# LB02 – Leistungsbeurteilung Vue.js Projekt coden

**Hinweis:** Die Leistungsbeurteilung findet Ende Juni statt. Dieses Dokument gibt Ihnen bereits jetzt einen vollständigen Überblick, damit Sie sich optimal vorbereiten können.

## Was Sie erwartet

Sie erhalten zu Beginn der Leistungsbeurteilung:

- Ein **Starter-Projekt** (Vue.js bereits konfiguriert, `npm install` noch nötig)
- Ein **Figma-Design** mit dem zu bauenden Interface
- Eine **API-URL** mit den Daten, die Sie abrufen und anzeigen sollen
- Ein **kurzes Anleitungsblatt** mit der groben Vorgehensweise

Die Art der Aufgabe ist Ihnen bekannt – sie ist ähnlich aufgebaut wie die Live-Coding-Anleitungen aus dem Unterricht. Der Inhalt (Design, Daten) ist neu, das Muster kennen Sie bereits.

## Was Sie umsetzen müssen

| Anforderung | Beschreibung |
|---|---|
| **Komponenten** | 2 eigene `.vue`-Dateien (ohne `App.vue`) erstellen oder bearbeiten |
| **Fetch** | Daten von der gegebenen API mit `async/await` und `fetch()` laden |
| **Anzeige** | Geladene Daten mit `v-for` als Liste oder Kacheln (=rechteckige Divs) darstellen |
| **Props** | Daten von einer Eltern- an eine Kind-Komponente übergeben |
| **Reaktivität** | `ref()` und Bindings (`{{ }}`, `:`, `v-bind`) korrekt einsetzen |
| **Ladezustand** | Loading-Anzeige und Fehlerhandling mit `v-if / v-else-if / v-else` |
| **Event** | `@click` oder `v-on`-Event mit einer Funktion |
| **Styling** | Komponenten mit Flexbox oder CSS-Grid layouten und nach dem Design gestalten (Fonts, Farben) (keine Responsiveness - Mobile only) |
| **Transition** | einen Übergang zwischen zwei Zuständen mit CSS-Transition animieren |
| **Bonus** | Eine Keyframe-Animation einbauen (wird vorgeschlagen) |

## Zeitplan

| Zeit | Phase |
|---|---|
| 10 Min. | Starterpaket mit Code und Figma downloaden, Code starten mit `npm install` & `npm run dev`, Anleitungsblatt studieren, Fragen stellen – **kein Coding, keine Tastatur** |
| ~60 Min. | Coding |
| 5 Min. | Abgabe vorbereiten und in Moodle hochladen |

**Wichtig:** 60 Minuten effektive Coding-Zeit sind knapp. Schauen Sie regelmässig auf die Uhr. Wenn etwas zu lange nicht funktioniert: kommentieren Sie es aus und arbeiten Sie weiter. Nicht hängen bleiben.

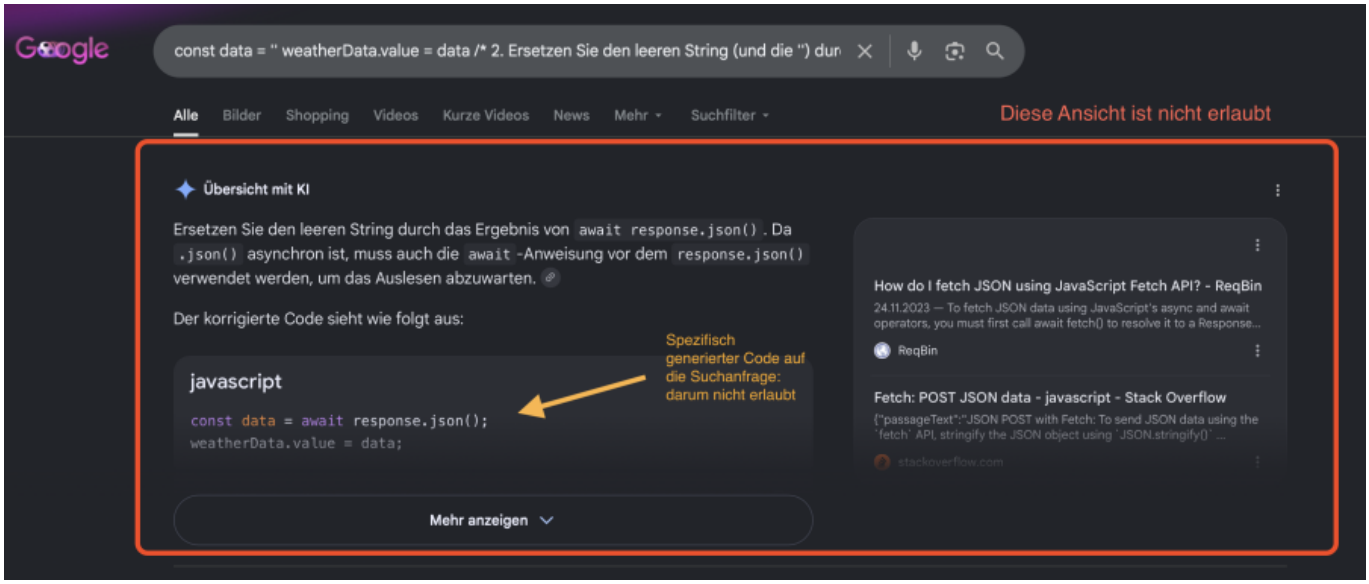
# Regeln & Fairness

| Erlaubt | Nicht erlaubt |
|---|---|
| Google (nur mit Web) aktiviert / DuckDuckGo (keine KI-Funktionen → Erklärung unter dieser Tabelle) | Bing, Google Suche mit „KI Übersicht“, ChatGPT, Claude, Gemini, Copilot, Junie oder andere KI |
| MDN Web Docs (mdn.dev) | Code von anderen Lernenden oder externen Personen (live-Kopien, Duplikate) |
| Vue.js Dokumentation (vuejs.org) | Figma-to-Code oder Design-To-Code Plugins |
| Moodle-Inhalte (moodle.bzz.ch) und eigene Notizen | KI-Extensions im Editor |
| Code-Snippets (= Autocomplete im Editor ≤ 7 Zeilen) | Chats-Fenster offen lassen (Teams, WhatsApp, Chatbots etc.) |

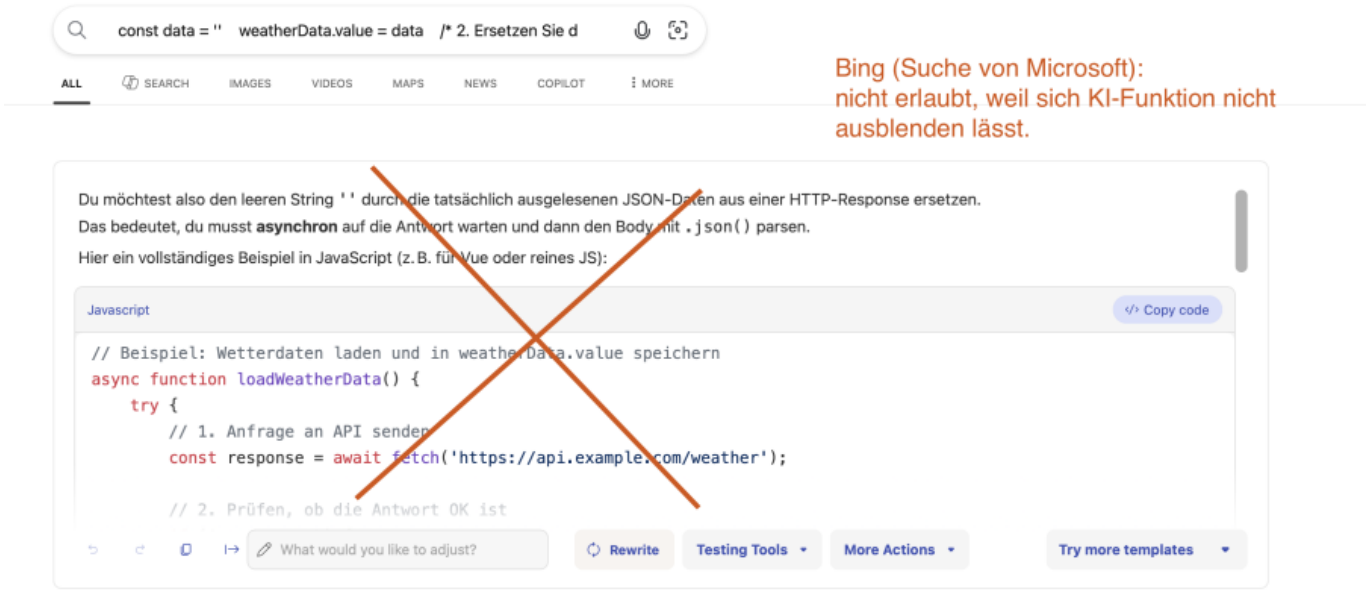
## Websuch-Maschinen

### Nicht erlaubt:

#### Google Suche mit KI-Übersicht aktiv

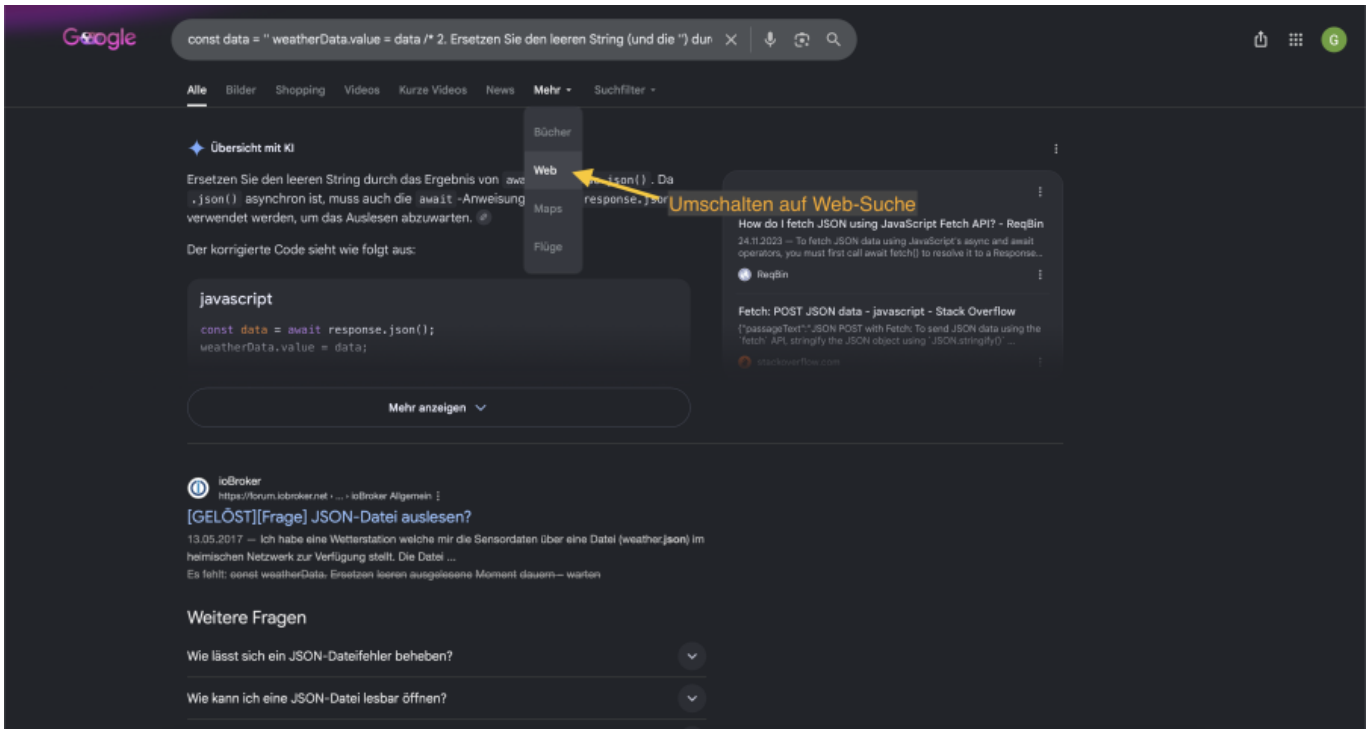



**Bing Suche (Microsoft)**

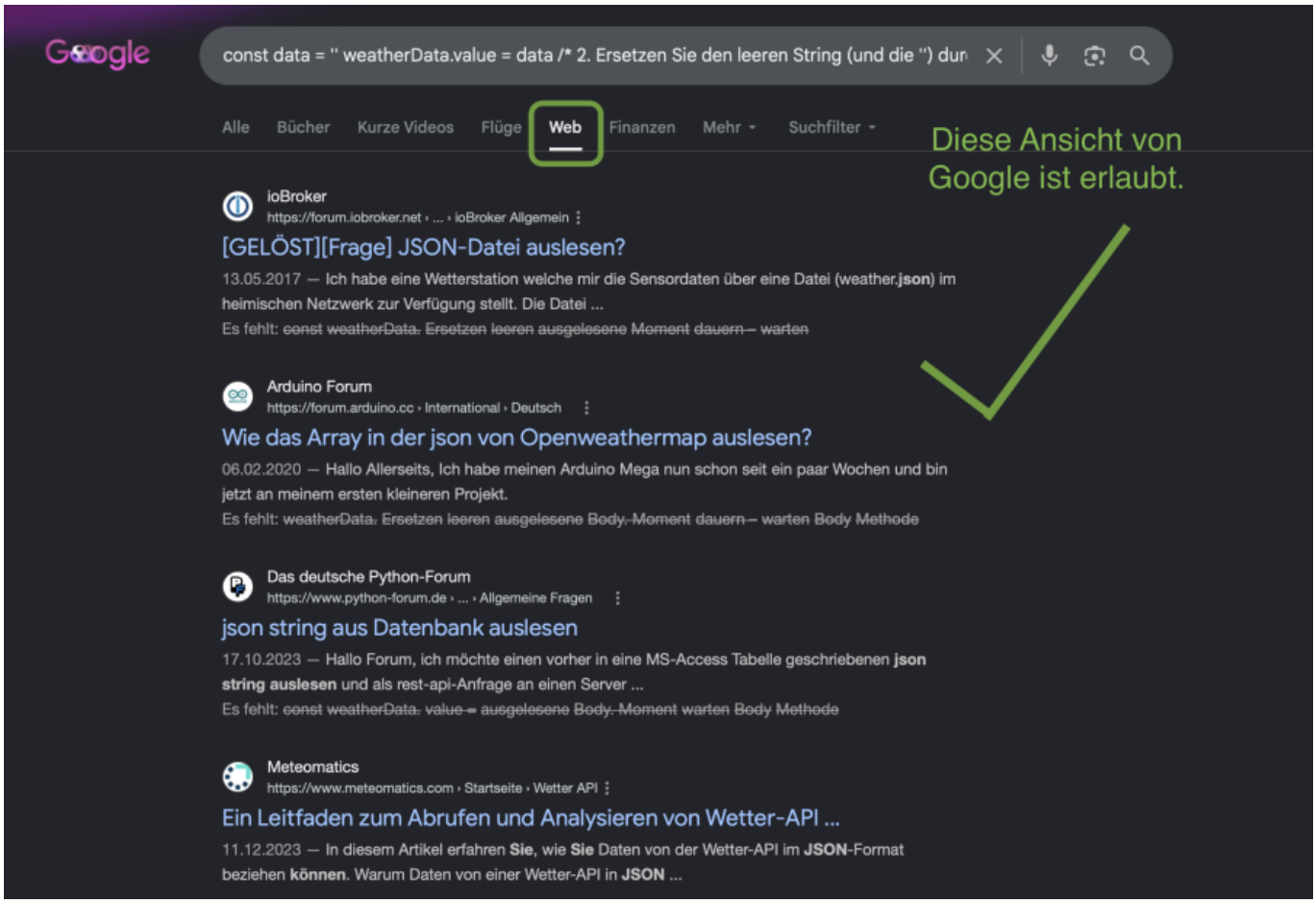



## Erlaubt:

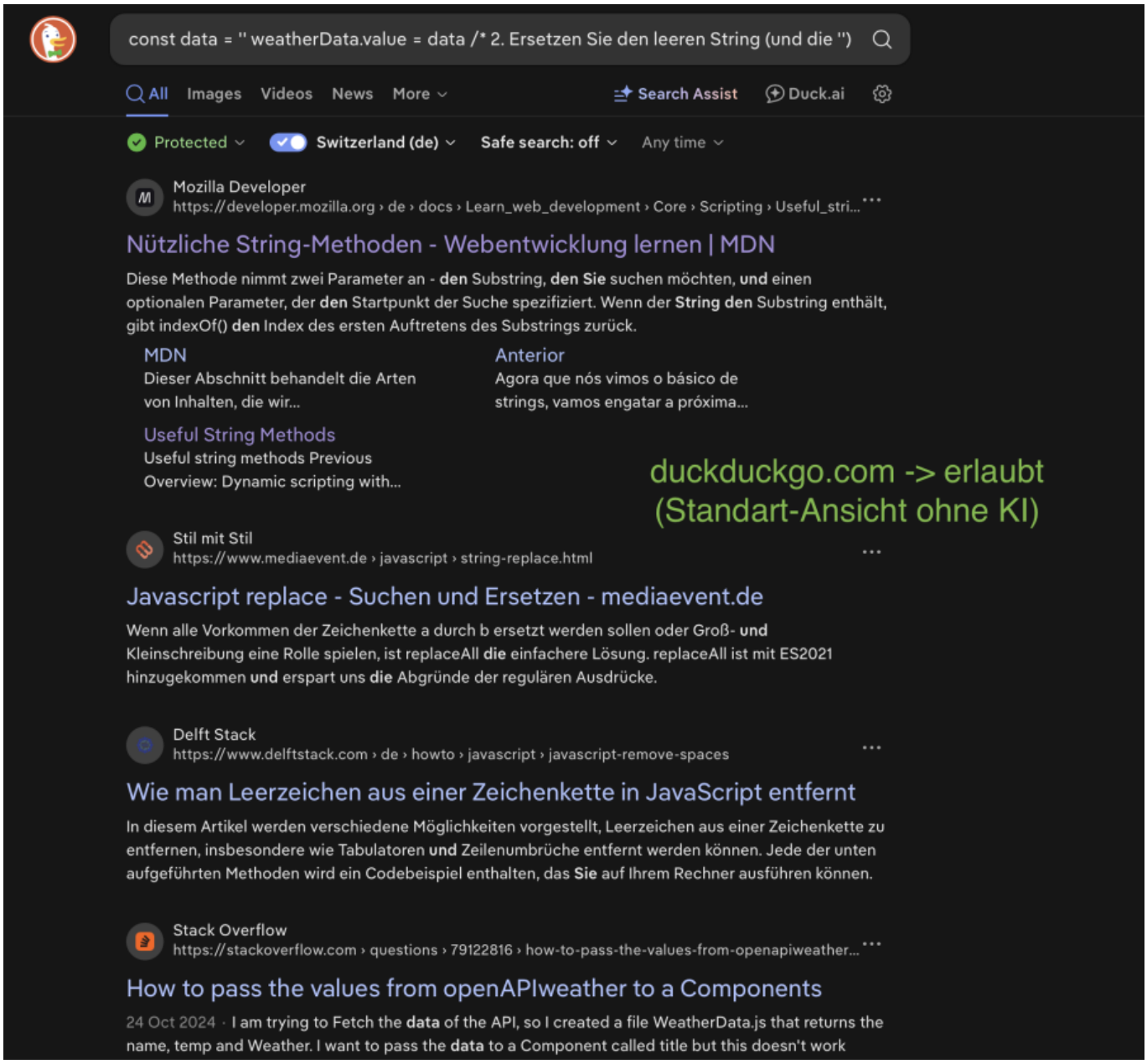
**Google Suche: KI-Übersicht abschalten**




**Google Suche: ohne KI Übersicht**



**Duckduckgo: ohne KI Übersicht**




**Zusätzliche Hinweise:**

- Fehlermeldungen im Editor und im Browser lesen und verstehen (→ LU07)
- Prettier (VS Code) oder Auto-Format (Webstorm) aktiv lassen – sauberer Code macht Bugs sichtbarer
- Computer **vollständig geladen** (Akku) mitbringen
- **Bildschirm hell einstellen**, damit die Lehrperson sehen kann, was Sie tun
- Keine Screen-Filter (Polarfilter o.ä.), die den Blick von der Seite blockieren → falls nicht entfernbar: Schulcomputer im Voraus einrichten und üben

# Konsequenzen bei Betrug

Wird der Einsatz von AI oder das Kopieren von Code erkannt oder begründet vermutet: In diesem Fall erfolgt eine mündliche Nachprüfung (LB04), die 50% der Modulnote ausmacht.

# Abgabe

Ihr Projekt als `.zip`-Datei in Moodle hochladen:

```
vorname_nachname_klassenname_LB02_M291.zip
```

**Wichtige Regeln:**

- `node_modules`-Ordner **nicht** einschliessen → Nicht umgesetzt, kostet Punkte
- `.idea`-Ordner **nicht** einschliessen → Nicht umgesetzt, kostet Punkte
- Keine Umlaute oder Sonderzeichen im Dateinamen: ä → ae, ö → oe, ü → ue, é → e usw. → Nicht umgesetzt, kostet Punkte
- **Nur** den Vue.js-Projektordner zippen – nichts anderes → Nicht umgesetzt, kostet Punkte

# Vorbereitung – was Sie jetzt tun sollten

Nutzen Sie die verbleibende Zeit, um sich gezielt vorzubereiten. Die folgenden Punkte sind prüfungsrelevant:

- `npm install` und `npm run dev` im Terminal im Projektordner ausführen können
- Das Vue-Projekt im Browser anzeigen (via `localhost:5173`)
- Neue Komponenten (`.vue`-Files) erstellen und importieren können
- Grundstruktur von Komponenten erzeugen (`<script setup>`, `<template>`, `<style>`)
- `fetch()` mit `async/await` und `onMounted` anwenden können
- Daten mit `v-for` und Props darstellen können
- `v-if / v-else-if / v-else` für Ladezustand und Fehler einsetzen können
- ZIP-Datei korrekt erstellen (ohne `node_modules`) und in Moodle hochladen üben
- Inhalte zu Vue.js, Frameworks, CSS repetieren

# Fragen

Zu Beginn der Prüfung haben Sie **10–15 Minuten** Zeit, Figma-Design und Anleitungsblatt zu lesen und Fragen zu stellen. Diese werden im Plenum beantwortet. Nach dem Start der Prüfung ist keine individuelle Unterstützung durch die Lehrperson mehr möglich – üben Sie deshalb das selbstständige Debuggen.

Bei Fragen zum Briefing wenden Sie sich jetzt an die Lehrperson – nicht erst kurz vor der Leistungsbeurteilung.

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

Permanent link:
**https://wiki.bzz.ch/de/modul/m291/leistungsbeurteilungen/02_lb/a_briefing?rev=1781805236**

Last update: **2026/06/18 19:53**