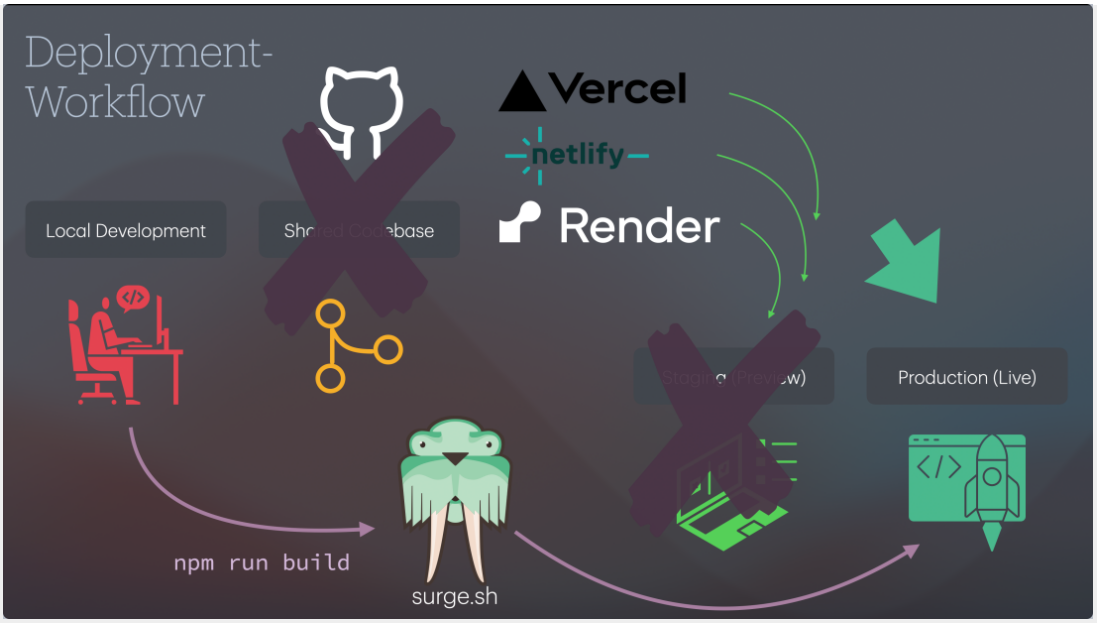


wird. Development → Code push zu GitHub → Vercel/Netlify/Render → Preview → Production (Live)

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:



Vue.js mit npm run build und surge.sh deployen

==== 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
```

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.