012-Druckansicht-Wartungsbericht

Frontend-Aufgabe
Anforderungsniveau: Einfach

1 Ziel der Aufgabe

Wartungsberichte werden am Bildschirm gelesen und anschließend als Dokument abgelegt oder unterschrieben. Sie bauen aus einem vorgegebenen JSON-Bericht eine Bildschirmansicht, die sich ohne zweite, separat gepflegte Komponente als mehrseitiges A4-Dokument drucken lässt. Geprüft wird, ob Sie Druck-Stylesheets beherrschen, Seitenumbrüche bewusst steuern und eine einzige Datenquelle für beide Darstellungen behalten.

Veranschlagte Bearbeitungszeit: 2–3 Stunden.

Verwendete Techniken:

- React mit TypeScript (Next.js oder Vite), Daten aus einer lokalen JSON-Datei
- CSS-Druck-Stylesheets: `@media print` und `@page`
- Seitenumbruch-Steuerung mit `break-inside` und wiederholtem Tabellenkopf
- Eine Komponente für Bildschirm und Druck – der Unterschied entsteht nur über CSS

2 Aufgabenbeschreibung

1. **Projekt aufsetzen**

   Legen Sie ein React-Projekt mit TypeScript an (Next.js mit App Router oder Vite mit React). Styling frei wählbar, solange Sie echte Druckregeln schreiben können.

```
npm create vite@latest wartungsbericht -- --template react-ts
cd wartungsbericht && npm install && npm run dev
```

2. **Datenquelle anlegen**

   Legen Sie die Datei `src/data/bericht.json` mit dem folgenden Aufbau an. Ergänzen Sie `positionen` auf rund 25 erfundene Einträge, damit der Druck über mehr als eine Seite läuft. Das Feld `status` kennt genau drei Werte, die alle vorkommen sollen: `OK`, `NIO` (nicht in Ordnung) und `NP` (nicht prüfbar).

```
{
  "berichtsnummer": "WB-2025-0147",
  "erstelltAm": "2025-11-04T09:15:00Z",
  "anlage": {
    "bezeichnung": "Waermepumpe Aussenbereich Nord",
    "hersteller": "Musterthermik", "typ": "MT-250-A",
    "seriennummer": "AAAA-0000-BBBB",
    "standort": "Musterstadt, Beispielweg 12, Technikraum UG"
  },
  "techniker": { "name": "M. Krause", "betrieb": "FireFleeb" },
  "positionen": [
    { "nr": 1, "gewerk": "Hydraulik",
```

```
      "taetigkeit": "Anlagendruck geprueft und nachgefuellt",
      "messwert": 1.8, "einheit": "bar",
      "status": "OK", "bemerkung": "" },
    { "nr": 2, "gewerk": "Elektrik",
      "taetigkeit": "Klemmen Schaltschrank nachgezogen",
      "messwert": null, "einheit": null,
      "status": "NIO", "bemerkung": "Klemme L2 oxidiert" }
  ],
  "gesamtbewertung": "Anlage betriebsbereit, Nacharbeit an Klemme L2 offen",
  "unterschriften": { "techniker": "M. Krause", "kunde": null }
}
```

3. **Bildschirmansicht bauen**

   Stellen Sie den Bericht als eine Seite dar, gegliedert in klar erkennbare Abschnitte:

   - Kopf mit Berichtsnummer, Erstellungsdatum, Techniker und Betrieb
   - Anlagenblock (Bezeichnung, Hersteller, Typ, Seriennummer, Standort)
   - Tabelle der Positionen mit Nr., Gewerk, Tätigkeit, Messwert mit Einheit, Status und Bemerkung
   - Abschlussblock aus Gesamtbewertung und einem Unterschriftenfeld für Techniker und Kunde

   Das JSON importieren Sie direkt; ein Ladezustand ist für diese Aufgabe nicht gefordert.

4. **Druckansicht aus derselben Komponente**

   Bildschirm und Druck zeigen denselben DOM-Baum; der Unterschied entsteht ausschließlich in CSS. Bedienelemente werden im Druck ausgeblendet, der Bericht nutzt die volle Seitenbreite.

```
@page {
  size: A4 portrait;
  margin: 18mm 15mm 20mm 15mm;
}

@media print {
  .no-print { display: none !important; }
  body { background: #fff; }
  .bericht { max-width: none; margin: 0; box-shadow: none; }
}
```

5. **Seitenumbrüche steuern**

   Der Druck soll sich wie ein gesetztes Dokument verhalten, nicht wie eine abgeschnittene Webseite:

   - Der Kopf der Positionstabelle wiederholt sich auf jeder Druckseite, über die sich die Tabelle erstreckt. Lösen Sie das über den Tabellen-`thead`, nicht über ein zweites, separat gepflegtes Kopf-Markup.
   - Eine Positionszeile wird nie mitten im Text umbrochen.
   - Der Abschlussblock bleibt zusammen und rutscht bei Bedarf komplett auf die nächste Seite.

```
@media print {
  thead { display: table-header-group; }
  tr { break-inside: avoid; }
  .abschluss { break-inside: avoid; }
}
```

6. **Druckauslösung**

   Ein Button „Bericht drucken“ ruft `window.print()` auf. Er gehört selbst nicht in den Ausdruck. Ohne Druckdialog prüfen lässt sich das Ergebnis in den DevTools unter `Rendering` mit `Emulate CSS media type: print`.

3 Anforderungen

- React mit TypeScript, keine Verwendung von `any`. Die Struktur des Berichts ist als Typ beschrieben, `null`-Werte (Messwert, Kundenunterschrift) sind im Typ abgebildet und in der Darstellung sauber behandelt.
- Genau eine Komponente rendert den Berichtsinhalt. Eine zweite, parallel gepflegte Druckkomponente gilt als nicht erfüllt.
- Der Ausdruck umfasst mindestens zwei A4-Seiten, ohne abgeschnittene Zeilen, ohne leere Seiten und ohne horizontalen Überlauf.
- Der wiederholte Tabellenkopf stammt aus dem `thead`; die vom Browser erzeugte Kopf- und Fußzeile zählt dafür nicht.
- Der Status ist nicht allein über Farbe erkennbar: das Textkürzel steht in jedem Fall in der Zeile, damit der Ausdruck in Schwarzweiß lesbar bleibt.
- Semantisches HTML (`table`, `thead`, `th` mit `scope`, Überschriftenhierarchie).

4 Abgabe

- Git-Repository mit nachvollziehbaren Commits, `node_modules` nicht eingecheckt.
- README mit Startanleitung und drei bis fünf Sätzen dazu, wie die Druckansicht entsteht.
- `beispiel-bericht.pdf` – ein aus Ihrer Anwendung gedruckter mehrseitiger Bericht.

5 Bewertungskriterien

- **Druckergebnis** – Stimmen Seitenränder, Umbrüche, wiederholter Tabellenkopf und Lesbarkeit in Schwarzweiß?
- **Eine Quelle, zwei Ausgaben** – Gibt es wirklich nur einen Berichtsbaum, und ist die Trennung zu den Druckregeln sauber?
- **Qualität der CSS-Regeln** – Gezielte Regeln statt breiter `!important`-Fluchten.
- **Einhaltung der Vorgaben** – Typisierung ohne `any`, semantisches Tabellen-Markup, Druckbutton nicht im Ausdruck.