LU10a - Statische Webseiten veröffentlichen (Nachtrag)

Definitionen / Unterscheidung

Browser • Applikation, um Webinhalte (HTML, CSS, JS) darzustellen	**Webserver** • Nimmt HTTP(S)-Requests entgegen und liefert Antworten zurück an den Browser • Die Inhalte können statisch oder dynamisch sein
Frontend • Der Teil der Webseite, der an den Aufrufer ausgeliefert wird und im Browser läuft	**Backend** • Läuft auf dem Server und verarbeitet Anfragen • Stellt Daten oder Funktionen über APIs bereit
Statische Webseite • Besteht aus fertigen Dateien (HTML, CSS, JS), die direkt ausgeliefert werden	**Dynamische Webseite** • Inhalte werden abhängig vom Request zur Laufzeit erzeugt (.php→ .html, …)

SSR vs. Entkoppelte Frontend/Backend-Architektur

In modernen Webprojekten wird häufig eine klare Trennung von Frontend und Backend umgesetzt:

Frontend: z. B. mit Angular, React oder Vanilla JS Backend: REST- oder GraphQL-API (z. B. in Node.js, Java, Python)

Diese Architektur bietet klare Vorteile:

- Lose Kopplung (Frontend und Backend unabhängig entwickelbar)
- Austauschbarkeit (z. B. Frontend-Technologie wechseln)
- Einfache Deployments (statische Dateien + API)
- Gute Skalierbarkeit

Static vs dynamic hosting

Ob etwas statisch deploybar ist, hängt nicht von den Features ab, sondern davon, ob der Server zur Laufzeit am Rendering beteiligt ist.

Es ist also eine Architekturfrage, ob man Server-Side-Rendering (SSR) einsetzt oder nicht.

Salopp gesagt, können die meisten Webapplikationen in ein statisches Frontend und ein (allfälliges) dynamisches Backend aufgeteilt werden.

Statische Webseiten sind möglich bei

- Vanilla
- Angular
- React
- Vite
- Astro (default)

Ausnahmen sind

- PHP
  - .php
- Next.js mit API/SSR
  - getServerSideProps
  - Routing (route.js, route.ts, „/api/*“)
  - NextResponse
  - cookies()
  - headers()
- Express mit Templates
  - .ejs
  - .pug
  - .hbs

Next.js

`next.config.js`

```
const nextConfig = {
  output: 'export',
};

module.exports = nextConfig;
```

Nach dem erfolgreichen Ausführen von folgendem Befehl, sind die fertigen Dateien unter `/out` verfügbar.

```
npm run build
```

```
npx serve out
```

Mögliche Fehler:

- „Route Handlers are not supported with output: export“
- „Dynamic server usage“